

Titre: QED-Tutrix : système tutoriel intelligent pour l'accompagnement des
élèves en situation de résolution de problèmes de démonstration en
géométrie plane
Title:

Auteur: Nicolas Leduc
Author:

Date: 2016

Type: Mémoire ou thèse / Dissertation or Thesis

Référence: Leduc, N. (2016). QED-Tutrix : système tutoriel intelligent pour l'accompagnement
des élèves en situation de résolution de problèmes de démonstration en
Citation: géométrie plane [Ph.D. thesis, École Polytechnique de Montréal]. PolyPublie.
<https://publications.polymtl.ca/2450/>

 Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie:
PolyPublie URL: <https://publications.polymtl.ca/2450/>

**Directeurs de
recherche:** Michel Gagnon, & Philippe R. Richard
Advisors:

Programme: Génie informatique
Program:

UNIVERSITÉ DE MONTRÉAL

QED-TUTRIX : SYSTÈME TUTORIEL INTELLIGENT POUR
L'ACCOMPAGNEMENT D'ÉLÈVES EN SITUATION DE RÉOLUTION DE
PROBLÈMES DE DÉMONSTRATION EN GÉOMÉTRIE PLANE

NICOLAS LEDUC
DÉPARTEMENT DE GÉNIE INFORMATIQUE ET GÉNIE LOGICIEL
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

THÈSE PRÉSENTÉE EN VUE DE L'OBTENTION
DU DIPLÔME DE PHILOSOPHIÆ DOCTOR
(GÉNIE INFORMATIQUE)
NOVEMBRE 2016

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Cette thèse intitulée :

QED-TUTRIX : SYSTÈME TUTORIEL INTELLIGENT POUR
L'ACCOMPAGNEMENT D'ÉLÈVES EN SITUATION DE RÉOLUTION DE
PROBLÈMES DE DÉMONSTRATION EN GÉOMÉTRIE PLANE

présentée par : LEDUC Nicolas

en vue de l'obtention du diplôme de : Philosophiæ Doctor

a été dûment acceptée par le jury d'examen constitué de :

M. DESMARAIS Michel C., Ph. D., président

M. GAGNON Michel, Ph. D., membre et directeur de recherche

M. RICHARD Philippe R., Ph. D., membre et codirecteur de recherche

Mme CARON France, Ph. D., membre

Mme PY Dominique, Ph. D., membre externe

DÉDICACE

*À la mémoire de mon amie Martine Jeffrey
qui, par son courage, m'a insufflé l'intrépidité
requise pour entreprendre une telle aventure.*

*À mon filleul Thomas Leduc
qui, par son entrain, m'a transmis l'énergie
nécessaire afin de parachever cette odyssée.*

REMERCIEMENTS

Je tiens premièrement à remercier Michel, mon directeur de recherche, ainsi que mon codirecteur Philippe qui m'ont soutenu tout au long de mes études doctorales. Ils ont fait preuve d'une patience remarquable et ont toujours eu une confiance inébranlable en moi, même lorsque je songeais à tout abandonner. Ils m'ont donc permis de mener à terme ce projet et de faire partie d'une équipe de recherche exceptionnelle.

Je veux ensuite remercier les élèves de 4^e et de 5^e secondaire de l'École d'Éducation Internationale de Laval et leurs enseignants, Huguette et Stéphane ainsi que les élèves de 4^e secondaire de l'Académie Ste-Thérèse et leurs enseignantes, Jenny et Sylvie. Leur générosité et leur perspicacité ont contribué au succès du projet. Sans eux, QED-Tutrix n'aurait jamais vu le jour, car il est inspiré des interventions de ces enseignants auprès des élèves.

Je désire aussi souligner l'apport du CRSH qui a subventionné le projet. Sans ce financement, il m'aurait été impossible de m'investir à temps plein dans mes études doctorales. J'aimerais également mettre en valeur la contribution des membres du jury qui ont accepté d'évaluer la qualité ainsi que la validité scientifique du travail réalisé.

Je m'en voudrais de passer sous silence le soutien constant de ma famille. Mes parents, Jean-Claude et Louise, m'ont toujours encouragé à persévérer, malgré les obstacles que j'ai dû affronter, et ont su respecter mon rythme de travail sans me mettre de pression. Mes frères, Mathieu et Olivier, ainsi que leurs familles respectives ont toujours été là pour me remonter le moral et pour organiser des après-midi de jeux de société fort appréciés.

J'exprime ma reconnaissance à mes amis, Jean-François et Valérie, qui, malgré leurs problèmes personnels, ont toujours été présents et à l'écoute de mes plaintes et de mes angoisses sans jamais me juger. Les diners et soupers improvisés qu'ils organisent sont toujours une bonne occasion de relaxer. Bravo aussi à ma collègue, et à ma nouvelle amie, Michèle qui a enduré mon humeur changeante et qui a réussi à adoucir mon caractère.

Je tiens enfin à mentionner le travail remarquable de l'équipe médicale qui m'aide à gérer ma maladie. Je pense plus particulièrement à Dr Massicotte, rhumatologue, M. Lussier, psychologue, et Dre Chabane qui sont toujours prêts à répondre à mes questions et à m'encourager. Le travail de l'équipe de l'Hôpital Juif de Réadaptation, en particulier Roula et Shirlene, est aussi digne de mention. Sans eux, il y a longtemps que je ne sortirais plus de la maison et que j'aurais abandonné le projet.

RÉSUMÉ

Au cours des dernières années, le système scolaire québécois impose une pression croissante sur les enseignants. En effet, ceux-ci doivent gérer des classes de plus en plus nombreuses tout en maintenant un soutien adéquat à l'apprentissage des élèves. Dans ce contexte, l'utilisation de systèmes tutoriels intelligents qui sont en mesure d'assister l'enseignant dans son travail pourrait permettre à ce dernier de consacrer plus de temps aux élèves qui en ont véritablement besoin. Malheureusement, dans le domaine de l'enseignement des preuves en géométrie, l'offre de systèmes tutoriels est limitée. De plus, ceux actuellement proposés forcent l'élève à travailler selon un ordre déterminé et ils ne fournissent pas de soutien dans le cadre d'une exploration libre du problème. Ils l'obligent aussi à rédiger des preuves formelles qui ne sont pas adaptées aux exigences des enseignants du secondaire.

Partant de ce constat, nous avons établi l'objectif principal de notre projet qui consiste à proposer un système tutoriel intelligent qui assiste l'élève dans une démarche d'exploration plutôt que de rédaction dans le cadre de l'élaboration d'une preuve en géométrie. Dans le but de l'atteindre, nous proposons le système QED-Tutrix qui a été conçu à la suite d'une analyse des interventions d'enseignants réels observés. Il permet à l'enseignant ou au didacticien de construire un ensemble de preuves acceptables pour un problème donné en fonction de l'objectif d'apprentissage visé. L'élève peut ensuite tenter de résoudre le problème choisi par l'enseignant en utilisant les différents outils offerts à l'interface de QED-Tutrix. Celui-ci a accès à une figure dynamique afin de découvrir des conjectures, à un répertoire d'énoncés pour composer sa preuve et à une démonstration qu'il doit compléter. Tous les énoncés proposés sont analysés par le système qui génère des rétroactions à l'intérieur d'une fenêtre de clavardage afin de guider l'élève lors de l'exploration et de la résolution du problème.

L'élaboration de QED-Tutrix a été réalisée par une équipe multidisciplinaire composée d'experts en didactique et en informatique. Le système a été construit itérativement par la mise en oeuvre du paradigme de la conception dans l'usage qui est constitué d'une succession de plusieurs cycles de recherche et de développement. Chaque cycle se clôture par une expérimentation qui vise à valider le travail accompli et à recueillir des informations qui sont réinvesties dans le cycle suivant. Une première version du système (GeoGebraTUTOR) a donc été créée afin d'étudier notamment les interventions d'enseignants réels qui ont inspiré l'élaboration de la seconde version (QED-Tutrix), qui est décrite et analysée dans cette thèse.

Nous ne prétendons pas que la version actuelle du système a un effet mesurable sur les résultats scolaires, car nous visons, pour l'instant, à permettre à un élève de travailler en

conformité avec des théories didactiques reconnues. En effet, la conception de QED-Tutrix s’ancre principalement dans la théorie des situations didactiques qui permet de représenter une situation didactique par une relation élève-milieu. Nous utilisons cependant une version étendue de cette théorie dans laquelle un agent tutoriel, qui joue le rôle d’un enseignant virtuel, peut agir sur la relation élève-milieu. De plus, nous désirons offrir un système tutoriel qui est un véritable espace de travail géométrique, c’est-à-dire qu’il permet à l’élève de résoudre des problèmes en mettant en oeuvre les trois démarches définies dans cet espace.

Ces théories didactiques ainsi que les résultats de nos observations ont été implantés dans QED-Tutrix. Il en est résulté un système comportant quatre couches logicielles principales. La première permet de modéliser l’ensemble des démonstrations possibles pour résoudre un problème donné. Pour chaque problème, l’enseignant inscrit toutes les inférences, ou pas de démonstration, qui sont acceptables pour sa résolution selon l’objectif d’apprentissage visé. Chaque inférence contient une justification qui est utilisée pour produire un conséquent à partir de l’ensemble de ses antécédents. Il est possible de les combiner afin d’obtenir un graphe contenant toutes les solutions valides, car les conséquents peuvent être recyclés pour former les antécédents d’autres inférences. Le parcours du graphe contenu dans cette première couche permet donc d’énumérer les différentes solutions au problème représenté.

Afin de proposer une aide qui respecte l’état cognitif de l’élève lors de l’exploration d’un problème, il est essentiel de conserver la chronologie de ses actions. Nous l’avons donc modélisée à l’intérieur de la deuxième couche de notre système. Celle-ci contient des données dynamiques qui sont mises à jour au cours de la résolution d’un problème, à l’opposé du graphe qui est statique, et elle se superpose à ce dernier. En effet, nous indiquons, pour chaque noeud du graphe, le temps d’activation le plus récent qui correspond à l’écriture de l’énoncé qui lui est attaché. Cette approche se démarque de celle des autres systèmes, car ces derniers n’utilisent pas la chronologie des actions, étant donné qu’ils imposent une séquence de résolution.

Pour être en mesure de suggérer différentes pistes de solution à un élève bloqué dans son processus de résolution, nous avons choisi de traiter les inférences selon un ordre de priorité. Ce classement est réalisé par la troisième couche de QED-Tutrix, qui utilise les données des deux couches précédentes. Pour l’obtenir, nous recherchons d’abord la solution la plus avancée à l’aide d’une heuristique originale, que nous avons élaborée et qui permet d’éviter d’énumérer toutes les solutions. Nous affectons ensuite des priorités plus élevées aux inférences faisant partie de la solution déterminée et qui ont été travaillées récemment par l’élève, afin de respecter son état cognitif. La proposition d’autres pistes nous démarque des systèmes tutoriels traditionnels qui offrent de l’aide uniquement pour compléter une solution optimale.

La liste ordonnée d'inférences est utilisée par la dernière couche du système, soit celle qui produit les différentes rétroactions. Premièrement, QED-Tutrix offre des rétroactions instantanées, en réponse à l'écriture de chaque énoncé, sous forme d'émoticônes et de messages courts. Il permet aussi d'encoder des erreurs courantes afin de leur associer des messages précis. Il offre enfin une aide à la prochaine étape qui est inspirée des interventions des enseignants réels observés. Cette dernière forme d'aide a été modélisée par une machine à états finis qui traite séquentiellement les inférences ordonnées dans la liste et produit une série d'indices permettant de les compléter. Des messages doivent être composés pour chacune des inférences, mais des mécanismes ont été implantés afin de réduire leur nombre. Les rétroactions offertes sont comparables à celles d'autres systèmes tutoriels.

Environ 450 inférences ont été produites et près de 900 messages composés afin d'implanter les cinq problèmes actuellement offerts dans notre système. Son fonctionnement a d'abord été vérifié par un expert indépendant. Celui-ci a confirmé que les messages produits étaient conformes à la structure déterminée, mais que l'évaluation de la solution la plus avancée était parfois problématique. QED-Tutrix a ensuite été utilisé par des élèves de 4^e secondaire. Ceux-ci ont généralement trouvé le système utile et ont apprécié l'expérience. L'analyse des enregistrements nous a permis de constater que la structure des messages générée permet d'aider certains élèves. De plus, nous avons observé la mise en oeuvre des différentes démarches, ce qui confirme le statut d'espace de travail géométrique de QED-Tutrix. L'efficacité de ce dernier est, par contre, limitée dans le cas d'élèves plus faibles, car la structure des messages est calibrée afin d'aider des élèves moyens. Le problème concernant l'évaluation de la solution la plus avancée a aussi provoqué la production de messages incohérents avec la stratégie de l'élève. Dans le but d'augmenter l'efficacité du système, nous envisageons, entre autres, de proposer des profils de tuteurs et d'élèves.

Malgré les lacunes qui ont été détectées, il n'en demeure pas moins que QED-Tutrix est un système tutoriel innovateur. En effet, dans le domaine des preuves en géométrie, il est le seul à utiliser des émoticônes et à proposer différentes pistes de solution. De plus, son élaboration itérative, par une équipe multidisciplinaire, permet d'obtenir un système respectueux du travail de l'élève, ce qui se démarque de l'approche traditionnelle qui consiste à tenter de reproduire le raisonnement d'un expert. Les étapes suivantes de conception visent à intégrer des rétroactions sous forme de problèmes connexes et à proposer une aide à la construction de la figure. Notre système pourrait facilement être adapté au traitement des démonstrations en logique de premier ordre. Une adaptation pour le traitement du raisonnement sous forme d'argumentation non formelle pourrait aussi être envisagée. Enfin, la suggestion de diverses pistes de solution pourrait être implantée dans d'autres systèmes tutoriels.

ABSTRACT

In the past years, Quebec's school system imposes a growing amount of pressure on its teaching staff. They must juggle classes with more and more students while assuring the quality of their teaching to each of them. In this context, the use of intelligent tutoring systems which can assist the teacher in his or her work could allow the teacher to dedicate more energy to each student when it's needed. However, the offer for tutoring systems for the learning of proof is limited. Moreover, the available systems force the student to work according to a determined order and they don't provide help in the context of a free exploration of the problem. They also force the student to write formal proofs when high school teachers rarely demand.

With this assessment in mind, we established the principal objective for our project which aims at offering an intelligent tutoring system that assists the student in an exploratory approach when solving geometry proofs instead of a formal proof writing approach. The system we offer is QED-Tutrix which was designed taking into account actual teacher interventions observed in a classroom environment. QED-Tutrix allows the teacher or didactician to construct a number of admissible proofs for a given problem according to the learning goals. The student can then try to solve the problem chosen by the teacher by using the different tools QED-Tutrix puts at his disposal. The student has access to a dynamic geometric figure he can work with in order to make conjectures, as well as to a repertoire of statements to create their proof and an interactive written proof they can use to complete their proof. All the statements he provides are analyzed by the system which then generates feedback through a chat window in order to guide the student during the exploration and solving of the problem.

QED-Tutrix's elaboration was carried out by a multidisciplinary team comprised of experts in didactics of mathematics and in computer science. The system is built in an iterative manner by adopting a design in use approach which consists of a series of many cycles of research and development. Each cycle is ended with an experimentation which aims at validating the work accomplished and at collecting information to be reinvested in the following cycle. A first version of the system (GeoGebraTUTOR) was created to study, among other things, real teacher interventions which inspired the implementation of the second version (QED-Tutrix). This last version is described and analyzed in the following thesis.

We do not claim that the present QED-Tutrix version has measurable effects on academic results, since our aim at the moment is to make sure it allows the student to work in a fashion put forward by known didactic principles. Indeed, QED-Tutrix's conception is rooted mainly

in the didactical situation's theory which represents a didactical context by a student-milieu interaction. However, we use an extended version of this theory in which a tutorial system playing a virtual teaching role may influence this student-milieu interaction. Moreover, we aim at offering a tutoring system that is a true geometrical workspace, meaning that it allows the student to solve problems by engaging in three mathematical processes described in the geometrical workspace model.

The didactic theories and conclusions drawn from our observations were implemented in QED-Tutrix. This resulted in a system made of four main software layers. The first of these layers is for the organization of the different proof solutions for a given problem. For each solution, the teacher or didactician registers all the inferences or proof steps which are admissible according to a specific learning context. Each inference includes a justification that is used to produce a result stemming from a group of premises. It is possible to combine the different inferences in order to generate a graph of all the different admissible solutions, since the results of one inference can be recycled as a premise for another. The different pathways of this graph which is the output of the first software layer allow for the account of the different solutions to each implemented problem.

In order to offer help that takes into account the cognitive state of the student exploring the problem, it is essential to keep track of the chronology of his or her actions. This memory was implemented in the second layer of the system. It contains information that is dynamic and evolves as problem solving occurs. It also overlays the solution graph which is static. Therefore, we indicate for each of the graphs nodes, the most recent activation time associated with the writing of the statement attached to it. This approach stands out from the way other systems operate and in which chronology is usually not taken into account since a solving sequence is imposed.

In order to be able to suggest alternative paths to a student who is stuck in his or her solving process, we chose to treat inferences according to an order of priority. This ranking is carried out by QED-Tutrix's third layer which uses the data from the first two layers. In order to achieve this ordering, we look for the most advanced solution with the help of an original heuristic, elaborated for this project, which spares the system from running through all the admissible solutions. We then assign the highest priorities to the inferences which are part of the identified solution and that has been worked on recently by the student keeping his or her cognitive state in mind. The ability to suggest other solution option distances us from traditional tutorial systems which offer help only to complete an optimal solution.

This list of ranked inferences is used by the fourth and last layer of the system, meaning the layer which generates various feedback. Firstly, QED-Tutrix replies to the writing of each

statement with instant feedback in the shape of emojis or short messages. It also allows the programming of particular messages associated with known common mistakes. Lastly, this layer of the system offers help with the next step which is inspired by actual teacher interventions. This last form of help was modelled by a finite state machine that sequentially treats the ranked inferences from the list and produces a series of hints to help the student complete them. Messages must be created for each of the inferences, but mechanisms are implemented in order to reuse messages according to the inferences content, limiting the number of entries. The feedback offered by these messages is similar in form to the feedback offered by other tutorial systems.

Approximately 450 inferences were produced and close to 900 messages were created in order to implement the five problems currently available in QED-Tutrix. Its operating has been verified by an independent expert, which confirmed that the output of messages is true to the identified structure, but the evaluation of the most advanced solution is sometimes problematic. QED-Tutrix was then put through a second trial in a class of 4th year of high school. The students generally found the system to be useful and appreciated the experience. The analysis of the session recordings revealed that the generated messages help some students. Also, we observed different mathematical processes which confirmed QED-Tutrix's geometrical workspace status. However, the efficiency of QED-Tutrix is limited when helping students with less mathematical abilities since the message structure is built with the average student as a reference. Problems with the identification of the most advanced solution also lead to incoherence between messages and student strategies. In order to enhance the system's efficiency in helping the student solve problems, we contemplate, among other things, to differentiate tutorial profiles according to students solving profiles.

In spite of witnessing shortcomings, QED-Tutrix is an innovative tutorial system. Indeed, in the field of geometry proofs, it is the only automated tutoring system to use emojis and to suggest alternative solution paths. Moreover, the iterative and multidisciplinary approach adopted for its design and development stands out from a traditional approach which aims at reproducing expert reasoning. The next design steps aims at including feedback in the form of related problems and to provide help with building the geometrical figure. Our system could easily be adapted to handle first order logical proofs. An adjustment to process non-formal argumentation could also be considered. Finally, the suggestion of alternative solution paths could be implemented in other tutorial systems.

TABLE DES MATIÈRES

DÉDICACE	iii
REMERCIEMENTS	iv
RÉSUMÉ	v
ABSTRACT	viii
TABLE DES MATIÈRES	xi
LISTE DES TABLEAUX	xv
LISTE DES FIGURES	xvi
LISTE DES ALGORITHMES	xvii
LISTE DES SIGLES ET ABRÉVIATIONS	xviii
LISTE DES ANNEXES	xix
CHAPITRE 1 INTRODUCTION	1
1.1 Enjeux concernant l'enseignement de la démonstration	1
1.2 Concepts relatifs aux démonstrations	4
1.3 Objectifs de recherche	6
1.4 Organisation de la thèse	8
CHAPITRE 2 REVUE DE LITTÉRATURE	10
2.1 Introduction	10
2.2 Ressources en ligne pour l'apprentissage des mathématiques	11
2.2.1 Apprentissage autonome	11
2.2.2 Gestion de curriculum et modélisation de l'apprenant	13
2.2.3 Micromondes	15
2.2.4 Assistant à la démonstration et preuve automatique	17
2.3 Systèmes tutoriels en géométrie	20
2.3.1 Geometry Tutor	20
2.3.2 Geometry Explanation/Cognitive Tutor	22

2.3.3	Advanced Geometry Tutor	24
2.3.4	ANGLE	26
2.3.5	Baghera	28
2.3.6	DEFI	30
2.3.7	Cabri-Euclide	31
2.3.8	Mentoniezh	33
2.3.9	Géométrie	35
2.3.10	Turing	37
2.4	Autres systèmes tutoriels pertinents	39
2.4.1	Apprentissage de la physique	40
2.4.2	Interactions en langue naturelle	42
2.4.3	Preuves logiques	44
2.4.4	Modélisation basée sur les contraintes	46
2.4.5	Outils de création	47
2.5	Principales caractéristiques des systèmes analysés	50
2.6	Conclusion	53
CHAPITRE 3 SURVOL DU SYSTÈME ET FONDEMENTS DIDACTIQUES . . .		55
3.1	Introduction	55
3.2	Aspects ergonomiques de QEDX	56
3.2.1	Fondements didactiques	57
3.2.2	Interface utilisateur	61
3.3	Architecture logicielle de QEDX	68
3.3.1	Graphe HPDIC	69
3.3.2	Modèle Itératif de l'Apprenant (MIA)	70
3.3.3	Évaluation des Démonstrations / Ordonnancement des Inférences (EDO)	70
3.3.4	Génération des Messages Discursifs (GMD)	72
3.4	Méthodologie	73
3.5	Conclusion	77
CHAPITRE 4 REPRÉSENTATION DES DÉMONSTRATIONS (HPDIC) ET MODÉ- LISATION DE L'APPRENTISSAGE (MIA)		79
4.1	Introduction	79
4.2	Problématique et enjeux	80
4.2.1	Problème du rectangle et classification des énoncés	80
4.2.2	Principaux styles de preuve	81
4.2.3	Revue sommaire des modèles de l'apprenant	85

4.3	Définitions formelles du HPDIC et du MIA	87
4.3.1	Graphe HPDIC	87
4.3.2	Modèle Itératif de l'Apprenant (MIA)	95
4.4	Résultats relatifs aux graphes et simulation du MIA	98
4.4.1	Statistiques des graphes HPDIC implantés	98
4.4.2	Validation du MIA	102
4.5	Discussion	104
4.6	Conclusion	108
 CHAPITRE 5 ÉVALUATION DES DÉMONSTRATIONS ET ORDONNANCEMENT		
	DES INFÉRENCES (EDO)	110
5.1	Introduction	110
5.2	Retour sur les principales stratégies d'aide recensées	112
5.3	Contrôle de l'explosion combinatoire des solutions	115
5.3.1	Traitement des énoncés	115
5.3.2	Décomposition d'un graphe en arbres	119
5.4	Évaluation de la progression	122
5.4.1	Méthode par Énoncé (ME)	122
5.4.2	Méthode par Noeud (MN)	123
5.4.3	Méthode par Propagation (MP)	125
5.5	Ordonnancement des inférences	130
5.5.1	Observations concernant la stratégie des enseignants	130
5.5.2	Retour sur quelques définitions formelles du HPDIC et du MIA . . .	132
5.5.3	Description de l'algorithme d'ordonnancement proposé	133
5.6	Résultats obtenus et validations réalisées	138
5.6.1	Réduction de l'explosion combinatoire	139
5.6.2	Données relatives aux arbres extraits des graphes implantés	139
5.6.3	Comparaison des méthodes d'estimation de l'avancement	143
5.6.4	Validation de l'algorithme d'ordonnancement	149
5.7	Discussion	150
5.8	Conclusion	155
 CHAPITRE 6 GÉNÉRATION DES MESSAGES DISCURSIFS (GMD)		
6.1	Introduction	159
6.2	Interactivité des systèmes existants	161
6.3	Analyse des interactions enseignants-élèves	165
6.3.1	Problème du rectangle	165

6.3.2	Description de l'expérimentation initiale	167
6.3.3	Stratégie de l'enseignant typique observé	169
6.3.4	Structure des messages	170
6.4	Mise en oeuvre des rétroactions de QEDX	174
6.4.1	Rétroactions instantanées	175
6.4.2	Machine à états finis produisant les messages d'aide discursifs	179
6.5	Évaluation du GMD	187
6.5.1	Rédaction des messages d'aide	187
6.5.2	Résultats des simulations du GMD	191
6.5.3	Expérimentation en classe	192
6.6	Discussion	196
6.7	Conclusion	202
CHAPITRE 7	RÉSULTATS CONNEXES ET DISCUSSION GÉNÉRALE	204
7.1	Introduction	204
7.2	Résultats se rapportant à l'ergonomie de QEDX	204
7.2.1	Commentaires relatifs à l'interface	205
7.2.2	Précision du statut d'ETG	206
7.3	Discussion	208
7.3.1	Aspects ergonomiques de l'interface	208
7.3.2	Examen de l'architecture informatique	210
7.3.3	Décisions méthodologiques	212
7.4	Conclusion	214
CHAPITRE 8	CONCLUSION GÉNÉRALE	215
8.1	Synthèse des travaux	215
8.2	Limites de la solution proposée	219
8.3	Améliorations futures	221
RÉFÉRENCES		224
ANNEXE		243

LISTE DES TABLEAUX

Tableau 1.1	Éléments d'une inférence	5
Tableau 2.1	Caractéristiques des principaux systèmes tutoriels recensés	51
Tableau 3.1	Caractéristiques des deux phases expérimentales	75
Tableau 4.1	Hypothèses, résultats Intermédiaire et Conclusion (HIC) permettant de résoudre le problème du rectangle	81
Tableau 4.2	Propriétés et Définitions (PD) permettant de résoudre le problème du rectangle	82
Tableau 4.3	Éléments d'une inférence	82
Tableau 4.4	Ensemble des solutions au problème du rectangle	95
Tableau 4.5	Données concernant la structure des cinq graphes HPDIC implantés .	99
Tableau 4.6	Statistiques associées aux solutions des cinq problèmes implantés . . .	100
Tableau 4.7	Solutions uniques pour le problème du rectangle	101
Tableau 4.8	Pourcentage d'avancement pour chaque solution à chaque étape d'une résolution simulée	102
Tableau 5.1	Listes des temps d'activation à la suite d'une séquence d'actions . . .	134
Tableau 5.2	Statistiques relatives aux arbres HPDIC extraits pour cinq problèmes	140
Tableau 5.3	Production d'arbres à partir d'une liste ordonnée de solutions	142
Tableau 5.4	Longueur des démonstrations pour le problème du rectangle telle que calculée par ME, MN et MP	145
Tableau 5.5	Pourcentage d'avancement calculé par ME pour chaque solution à chaque étape d'une séquence typique de résolution	146
Tableau 5.6	Pourcentage d'avancement estimé par MN pour chaque solution à chaque étape d'une séquence typique de résolution	147
Tableau 5.7	Pourcentage d'avancement estimé par MP pour chaque solution à chaque étape d'une séquence typique de résolution	148
Tableau 6.1	Séquence des messages associés à l'inférence de la figure 6.3	172
Tableau 6.2	Métrique rattachée à chacun des verdicts possibles	178
Tableau 6.3	Données reliées aux messages créés pour les cinq problèmes implantés	188
Tableau 6.4	Statistiques relatives aux tests réalisés par l'expert	193

LISTE DES FIGURES

Figure 1.1	Problème du rectangle	4
Figure 3.1	Carte des interactions didactiques	58
Figure 3.2	Cycles de conception de QEDX	59
Figure 3.3	Fenêtre principale	62
Figure 3.4	Onglet <i>Figure</i>	64
Figure 3.5	Onglet <i>Phrases</i>	65
Figure 3.6	Onglet <i>Rédaction</i>	66
Figure 3.7	Architecture logicielle de QEDX	68
Figure 3.8	Interface du logiciel ScreenFlow	76
Figure 4.1	Problème du rectangle	80
Figure 4.2	Représentations usuelles de la démonstration	83
Figure 4.3	Transformation d’une inférence en arbre HPDIC	88
Figure 4.4	Graphe HPDIC pour le problème du rectangle	89
Figure 5.1	Problème du rectangle	116
Figure 5.2	Solutions pour le problème du rectangle	118
Figure 5.3	Arbres extraits du graphe pour le problème du rectangle	121
Figure 5.4	Trois solutions possibles au problème du rectangle	123
Figure 5.5	Exemple du calcul de l’avancement des solutions sur un arbre HPDIC	127
Figure 5.6	Temps d’activation associés à une séquence d’actions	133
Figure 5.7	Chronologie d’actions typique produisant une démonstration	144
Figure 6.1	Organisation des messages sur deux niveaux	164
Figure 6.2	Problème du rectangle	166
Figure 6.3	Encodage d’une inférence acceptée pour le problème du rectangle	167
Figure 6.4	Interface de GGBT lors de la première expérimentation (2010)	169
Figure 6.5	Interface actuelle de QEDX	176
Figure 6.6	Schéma général de la machine à états finis générant les messages	181
Figure 6.7	Structure interne de la machine illustrée à la figure 6.6	183
Figure 6.8	Exemples de messages générés pour le problème du rectangle	192
Figure 6.9	Organisation des messages selon deux niveaux intégrant un cycle	201

LISTE DES ALGORITHMES

Algorithme 4.1 Énumération de toutes les solutions du graphe HPDIC	93
Algorithme 4.2 Validation d'une solution afin de s'assurer qu'elle est complète	94
Algorithme 5.1 Création d'une forêt d'arbres à partir d'un graphe HPDIC	120
Algorithme 5.2 Ordonnancement des inférences pour un graphe HPDIC	136

LISTE DES SIGLES ET ABRÉVIATIONS

AGT	Advanced Geometry Tutor
CBM	Constraint Based Modeling / modèle basé sur les contraintes
CPT	Carnegie-Mellon Proof Tutor
CTAT	Cognitive Tutor Authoring Tools
DC	Diagram Configuration
DEFI	Démonstration et Exploration de la Figure Interactives
EDOI	Évaluation des Démonstrations et Ordonnancement des Inférences
EIAH	Environnements Interactifs d'Apprentissage Humain
ETG	Espace de Travail Géométrique
ETM	Espace de Travail Mathématique
FLOT	Formations en Ligne Ouvertes à Tous
FSM	Finite State Machine / machine à états finis
GGBT	GeoGebraTUTOR
GMD	Génération des Messages Discursifs
HPDIC	Hypothèse, Propriété, Définition, résultat Intermédiaire, Conclusion
IRT	Item Response Theory / théorie de réponse aux items
KC	Knowledge Component / composant de connaissance
MELS	Ministère de l'Éducation, du Loisir et du Sport
MIA	Modèle Itératif de l'Apprenant
MTT	Model Tracing Tutor / tuteur par traçage de modèle
QEDX	QED-Tutrix
TSD	Théorie des Situations Didactiques

LISTE DES ANNEXES

Annexe A	Problèmes implantés dans QEDX	243
----------	---	-----

CHAPITRE 1 INTRODUCTION

«Ils ne savaient pas que c'était impossible alors ils l'ont fait.»

— Mark Twain

Au cours des siècles, les développements mathématiques nous ont permis d'étendre notre compréhension du monde qui nous entoure en plus d'avoir un impact important sur notre qualité de vie. En effet, toutes les technologies modernes, allant de l'ordinateur aux téléphones intelligents en passant par les GPS, sont issues de la mise en oeuvre de différents modèles mathématiques. Ils permettent aussi, entre autres, de fixer le prix de l'essence, de prévoir les horaires d'autobus et de déterminer votre prime d'assurance. Malgré cette omniprésence des mathématiques dans notre vie, il n'en demeure pas moins que plusieurs élèves du secondaire redoutent cette matière. Or, c'est principalement aux enseignants que revient la responsabilité de transmettre aux élèves une perception positive des mathématiques, notamment de la démonstration, mais il semble que plusieurs échouent malgré toute leur bonne volonté.

1.1 Enjeux concernant l'enseignement de la démonstration

À la défense des enseignants, il est reconnu que certains sujets des mathématiques, comme les démonstrations, sont plus difficiles à enseigner. En effet, Duval (1995) déclare que «l'accès à l'activité de démonstration constitue un seuil décisif que, jusqu'à présent, l'enseignement ne semble pas avoir réussi à faire franchir à la plupart des élèves». Ce seuil dont il est question concerne le passage du raisonnement argumentatif, dont la valeur de la conclusion dépend de la pertinence des énoncés, au raisonnement déductif, qui repose sur une structure formelle. L'accompagnement adéquat de l'élève qui navigue ce passage houleux est donc crucial (Tall, 1991). Ce malaise concernant l'enseignement de la démonstration se manifeste au Québec par une valse-hésitation entre la valorisation et la mise de côté des preuves au cours des quarante dernières années (Caron et René de Cotret, 2007). De plus, Stéphane Cyr, dans une entrevue accordée à Gervais (2011), soutient qu'au Québec on adopte une approche plus pragmatique de l'enseignement des mathématiques par rapport à l'approche plus élitiste préconisée en Europe. L'enseignement des preuves est, par conséquent, un sujet délicat au Québec.

À cette complexité intrinsèque de l'enseignement de la démonstration s'ajoute l'évaluation difficile des acquis. D'une part, le Programme de formation de l'école québécoise (Ministère de l'Éducation, du Loisir et du Sport, 2009) actuellement en vigueur inclut la compétence consistant à déployer un raisonnement mathématique qui fait référence au raisonnement dé-

ductif. D'autre part, le Cadre d'évaluation des apprentissages (Ministère de l'Éducation, du Loisir et du Sport, 2011) précise des critères d'évaluation concernant la structure et la justification des preuves produites par l'élève. Si l'on y suggère, entre autres, d'observer si l'élève laisse des traces claires et complètes de son raisonnement, les critères d'évaluation laissent sous-entendre que c'est la démonstration achevée qui fera office de traces de son raisonnement. Pourtant, le verbe *déployer* fait référence à l'ensemble du raisonnement déductif et non seulement à la preuve résultante. Il est alors légitime de se demander comment l'élève peut rendre explicite son processus de résolution afin que l'enseignant soit en mesure d'évaluer sa maîtrise de la compétence. La solution idéale implique une observation individuelle de chaque élève lors de la résolution d'un problème de démonstration, ce qui est impraticable dans un contexte d'enseignement en classe.

Il nous apparaît donc que l'enseignement et l'évaluation adéquate du raisonnement déductif requièrent, de la part de l'enseignant, une durée d'intervention minimale auprès de chaque élève pris isolément. Dans ce contexte, une réduction du nombre d'élèves par classe pourrait favoriser un meilleur apprentissage. Pourtant, on observe, depuis plusieurs années, que presque toutes les commissions scolaires québécoises comptent des classes surchargées, ce qui permet d'économiser par rapport à l'ouverture de nouvelles classes (Allard, 2008). Afin de réduire les dépenses reliées aux débordements, le ministre a proposé à la fois d'augmenter le nombre maximal d'élèves par classe et de comptabiliser comme étant normaux les élèves ayant de sérieux troubles de comportements (Vailles, 2015). Cette dernière politique pourrait faire augmenter le nombre d'élèves problématiques par classe, ce qui forcera l'enseignant à allouer plus de temps à la gestion de sa classe. Le temps disponible au soutien de l'apprentissage individuel est alors susceptible de diminuer substantiellement, ce qui pourrait mettre en péril la réussite de certains élèves. L'enseignement du raisonnement déductif risque donc de se solder par un échec si l'on ne fournit pas des ressources appropriées à l'enseignant.

Dans le but d'assister l'enseignant dans sa tâche, nous avons naturellement pensé qu'un outil informatique serait tout indiqué. Nous nous sommes premièrement intéressé aux micromondes de géométrie, comme GeoGebra (Hohenwarter, 2013; International GeoGebra Institute, 2015) et Cabri-Géomètre (Baulac, 1990; CABRILOG SAS, 2009), qui permettent, entre autres, de manipuler des figures dynamiques afin d'en étudier les propriétés. Cependant, ils n'offrent aucun soutien à l'élève concernant le travail à réaliser, ce qui ne permet pas d'alléger la tâche de l'enseignant. De plus, ce dernier doit être en mesure de répondre à des questions techniques concernant l'utilisation du logiciel. Il existe aussi des environnements d'assistance à la démonstration, par exemple, Cabri-Euclide (Luengo, 2005) ou Baghera (Balacheff et al., 2003). Ceux-ci permettent de construire librement une preuve et peuvent fournir des rétroactions concernant chacun des pas proposés ou des annotations sur la preuve finale. Toutefois,

il s'agit, comme pour les micromondes, d'outils d'exploration qui offrent une aide limitée à l'élève et qui ne sont pas en mesure de le contraindre à résoudre le problème posé. Malgré les nouvelles occasions d'apprentissage qu'offrent ces systèmes, ils ne sont pas aptes à guider l'élève dans son processus de résolution.

Il nous a alors semblé qu'un système tutoriel intelligent, au sens de VanLehn (2006), serait plus approprié pour appuyer l'enseignant. Ainsi, dans le cadre de cette thèse, nous définissons le *système tutoriel intelligent* comme étant un système qui soutient l'élève en mettant en oeuvre diverses rétroactions lors de l'exécution, par ce dernier, des différentes étapes requises pour résoudre un problème particulier. Ces systèmes ne s'intéressent donc pas seulement à la réponse produite par l'élève, mais ils lui fournissent de l'aide adaptée pour y arriver. Par contre, ceux qui ne gèrent que la séquence de problèmes à proposer en ne validant que les réponses offrent plutôt de l'enseignement assisté par ordinateur selon VanLehn (2006). De même, les systèmes qui permettent à l'élève de corriger leur réponse en leur donnant une rétroaction uniquement sur celles qui sont erronées ne sont pas considérés comme des systèmes tutoriels intelligents d'après notre définition, car ils ne soutiennent pas systématiquement le processus de résolution. Cette définition est certes restrictive, mais nous permet de cibler précisément la catégorie de systèmes qui nous intéresse.

Nous avons donc recherché des systèmes tutoriels intelligents qui proposent des problèmes de démonstration en géométrie euclidienne. La majorité des systèmes recensés (voir chapitre 2) se servent des moteurs de déduction formels afin de répertorier les solutions valides pour un problème donné. Par conséquent, leur niveau d'exigence en ce qui concerne la production d'une preuve n'est généralement pas compatible avec le paradigme utilisé au Québec, défini par Richard et Fortuny (2007) comme étant le paradigme de la géométrie cognitive. Dans un tel paradigme, certains enseignants autorisent parfois l'usage d'arguments informels ou permettent d'omettre des étapes accessoires de la preuve, ce qui n'est pas acceptable en géométrie formelle. Cette disparité entre les paradigmes limite donc l'utilité de ces systèmes. Il est aussi intéressant de noter que le Geometry Tutor (Anderson et al., 1985), qui est un pionnier et une référence dans le domaine des systèmes tutoriels en géométrie, ne traite désormais que des preuves numériques. En effet, cette transition a été réalisée afin de respecter les nouveaux objectifs du programme de formation Américain (Aleven et al., 1998). Comme nous le montrerons en détail dans le chapitre 2, il s'avère que les principaux systèmes recensés ne répondent pas précisément aux besoins des enseignants québécois.

L'adaptation d'un logiciel existant pour qu'il soit en mesure de traiter la géométrie telle qu'enseignée au Québec a enfin été envisagée. Nous avons alors analysé la façon dont chacun des systèmes permet de déployer un raisonnement mathématique afin de choisir celui qui est

le plus susceptible d'assister adéquatement l'enseignant. Nous avons constaté que la majorité des systèmes forcent l'élève à construire sa preuve dans un ordre déterminé. Par conséquent, ils ne permettent pas une exploration libre du problème, qui est une étape importante dans le processus de résolution. Lorsque l'exploration est possible, comme dans Mentoniez (Py, 1996), aucun indice concernant la prochaine étape à exécuter n'est offert. De plus, aucun des systèmes recensés ne propose différentes pistes de solution comme le fait un enseignant. Ils considèrent que l'élève se doit de produire la solution optimale telle que déterminée par l'expert. En fait, généralement, l'objectif de ces systèmes est d'enseigner à l'élève à rédiger une démonstration à la manière d'un expert et non de promouvoir l'apprentissage par la découverte. Nous arrivons alors à la conclusion que les systèmes tutoriels offerts actuellement ne sont pas aptes à épauler convenablement l'enseignant québécois en classe de géométrie.

1.2 Concepts relatifs aux démonstrations

En réponse à ces constats, nous avons décidé de concevoir un système tutoriel intelligent adapté au contexte d'enseignement québécois, mais il nous a d'abord fallu préciser le genre de problèmes que notre système devrait traiter. Nous avons, d'une part, consulté les objectifs du volet géométrie du programme québécois de mathématiques au secondaire et avons constaté que l'un d'eux consiste en la production, par les élèves, de démonstrations en géométrie euclidienne (Ministère de l'Éducation, du Loisir et du Sport, 2009). Afin d'obtenir des exemples plus explicites de problèmes, nous nous sommes, d'autre part, référé aux manuels scolaires et aux exercices proposés par les enseignants. Nous avons alors extrait un échantillon de cinq problèmes (annexe A) qui offrent une diversité didactique, mais aussi des défis informatiques. Nous présentons donc, dans cette section, un problème typique qui pourrait être posé à des élèves de quatrième secondaire ainsi que des définitions utiles à la compréhension de cette thèse. Nous avons nommé le problème choisi et présenté à la figure 1.1, le *problème du rectangle* (annexe A.1). Celui-ci ne fait pas intervenir de valeurs numériques, ce qui demande, de la part de l'élève, d'exercer sa compétence en démonstration. Ce problème sera, de plus, employé tout au long de cette thèse afin d'illustrer différents concepts.

Démontrer qu'un quadrilatère ayant trois angles droits est un rectangle.

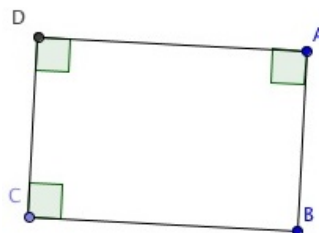


Figure 1.1 Problème du rectangle

Une démonstration acceptable, parmi toutes celles qui sont possibles, pour le *problème du rectangle* peut s'écrire comme suit :

- Comme **AB est perpendiculaire à AD**, et **AD est perpendiculaire à CD**, alors, selon la propriété qui dit «**Deux droites perpendiculaires à une même troisième sont parallèles entre elles**», **AB est parallèle à CD**.
- Comme **BC est perpendiculaire à CD**, et **AB est parallèle à CD**, alors, selon la propriété qui dit «**Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre**», **AB est perpendiculaire à BC**.
- Comme **AB est perpendiculaire à AD**, **AD est perpendiculaire à CD**, **BC est perpendiculaire à CD**, et **AB est perpendiculaire à BC**, alors, selon la propriété qui dit «**Un rectangle est un quadrilatère ayant quatre angles droits**», **ABCD est un rectangle**.

Chacune des phrases de la démonstration peut aussi être qualifiée d'*inférence*, qui est un concept central dans le cadre de notre travail. Toutes les inférences ont la même structure, soit un ensemble d'antécédents sur lesquels une justification est appliquée pour produire un conséquent. Le tableau 1.1 montre les éléments de l'inférence qui correspondent à la deuxième phrase de la démonstration proposée.

Tableau 1.1 Éléments d'une inférence

Antécédents	BC est perpendiculaire à CD
	AB est parallèle à CD
Justification	Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre
Conséquent	AB est perpendiculaire à BC

Une démonstration est donc constituée d'un certain nombre d'inférences. Le conséquent d'une inférence peut être utilisé en tant qu'antécédent pour une autre, ce qui forme un résultat intermédiaire. Une hypothèse est un antécédent qui n'est le conséquent d'aucune inférence et qui ne demande pas à être démontré dans le contexte du problème posé. Réciproquement, la conclusion est un conséquent qui n'est pas utilisé comme antécédent dans d'autres inférences associées au problème et qui répond à la question posée. Une démonstration est ainsi un ensemble d'inférences qui permettent de relier les hypothèses à la conclusion et qui ne contient pas de conséquents inutilisés, sauf la conclusion, ou d'antécédents non prouvés, à l'exception des hypothèses.

Lors de la rédaction d'une démonstration, nous utilisons habituellement le chaînage avant ou arrière. Le fait de n'utiliser que des hypothèses ou des résultats intermédiaires prouvés comme antécédents de chaque inférence ajoutée à la démonstration, pour éventuellement arriver à la conclusion, se nomme le *chaînage avant*. Le *chaînage arrière* consiste à faire le chemin inverse, c'est-à-dire que l'on part d'une inférence ayant comme conséquent la conclusion et l'on ajoute, à rebours, des inférences pour prouver les antécédents qui ne sont pas des hypothèses du problème, jusqu'à ce que tous les antécédents non prouvés soient des hypothèses. Enfin, si nous ne sommes pas contraints de résoudre un problème en chaînage avant ou arrière, nous disons alors qu'une exploration libre est possible.

1.3 Objectifs de recherche

Notamment, l'étude des cinq problèmes retenus conjuguée à nos constatations concernant les difficultés associées à l'enseignement de la preuve ainsi qu'au soutien limité offert par les systèmes tutoriels intelligents accessibles nous a donné l'occasion de préciser les objectifs de notre projet dont le principal (O_p) est le suivant :

- O_p : Proposer un système tutoriel intelligent qui soutient l'élève dans une démarche d'exploration, plutôt que de rédaction, dans le cadre de l'élaboration d'une preuve en géométrie.

Cet objectif contient deux concepts principaux relatifs à notre projet. Il indique, d'une part, que nous visons la mise en oeuvre d'un système tutoriel intelligent, soit un système informatique qui est en mesure de réagir aux actions de l'élève dans le but de l'aider à bâtir sa démonstration. Il ne suffit donc pas de produire un micromonde qui facilite la rédaction d'une preuve, mais bien de proposer des indices pertinents capables de relancer le processus de résolution d'un élève bloqué. D'autre part, l'objectif spécifie que notre système doit fournir une assistance tout au long du processus de résolution, ce qui inclut la démarche d'exploration. En effet, nous voulons autoriser et même encourager l'élève à explorer diverses pistes de solution avant de rédiger sa preuve finale, ce qui n'est pas permis dans d'autres systèmes tutoriels. L'atteinte de cet objectif nous permettra ainsi de proposer un système innovateur.

Afin de détailler la structure du système mentionné dans l'objectif principal, nous nous sommes inspiré de la division d'un environnement d'enseignement assisté par ordinateur en cinq composants, suggérée par Beck et al. (1996). Nous avons alors interprété les définitions proposées dans le contexte du système tutoriel intelligent planifié pour produire les objectifs secondaires suivants :

- O_{s1} : Construire un système tutoriel qui met en oeuvre des théories didactiques reconnues, en faisant appel à une équipe multidisciplinaire.

- O_{s2} : Identifier une manière adéquate d'encoder l'ensemble des démonstrations acceptables, qui ne sont pas nécessairement formelles, pour un problème donné.
- O_{s3} : Modéliser l'aspect dynamique de la démarche de résolution de l'élève.
- O_{s4} : Interpréter la chronologie des actions de l'élève afin de reconnaître sa stratégie de résolution.
- O_{s5} : Apporter le soutien nécessaire à l'élève par l'entremise de messages discursifs inspirés des interventions d'enseignants réels.

Il est, en effet, relativement aisé d'établir une relation entre nos objectifs et trois des composants proposés par Beck et al., soit les connaissances du domaine (O_{s2}), le modèle de l'élève (O_{s3}) et le modèle de communication (O_{s5}). Pour le quatrième composant, soit le modèle de l'expert, nous avons interprété la définition dans un sens plus large pour générer O_{s4} . Ainsi, il s'agit d'un modèle qui possède non seulement une expertise dans la production de démonstrations, mais qui est aussi en mesure de déterminer la stratégie de résolution de l'élève à la manière d'un expert. Le modèle pédagogique, qui est le dernier composant, dépasse le cadre du projet actuel, car il correspond à la gestion des itinéraires de problèmes, donc aucun objectif ne lui est rattaché. Enfin, O_{s1} est un objectif qui concerne plutôt la méthodologie ainsi que les contraintes relatives à la mise en oeuvre du système et n'est, par conséquent, relié à aucun composant décrit par Beck et al. (1996).

Même s'il est possible de faire un rapprochement entre les composants d'un système tutoriel et nos objectifs secondaires, ces derniers permettent néanmoins de préciser la nature et l'architecture de notre système. Tout d'abord, O_{s1} spécifie qu'il s'agit d'un projet multidisciplinaire dans lequel nous devons travailler en étroite collaboration avec des experts en didactique. Par conséquent, les membres de l'équipe doivent avoir des interactions constantes et éviter de travailler en vase clos. Ensuite, la géométrie cognitive est prise en compte dans O_{s2} lorsque l'on indique que les preuves ne sont pas nécessairement formelles. Troisièmement, l'aspect dynamique de la démarche (O_{s3}) et la chronologie des actions (O_{s4}) font référence au processus exploratoire de la résolution. En effet, comme l'ordre de travail n'est pas imposé, nous devons analyser la séquence d'actions de l'élève pour comprendre son cheminement. La dernière information digne de mention est relative aux rétroactions du système (O_{s5}). Ainsi, on précise qu'elles sont basées sur les interventions d'enseignants réels, donc elles devraient être plus riches que celles des systèmes traditionnels en proposant, par exemple, diverses pistes de solution. Bref, ces objectifs permettent de cerner les principaux aspects originaux de notre système.

1.4 Organisation de la thèse

Dans cette thèse, nous décrivons toute la démarche de recherche qui nous a amené à définir l'architecture logicielle actuelle de QED-Tutrix (QEDX), soit un système tutoriel intelligent que nous avons mis en oeuvre afin de soutenir l'élève lors de l'élaboration de démonstrations. Nous enrichissons cette description technique avec des résultats d'analyses didactiques qui ont gouverné sa réalisation. Toutefois, avant de nous lancer dans le vif du sujet, nous présentons, dans le chapitre suivant, une revue des principaux systèmes tutoriels offerts pour la géométrie, afin de démontrer qu'aucun ne permet d'atteindre entièrement les objectifs que nous nous sommes fixés. Nous avons ainsi analysé en détail dix systèmes tutoriels en géométrie ainsi que cinq autres systèmes populaires se rapprochant de nos travaux. Nous nous sommes intéressé, entre autres, à la modélisation du domaine et de l'apprenant, aux rétroactions offertes, à la séquence des messages affichée, aux théories sur lesquelles les systèmes sont basés, etc. Un tableau synthèse (tableau 2.1) qui résume les principales caractéristiques techniques des différents systèmes est proposé à la fin du chapitre 2.

Les quatre chapitres suivants traitent du coeur de notre travail, soit de l'architecture de QEDX telle qu'elle a été mise en oeuvre au terme du second cycle de son élaboration (figure 3.2). Étant donné que les différentes couches de notre système sont relativement indépendantes, nous avons pris l'initiative de les présenter dans des chapitres autonomes. Cette structure a l'avantage de permettre au lecteur de ne consulter que les chapitres qui abordent les modules qui l'intéressent, sans avoir à parcourir la totalité de la thèse. Néanmoins, le premier de ces chapitres, soit le chapitre 3, se veut une synthèse de nos travaux dans lequel nous présentons les relations qui existent entre les aspects didactiques et informatiques de notre projet. Nous y exposons, d'une part, les principales théories didactiques ainsi que les deux expérimentations réalisées qui ont permis de concevoir et de valider QEDX. D'autre part, nous donnons une description sommaire de l'interface et de l'architecture interne de notre système tout en faisant ressortir les aspects didactiques reliés à ceux-ci. Ce chapitre porte donc notamment sur l'objectif secondaire O_{s1} , soit l'aspect multidisciplinaire du projet.

Les trois autres chapitres (4, 5 et 6) exposent les aspects plus techniques des différentes couches (figure 3.7). Le chapitre 4 présente une définition formelle des graphes HPDIC (Hypothèse, Propriété, Définition, résultat Intermédiaire et Conclusion). Ceux-ci permettent de représenter l'ensemble des solutions à un problème de preuve telles qu'elles pourraient être acceptées par un enseignant. Nous nous intéressons, plus particulièrement, à l'encodage des inférences et à l'extraction de l'ensemble des solutions, ce qui correspond essentiellement à la modélisation du domaine, soit à l'objectif O_{s2} . Nous abordons aussi la sauvegarde de la chronologie des actions discursives d'un élève à l'interface de QEDX afin de lui fournir une

aide qui respecte son état cognitif. Ce dernier module peut être relié à l'objectif O_{s3} , car il traite d'un modèle dynamique de l'apprenant. Le chapitre suivant, soit le chapitre 5, apporte des réponses quant à l'objectif O_{s4} . En effet, il présente la couche EDOI (Évaluation des Démonstrations et Ordonnancement des Inférences) qui permet, en fait, d'interpréter les actions de l'élève et de tenter de déduire sa stratégie de résolution. De plus, on y propose différentes techniques qui ont été mises en oeuvre afin de contenir l'explosion combinatoire concernant l'extraction de la solution la plus avancée. Finalement, l'objectif O_{s5} est abordé dans le chapitre 6 traitant du GMD (Génération des Messages Discursifs). Dans celui-ci, on expose premièrement les principaux résultats des analyses didactiques qui nous ont permis de construire notre modèle d'intervention. Nous proposons ensuite des rétroactions immédiates ainsi qu'une structure de messages gérée par une machine à états finis afin de soutenir l'élève lors de la résolution d'un problème. Il est à noter que ces trois derniers chapitres incluent chacun des résultats et une discussion distincts qui se limitent aux sujets abordés dans le chapitre concerné.

Nous proposons, par la suite, des résultats ainsi qu'une discussion générale qui traitent de sujets qui ne sont rattachés à aucune couche particulière de QEDX et qui n'ont donc pas été spécifiquement abordés dans les chapitres précédents. Nous présentons d'abord des résultats concernant l'évolution de l'interface à la suite des différents tests et expérimentations en plus de la mise en oeuvre, par les élèves, de diverses démarches à l'interface de QEDX. Ensuite, dans une discussion générale, nous traitons, entre autres, des occasions d'apprentissage insoupçonnées offertes par notre système, des avantages ainsi que des problèmes liés à notre choix d'architecture logicielle et des implications de nos principales décisions méthodologiques. Cette thèse se termine par une conclusion dans laquelle nous faisons tout d'abord une synthèse de nos travaux relativement aux objectifs que nous nous sommes fixés. Nous exposons ensuite les limites de notre système pour enfin proposer des pistes de recherche qui permettront de poursuivre le développement de QEDX.

CHAPITRE 2 REVUE DE LITTÉRATURE

*«Ceux qui ne connaissent pas l'histoire
sont condamnés à la répéter.»*

— Edmund Burke

2.1 Introduction

Notre projet consiste à produire un système tutoriel pour aider les élèves du secondaire à construire des preuves en géométrie euclidienne. À la suite d'une analyse didactique à priori des enjeux relatifs à l'enseignement et à l'apprentissage de la démonstration, menée par les experts en didactique de l'équipe, une liste de caractéristiques souhaitables pour un système soutenant l'exercice de la démonstration a été établie. Parmi celles-ci, deux caractéristiques se sont avérées d'une importance particulière, soit de permettre une exploration libre des différentes solutions et d'offrir une aide qui propose différentes pistes de solution dans le cas d'un blocage du processus de résolution. Ces deux caractéristiques ont pour but d'imiter l'environnement de résolution en classe, soit l'écriture des énoncés sans contraintes sur papier et les interventions des enseignants qui s'adaptent au gré de la discussion avec l'élève. Une revue de la littérature informatique a alors été réalisée afin de trouver des systèmes qui pourraient répondre à ces exigences. Nous présentons un résumé de nos découvertes dans ce chapitre. Nous avons choisi de présenter, de façon plus élaborée, les systèmes qui nous semblaient les plus pertinents, soit parce qu'ils concernent directement les démonstrations en géométrie, soit parce qu'ils utilisent une stratégie qui peut être pertinente pour notre projet. À la fin du chapitre, nous proposons un tableau synthèse (tableau 2.1) des principales caractéristiques des quinze systèmes choisis. Il faut enfin savoir que l'analyse des différentes approches de modélisation est présentée tout au long du chapitre dans les sections des systèmes concernés.

Le chapitre débute par une description des différentes ressources électroniques disponibles pour l'apprentissage des mathématiques. Nous présentons les ressources allant des sites pour discuter avec des tuteurs humains jusqu'aux approches par curriculum qui permettent de guider l'élève au cours de ses apprentissages. Relativement à ces derniers, nous proposons une brève analyse des stratégies traditionnelles pour modéliser l'apprenant et nous montrons qu'elles ne peuvent être utilisées dans le cadre de notre projet actuel. Nous présentons ensuite les micromondes et les systèmes de preuve automatique en géométrie. Ceux-ci sont utiles pour l'apprentissage de la géométrie, mais n'offrent pas d'aide interactive. Par la suite, nous ana-

lysons dix systèmes tutoriels intelligents pour l'apprentissage de la preuve en géométrie. Pour chaque système, nous retraçons brièvement son histoire et nous décrivons son fonctionnement. Une analyse plus technique est ensuite proposée et nous concluons par une brève critique. Le chapitre se poursuit par une description de cinq systèmes qui proposent des approches de modélisation différentes. Il s'agit de systèmes qui nous semblent les plus pertinents dans le cadre de notre projet, mais qui ne s'appliquent pas nécessairement à l'apprentissage de la géométrie. Ces derniers sont étudiés en adoptant les mêmes critères que ceux utilisés pour l'analyse des systèmes tutoriels en géométrie. Un tableau synthèse (tableau 2.1) qui contient les principales caractéristiques, qui sont relatives à notre projet, des quinze systèmes analysés est enfin présenté. En guise de conclusion pour ce chapitre, nous proposons une synthèse de notre analyse qui vise à justifier la pertinence du système conçu dans le cadre de ce projet.

2.2 Ressources en ligne pour l'apprentissage des mathématiques

Dans cette section, nous présentons une description générale des ressources offertes en ligne pour l'apprentissage des mathématiques sans nous restreindre à la géométrie. Dans la partie concernant les ressources génériques, nous donnons un aperçu des outils disponibles pour un apprenant autonome. Si ce dernier connaît ses besoins, ces outils lui permettent habituellement d'obtenir une aide ponctuelle adéquate. Par la suite, nous discutons des approches par curriculum. Celles-ci proposent à l'apprenant de suivre un parcours plus adapté à son profil et à ses connaissances. Nous y intégrons aussi une analyse des modèles de l'apprenant permettant de représenter les connaissances face à un domaine. Pour les étudiants qui sont autonomes et qui veulent expérimenter en géométrie, des micromondes sont offerts et sont le sujet de la troisième partie de cette section. Enfin, nous présentons les systèmes d'aide à la démonstration ainsi que les systèmes de preuve automatiques. Ces derniers, ainsi que les micromondes, sont utilisés dans certains systèmes tutoriels intelligents, mais n'offrent pas de systèmes d'aide intrinsèque.

2.2.1 Apprentissage autonome

La ressource idéale pour aider un élève en mathématiques, ou dans tout autre domaine, est souvent l'enseignant. Cependant, le temps qu'il peut accorder à chaque élève est limité et il n'est habituellement pas présent lorsque l'élève travaille sur des problèmes à la maison. Il arrive, par contre, que des élèves aient besoin d'aide supplémentaire et c'est pour cette raison que des sites comme tutor.com (Tutor.com, Inc., 2015) ou Succès Scolaire (Succès Scolaire, 2014) existent. Ceux-ci offrent, moyennant une certaine somme d'argent, un accès en ligne à des tuteurs humains qui sont en mesure d'offrir de l'aide aux devoirs ou d'aider l'élève à

se préparer aux examens. Pour les personnes qui n'ont pas les moyens de se payer un tuteur privé, il est possible de poser ses questions sur des sites comme Yahoo! Answers (Yahoo! Inc., 2015) afin d'obtenir potentiellement une réponse rapidement. Il faut cependant être vigilant, car n'importe qui peut répondre à une question qui y est posée. Ainsi, la réponse obtenue peut parfois être erronée, incomplète ou trop complexe pour le niveau de compétence du demandeur. Tous ces sites permettent donc de mettre l'apprenant en contact avec d'autres personnes pour résoudre un problème, mais n'offrent pas de systèmes tutoriels intelligents.

En ce qui concerne l'aide automatique à la résolution de problèmes, la compagnie Mathway (Mathway, 2015) propose un site permettant de visualiser les étapes pour résoudre des problèmes de mathématiques typiques contenus dans les devoirs. L'élève choisit le type de question à laquelle il doit répondre et inscrit l'énoncé. Le logiciel résout ensuite le problème et affiche les étapes de la solution ainsi que des explications pour chacune. Celui-ci utilise un moteur de résolution générique, mais le genre de problèmes qu'il peut résoudre, quoiqu'assez exhaustif, est limité. De plus, il ne propose qu'une seule solution, ce qui limite son utilité pour les élèves qui auraient produit une solution différente. En effet, ils ne sont pas en mesure de comparer leurs solutions à celles du logiciel pour trouver leurs erreurs. Nous ne pouvons donc pas considérer ce site comme un système tutoriel intelligent, même si son interface, qui imite un environnement papier/crayon, est intéressante, car il n'offre pas de soutien interactif lors de la résolution de problèmes.

Si l'objectif n'est pas de résoudre un problème particulier, mais plutôt combler des lacunes théoriques ou approfondir la matière, des vidéos éducatives peuvent être utiles. Le site «The Great Courses» (The Teaching Company, LLC. , 2015), par exemple, offre des cours complets sur différents sujets. En faisant des recherches sur YouTube (YouTube, LLC, 2015), il est souvent possible de trouver une vidéo sur le sujet de son choix et qui correspond à son style d'apprenant. On peut aussi penser à Khan Academy (Khan Academy, 2015) qui doit son succès à la diffusion, sur YouTube, de vidéos éducatives produites par son fondateur. Les vidéos ajoutent un peu de vie aux notions présentées, mais restent souvent très proches des lectures que l'on peut faire sur le sujet. Il faut aussi être vigilant, car les vidéos ne sont pas toutes produites par des experts du sujet et, même si elles sont attrayantes, peuvent comporter des erreurs. Les vidéos sont donc un autre moyen de remplacer l'enseignant en cas de besoin. Nous les considérons cependant comme de l'enseignement assisté par ordinateur et non comme des systèmes tutoriels intelligents.

Pour ceux qui préfèrent un environnement d'apprentissage plus structuré, il est possible de suivre des cours en ligne. Par exemple, la plateforme Moodle (Moodle Pty Ltd, 2015) est principalement destinée aux enseignants qui désirent mettre en ligne des documents et des

vidéos pour les étudiants inscrits à leur cours. Elle offre différents outils pour la gestion des apprentissages et peut permettre, par l’entremise d’un forum de discussion, un mode d’apprentissage collaboratif. Elle est donc habituellement utilisée en tant que prolongement des cours magistraux, mais permet aussi d’offrir des formations en ligne ouvertes à tous (FLOT). Par contre, des plateformes comme edX (edX Inc., 2015) et Coursera (Coursera Inc., 2015) sont spécifiquement destinées à gérer des FLOT avec un nombre d’usagers plus important. Elles sont moins flexibles que Moodle, mais on y retrouve les principales fonctionnalités et une interface plus conviviale. Il est possible de suivre des cours crédités ou d’être un simple observateur afin d’avoir accès aux contenus. Tous les cours offerts sont construits et gérés par des groupes d’enseignants. Par conséquent, ces plateformes, même si elles contiennent des outils d’analyse, n’offrent généralement pas de système d’aide intelligent pour guider l’étudiant dans son cheminement ou pour la résolution de problèmes.

2.2.2 Gestion de curriculum et modélisation de l’apprenant

Toutes les ressources recensées précédemment ne contenaient pas d’agents intelligents pour aider ou guider l’étudiant qui devait, par conséquent, gérer lui-même son processus d’apprentissage. Il existe, par contre, plusieurs sites qui permettent de proposer un cheminement adapté aux compétences de l’utilisateur dans un domaine, ou selon un curriculum prédéfini. ELM-ART (Brusilovsky et al., 1996; Weber et Brusilovsky, 2001; Weber, 2014), qui permet d’apprendre le LISP, fut l’un des premiers sites offrant un système de gestion intelligent du contenu qui prend en considération les connaissances de l’utilisateur. Il est présenté comme un manuel interactif dans lequel il est possible de résoudre des problèmes et de mettre en application les notions par le biais d’outils interactifs. L’analyse des interactions permet à ELM-ART de proposer un cheminement d’apprentissage adapté. Dans le domaine des mathématiques, ALEKS (Claude Falmagne et al., 2006; Sabo et al., 2013; McGraw-Hill Global Education Holdings, LLC., 2015) offre un cheminement basé sur une séquence de problèmes à résoudre. Le système présente un problème et analyse la réponse obtenue et l’état estimé des connaissances de l’élève pour choisir le problème optimal, soit celui qui maximise la probabilité d’un apprentissage, qu’il propose ensuite. Les problèmes sont donc de difficulté croissante, mais respectent l’état cognitif de l’élève en lui offrant un défi accessible. Pour sa part, le système ActiveMath (Melis et al., 2009; DFKI & Saarland University, 2007) utilise les informations contenues dans le web sémantique pour construire des cours de mathématiques adaptés aux compétences et aux besoins de l’utilisateur. Les ressources proposées évoluent donc et dépendent de ce qui est offert, au moment de la requête, sur le web sémantique. La modélisation du domaine est aussi mise à jour afin de la raffiner. Enfin, Wayang Outpost (Arroyo et al., 2004), mise sur la motivation en proposant des activités d’apprentissage dyna-

miques et attrayantes. Il offre de l'aide à la résolution sous forme de vidéos et d'animations et contient un algorithme qui permet de produire un itinéraire de problèmes adapté. Ces quelques exemples témoignent de la diversité des approches utilisées par les systèmes basés sur un curriculum.

Malgré cette diversité, tous ces systèmes ont un point en commun, celui de contenir une modélisation des connaissances de l'apprenant afin de pouvoir l'orienter. Pour y arriver, plusieurs modèles peuvent être employés, dont les principaux sont présentés dans Desmarais et Baker (2012) et dans Chrysafiadi et Virvou (2013). Le plus simple, mais aussi le plus limité, consiste à établir, sur une échelle binaire, les concepts du domaine que l'étudiant maîtrise. Cependant, il n'est pas possible d'identifier ceux dont il n'a qu'une maîtrise partielle et de la quantifier. Pour avoir un portrait plus complet de la capacité de l'élève à résoudre des problèmes, il est possible d'ajouter, au modèle de base, des noeuds correspondants à des concepts erronés. À partir de ces modèles, on peut proposer des problèmes qui lui permettront de maîtriser de nouveaux concepts, mais aussi de corriger les concepts erronés. Une autre avenue est de regrouper les élèves selon des caractéristiques communes et de développer une stratégie d'enseignement pour chacun des stéréotypes. ALEKS (Claude Falmagne et al., 2006; Sabo et al., 2013; McGraw-Hill Global Education Holdings, LLC., 2015) utilise une stratégie similaire en représentant les élèves selon un espace de connaissances (KS), soit un ensemble de problèmes qu'ils sont en mesure de résoudre. Deux élèves ayant réussi un même ensemble de problèmes partageront le même espace de connaissances, donc le système leur proposera un cheminement identique.

En appliquant de la logique floue, c'est-à-dire une échelle de compétence continue, aux modèles précédents, il est possible d'avoir une idée plus précise des compétences de l'élève. Le problème qui se pose alors est d'évaluer ce degré de compétence. Le projet Pépité (Jean-Daubias, 2002, 2000) avait fait un travail dans ce sens pour évaluer les compétences des élèves en algèbre. Le système utilisait les réponses des élèves à un test et appliquait une grille d'analyse pour en tirer des conclusions. Dans le même ordre d'idée, la théorie de réponse aux items (IRT) (Sijtsma et Junker, 2006) est une méthode formelle qui permet d'inférer les connaissances à partir d'un test. Il est aussi possible de proposer des tests adaptatifs afin de minimiser le nombre de questions nécessaires pour produire un modèle représentatif. Les réseaux bayésiens peuvent aussi être utilisés pour modéliser et mettre à jour le réseau de compétences de l'apprenant. Ils peuvent permettre de modéliser des compétences abstraites qui découlent de compétences observables. Il est aussi légitime de se demander quel genre de compétences ces techniques permettent de modéliser. Une approche classique consiste à modéliser les connaissances factuelles et procédurales. Cependant, le concept de composant de connaissance (KC) (Aleven et Koedinger, 2013) permet d'étendre la modélisation à des

concepts abstraits. Les KC représentent en fait toutes les variables cognitives qui peuvent influencer la réussite d'un élève. Ils ne sont pas une réponse définitive, mais une autre manière d'aborder la structure du modèle de l'apprenant.

Les modèles présentés sont les principales approches utilisées pour la gestion de curriculum. En effet, ils servent, en général, à proposer un itinéraire de problèmes et non à offrir une aide à la résolution d'un problème particulier. Dans ces systèmes, c'est la réponse aux questions qui importe et non la démarche suivie. Il est cependant possible d'utiliser les données du modèle de l'apprenant pour offrir une aide à la résolution mieux adaptée aux compétences de l'élève. Toutefois, dans le cadre de la phase actuelle de conception du système, nous nous intéressons au processus de résolution d'un problème particulier, sans chercher à dresser un portrait global des compétences de l'élève. Nous laissons donc à l'enseignant le soin du choix des problèmes à résoudre en fonction du cheminement de l'élève. Ainsi, les modèles de l'apprenant employés par les systèmes de gestion du curriculum ne sont pas pertinents pour la réalisation du projet actuel, mais pourront être utiles lors de l'intégration de la gestion des problèmes dans une phase ultérieure de développement.

2.2.3 Micromondes

Nous avons décidé de ne pas nous préoccuper de l'itinéraire des problèmes, pour l'instant, et de déléguer cette tâche à l'enseignant, comme dans la situation traditionnelle d'enseignement. Cependant, une fois le problème choisi, nous voulons offrir la possibilité de le résoudre de manière interactive et proposer une aide intelligente. Nous nous sommes donc intéressés aux micromondes de géométrie dans le but de permettre à l'élève de manipuler une figure dynamique, ce qui n'est pas possible dans l'environnement papier/crayon. Nous pouvons décrire un micromonde comme étant un univers isolé dans lequel on intègre les règles d'une partie de la réalité et qui permet d'expérimenter librement. Par exemple, un micromonde en géométrie met en place les règles de la géométrie euclidienne et permet à l'utilisateur de créer des figures et de les manipuler dynamiquement. L'environnement Logo (Papert, 1980) est un pionnier dans ce domaine. Il permet, à l'aide de primitives simples, de créer des objets complexes. Il offre aussi un environnement de programmation qui permet aux utilisateurs de s'initier à la programmation procédurale. Il est assez flexible pour accommoder un apprenant en constante évolution. De nos jours, les micromondes de géométrie sont plus spécialisés et offrent des interfaces conviviales. Il y a, par exemple, le logiciel Geometer's Sketchpad (McGraw-Hill Education, 2014) qui offre non seulement des outils pour la géométrie dynamique, mais aussi, pour les mathématiques en général. Ce logiciel est attrayant, car il permet de construire des objets géométriques semblables à ceux que l'on retrouve dans

les manuels scolaires, c'est-à-dire sans artéfacts relatifs à leur implantation informatique. D'autres systèmes, comme GeoGebra (Hohenwarter et Fuchs, 2004; Hohenwarter, 2013; International GeoGebra Institute, 2015), gèrent les objets géométriques différemment et sont ainsi notamment susceptibles d'associer plusieurs noms à un segment unique, ce qui peut causer certains désagréments dans des scénarios d'utilisation précis. Néanmoins, ce dernier évolue rapidement, car il est gratuit et son code source est accessible. Cabri-Géomètre (Baulac, 1990; Kordaki et Mastrogiannis, 2006; CABRILOG SAS, 2009) offre des fonctionnalités semblables à GeoGebra, mais celui-ci est payant. Nous avons aussi découvert iGeom (Isotani et de Oliveira Brandão, 2008), qui est un micromonde de géométrie conçu pour fonctionner sur des ordinateurs moins performants accessibles dans des pays sous-développés. Il offre tout de même la possibilité d'écrire des scripts dans lesquels la récursivité peut être utilisée. Il contient un moteur pour corriger des exercices de constructions produites à son interface par des enseignants. DrDoodle (Winterstein et al., 2004) est, pour sa part, un outil interactif pour produire des démonstrations en géométrie dans des espaces métriques. Il permet de construire des preuves figurales, ou sans mots, mais n'offre pas de moteur de vérification et c'est pour cette raison que nous le considérons comme un micromonde.

Les principaux micromondes modernes dédiés à la géométrie offrent des fonctionnalités semblables. Premièrement, des outils permettant de réaliser des constructions s'apparentant à celles produites à l'aide d'une règle et d'un compas sont disponibles. Toutefois, l'utilisation de ces outils génère, par défaut, des figures dynamiques qui possèdent un certain nombre de propriétés invariantes dans le mouvement, contrairement à celles dessinées sur papier. Par exemple, en construisant un angle droit en utilisant la procédure appropriée, celui-ci restera un angle droit, peu importe les transformations appliquées aux droites le composant. Cette caractéristique est intéressante, car elle permet aux élèves de découvrir des propriétés d'une figure produite par l'enseignant. On peut aussi faire explorer les procédures de construction, car les élèves peuvent vérifier si leur construction est robuste, ou invariante, malgré les transformations. De plus, on retrouve des outils de mesure pour les segments, les angles, etc. qui peuvent soutenir le travail des élèves dans leur processus d'exploration. Il est aussi possible de consulter des *oracles* qui mettent en oeuvre différentes heuristiques afin d'extraire diverses informations sur la figure. Ceux-ci peuvent être utiles pour produire ou vérifier des conjectures. Il faut cependant être prudent, car leur diagnostic n'est pas infallible. En effet, ils n'utilisent habituellement pas de méthode formelle, mais plutôt des algorithmes numériques, pour tirer leurs conclusions. Il est donc, en théorie, possible de produire, entre autres, des constructions qui passeront les tests numériques, mais dont le résultat ne peut être prouvé formellement. La représentation des objets géométriques sous forme d'équation algébrique est aussi une fonctionnalité répandue. Elle peut permettre à un élève du niveau adéquat de comprendre le lien

entre ses constructions et les équations associées. Les équations permettent aussi de détecter des problèmes qui ne sont pas visibles directement sur la figure, par exemple l’alignement de points, à cause de la résolution limitée de l’écran. Certains micromondes offrent même un moteur pour manipuler les expressions algébriques et permettent de produire un graphe de l’équation résultante. Chaque logiciel offre aussi un langage de script qui permet d’automatiser certaines opérations ou de créer facilement des constructions complexes. Certaines fonctionnalités vont au-delà de la géométrie euclidienne, mais sont accessibles au besoin.

Toutes ces caractéristiques rendent les micromondes intéressants pour l’enseignement de la géométrie d’une manière différente et ils offrent beaucoup plus de possibilités que l’environnement papier/crayon classique. Cependant, leur efficacité dépend des tâches proposées et du contrôle de l’enseignant face à l’engagement de l’élève. En effet, ces logiciels ne contiennent aucune mesure de contrôle qui permette de restreindre les actions de l’élève à la tâche proposée. De plus, ils n’offrent aucune aide ni aucun tutoriel pour aider l’élève à résoudre le problème posé. Ils ne forment donc pas des systèmes tutoriels intelligents, mais ils sont tout de même intéressants dans le cadre de notre projet, car ils forment une base sur laquelle il est possible d’ajouter de l’aide intelligente.

2.2.4 Assistant à la démonstration et preuve automatique

Construire et manipuler des figures géométriques dynamiques qui possèdent certaines propriétés à l’intérieur d’un micromonde peut s’avérer être un défi intéressant et amusant. Cependant, toutes les observations que l’on peut faire par rapport aux caractéristiques invariantes d’une figure dynamique ne sont que des conjectures qu’il faut prouver. Pour aider les utilisateurs dans ce processus, les concepteurs de logiciels ont produit des systèmes de preuve automatique. Par exemple, il existe des systèmes comme Coq (Coq Development Team, 2015) ou Isabelle (Li et Li, 2012; Wenzel, 2015; University of Cambridge et Technische Universität München, 2015) qui sont des moteurs de déduction génériques qui sont en mesure de traiter de la logique de premier ordre, mais aussi d’ordres supérieurs. Il y a aussi GCLCProver (Janićić et Quaresma, 2006) qui est destiné plus spécifiquement aux preuves en géométrie. Étant donné que leur utilisation nécessite la connaissance d’un langage qui est difficile à maîtriser pour un élève de niveau secondaire, des programmeurs les ont intégrés à l’intérieur de logiciels de géométrie dynamique.

Parmi les systèmes intéressants que nous avons recensés, il y a premièrement GeoView (Bertot et al., 2004) qui permet d’afficher une figure qui illustre un théorème encodé dans Coq pourvu que l’on utilise le dictionnaire de GeoView pour alimenter le moteur de déduction. À l’inverse, GeoProof (Narboux, 2007, 2006) permet de construire une figure dans une interface

de géométrie dynamique et traduit ses caractéristiques dans le langage de Coq. Par la suite, l'utilisateur peut inscrire une liste d'hypothèses et une conjecture et demander si celle-ci peut être prouvée, de manière automatique par le moteur interne ou de manière interactive avec l'interface Coq. En combinant les deux derniers systèmes, nous obtenons le Java Geometry Expert (JGEX) (Ye et al., 2011), qui est une version évoluée du Geometry Expert (GEX) (Chou et al., 2000). En effet, JGEX permet d'injecter les caractéristiques d'une figure dynamique dans le moteur de déduction et d'afficher une nouvelle figure faisant état des résultats obtenus. Il est aussi possible de créer manuellement des animations sur la figure pour produire des preuves sans mots. Le MMP/Geometer (Gao et Lin, 2004) est semblable à JGEX, car il est une version dérivée de ce dernier. Par contre, il permet d'entrer les énoncés selon quatre modes distincts, y compris en langue naturelle, et est en mesure de faire les traductions entre ces quatre représentations. Tous ces systèmes utilisent des méthodes similaires pour produire des preuves qui sont formelles et qui sont souvent difficiles à comprendre. Ils ne sont donc pas compatibles avec le genre de preuve qui est demandé aux élèves.

D'autres systèmes utilisent les moteurs de preuve automatique d'une manière différente. Par exemple, Cinderella (Kortenkamp et Richter-Gebert, 2004; Kortenkamp, 1999, 2013), un micromonde qui offre des fonctionnalités comparables à GeoGebra, contient un moteur de preuve automatique pour gérer les constructions. En effet, il n'est pas accessible à l'utilisateur, mais Cinderella s'en sert, par exemple, pour éviter d'attribuer des noms distincts à un élément unique. Geometry Explorer (Wilson et Fleuriot, 2005; Hvidsten, 2005) permet, comme dans GeoProof, de construire une figure qui est employée pour définir les hypothèses. Par la suite, il utilise un moteur de déduction automatique pour produire une preuve qui peut être facilement comprise par un humain. Il peut aussi générer les figures correspondantes aux différents pas de la preuve. Cependant, cette preuve n'utilise pas les justifications de la géométrie euclidienne. Le logiciel permet tout de même de voir différentes preuves possibles sous forme de graphe, ce qui est une caractéristique intéressante.

Enfin, le logiciel qui se rapproche le plus de notre projet est Chypre (Bernat, 1993), soit un logiciel d'aide au raisonnement. Le but de l'auteur était de proposer un système qui permette aux élèves d'explorer le problème sans s'encombrer de détails inutiles. Il a donc choisi de se concentrer sur les figures pertinentes et les résultats qu'il juge importants à la suite de l'observation du fonctionnement de l'expert. Le logiciel permet de construire une figure et d'entrer des énoncés en tant qu'hypothèse ou en tant que conjecture. Pour chaque énoncé qui est soumis au système, un réseau de déduction local est produit, selon certaines règles qui sont déduites de l'observation de l'expert, et est ajouté au réseau de déduction du problème. S'il est possible de relier la conjecture aux hypothèses, celle-ci est donc prouvée. Chaque fois qu'une conjecture est prouvée, le résultat est propagé dans le réseau pour mettre

à jour le statut de tous les énoncés. L'élève, à mesure qu'il inscrit des énoncés, voit leur statut se modifier. Le système peut aussi fournir une trace des déductions produites par l'élève. Celle-ci peut cependant contenir des déductions inutiles à la production d'une preuve finale et c'est à l'élève de faire le tri. Ce système est intéressant, car il représente les preuves sous forme d'un graphe, qui n'est malheureusement pas visible à l'interface, et il produit des preuves qui ressemblent à ce qui est exigé par les enseignants du secondaire. Toutefois, les détails concernant la mise en oeuvre du logiciel Chypre ne sont pas divulgués. De plus, aucune aide intelligente n'a été implantée. Il est aussi possible d'inscrire des conjectures qui ne correspondent pas à la réalité de la figure et le système les accepte, car les auteurs veulent permettre, entre autres, des raisonnements par l'absurde.

Les systèmes de preuve automatique recensés sont donc généralement incompatibles avec le genre de système que nous voulons concevoir. En effet, ils permettent de vérifier des conjectures ou de déduire de nouveaux faits automatiquement, mais n'offrent aucun support pour l'élaboration d'une preuve. De plus, à l'exception des preuves générées par le logiciel Chypre, les preuves produites ne ressemblent généralement pas à celles que l'on retrouve dans les manuels scolaires. Il est donc difficile d'utiliser ces systèmes sans y apporter des modifications importantes pour y introduire un système d'aide intelligent adapté. En effet, dans notre cas, il nous faut un système qui accepte un ensemble de solutions qui correspondent aux connaissances de l'élève. Cependant, il s'agit d'une avenue que nous pourrions explorer plus en profondeur lors de phases ultérieures du projet, dans le but d'accepter des preuves qui ne seraient pas prises en compte par notre système, mais qui sont tout de même valides. En ce sens, il sera intéressant de suivre l'évolution de GeoGebra dans lequel on travaille actuellement à intégrer un système de preuve automatique (Botana et al., 2015).

Le but de cette section était de faire un survol des ressources disponibles pour l'apprentissage des mathématiques avant de passer aux véritables systèmes tutoriels intelligents en géométrie. Nous avons premièrement recensé des offres de service de tuteurs et des cours offerts en ligne sur différentes plateformes. Toutes ces ressources étaient gérées par des humains et n'offraient pas de système d'aide intelligente. Cependant, certains logiciels permettent de gérer le cheminement à l'intérieur d'un curriculum de façon automatique, donc de générer un itinéraire de problèmes selon les connaissances modélisées de l'élève. Ce genre de système, ainsi que les modèles de l'apprenant associés, n'est pas pertinent pour notre projet, car notre but est d'aider l'élève au cours de la résolution d'un problème choisi par l'enseignant. Nous avons ensuite présenté les micromondes ainsi que les systèmes de preuve automatique. Il s'agit d'outils intéressants pour l'exploration et la découverte. Par contre, ils n'offrent pas d'aide à la résolution ni de moyens pour s'assurer que l'élève réalise l'activité proposée par l'enseignant. En ce qui concerne les démonstrateurs automatiques, à part Chypre, les preuves

produites par ces moteurs ne correspondent pas à ce qui est attendu de la part des élèves. Ils peuvent tout de même permettre de valider des pas de démonstration et pourront être utiles pour une prochaine phase du projet. Nous avons par contre intégré un micromonde à notre système afin de donner accès à l’exploration de la figure dynamique et permettre à l’élève de découvrir des conjectures qu’il pourra prouver par la suite. Dans la section suivante, nous présentons les principaux systèmes tutoriels intelligents en géométrie et nous allons voir comment ils ont adapté certains modèles et systèmes présentés précédemment.

2.3 Systèmes tutoriels en géométrie

Jusqu’à présent, tous les systèmes présentés n’offrent pas d’aide intelligente à la résolution d’un problème particulier de démonstration en géométrie. Les micromondes peuvent cependant être utiles lors d’activités d’exploration pour découvrir et tester des conjectures. Ils forment donc une base intéressante sur laquelle il est possible de construire un système tutoriel intelligent, comme le montre notre recension des logiciels. Dans cette section, nous présentons les dix systèmes ou familles de systèmes qui offrent une aide intelligente à la démonstration, soit ceux qui sont les plus pertinents dans le cadre de notre projet. Ils n’offrent pas tous le même type d’aide ni le même degré d’interactivité ou de liberté dans le processus de résolution. En ce qui concerne la figure, nous avons recensé des systèmes n’offrant qu’une figure statique jusqu’à ceux permettant la construction d’une figure dynamique soutenue par une aide intelligente. Pour nous y retrouver, nous avons construit un tableau synthèse (tableau 2.1) qui peut être consulté à la fin de ce chapitre. Les dix systèmes choisis sont présentés dans un ordre qui nous semble logique selon leurs caractéristiques et selon les groupes de recherche qui les ont conçus. Pour chacun des systèmes, nous débutons par une mise en contexte en énumérant les principaux systèmes qui font partie de la même famille. Nous poursuivons par une description de l’interface et de ses possibilités du point de vue de l’utilisateur. Une analyse détaillée du fonctionnement interne, basée sur les objectifs de notre recherche, est ensuite présentée. Pour chaque système, nous terminons par une brève critique dans le but d’expliquer en quoi il ne correspond pas à nos objectifs.

2.3.1 Geometry Tutor

Au cours des années 1980, des chercheurs de l’université Carnegie-Mellon ont développé le Geometry Tutor (Anderson et al., 1985; Guin et al., 1991; Ritter et al., 2010), qui est probablement le premier système tutoriel intelligent pour l’enseignement des preuves en géométrie. Celui-ci est basé sur la même théorie cognitive que le LISP Tutor (Anderson et Reiser, 1985; Corbett et al., 2000) qui a été conçu en parallèle. Cette théorie a aussi donné naissance à l’Al-

gebra Tutor (Koedinger et al., 1997). Ces trois tuteurs, ayant une base commune, offrent des interactions semblables. En effet, ils sont tous en mesure de suivre la progression de l'élève, d'indiquer ses erreurs et de lui proposer la prochaine étape de résolution à la demande. Au cours des années, l'interface du Geometry Tutor s'est améliorée (Ritter et al., 2010) et le système d'aide s'est raffiné, mais ses fondations sont restées sensiblement les mêmes. Dans la version la plus récente, on propose un problème de preuve auquel est associée une figure statique. Dans une autre fenêtre, les hypothèses et la conclusion sont inscrites de façon à créer les bases d'un arbre de preuve que l'élève doit construire, comme c'était le cas dans la version originale. Pour résoudre le problème, l'élève doit créer de nouveaux noeuds dans l'arbre à partir d'un noeud ou d'une combinaison de noeuds existants. Le système force donc l'élève à travailler en chaînage avant ou arrière, car il n'est pas possible de créer un nouveau noeud qui n'est relié à aucun autre pour proposer une conjecture. Dans chaque noeud créé, l'élève doit inscrire un résultat intermédiaire et la justification. L'énoncé de résultat est construit à l'aide de symboles dans une barre d'outils et par la sélection d'éléments sur la figure. Ceux qui sont relatifs à un énoncé sélectionné sont mis en surbrillance. Les justifications sont choisies à l'aide d'un glossaire qui contient l'énoncé des règles ainsi que des exemples d'application. Lorsque l'élève inscrit une information erronée, la couleur de fond passe au rouge afin d'offrir une rétroaction immédiate. S'il s'agit d'une erreur courante qui est reconnue par le système, un message apparaît pour indiquer la source probable de l'erreur. Dans le cas où un élève serait bloqué dans son processus de résolution, il est possible de demander de l'aide pour déterminer la prochaine étape. Dans ce cas, il appuie sur un bouton qui lui offre un premier message relativement vague concernant son plan de résolution. Il peut alors demander à nouveau de l'aide et obtenir un message un peu plus précis. En répétant le processus, jusqu'à cinq fois, on obtient un message qui énonce précisément l'étape à réaliser. Le système considère que le problème est résolu lorsqu'il existe un chemin, qui ne contient aucune erreur, qui relie les hypothèses à la conclusion. L'élève a alors une représentation graphique de sa preuve et peut passer au problème suivant.

En ce qui concerne le fonctionnement interne du système, il est principalement basé sur le traçage de modèle, soit la théorie cognitive ACT-R (Anderson et al., 1990; Anderson, 1996; Anderson et Schunn, 2000). Celle-ci définit les savoirs comme un ensemble de connaissances déclaratives ainsi que des règles de production. Une connaissance déclarative est un ensemble de faits qu'une personne connaît par rapport à ce qu'elle observe. Par exemple, en voyant la chaîne « $3 + 7$ », notre cerveau peut instantanément la remplacer par la valeur 10, sans effort de notre part, car nous connaissons notre table d'addition. Ces connaissances déclaratives sont utilisées comme prémisses dans les règles de production. Ces dernières encodent les stratégies de résolution de problème sous forme d'énoncés conditionnels. Par exemple, en

géométrie on peut avoir une règle qui dit «SI le but est de prouver que deux triangles sont congrus et qu'un angle est congru, ALORS ajouter un but secondaire qui consiste à prouver la congruence des deux côtés adjacents à l'angle». Elles encodent autant le résultat d'application de théorèmes que les stratégies de résolution du problème. Ces règles sont apprises par les élèves et se raffinent en observant différents exemples d'application pour tendre vers la règle idéale régissant le domaine. Ce sont ces règles idéales, concernant la résolution de problèmes en géométrie, qui ont été implantées dans Geometry Tutor. En appliquant ces règles sur un énoncé, il est possible de résoudre le problème à la manière d'un expert. On peut aussi introduire des règles erronées pour modéliser les erreurs fréquentes observées et permettre au système de proposer une aide appropriée. Le système modélise le cheminement de l'apprenant comme étant l'application des règles de production ainsi que l'état supposé de sa mémoire de travail. À tout moment, il est donc en mesure de déterminer la règle de production optimale qui permettra à l'élève de poursuivre son travail. Cependant, le système accepte aussi que l'élève travaille en utilisant des règles qui ne sont pas optimales et sera en mesure de suivre sa progression. Le Geometry Tutor (Anderson et al., 1985; Guin et al., 1991; Ritter et al., 2010) a été validé par la procédure standard de prétest et post-test. Cependant, en ce qui concerne la théorie ACT-R, elle a été mise à l'épreuve en utilisant différents capteurs biométriques, y compris les IRM (Anderson et al., 2004). Le Geometry Tutor est un système intéressant, car il est utilisé avec succès depuis plusieurs années et qu'il offre une représentation graphique de la preuve sous forme d'arbre. Cependant, même si la théorie ACT-R est un modèle solide, elle ne permet pas de prendre en compte les initiatives de l'élève qui sortent du cheminement rigoureux et néglige donc l'aspect exploratoire de la résolution de problème, ce que nous avons choisi de considérer avec ce projet. Le système ne propose pas, non plus, d'autres pistes de solution dans le cas d'un blocage, car il se concentre sur l'application de la règle optimale ce qui n'est pas l'idéal dans un environnement favorisant l'exploration.

2.3.2 Geometry Explanation/Cognitive Tutor

Quelques années après l'élaboration du Geometry Tutor (Anderson et al., 1985; Guin et al., 1991; Ritter et al., 2010) enseignant les preuves, le curriculum PACT de l'université Carnegie-Mellon a été modifié afin de donner une plus grande place aux problèmes concrets, au détriment des preuves. Le PACT Geometry Tutor (Aleven et al., 1998; Aleven et Koedinger, 2000) a donc été créé et, dans sa première version, ne demandait que la valeur numérique de différents éléments d'une figure statique fournie. Cependant, lors de l'évaluation de l'efficacité du logiciel, les concepteurs se sont aperçus que certains élèves utilisent des heuristiques basées sur leurs perceptions sans pouvoir fournir de justification ce qui mène à des connaissances superficielles. Le système a donc été modifié dans le but de demander non seulement la réponse,

mais aussi la justification qui lui est rattachée, ce qui a produit le Geometry Cognitive Tutor (Aleven et Koedinger, 2002; Aleven et al., 2006; Roll et al., 2014). Dans celui-ci, les justifications sont choisies à partir d'un glossaire, mais une version nommée Geometry Explanation Tutor (Aleven et al., 2001a,b) demandait des justifications en langue naturelle. Tous ces logiciels partagent donc une interface semblable et seul le traitement des justifications diffère. À l'écran, on a premièrement une fenêtre qui affiche l'énoncé du problème, accompagné d'une figure statique, qui contient une zone semblable à une feuille de calcul pour y inscrire ses réponses et justifications. On retrouve aussi une fenêtre montrant une vue schématique de la figure, une autre donnant accès au glossaire contenant des définitions ainsi que des exemples et une dernière permettant d'afficher des messages. Enfin, le degré de maîtrise de différents concepts est affiché. Seules les informations inscrites dans la feuille de calcul simulée sont analysées par le logiciel et une rétroaction immédiate est fournie à l'élève. Ce dernier doit donc entrer des valeurs et des justifications afin de compléter le tableau et peut consulter le glossaire à sa guise. S'il inscrit une valeur incorrecte, le système le signale et un message peut s'afficher s'il s'agit d'une erreur courante reconnue. L'élève peut aussi demander de l'aide s'il est bloqué dans son processus de résolution. Dans ce cas, le système propose une aide hiérarchique contenant un total de sept messages allant du plus générique au plus précis. Pour chaque action, bonne ou mauvaise, le niveau de maîtrise des différents concepts est ajusté en conséquence. Un élève qui abuse du système d'aide en demandant toujours l'aide la plus précise verrait ses niveaux diminuer, car on considère qu'il ne maîtrise pas la matière pour résoudre le problème posé. Ces niveaux sont utilisés pour construire l'itinéraire de problèmes à proposer à l'élève, mais servent aussi d'outil de diagnostic pour l'enseignant.

Ces systèmes, étant des dérivés du Geometry Tutor (Anderson et al., 1985; Guin et al., 1991; Ritter et al., 2010), sont aussi basés sur la théorie cognitive ACT-R (Anderson et al., 1990; Anderson, 1996; Anderson et Schunn, 2000) et sur le traçage de modèle. En effet, chaque ligne du tableau, associant une valeur à sa justification, équivaut à un pas de preuve en géométrie euclidienne. Les concepteurs ont donc encodé les connaissances et les règles du domaine pour être en mesure de résoudre ce nouveau type de problème. Ils ont aussi produit les règles concernant les erreurs les plus courantes afin de leur associer des messages d'aide appropriés. Le système applique donc les différentes règles pour résoudre un problème et utilise le graphe produit pour modéliser le cheminement de résolution de l'élève. Ce dernier n'a d'autre choix que de travailler en chaînage avant, car il ne lui est pas permis de deviner les valeurs selon sa perception de la figure. Le système est en mesure de suivre sa progression et de lui proposer une aide appropriée ou l'application de la prochaine règle optimale en cas de besoin. L'aide concernant les justifications est proposée de façon similaire à celle pour les valeurs numériques, car les justifications proviennent d'un glossaire sauf pour le Geometry

Explanation Tutor (Aleven et al., 2001a,b). Dans ce dernier, les justifications doivent être écrites en langue naturelle et doivent être complètes, car l'identification seule du nom de la règle ne suffit pas. Les concepteurs ont donc implanté un système d'analyse de la langue naturelle. Celui-ci convertit premièrement le texte inscrit en un ensemble de concepts basé sur une ontologie adaptée aux justifications géométriques qu'ils désirent modéliser. Ils ont ensuite construit une hiérarchie qui contient différentes combinaisons des concepts identifiés dans une réponse jugée complète et ont associé des messages d'aide à chaque noeud. Donc, pour chaque justification incomplète fournie par l'élève, on lui propose un correctif lui permettant de la compléter. Ce système propose un correctif pour les phrases individuelles et ne prend pas en compte l'ensemble du dialogue, ce qui ne permet pas de détecter une solution stagnante. Le classificateur du Geometry Explanation Tutor a été comparé au travail des experts et sa performance est généralement acceptable. Les systèmes ont aussi été testés à l'aide de prétest et post-tests et il a été observé une amélioration concernant la compréhension lorsque les élèves doivent fournir des justifications. Ces systèmes sont donc importants, car ils permettent d'approfondir les connaissances dans le cas où la résolution pourrait être possible, avec un peu de chance, en utilisant des heuristiques ad hoc. Par contre, notre système ne doit pas se limiter aux problèmes numériques et doit proposer des problèmes riches et permettre une exploration qui n'est pas contrainte à une séquence déterministe. L'approche d'analyse de la langue naturelle est cependant intéressante, mais elle ne fait pas partie de nos objectifs.

2.3.3 Advanced Geometry Tutor

Le Geometry Tutor (Anderson et al., 1985; Guin et al., 1991; Ritter et al., 2010) permet de construire, en utilisant le chaînage avant ou arrière, une preuve qui ne requiert pas la construction de nouveaux points ou segments sur la figure statique fournie. Cependant, il arrive que certains problèmes proposés en classe nécessitent des constructions supplémentaires, donc Matsuda a créé l'Advanced Geometry Tutor (AGT) (Matsuda et VanLehn, 2005; Matsuda, 2004) pour enseigner une stratégie de preuve intégrant des constructions. En fait, il souhaitait savoir s'il était plus facile d'élaborer de telles preuves en chaînage avant ou arrière. Le logiciel proposé dispose d'une première fenêtre qui présente l'énoncé du problème ainsi qu'une figure statique. Cependant, cette figure est utilisée pour compléter des énoncés, en sélectionnant les points appropriés, et c'est sur celle-ci que les constructions apparaissent. Dans le cadre de l'AGT, les constructions consistent essentiellement à relier, par un segment, deux points déjà existants. Le logiciel offre aussi une fenêtre montrant la preuve en deux colonnes que l'élève doit compléter. Celui-ci doit proposer des énoncés, ainsi que les justifications et prémisses associées, en chaînage avant ou arrière selon la version d'AGT qu'il utilise. Une troisième fenêtre présente un navigateur de postulats dans lequel la figure utilisée à titre d'exemple peut

s'adapter pour correspondre à celle associée au problème à résoudre. Les différentes étapes pour produire une inférence sont aussi affichées. Celles-ci correspondent à toutes les étapes qui touchent autant les actions physiques que les démarches intellectuelles que l'élève doit utiliser pour progresser dans la résolution du problème. Elles sont représentées sous forme de hiérarchie et l'étape courante, qui correspond aux instructions du tuteur, est affichée en gras. L'objectif est de faire prendre conscience du processus de résolution et d'imposer une stratégie systématique de travail. Enfin, le tuteur discute avec l'élève à l'intérieur d'une fenêtre de messages interactive. En effet, l'élève doit confirmer la lecture de chaque message soit en exécutant une action ou en appuyant sur le bouton «OK». Il lui est aussi possible d'accéder à la chronologie des messages du tuteur. Ce dernier utilise quatre types de messages pour interagir avec l'élève. Lors d'une première erreur, il lui propose d'essayer à nouveau. S'il fait une seconde erreur consécutive, alors le tuteur lui propose un message qui dépend de son niveau de compétence. Trois niveaux sont possibles, soit d'encourager l'élève à travailler sur l'étape courante, de lui dicter les actions à réaliser ou de lui démontrer ce qu'il devait faire. À chaque erreur consécutive, le message est plus précis et la compétence de l'élève est ajustée selon le message qui lui a permis de réaliser l'action escomptée.

Les messages du tuteur sont basés sur une solution du problème généré par GRAMY (Matsuda et VanLehn, 2004), un système de preuve formelle automatique intégrant les constructions. Celui-ci travaille avec les postulats de la géométrie euclidienne pour construire une preuve. Les auteurs ont fait l'hypothèse que les constructions simples sont habituellement associées à un pas de preuve. L'emploi d'un postulat, dans AGT, peut donc s'accompagner d'une construction lorsque c'est nécessaire. La stratégie de GRAMY est d'appliquer les postulats en chaînage avant, sur une figure de départ, jusqu'à atteindre la conclusion fournie ou jusqu'à ce qu'aucun nouveau fait ne puisse être généré. Dans ce dernier cas, il applique des postulats en chaînage arrière et s'arrête lorsqu'une construction est nécessaire, c'est-à-dire que certaines prémisses qui ne font pas partie de la figure peuvent y être ajoutées. Cette opération ajoute de nouveaux faits qui peuvent être utilisés par le moteur en chaînage avant. Une combinaison de chaînages permet donc à GRAMY de produire une preuve intégrant des constructions. Dans AGT, on récupère la preuve la plus courte pour un problème donné et pour chacun des pas de résolution, on lui attache la liste des étapes à réaliser pour appliquer le postulat. On ordonne ensuite les pas selon que l'on désire que l'élève travaille en chaînage avant ou arrière. L'agent tutoriel de l'AGT parcourt enfin cette liste qui indique, en détail, toutes les étapes que l'élève doit réaliser et produit des messages appropriés. L'élève est donc forcé à travailler dans un ordre préétabli sur la solution la plus courte au problème posé. La chronologie des actions étant déterminée à l'avance, le système tutoriel n'a qu'à connaître la dernière action de l'élève pour lui proposer des messages d'aide. L'AGT n'offre pas de bouton

pour obtenir une explication concernant une erreur ou un indice par rapport à la prochaine étape à réaliser. Seules des actions erronées permettent d'obtenir de l'aide. Deux versions de l'AGT, l'une employant le chaînage avant et l'autre, le chaînage arrière, ont été utilisées pour comparer la performance de ces deux stratégies pour la résolution de problèmes à l'aide de prétest et post-test. Par contre, il n'y a pas eu d'étude sur l'efficacité du logiciel par rapport à une approche classique. En résumé, l'AGT est intéressant, car il propose des preuves contenant des constructions simples, il offre un navigateur de postulats qui s'ajuste au problème posé et il est en mesure d'ajuster son niveau d'intervention à la compétence de l'élève. Par contre, le système est très directif et n'offre pas la possibilité d'explorer le problème ou même de travailler sur une solution qui n'est pas optimale, ce qui est incompatible avec nos objectifs. En fait, il offre plutôt des exemples guidés qu'un système tutoriel intelligent.

2.3.4 ANGLE

Les logiciels utilisant une approche formelle, comme le Geometry Tutor (Anderson et al., 1985; Guin et al., 1991; Ritter et al., 2010), sont en mesure d'aider l'élève à préparer la rédaction de sa preuve. En effet, ils le contraignent à travailler selon un ordre précis et l'obligent à fournir l'ensemble des inférences nécessaires. Cependant, en analysant la stratégie utilisée par les experts, Koedinger et Anderson (1990) se sont aperçus que ceux-ci préparaient un plan avant d'ajouter les éléments manquants à leur démonstration. Ils ont donc conçu le logiciel ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991) afin d'apprendre aux élèves à planifier leur démonstration et les aider à reconnaître les éléments importants d'une figure. Dans ce contexte, ils ont développé une théorie concernant les «Diagram Configurations» (DC) (Koedinger et Anderson, 1990), soit les configurations qui semblent être utilisées par les experts pour produire un plan de résolution. Il s'agit, en fait, de configurations remarquables dans une figure qui ont été judicieusement déterminées. Par exemple, nous pourrions identifier comme étant congrus deux triangles ayant un côté en commun et nous laissant croire que les autres côtés homologues sont congrus, ce qui est décrit comme un énoncé complet dans la théorie. Une fois identifiés, nous savons que les angles correspondants seront congrus et qu'il en sera de même pour les côtés. Cette observation nous permet de produire six énoncés partiels qui découlent de l'énoncé complet. Pour être en mesure de prouver la congruence, il suffit de prouver un sous-ensemble particulier des énoncés partiels, ce qui correspond aux moyens de prouver le DC. Un plan de résolution contient donc un ensemble de DC et pour compléter la preuve on doit ajouter les énoncés partiels qui permettent de les relier, ainsi que les justifications associées. ANGLE contient donc un moteur qui est en mesure de résoudre les problèmes en utilisant les DC. Pour créer l'arbre des solutions, le moteur commence par extraire les DC qu'il est en mesure de trouver dans la figure associée au problème et il

produit tous les énoncés partiels associés à ceux-ci. Il fusionne ensuite les énoncés semblables provenant de différents DC et des hypothèses du problème et propage le statut prouvé aux différents éléments. Si, à la fin du processus de propagation, la conclusion est prouvée, cela signifie qu'il existe au moins une démonstration pour le problème posé. Cette stratégie permet de réduire l'espace de recherche des solutions, par rapport à l'approche formelle, et semble correspondre à la stratégie utilisée par les experts. Les DC permettent aussi une modélisation de l'espace des solutions plus naturelle qui comprend les éléments de la planification.

L'interface du système ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991) a, par conséquent, été conçue afin de permettre la planification et la résolution de problèmes de preuve géométrique à l'aide des DC (Koedinger et Anderson, 1990). Il offre une fenêtre unique dans laquelle on retrouve, au départ, l'énoncé du problème ainsi que la figure correspondante dans le coin supérieur gauche. À gauche, une série de boutons est disponible pour créer différents objets et interagir avec le logiciel. Le reste de la fenêtre est au départ une feuille vierge dans laquelle l'élève pourra construire un arbre de preuve en utilisant les DC. Pour construire un DC, l'élève doit cliquer sur le bouton approprié et choisir les éléments de la figure qui correspondent. Si l'élève choisit un DC qui ne se trouve pas dans la figure ou s'il choisit les mauvais éléments, un message lui indiquant son erreur apparaît. Les DC acceptés par le système peuvent ensuite être placés dans la fenêtre à l'endroit choisi par l'élève. ANGLE n'impose aucune contrainte par rapport à l'ordre d'ajout ou l'emplacement des différents éléments. L'élève peut aussi ajouter des énoncés partiels, de façon similaire, en utilisant le bouton approprié. Il doit ensuite relier les différents éléments pour former un arbre de preuve. ANGLE vérifie si les liens sont valides et indique les erreurs à l'aide de messages. Enfin, l'élève, à l'aide d'un autre bouton, doit ajouter une justification pour chacun des liens créés afin de compléter sa preuve. ANGLE laisse cependant une totale liberté à l'élève qui peut utiliser les différents outils dans un ordre quelconque et il peut même ajouter des pas de preuve qui ne seront peut-être pas utiles dans sa preuve finale. La sélection d'un élément permet de l'afficher en surbrillance sur la figure originale. Lorsque l'élève est bloqué, il peut demander un indice en appuyant sur un bouton. ANGLE offre quatre niveaux d'indices allant du plus générique au plus précis. L'aide offerte dépend de la progression de l'élève, mais se produit toujours en chaînage avant. Par exemple, s'il manque des DC pour avoir un plan complet, le système va suggérer de construire le prochain DC manquant. Dans le cas où tous les DC nécessaires seraient présents, ANGLE propose de travailler sur les énoncés partiels et ainsi de suite. Le système a été comparé à Geometry Tutor (Anderson et al., 1985; Guin et al., 1991; Ritter et al., 2010) à l'aide d'une procédure de prétest et post-test et son efficacité est similaire. Ce logiciel est intéressant, car il offre à l'élève une totale liberté et lui permet d'appliquer une phase exploratoire sous forme de découverte des DC contenus dans la figure de base. De plus,

les DC proposent une stratégie de résolution semblable à celle utilisée par les experts et la démonstration est présentée sous forme d'arbre. Par contre, dans le cadre de l'enseignement de la preuve au secondaire, le paradigme utilisé est plus près de la géométrie formelle que des DC et notre objectif n'est pas de forcer l'élève à imiter les stratégies d'un expert, mais bien de lui permettre de développer les siennes. L'aide en chaînage avant, qui ne respecte pas toujours l'état cognitif de l'apprenant, et l'absence de figure dynamique constituent deux autres lacunes de ce système.

2.3.5 Baghera

Le but du projet Baghera (Webber et al., 2001; Balacheff et al., 2003) est d'offrir une architecture informatique, pour la conception d'environnements d'apprentissage, qui est basée sur des théories didactiques. Pour valider le concept, les auteurs ont conçu un système multi-agent pour l'enseignement des preuves en géométrie. Il s'agit d'une plateforme web qui offre une interface autant pour les enseignants que pour les élèves. D'une part, l'enseignant est en mesure de créer de nouveaux problèmes ou de modifier ceux qui sont existants et de les faire parvenir aux élèves. Il lui est aussi possible de suivre, en temps réel, la progression du travail des élèves et de discuter avec eux au moyen d'une fenêtre de clavardage à l'intérieur d'une classe virtuelle. L'interface de l'enseignant est gérée par deux agents distincts, soit l'un qui s'occupe de l'interface et des accès et l'autre qui assiste l'enseignant dans la conception et la diffusion des problèmes. De son côté, l'élève a premièrement accès à une liste de problèmes regroupés par thème, soit un sac d'école virtuel créé par l'enseignant. Il peut alors choisir un problème pour faire apparaître l'interface de résolution. Dans celle-ci, on retrouve l'énoncé du problème accompagné d'une figure dynamique gérée par Cabri-Java (Cabri-Java Project, 2004). Cette dernière peut être manipulée pour permettre de découvrir des conjectures, mais il n'est pas possible d'y ajouter de nouvelles constructions. L'élève doit rédiger sa preuve dans une fenêtre, acceptant du texte libre, en utilisant le clavier, différents outils et un dictionnaire de théorèmes. Lorsqu'il croit avoir complété sa preuve, il la soumet au système d'analyse qui lui retourne ensuite une preuve annotée. Pour chaque inférence, un diagnostic local est fourni par rapport à la cohérence de celle-ci. On indique aussi si la preuve, dans son ensemble, est cohérente et complète, c'est-à-dire qu'elle relie les hypothèses à la conclusion. Le système d'analyse est aussi en mesure de détecter et d'indiquer les inférences qui sont des analogies, c'est-à-dire qu'elles sont semblables à d'autres, utilisées précédemment dans la preuve. L'interface ne contient aucun système pour soutenir et aider l'élève lors de l'écriture de sa preuve. Pour gérer les interactions de l'élève, trois agents ont été implantés. Le premier permet de gérer l'interface et les accès de l'élève et est responsable de communiquer les informations utiles aux autres agents. Un second s'occupe de prendre des décisions didactiques et pédago-

giques concernant le cheminement de l'élève en lui proposant des problèmes appropriés selon son profil d'apprenant. Le dernier agent sert de médiateur en choisissant le système d'analyse qui servira à extraire les informations pertinentes de la solution de l'élève et en interprétant la réponse obtenue.

Le principal intérêt de ce système est la théorie didactique, cKc (Balacheff et Margolinas, 2005), sur laquelle est basé son modèle de l'apprenant. Cette théorie définit les différentes conceptions (C), correctes ou erronées, de l'apprenant par la relation $C = (P, R, L, \Sigma)$. Cette dernière signifie que pour une conception donnée, on y associe un ensemble de problèmes (P) dans lesquels l'utilisation de celle-ci permet de les résoudre en partie. En effet, pour résoudre un problème il faut habituellement faire intervenir plusieurs conceptions. L'ensemble des opérateurs (R) permet de transformer, en respectant la conception, un problème. On peut penser, par exemple, à l'emploi de propriétés ou de définitions pour justifier une conjecture en géométrie. Pour exprimer les opérateurs et les problèmes, un système de représentation (L), soit un langage, est utilisé. Enfin, un système de contrôle (Σ) permet de savoir dans quels contextes il est légitime d'appliquer la conception et contient des mesures permettant d'éviter sa contradiction. Dans le système Baghera (Webber et al., 2001; Balacheff et al., 2003), un agent existe pour chaque élément du modèle cKc, par exemple, un agent est créé pour chaque opérateur établi. Selon les actions de l'élève, les agents correspondants sont activés et forment des coalitions. Ces dernières correspondent aux conceptions de l'élève décrites selon le modèle cKc. Le défi consiste donc à relier les actions de l'élève, sous forme d'une preuve discursive, aux différents agents implantés. Pour y arriver, les concepteurs de Baghera ont eu recours à un moteur de preuve automatique capable de vérifier une preuve formelle fournie en logique du premier ordre. Ce moteur autorise notamment les inférences implicites et permet d'ajouter des points manquants à une figure. L'agent médiateur est responsable de convertir la preuve discursive en logique du premier ordre. Malheureusement, l'interprétation de l'analyse issue du moteur de déduction n'était qu'au stade de projet dans la documentation recensée. Une validation du système multi-agent a été réalisée en comparant ses diagnostics à ceux d'experts et ceux-ci concordent dans la majorité des cas. Le principal intérêt de ce système est la mise en oeuvre d'un modèle de l'apprenant basé sur une théorie didactique en utilisant une architecture multi-agent. L'élève est aussi libre d'explorer le problème et de rédiger sa preuve dans un ordre arbitraire et a accès à une figure dynamique pour découvrir des conjectures. De plus, il offre une interface pour construire des problèmes et une classe virtuelle pour permettre à l'enseignant d'interagir avec les élèves. Par contre, même si le modèle multi-agent a été validé, il ne peut être utilisé, car l'analyse des rapports du moteur de déduction n'a pas été implémentée. Le principal problème de ce système est cependant l'absence totale de système tutoriel pour aider l'élève à rédiger sa preuve.

2.3.6 DEFI

Des expérimentations (Gras, 1988) visant à comprendre les difficultés des élèves à produire des preuves, il en est ressorti deux constats importants. Le premier concerne la figure qui est souvent perçue en tant que simple représentation de l'énoncé et dont les propriétés sont mal interprétées par les élèves. D'autre part, la rédaction de la preuve est difficile, car les élèves n'en comprennent pas toujours l'objectif, n'ont pas de plan précis pour la produire et peuvent confondre les éléments qui la composent. Dans le but d'aider à changer la perception des élèves face aux preuves, le logiciel DEFI (Démonstration et Exploration de la Figure Interactives) (Gras, 1988; Almouloud, 1992) a été développé. Au démarrage du logiciel, l'élève sélectionne d'abord un problème et l'énoncé correspondant est affiché. À partir de celui-ci, il doit construire une figure sur papier qui lui servira à répondre à des questions par la suite. Par contre, dans la version Cabri-DEFI (Luengo et Balacheff, 1998), la figure est construite à l'écran à l'aide de Cabri-Géomètre (Baulac, 1990; Kordaki et Mastrogiannis, 2006; CABRILOG SAS, 2009) et peut ensuite être validée par DEFI. Ce dernier utilise des oracles pour vérifier si la figure est complète et si elle respecte l'énoncé et fournit enfin un verdict à l'élève. Lorsque celui-ci a terminé de construire sa figure, il passe au module d'exploration de la figure dont l'objectif est de l'aider à planifier sa preuve. L'exploration est mise en oeuvre par une série de questions, en mode texte, auxquelles l'élève doit répondre en se reportant à la figure construite précédemment. Elles ont pour but d'aider l'élève à découvrir un plan adéquat pour résoudre le problème et de s'assurer qu'il est en mesure de le mettre en oeuvre. Si l'élève ne donne pas de réponses satisfaisantes, DEFI lui propose de revoir les hypothèses et de recommencer l'exploration de la figure. Lorsque son plan a été validé, il passe ensuite au module de démonstration qui lui permet de rédiger sa preuve. Dans ce cas, il doit, pour chaque pas de preuve, d'abord fournir la conclusion de celui-ci. Il choisit un énoncé parmi ceux qui sont offerts et ajoute ensuite le nom des éléments géométriques permettant de le compléter. Par la suite, il sélectionne la justification et termine par les hypothèses. Le pas est analysé par DEFI et une rétroaction concernant la cohérence entre hypothèse, justification et conclusion est affichée. L'élève entre ses pas de preuve jusqu'à ce qu'il ait prouvé la conclusion. À tout moment, il a accès à un fichier qui lui indique la progression du pas de preuve courant, soit l'ensemble des énoncés soumis. Un autre fichier contient les énoncés qui ont déjà été prouvés par l'élève ainsi que ceux que ce dernier a affirmé être en mesure de prouver par ses réponses dans le module d'exploration de la figure.

Pour modéliser le domaine relié à un problème, DEFI (Gras, 1988; Almouloud, 1992) utilise deux stratégies différentes, soit une pour chaque module. Dans le cadre de l'exploration de la figure, la hiérarchie des questions est programmée par un expert et est unique à chaque

problème. La structure est, en général, construite de manière à proposer une exploration en chaînage arrière en découpant le but principal en buts secondaires. Elle représente un nombre limité de stratégies de résolution et peut contenir des cycles. Il est donc possible qu'une stratégie valide ne soit pas reconnue par le système qui, par conséquent, forcera l'élève à modifier son plan. Les réponses associées à des stratégies non reconnues peuvent même amener le système à poser une série de questions qui forment une boucle. En ce qui concerne le module de démonstration, les preuves acceptables, qui correspondent aux stratégies proposées par l'expert, sont implantées sous forme d'un ensemble d'inférences. Pour évaluer l'inférence proposée par l'élève, le système vérifie premièrement si toutes les hypothèses ont été démontrées et la compare ensuite à l'ensemble des inférences valides pour identifier celle qui ressemble le plus à celle qui a été proposée. Les connaissances de l'élève sont, par conséquent, modélisées par une liste d'énoncés démontrés qui peuvent être utilisés en tant qu'hypothèse, ce qui l'oblige à produire une preuve en chaînage avant. À part une rétroaction plus ou moins précise lorsque l'élève termine un pas de preuve, le système n'offre aucune aide pour le guider dans la production de sa preuve. Le système, qui a été conçu pour répondre à un besoin réel identifié lors de séances d'observations en classe, ne semble s'appuyer sur aucune théorie didactique. En ce qui concerne l'évaluation de DEFI, les auteurs ont utilisé essentiellement une analyse qualitative. En effet, ils ont organisé des séances d'observation et fait passer des tests pour voir si le logiciel avait une influence sur le processus de démonstration, sans pour autant quantifier le gain réel sur la note finale de l'élève. Le principal intérêt de ce système est qu'il force l'élève à réfléchir sur sa stratégie en utilisant la figure qu'il a construite avant de se lancer dans l'écriture de sa preuve formelle. La version Cabri-DEFI (Luengo et Balacheff, 1998) permet, en plus, d'avoir accès à une figure dynamique dont la validité peut être vérifiée. Cependant, la structure rigide du raisonnement, autant au niveau de l'exploration que de la preuve, ainsi que l'absence de tuteur en mesure de guider l'élève lors de l'élaboration de sa démonstration sont deux lacunes majeures de ce système. De plus, les solutions acceptables sont relativement limitées.

2.3.7 Cabri-Euclide

L'objectif du système Cabri-Euclide (Luengo, 2005, 1997) est d'offrir un environnement expérimental afin de permettre aux élèves de construire leurs connaissances concernant la démonstration. Pour y arriver, ce système accepte les conjectures, qui peuvent même se révéler fausses, car il ne force pas l'élève à reproduire une preuve formelle déterminée. La conception de son interface, ainsi que le choix des interactions, s'appuie sur différentes théories didactiques qui ont été élaborées, entre autres, par Brousseau (1998), Duval (1995) et Balacheff (1987). Ce système met en oeuvre une interaction entre Cabri-Géomètre (Baulac, 1990; Kor-

daki et Mastrogiannis, 2006; CABRILOG SAS, 2009), un logiciel de géométrie dynamique, et Euclide, le système discursif d'analyse de preuve développé dans le cadre du projet. L'interface de Cabri-Euclide offre trois fenêtres principales, soit une première qui permet de construire une figure dynamique, une seconde qui permet d'écrire la preuve et une dernière qui présente le graphe de la preuve en cours d'élaboration. L'élève, devant résoudre un problème proposé par son enseignant, doit commencer par construire une figure, qui respecte les contraintes imposées, dans la fenêtre de Cabri-Géomètre. Celle-ci doit être cohérente par rapport aux énoncés que l'élève a l'intention d'utiliser, car Euclide vérifie systématiquement si les énoncés sont valides sur la figure en faisant appel aux oracles. Lorsque la véracité d'un énoncé ne peut être vérifiée, l'oracle de Cabri-Géomètre peut, dans certains cas, fournir un contre-exemple. Lorsque l'élève croit avoir une figure suffisante pour résoudre le problème, il peut proposer des énoncés à Euclide. Pour chaque énoncé, l'élève doit indiquer son statut, soit qu'il s'agisse d'une hypothèse, d'une conclusion, d'une conjecture, etc. Il peut ensuite y ajouter une ou plusieurs preuves qui lui sont associées. Il choisit donc un théorème ainsi que les prémisses nécessaires et Euclide vérifie la cohérence locale de l'inférence ainsi que le statut des énoncés reliés. Le système produit une rétroaction sous forme d'un message pour faire part à l'élève de toute incohérence dans sa proposition. L'élève peut donc entrer un ensemble d'énoncés, les justifier, et voir la liste de ceux-ci qui sont regroupés selon leur statut. Lors de l'ajout d'une inférence qui permet, par exemple de prouver une conjecture, il importe à l'élève de mettre à jour le statut de l'énoncé, mais Euclide émet un message si celui-ci choisit un statut invalide. Pour avoir une preuve complète, la conclusion doit pouvoir prendre le statut prouvé. La fenêtre du graphe, qui ne peut être manipulée, est construite dynamiquement en utilisant les informations concernant les différents énoncés. Il est à noter que pour tous les messages du système, un bouton permettant de demander une justification est accessible.

À l'intérieur de Cabri-Euclide (Luengo, 2005, 1997), les inférences sont modélisées en utilisant le LDL (Allen et al., 1992), soit un langage qui permet de représenter les figures géométriques sous forme de prédicats et de les manipuler. Un arbre syntaxique LDL est associé à chaque définition, ou théorème, ce qui permet de valider la cohérence d'une inférence en vérifiant l'isomorphisme entre l'arbre de l'inférence et celui de la définition. Cette représentation permet aussi à l'utilisateur de créer de nouvelles définitions à partir d'inférences qu'il a construites et lui évite l'obligation d'apprendre un langage formel. Les diagnostics du système sont cependant purement locaux à une inférence et sont déduits de la comparaison des arbres associés. En ce qui concerne l'état de l'apprenant, il est modélisé par l'ensemble des énoncés, des inférences et des définitions qu'il a ajoutés dans son espace de travail, ce qui correspond à l'état de sa preuve. Le système ne contient, par ailleurs, aucun énoncé de problèmes. Ces derniers sont, par conséquent, remis aux élèves sous format papier. Cabri-Euclide ne contient donc aucune

procédure pour s'assurer que l'élève respecte le contrat didactique et qu'il travaille sur le problème posé. Sans informations concernant la tâche à réaliser, il ne lui est pas possible de proposer une étape utile pour le cheminement de l'élève vers une solution. Par contre, l'élève a toujours accès à l'arbre de sa preuve pour lui permettre de voir ce qu'il lui reste à accomplir. En fait, il s'agit d'un micromonde intégrant une aide intelligente pour l'élaboration de preuves. Dans le cadre de l'évaluation du système, les auteurs souhaitaient vérifier l'impact de celui-ci sur le travail de l'élève et de ses conceptions face aux preuves. Il a donc utilisé une approche qualitative et a procédé à une série d'observations pour en arriver à la conclusion que le logiciel semble avoir un impact généralement positif auprès des élèves. Le principal intérêt de ce système, par rapport à notre recherche, est qu'il offre un environnement permettant une exploration libre et qu'il est même en mesure de traiter les conjectures fausses. De plus, il intègre une figure dynamique qu'il utilise pour valider les énoncés ainsi qu'un graphe de la preuve qui aide l'élève à comprendre la structure de la preuve. Par contre, même si Cabri-Euclide offre un processus de vérification des inférences et propose des messages qui indiquent les erreurs, il reste un micromonde dans lequel aucune aide n'est prodiguée pour orienter un élève en difficulté. De plus, il n'offre pas de mécanismes pour s'assurer que l'élève réalise la tâche qui lui a été proposée ou qu'il a introduit correctement les données du problème, ce qui peut être problématique pour une utilisation non supervisée en classe.

2.3.8 Mentoniez

Pour sa part, le système Mentoniez (Py, 1994, 1996, 2001) a été conçu dans le cadre d'un projet qui visait à produire un logiciel complet d'aide à la démonstration. Celui-ci se distingue, car il découpe l'activité de démonstration en quatre phases, soit la lecture de l'énoncé, l'exploration de la figure, l'élaboration de la preuve et la rédaction finale. Un élève qui désire utiliser le logiciel doit tout d'abord choisir un problème qui aura été préalablement construit par son enseignant. On lui présente alors une interface lui demandant d'inscrire les hypothèses ainsi que la conclusion du problème, ce qui correspond à l'étape de la lecture de l'énoncé. Il doit choisir des squelettes de phrases qu'il complète et qu'il place dans la catégorie appropriée. Si des énoncés invalides sont proposés, un message d'erreur est affiché. Lorsqu'il croit avoir terminé, il peut passer à l'étape de l'exploration de la figure si aucune erreur n'est détectée. Dans le cas contraire, le système lui indique les problèmes détectés, soit des énoncés manquants, superflus ou mal classés. La deuxième étape a pour objectif de permettre à l'élève de planifier sa preuve en analysant la figure. Le logiciel offrant uniquement une interface en mode texte, cette étape a été implémentée sous forme d'une série de questions portant sur la figure que l'élève a dessinée sur papier. La stratégie d'exploration se déroule en chaînage arrière, mais elle permet des retours arrière, ce qui offre la possibilité de modifier sa stratégie

en cours de route. Lorsque l'élève atteint les hypothèses, il passe à l'étape de l'élaboration de sa preuve. Lors de celle-ci, ce dernier propose des inférences qui font partie de sa preuve, sans être contraint à respecter un ordre déterminé. Chaque inférence est comparée à celles qui sont valides dans le cadre du problème et, si elle n'est pas trouvée, un message d'erreur est affiché. Le système classe automatiquement les énoncés utilisés par l'élève selon qu'il s'agit d'une hypothèse (ou propriété démontrée), d'une conjecture ou d'une propriété à démontrer et met à jour les statuts dynamiquement. Lorsque la conclusion du problème devient une propriété prouvée, l'élève passe à la rédaction finale. Pour réaliser sa rédaction, il doit premièrement placer les inférences utilisées lors de l'élaboration de sa preuve en ordre afin d'obtenir une preuve en chaînage avant. Par la suite, il doit composer les phrases qui correspondent à chacune des inférences choisies pour la rédaction. L'élève a accès aux éléments de son inférence ainsi qu'à un ensemble de mots permettant de compléter la phrase. S'il fait une erreur dans l'ordonnancement ou lors de la construction de la phrase, le système affiche un message d'erreur. Une fois que toutes les phrases ont été construites correctement, il a une preuve complète et a donc terminé de résoudre le problème.

Sur le plan technique, le HDL, créé dans le cadre de l'élaboration de Mentoniez (Py, 1994, 1996, 2001), est un langage logique de premier ordre qui permet de représenter les connaissances géométriques du système. Pour ajouter un problème dans le système, l'enseignant doit fournir les hypothèses ainsi que la conclusion sous forme de clauses HDL. Il peut aussi indiquer des pas implicites, pour accepter les raccourcis inférentiels, ou interdits. Un moteur de déduction formel est ensuite utilisé, jusqu'à atteindre la saturation, pour produire l'ensemble des démonstrations possibles dont on en conserve qu'un nombre déterminé, en privilégiant les plus courtes. En ce qui concerne les messages du système, ceux-ci sont générés à partir d'un catalogue d'erreurs courantes, mais un module a aussi été étudié afin de modéliser la formation des erreurs pour offrir de meilleurs diagnostics. De plus, un algorithme de reconnaissance de plan, inspiré des travaux de Kautz (1991, 1987), a été mis en oeuvre pour obtenir des diagnostics plus précis. En effet, il permet de déterminer la preuve courante sur laquelle travaille l'élève et permet donc d'interpréter le contexte de l'erreur. Pour y arriver, le système conserve un historique chronologique des actions de l'élève. Cette approche se distingue des méthodes classiques (Carberry, 2001; Gal et al., 2012; Sukthankar et al., 2014), par exemple les réseaux bayésiens ou l'analyse lexicale. En effet, ces dernières s'appliquent surtout à des domaines dans lesquels la séquence des actions est importante et permettent de traiter des erreurs d'exécution. Celles-ci sont génériques et se prêtent bien, par exemple, à l'interprétation des actions d'un utilisateur à l'interface d'un logiciel dans le but de lui proposer de l'aide adaptée, comme dans le cas d'EdCoach (Desmarais et al., 1993) ou du projet Lumière (Horvitz et al., 1998). Dans notre cas, la stratégie la plus pertinente, étant donné les objectifs

de notre projet, nous semble être celle proposée dans Mentoniez (Py, 1994, 1996, 2001), car elle permet de représenter les changements de plan de l'élève. Mentoniez a été évalué de manière qualitative auprès des élèves et des enseignants et a reçu une évaluation positive. Ce logiciel est intéressant, car il permet une phase d'élaboration de la preuve qui offre une totale liberté à l'élève et lui permet de travailler sur différentes preuves possibles. De plus, il découpe le travail en quatre phases, ce qui permet aux élèves d'attaquer un problème de façon systématique. Par contre, le travail sur la figure s'effectue sur papier, mais un prototype intégrant TALC (Desmoulin, 1994; Macrelle et Desmoulin, 1998), un système de construction et de vérification de figure, a été développé. La principale lacune du système est qu'il n'offre pas d'aide à la prochaine étape, donc un élève, sans une aide-externe, pourrait ne pas être en mesure de compléter le problème. Aussi, quoiqu'il admette les raccourcis inférentiels, les démonstrations demandées restent formelles et ne permettent pas d'intégrer le paradigme utilisé dans l'enseignement secondaire. Enfin, sa conception, même si elle est issue d'observations et a permis d'obtenir un logiciel intéressant, ne semble pas être basée sur des théories didactiques ou cognitives.

2.3.9 Géométrie

Gressier, depuis les années 1990, développe Géométrie (Gressier, 2015) qu'il décrit comme un assistant à la démonstration. Il offre, de plus, la possibilité de créer des exercices de construction et de démonstration. L'interface, au chargement du logiciel, ressemble aux micromondes de géométrie comme GeoGebra (Hohenwarter et Fuchs, 2004; Hohenwarter, 2013; International GeoGebra Institute, 2015). En effet, en cliquant sur des boutons, il est possible de construire différents éléments admis en géométrie euclidienne et ainsi obtenir une figure dynamique. Par contre, pour y parvenir, l'utilisateur doit compléter des phrases à trous en indiquant les éléments qu'il désire utiliser. Il offre aussi la possibilité d'appliquer des transformations, de travailler avec des vecteurs et de produire des courbes de variations. Ce logiciel se démarque cependant des autres micromondes de géométrie, car il est en mesure de produire une liste de déductions issues de l'analyse de la figure à mesure que l'utilisateur la construit. Par exemple, si l'on construit deux paires de droites parallèles, des énoncés concernant, entre autres, l'appartenance des points aux droites apparaîtront, mais on retrouvera aussi une proposition démontrable concernant le parallélogramme construit. Lorsque la figure est complète, il est possible de passer en mode de preuve. L'interface se transforme alors et présente à nouveau la fenêtre des données ainsi que celle des propositions auxquelles s'ajoute une fenêtre permettant de construire une inférence. L'utilisateur peut alors placer une proposition en tant qu'énoncé à démontrer et Géométrie lui propose les justifications ainsi que les énoncés compatibles. Il choisit donc sa justification et les énoncés requis avant de valider son inférence, ce qui met à

jour le statut des énoncés. Il construit sa preuve, une inférence à la fois, et peut afficher sa démonstration terminée sous forme de texte. Ces fonctionnalités s'apparentent à celles d'un micromonde, mais des outils sont disponibles pour produire des exercices. En effet, pour préparer un exercice de construction, un enseignant doit d'abord produire une figure à la manière d'un élève qui résout le problème. Il cache ensuite les constructions demandées à l'élève et identifie la figure de départ. L'élève peut alors ouvrir le fichier contenant l'exercice et tenter de réaliser la figure demandée. Même s'il n'utilise pas le même protocole que l'enseignant, Géométrix est en mesure de vérifier si ses constructions sont valides par rapport à l'énoncé et de lui indiquer ses erreurs. Il est possible de créer un exercice de démonstration de façon similaire. Dans ce cas, l'enseignant choisit, après avoir construit la figure, l'énoncé à démontrer, les données disponibles et les justifications acceptées. Dans ce cas, l'élève doit produire la preuve en chaînage avant et le logiciel l'avertit lorsqu'il fait des erreurs.

Étant donné qu'une partie de Géométrix (Gressier, 2015) est programmée en Prolog, cela nous porte à croire que les constructions géométriques ainsi que les inférences sont modélisées en utilisant des prédicats en logique de premier ordre. Toutes les démonstrations produites sont donc issues d'un moteur de déduction automatique utilisant les règles de géométrie formelle. Les théorèmes utilisés par le moteur de déduction peuvent être choisis par l'utilisateur, mais sont limités. Il arrive que certaines preuves ne soient pas possibles à l'interface du système, car les théorèmes nécessaires n'ont pas été implantés. Les connaissances de l'élève, en cours de résolution d'un exercice, sont représentées par l'historique de ses constructions, en ce qui concerne la figure, et par l'ensemble des inférences validées pour la partie reliée à la démonstration. En ce qui concerne les rétroactions lorsque l'élève fait une erreur, les possibilités offertes dans Géométrix sont nombreuses pour les exercices de construction. En effet, il est possible d'associer des messages d'erreur distincts pour chaque construction erronée que l'élève pourrait réaliser ainsi que pour chaque répétition d'une même erreur. L'enseignant peut aussi préparer un message, et même une animation, qui sera présenté à l'élève après un temps déterminé. La création de ces rétroactions demande cependant un travail fastidieux, mais nécessaire, car les messages créés automatiquement sont souvent difficiles à interpréter et peuvent faire référence à des éléments cachés. En ce qui concerne la preuve, le système ne permet pas d'adapter les messages. Ceux-ci font référence aux erreurs associées à l'inférence construite et peuvent par exemple indiquer si le nombre ou le type d'hypothèses est incorrect. Aucun système d'aide pour indiquer la prochaine étape, à un élève qui serait bloqué en cours de résolution, n'est offert. L'auteur du logiciel indique qu'il a consulté des enseignants pour l'élaboration du système, mais il ne fait référence à aucune théorie cognitive ou didactique. De plus, aucune documentation concernant des tests ou la validation du logiciel n'est accessible. Géométrix est un logiciel très intéressant dans le cadre d'une utilisation comme assistant à la

démonstration. En effet, les moteurs de déductions automatiques lors de la construction de la figure ainsi que l'interface de preuve qui s'adapte et qui permet de construire une preuve dans un ordre quelconque sont des caractéristiques intéressantes. Par contre, lorsqu'il est utilisé comme système tutoriel intelligent, ses capacités sont limitées. D'un côté, le traitement des erreurs concernant la figure est fastidieux et de l'autre les messages concernant la démonstration sont rudimentaires. Enfin, l'utilisation de la géométrie formelle n'offre pas la flexibilité nécessaire pour traiter les preuves acceptées par plusieurs enseignants du secondaire.

2.3.10 Turing

Au début des années 2000, deux systèmes similaires ont été conçus par une équipe interdisciplinaire, composée d'experts en didactique des mathématiques et en informatique. L'objectif de ceux-ci était de permettre à l'élève d'engager une conversation avec un agent tutoriel afin de l'aider à élaborer une démonstration. Le premier, AgentGeom (Cobo et al., 2007), qui a été rebaptisé par la suite geogebraTUTOR (Paneque et al., 2016), mais qui n'a aucun lien avec le GeoGebraTUTOR de notre projet, offre une interface web qui permet autant à l'enseignant qu'à l'élève d'interagir avec le logiciel. D'une part, l'enseignant est en mesure de composer de nouveaux problèmes de preuve, qui peuvent aussi faire intervenir des constructions sur une figure dynamique. Il peut ensuite les utiliser, ou utiliser ceux déjà implantés pour produire un itinéraire de problèmes qui seront proposés à l'élève. Ce dernier, de son côté, a accès à l'énoncé du problème, à une figure dynamique sur laquelle il peut construire de nouveaux éléments et à une fenêtre dans laquelle il peut proposer des énoncés pour construire sa démonstration. Le système analyse ses actions et les compare à celles qui sont valides et l'agent tutoriel peut lui faire des suggestions par l'intermédiaire d'une interface analogue à une fenêtre de clavardage. Le système Turing (El-Khoury et al., 2005; Richard et Fortuny, 2007) offre des fonctionnalités similaires, mais offre, en plus, une interface qui permet à l'expert de composer de nouveaux problèmes. Celui-ci est donc responsable de préparer l'énoncé du problème ainsi que la figure correspondante et de produire l'ensemble des inférences acceptables dans le cadre du problème. Dans ce contexte, le rôle de l'enseignant est limité au choix du problème à proposer à l'élève ainsi qu'à la mise au point des messages du tuteur pour des situations particulières. Pour l'élève, l'interface présente une figure dynamique, gérée par Cabri-Géomètre, une zone dans laquelle il peut construire ses énoncés, une fenêtre de clavardage qui simule une discussion avec le système tutoriel et une liste des actions réalisées. L'élève peut donc manipuler la figure et y ajouter des constructions pour en dégager des conjectures. Il peut alors les proposer au système en utilisant un langage déterminé qui se rapproche de la langue naturelle. Le système analyse l'énoncé produit et fournit une rétroaction immédiate, sous forme d'un message, si une erreur est détectée. Après une période prédéfinie d'inactivité de la part de

l'élève, le système lui fait des propositions de plus en plus précises dans le but de relancer son processus de résolution. Dans Turing, l'élève n'est pas contraint de suivre un cheminement en chaînage avant ou arrière, ce qui lui permet une exploration libre du problème. Une autre particularité de ce système est qu'il n'impose pas la production d'inférences complètes. Ainsi, l'élève peut proposer chacun des éléments, hypothèses, justification ou conclusion, dans un ordre quelconque, sans se soucier de la structure de l'inférence.

La conception de ces deux systèmes vise, en partie, à appliquer différentes théories didactiques, comme celles de Brousseau (1998), Lakatos (1984) et Duval (1995), dans le cadre de l'apprentissage des preuves en géométrie. Pour modéliser le domaine, les auteurs n'utilisent pas la géométrie formelle, car elle n'est pas assez flexible pour représenter, par exemple, des contradictions ou des arguments acceptables dans le cadre de l'enseignement secondaire. Les inférences acceptables sont donc codées manuellement, par l'expert, et sont reliées pour former un graphe orienté qui est appelé le graphe inférentiel. Un chemin dans ce graphe représente une solution possible au problème, donc le graphe contient l'ensemble des solutions acceptables. Pour modéliser l'apprenant, ces systèmes mémorisent l'ensemble des énoncés produits ainsi que la dernière action valide, car celle-ci est la clé qui permet au système tutoriel de choisir ses messages. Les deux systèmes utilisent une approche multi-agent dans le but d'obtenir une architecture plus flexible. Ceux-ci mettent en oeuvre différents agents pour gérer les interactions et les interfaces, mais ils ont tous les deux un agent tutoriel qui fonctionne de façon similaire. En effet, les messages affichés sont ceux liés à la dernière inférence sur laquelle l'élève a travaillé. Les messages produits sont donc construits par l'expert, et modifiés au besoin par l'enseignant, selon l'état de l'inférence, et forment une liste statique. Il n'est pas possible, pour l'élève, de demander de l'information supplémentaire par rapport aux messages du tuteur ou à la suite de l'identification d'une erreur. En ce qui concerne l'évaluation de Turing (El-Khoury et al., 2005; Richard et Fortuny, 2007), un prototype a été proposé à un nombre restreint d'élèves et une analyse qualitative, par rapport aux aspects didactiques, a été réalisée. Le projet actuel, QED-Tutrix, est en fait un descendant direct de Turing. En effet, QED-Tutrix met en oeuvre des bases théoriques presque identiques auxquelles des théories récentes issues de la didactique des mathématiques sont adjointes (voir section 3.2.1). Les graphes inférentiels, qui permettent de représenter des raisonnements non formels, la stratégie du tuteur ainsi que l'exploration libre du problème sont des aspects positifs du système dont notre travail s'inspire. Cependant, les graphes inférentiels deviennent difficilement compréhensibles pour des problèmes complexes. De plus, le système tutoriel ne permet pas de proposer d'autres stratégies à l'élève avant de lui donner la réponse. Enfin, le code source de Turing n'est pas accessible, ce qui nous oblige à développer un nouveau système original, à partir de zéro, et qui nous permet d'explorer une approche différente.

Dans cette section, nous avons présenté dix systèmes tutoriels qui traitent des preuves en géométrie. Pour chacun, nous avons d'abord décrit brièvement son interface et les interactions possibles pour ensuite présenter quelques détails techniques pertinents pour notre étude ainsi qu'une brève critique. Il en est ressorti que le système qui se rapproche le plus de nos objectifs est Turing, soit l'ancêtre du système actuel. En ce qui concerne les autres systèmes recensés, tous, à l'exception d'ANGLE, utilisent une forme de géométrie formelle, ce qui restreint le type de preuves qu'il est possible de traiter et ne permet pas d'accepter, par exemple, les contradictions. La modélisation dans ANGLE ne correspond cependant pas aux preuves typiques acceptées au secondaire. En ce qui concerne la modélisation de l'apprenant, seul le système Mentoniezsh conserve la chronologie des événements, car il l'utilise lors de la reconnaissance de plan. Malheureusement, ce dernier n'utilise pas ces informations pour proposer la prochaine étape à un élève bloqué, comme nous avons l'intention de le faire. En fait, la majorité des systèmes qui offrent des indices concernant la prochaine étape, le font en chaînage avant ou arrière. À notre grand étonnement, la moitié des systèmes présentés offre une exploration libre du problème. Par contre, la majorité n'offre qu'une rétroaction locale par rapport à l'inférence soumise par l'élève. En ce qui concerne l'évaluation, la méthodologie des prétests et post-tests pour évaluer l'effet sur la note de l'élève semble être typiquement américaine, alors que les Européens privilégient les analyses qualitatives. Une tendance semblable en ce qui concerne les bases théoriques est aussi visible. En effet, les Américains semblent privilégier les modèles cognitifs alors que les Européens utilisent habituellement des modèles didactiques. Ce recensement nous a permis de constater que même si plusieurs systèmes présentent des caractéristiques intéressantes, qui pourraient être implantées dans notre système, aucun n'offre la flexibilité et le type d'aide que nous recherchons. En fait, aucun système ne permet d'offrir différentes pistes de solution à un élève bloqué. Devant cette impasse, nous avons décidé d'étendre notre exploration à des systèmes dans des domaines connexes qui sont susceptibles de correspondre à nos objectifs et qui pourraient être adaptés à la géométrie. La section suivante présente cinq familles de systèmes dont les principes sont susceptibles de s'appliquer à notre recherche.

2.4 Autres systèmes tutoriels pertinents

N'ayant pas été en mesure de trouver un système tutoriel en géométrie, parmi ceux recensés à la section précédente, qui soit compatible avec les objectifs de notre recherche, nous pouvons présumer que notre projet est innovateur. Pour nous en assurer, et dans le but d'explorer différentes approches, nous avons analysé cinq familles de systèmes qui nous semblent pertinentes dans le cadre de notre recherche. Dans cette section, nous traitons autant de systèmes

que de modélisation ou de système de création. Cependant, pour chaque famille, nous avons choisi un système représentatif qui est analysé. Pour chacune, nous allons premièrement présenter les principaux systèmes qui en font partie et donner une brève description de chacun. Par la suite, lorsque nous le jugeons pertinent, une description générale du domaine et des principes sous-jacents s’appliquant aux systèmes associés est proposée. Par exemple, lors de la présentation de Deep Thought, nous en profiterons pour parler de l’exploration de données. Le système choisi est ensuite décortiqué en adoptant les mêmes critères qui nous ont permis d’analyser les systèmes tutoriels en géométrie. À la lumière des informations recueillies, nous terminons la présentation de chaque système par une brève critique.

2.4.1 Apprentissage de la physique

Andes (Conati et al., 2002; VanLehn et al., 2005, 2010) est un système tutoriel destiné à la résolution de problèmes de physique classique qui a un succès indéniable aux États-Unis. Son élaboration a débuté au milieu des années 1990 et le logiciel en est actuellement à sa troisième itération. L’objectif des concepteurs est d’offrir un système qui puisse guider l’élève dans la résolution d’un problème tout en lui laissant la liberté qu’offre un environnement papier/crayon. Les trois versions présentent une interface semblable qui permet à l’élève de dessiner des vecteurs sur une figure, de définir des variables et d’inscrire des équations dans le but de trouver la réponse au problème et de l’inscrire dans la case prévue. Une zone est aussi réservée pour l’affichage des messages du tuteur. La première version, Andes1 (Conati et al., 2002), se distingue par sa stratégie d’aide. En effet, dans le but de proposer l’étape la mieux adaptée au cheminement de l’élève, les auteurs ont mis en oeuvre une reconnaissance de plan basée sur des réseaux bayésiens. Bien que cette version ait eu un impact positif sur l’apprentissage des élèves, il s’est avéré que la stratégie d’aide ne correspondait pas à celle des experts. En effet, ces derniers ne tentent habituellement pas de deviner le plan de l’élève et procèdent plutôt en chaînage avant selon une procédure préétablie. Andes2 (VanLehn et al., 2005) utilise cette stratégie et contient des améliorations concernant la validation des équations fournies par l’élève. Dans ces deux premières versions, la création de variables se fait par l’intermédiaire de boîtes de dialogues, ce qui impose des contraintes, alors que dans Andes3 (VanLehn et al., 2010), elle peut être réalisée en langue naturelle. La modularité du système Andes a aussi permis de l’adapter au domaine des probabilités (Chi et VanLehn, 2007). Pour y arriver, les auteurs ont choisi des problèmes qui demandent un processus de résolution semblable à celui mis en oeuvre pour les problèmes de physique et ont mis à jour la base de connaissances en conséquence. Pour sa part, le système Pyrenees (VanLehn et al., 2004) utilise le moteur d’Andes, mais propose une stratégie de résolution en chaînage arrière. En effet, après avoir défini les variables, le système propose de produire une équation

qui lui permettra de trouver la réponse au problème. Les variables utilisées dans celle-ci deviennent des buts secondaires et l'on procède ainsi jusqu'à obtenir un système d'équations dans lequel il ne reste qu'à substituer les valeurs. Dans un autre registre, SE-Coach (Conati et VanLehn, 1999) est un logiciel qui utilise le moteur d'Andes pour proposer des exemples de résolution et inciter l'élève à expliquer les différentes étapes, à l'aide de menus, dans le but d'obtenir un apprentissage plus profond. Atlas-Andes (Rosé et al., 2001) propose une discussion, scénarisée, en langue naturelle pour atteindre le même objectif.

La véritable force d'Andes (Conati et al., 2002; VanLehn et al., 2005, 2010) se situe au niveau de son système d'aide. En effet, les auteurs ont réussi à produire des messages pertinents malgré la liberté qui est offerte à l'élève. Premièrement, en ce qui concerne les constructions et la définition des variables, celles-ci doivent être présentes dans la liste prédéterminée pour qu'elles soient considérées comme valides. Pour la validation des équations, le problème est plus complexe, car il est impossible de produire toutes les équations que l'élève pourrait écrire. Pour y arriver, les concepteurs utilisent une méthode de substitution de valeurs afin de vérifier leur validité et appliquent des transformations qui correspondent aux erreurs les plus courantes dans le but de poser un diagnostic. Andes propose trois types d'aide, soit une rétroaction immédiate, des informations concernant le problème identifié ainsi qu'une indication de l'étape suivante à réaliser. La rétroaction consiste à indiquer en rouge les erreurs de l'élève et à lui proposer un message dans le cas où le système jugerait qu'il s'agit d'une erreur d'inattention. Pour chaque élément en rouge, il est possible de demander des informations concernant la source de l'erreur. Andes offre trois niveaux de messages, de plus en plus précis, qui sont associés à chacun des diagnostics possibles, pour aider l'élève à corriger celle-ci. Ce dernier a aussi accès à trois niveaux de message lui permettant d'avoir des indications concernant la prochaine étape de résolution. Andes choisit alors la prochaine étape, en chaîne avant, qui n'a pas été produite par l'élève en se référant à l'arbre des solutions produit par un moteur de déduction appliqué à la base de connaissances. Cette dernière contient, en fait, les règles de production qui permettent de résoudre les problèmes de physique ainsi que les messages d'aide associés. La performance du système a été confirmée par des prétests et des post-tests ainsi que par une validation de la part des experts. Les bases théoriques sous-jacentes à l'élaboration du système n'ont toutefois pas été mentionnées explicitement. Andes est très intéressant, car il offre une semi-liberté (les variables doivent être définies au préalable) et que son système d'aide est très élaboré et efficace. De plus, son architecture est assez flexible pour qu'il puisse être adapté à d'autres domaines. Nous croyons cependant qu'il serait difficile de l'utiliser dans le domaine de la géométrie. En effet, le système est très efficace pour manipuler les équations dans lesquelles il est possible d'y substituer des valeurs, mais il n'est pas en mesure de manipuler des propriétés géométriques abstraites. De plus, sa

base de connaissances correspond à une représentation formelle du domaine ce qui n'est pas compatible avec nos objectifs. Enfin, sa politique d'aide à la prochaine étape, en chaînage avant, est valide pour la résolution de problèmes en physique, car une séquence d'actions précise doit être exécutée. Par contre, elle ne respecte pas l'état cognitif de l'apprenant dans le contexte de l'élaboration d'une preuve en géométrie.

2.4.2 Interactions en langue naturelle

En ce qui concerne les systèmes permettant un dialogue en langue naturelle, le plus important est AutoTutor (Graesser et al., 2004; Olney et al., 2010; D'Mello et Graesser, 2013), dont la conception a débuté au milieu des années 1990. Les auteurs se sont basés sur une analyse des conversations entre un tuteur et un élève ainsi que sur différentes théories pédagogiques et résultats de recherches pour concevoir leur système. Les mécanismes mis en oeuvre dans la version de base ont été utilisés dans de nombreux projets connexes (Nye et al., 2014). L'architecture a été appliquée, entre autres, à la physique (AutoTutor-3D), à la biologie (Guru Tutor), à l'apprentissage de la lecture (iSTART), etc. Ils ont même adapté AutoTutor au domaine du jeu sérieux dans le projet Operation ARIES, qui vise l'apprentissage de la pensée scientifique. Pour être en mesure d'utiliser le système sur une plateforme web, une version allégée a été développée (AutoTutor Lite (Wolfe et al., 2013)). Un groupe travaille aussi sur GnuTutor (Olney, 2009), une version ouverte, écrite en langage Java, basée sur une version préliminaire d'AutoTutor Lite. Le système WHY2 (Graesser et al., 2001), pour sa part, vise à aider l'élève dans la composition d'une dissertation scientifique en réponse à une question nécessitant une analyse approfondie. Il permet de détecter les erreurs dans la dissertation et il utilise Atlas ou AutoTutor pour gérer le dialogue qui amène l'élève à corriger chacune de ses erreurs pour finalement obtenir une réponse valide et complète. Pour les élèves du primaire, le système MyST (Ward et al., 2011; Becker et al., 2012) a été conçu afin de les aider à approfondir leurs connaissances scientifiques à l'aide d'un agent qui permet une conversation orale bidirectionnelle. Il est basé sur un scénario qui contient les sujets à traiter dans un ordre prédéterminé. Une fiche contenant les informations à présenter, les réponses possibles de l'élève ainsi que les réponses à fournir dans différentes situations est associée à chaque sujet. Il s'agit donc d'un système à base de règles, comme pour BEETLE II (Dzikovska et al., 2014), qui vise l'acquisition de connaissances concernant les circuits électriques et électroniques. Par contre, les interactions avec ce dernier se font par l'intermédiaire d'une fenêtre de clavardage. Le système STEVE (Elliott et al., 1999) offre, pour sa part, un environnement immersif pour l'acquisition de connaissances procédurales dans le but, par exemple, d'utiliser une pompe hydraulique. Il réagit aux erreurs de procédure selon des modèles prédéterminés. Enfin, le dernier système qui a attiré notre attention, TuTalk (Jordan et al., 2007), est une plateforme

qui permet de concevoir des dialogues à l'aide de machines à états hiérarchiques. En effet, il est possible qu'une transition soit liée à une machine secondaire et le système garde la trace de ces appels. Ceci permet donc de traiter de sujets secondaires, au besoin, et de reprendre le sujet principal à l'endroit de la bifurcation.

Comme la majorité des systèmes visant à offrir un dialogue en langue naturelle, l'architecture d'AutoTutor (Graesser et al., 2004; Olney et al., 2010; D'Mello et Graesser, 2013) contient des modules pour analyser et classer les entrées de l'utilisateur, construire des phrases valides et, ce qui nous intéresse dans le cadre de notre projet, gérer la conversation. Pour y arriver, il se base sur un scénario qui contient une question, demandant une réponse élaborée, une réponse idéale, une liste d'éléments requis, une liste de conceptions erronées ainsi que d'autres informations. Pour chaque élément contenu dans les deux listes, différentes questions à poser ou indices à fournir sont spécifiés. AutoTutor utilise une stratégie en cinq étapes pour traiter chaque scénario. Premièrement, il présente la question et analyse ensuite la réponse, habituellement incomplète, de l'élève. Une évaluation de la qualité de la réponse est fournie et s'engage ensuite un dialogue afin de permettre à l'élève de compléter sa réponse. La conversation se termine par un résumé du sujet traité et une validation de la compréhension de l'élève. Le dialogue permettant de compléter la réponse se base sur la liste des éléments requis et des conceptions erronées inscrites dans le scénario. Pour chaque élément requis manquant, un cycle, contenant quatre niveaux de messages dont le dernier dicte la réponse attendue, est enclenché, avant de passer à l'élément suivant. Ce dernier est déterminé en utilisant des critères dérivés de la théorie de la zone proximale de développement de Vygotsky (1978), mais il n'est pas imposé à l'élève au début de chaque cycle. Lorsqu'une conception erronée est détectée, le système affiche le message rattaché qui contient un correctif. AutoTutor met en oeuvre une machine à états finis, qui utilise l'historique de la conversation, pour générer le message suivant du tuteur. Le système, ainsi que les systèmes dérivés, ont été validés à l'aide de prétests et de post-tests, et, en général, un gain concernant l'apprentissage a été observé. Il a aussi réussi un test de Turing selon la perspective d'un observateur (Nye et al., 2014), donc il est en mesure d'agir partiellement de manière indissociable d'un humain dans son domaine d'expertise. La véritable force d'AutoTutor est associée à ses analyseurs syntaxiques élaborés qui sont en mesure d'analyser les scénarios ainsi que les textes soumis par les élèves pour en découvrir les similitudes, mais nous ne nous sommes pas attaqués à ce domaine. Même s'il est relativement simple, le module de gestion du dialogue paraît efficace. Par contre, AutoTutor, comme tous les autres systèmes semblables, n'est pas en mesure de traiter les problèmes admettant plusieurs solutions. En effet, les dialogues sont construits autour de la réponse idéale et permettent à l'élève d'en explorer tous ses éléments, sans pour autant lui laisser la liberté d'explorer une autre explication. Dans le cadre de notre projet,

la gestion d'un ensemble de solutions est un incontournable, donc, même si la performance de ces systèmes est indéniable, ils sont difficiles à adapter.

2.4.3 Preuves logiques

Notre recherche nous a aussi amené à analyser les principaux systèmes tutoriels destinés aux preuves mathématiques, autres que celles en géométrie. Au début des années 1990, le Carnegie-Mellon Proof Tutor (CPT) (Scheines et Sieg, 1994) a été conçu dans le but d'aider les élèves à produire des preuves en logique formelle. Il propose une interface en mode texte qui intègre une preuve en deux colonnes ainsi que l'arbre correspondant. Il permet de travailler en chaînage avant ou arrière. En ce qui concerne le tutorat, il offre premièrement une rétroaction immédiate pour indiquer les erreurs sur le pas courant. Il permet, de plus, à l'aide d'un moteur de déduction, d'aider l'élève à prouver un énoncé sélectionné. Le Logic-ITA (Lesta et Yacef, 2002), de son côté, traite le même type de preuve que le CPT, mais l'aide offerte se limite à une rétroaction par rapport à la validité locale du pas courant. Par contre, il offre des outils pour la gestion des problèmes et l'analyse de la performance des élèves. Pour sa part, le projet AProS (Sieg, 2007) vise à proposer une aide par rapport au pas suivant en se basant sur le travail réalisé par l'élève. Le défi est de produire un moteur de déduction formelle qui est en mesure d'accepter une preuve partielle, qui peut comporter des erreurs, et de la compléter. Ces trois systèmes, en plus d'être basés sur des déductions formelles, n'offrent pas une aide très élaborée. Par contre, le projet DIALOG (Benzmüller et al., 2003; Buckley et Benzmüller, 2005; Autexier et al., 2012) est prometteur, car il a pour objectif d'offrir de l'aide en langue naturelle pour la construction de preuves en théorie des ensembles. Une thèse décrit le module MENON (Tsovaltzi, 2010), qui permet de produire des dialogues socratiques qui sont issus de l'analyse d'un corpus de conversation. La stratégie utilisée se base sur une reconnaissance de motifs. Malheureusement, plusieurs modules du projet DIALOG sont incomplets. Il est donc impossible d'analyser la performance du système de façon globale. En revanche, le système le plus intéressant dans le domaine des preuves logiques, et que nous avons choisi d'analyser en détails, est Deep Thought (Barnes et Stamper, 2010; Stamper, 2010). Sa stratégie d'aide est basée sur des algorithmes d'exploration de données qui sont appliqués sur un corpus de preuves produites par des élèves. Enfin, du côté de l'argumentation, un domaine connexe, le système LARGO (Pinkwart et al., 2006, 2009; Scheuer et al., 2012) s'intéresse au domaine légal. Son interface qui permet de construire un graphe d'argumentation est son principal atout. Le système tutoriel utilise des règles ainsi que de l'appariement de graphes pour suggérer des corrections. En effet, dans un domaine mal défini, comme l'argumentation, il n'y a pas de réponses idéales, mais seulement des principes généraux à respecter, ce qui est trop permissif dans le cadre de la production de preuves.

Pour l'utilisateur, l'interface de Deep Thought (Barnes et Stamper, 2010; Stamper, 2010) offre la possibilité de construire une preuve logique, sous forme d'un graphe, en utilisant le chaînage avant ou arrière. Le système vérifie chacun des pas et retire automatiquement ceux qui sont invalides du graphe sans donner d'explications. Pour offrir une aide à la prochaine étape, Stamper (2010) a utilisé une approche par exploration de données basée sur le processus de décision de Markov, soit une stratégie innovatrice dans ce domaine. En effet, historiquement, l'exploration de données a été utilisée principalement pour extraire des informations et des statistiques concernant le comportement et les performances des élèves lors d'analyses à posteriori (Liao et al., 2012; Romero et Ventura, 2007; Peña-Ayala, 2014). On s'intéresse, par exemple, à extraire des statistiques pour prévoir la persévérance d'un élève selon différentes métriques. Le modèle produit est ensuite utilisé de façon prédictive, au moment prévu, de manière ponctuelle, en cours de trimestre. Un système qui suggère un cheminement basé sur ceux d'élèves ayant des profils similaires, identifiés notamment à l'aide de méthodes statistiques de partitionnement de données, est un autre exemple d'application. Par contre, il semble que Deep Thought soit un pionnier pour l'utilisation, en ligne, de l'exploration de données dans un système tutoriel. Sa stratégie consiste à modéliser une procédure de résolution sous forme d'un graphe dans lequel les noeuds représentent l'état du système et les arcs, les actions de l'élève. Il applique cette modélisation sur un corpus de solutions, combine les noeuds identiques et calcule différentes statistiques. Par exemple, il note la probabilité d'obtenir une preuve complète pour chacune des actions associées à un état. Le système est donc en mesure de proposer, selon le profil de l'élève, une série de quatre messages qui correspondent à l'action qui minimisera la probabilité de se retrouver dans une impasse, même si elle produit une preuve plus longue. Le système iList (Fossati et al., 2015), qui permet d'apprendre à manipuler des listes chaînées, s'est inspiré de cette modélisation et propose donc une stratégie d'aide similaire. La principale approche utilisée pour valider leur système consiste à calculer sa couverture par rapport au nombre de preuves utilisées pour construire le modèle. Par exemple, pour être en mesure de fournir de l'aide dans 75% des situations, il a été nécessaire de traiter environ 50 solutions. Ce logiciel est intéressant, car il permet de construire un système d'aide sans avoir recours à un expert. Le modèle employé peut aussi s'appliquer sur des preuves informelles, comme dans le cadre de notre projet, pourvu qu'il soit possible de décrire les différents états et actions possibles. Dans le cadre de Deep Thought, la contrainte de chaînage ainsi que le traitement d'inférences complètes permettent de limiter le nombre d'états possibles. Par contre, en éliminant ces contraintes, comme dans le projet actuel, le nombre d'états possibles se multiplie, ce qui diminue la probabilité d'avoir des états qui se répètent dans une preuve produite par différents élèves. Dans ce cas, la capacité du système à fournir de l'aide diminue de façon importante.

2.4.4 Modélisation basée sur les contraintes

Dans la majorité des systèmes tutoriels, on modélise l'élève à l'aide de différentes règles de production, bonnes ou erronées (traçage de modèle (MTT)), alors que dans SQL-Tutor (Mitrovic et al., 2002; Weerasinghe, 2013), on utilise des contraintes (modèle basé sur les contraintes (CBM)). En effet, pour ce dernier, l'idée n'est pas de construire tous les chemins possibles pour produire les solutions valides, mais bien de décrire les caractéristiques de la solution attendue. Par exemple, dans un problème concernant l'addition de fractions, une règle pourrait être : S'il faut calculer $a/b + c/d$ ET que la réponse est $(a + c)/n$ ALORS $b = d = n$. Un élève qui aurait inscrit $(b + d)$ au dénominateur ne respecterait pas cette contrainte, donc nous savons que sa réponse n'est pas valide et, de plus, nous avons identifié la source de son erreur. Dans un système MTT, il aurait été nécessaire, pour obtenir le même diagnostic, d'inclure une règle erronée permettant de modéliser l'erreur. Pour des problèmes simples, les deux approches semblent équivalentes. Cependant, dans des domaines pour lesquels les procédures sont mal définies, par exemple pour la construction de requêtes SQL, le MTT est difficilement applicable. Par contre, si les caractéristiques de la réponse attendue peuvent être décrites à l'aide de contraintes, alors la CBM est une piste à explorer. En ce qui concerne la validation des solutions, il faut savoir que le MTT peut refuser une réponse valide si elle n'utilise pas les règles modélisées. Au contraire, le CBM peut accepter des solutions invalides, si certaines contraintes ont été omises. Cependant, la principale différence entre les deux approches consiste en la capacité à offrir de l'aide en cours de résolution. En effet, dans le MTT, chaque action est analysée et une rétroaction immédiate est offerte en cas d'erreur. De plus, le système est en mesure de déterminer la règle suivante à appliquer et peut la proposer à l'élève. Dans le CBM, seule la réponse finale soumise au système est analysée et celui-ci retourne un ensemble de messages qui correspondent aux contraintes qui n'ont pas été respectées. Le système, n'ayant pas de modèle de résolution, n'est donc pas en mesure de proposer une action à un élève bloqué. De plus, la quantité de messages produits peut créer de la confusion par rapport aux correctifs à apporter à la solution. Même si le CBM semble plus limité en matière de stratégie d'aide, il n'en demeure pas moins que le modèle a été appliqué avec succès, validé par des prétests et post-tests, dans plusieurs systèmes tutoriels.

Dans le but de montrer les applications possibles du CBM, nous décrivons ici les principaux systèmes présentés dans un article relatant quinze années d'utilisation de la modélisation par contraintes (Mitrovic, 2012). Le premier système, qui fut développé à la fin des années 1990, est le SQL-Tutor (Mitrovic et al., 2002; Weerasinghe, 2013) et il permet d'apprendre à composer des requêtes SQL. Il présente une question qui correspond à une requête à composer, le schéma des tables à utiliser et un squelette de requête à remplir. L'élève doit donc compléter

les champs et peut ensuite soumettre sa réponse au système qui la valide en y appliquant les contraintes et qui affiche ensuite les messages associés aux contraintes violées. L'élève peut alors prendre connaissance de ses erreurs et les corriger avant de soumettre une nouvelle réponse. À la première version de SQL-Tutor, les auteurs ont ajouté un modèle de l'apprenant dans le but de proposer des itinéraires de problèmes. Pour y arriver, on indique, pour chaque contrainte, le niveau de maîtrise de l'élève. Il est alors possible de proposer des problèmes qui ne font intervenir qu'un nombre restreint de contraintes non maîtrisées. L'EER-Tutor permet, pour sa part, d'apprendre à produire des diagrammes entités/reliations. Il fonctionne sur le même principe que le SQL-Tutor, c'est-à-dire que la réponse de l'élève est validée à l'aide de contraintes. Cependant, lors d'un projet (Weerasinghe, 2013), des messages adaptatifs, qui ciblent les principales difficultés de l'apprenant, ont été intégrés. De plus, une version qui inclut des questions qui engagent l'élève dans une réflexion a aussi été produite. Le CBM a aussi été appliqué à la normalisation de schémas relationnels. Dans ce cas, le travail est divisé en un certain nombre d'étapes qui correspondent à un découpage de la procédure à appliquer. L'élève ne peut donc pas passer à l'étape suivante tant que la réponse de l'étape courante ne respecte pas les contraintes déterminées. Cette mise en oeuvre permet donc de montrer comment le CBM peut être appliqué à l'apprentissage de connaissances procédurales. En dehors du domaine des bases de données, J-LATTE est un système pour apprendre le Java qui permet de travailler à la fois sur la conception et sur le codage. Même si les applications de la modélisation à base de contraintes étaient limitées au départ, les progrès réalisés au cours des quinze dernières années la rendent de plus en plus intéressante. Par exemple, la modélisation de l'apprenant a permis à la fois de proposer des itinéraires de problèmes et de produire des messages mieux adaptés. La CBM est aussi un outil utile dans les situations où la MTT ne peut être utilisée. Cependant, dans le cadre de notre recherche, la procédure utilisée pour arriver à la conclusion est en fait la solution recherchée. Il est donc nécessaire d'énumérer toutes les procédures possibles pour obtenir l'ensemble des solutions. Dans ce cas, il est préférable d'utiliser MTT, ou une stratégie équivalente, car il permet de proposer une stratégie d'aide pour la prochaine étape.

2.4.5 Outils de création

Pour terminer, nous avons analysé différents outils d'aide à la création de systèmes tutoriels intelligents dans l'espoir d'en trouver un qui réponde à nos besoins et qui puisse nous éviter de repartir de zéro. Premièrement, dans le domaine de la gestion de curriculum, des outils tels que REDEEM (Ainsworth et al., 2003) et Eon (Murray, 2003) ont été créés. Le premier permet à un intervenant en enseignement d'organiser du matériel pédagogique existant pour l'utiliser au sein d'itinéraires d'apprentissage. Il permet aussi de recueillir des informations

sur la progression des élèves. Son interface est accessible au personnel ayant peu de connaissances en programmation. Eon, de son côté, offrait un ensemble complet d'outils pour la modélisation du domaine, sous forme d'ontologie, la création d'interfaces, la gestion des itinéraires d'apprentissage et l'aide, rudimentaire, à la résolution de problèmes. Ce système était prometteur, mais son élaboration s'est arrêtée, car le langage de programmation utilisé a été abandonné. Ces deux systèmes sont intéressants, mais ils n'offrent pas d'outils élaborés pour la gestion de l'aide en cours de résolution. ASPIRE (Mitrovic et al., 2009), par contre, est un outil qui permet de construire des problèmes avec des systèmes d'aide associés. Dans celui-ci, l'utilisateur décrit premièrement le domaine à l'aide d'une ontologie et modélise la structure des problèmes et des solutions. Il construit ensuite une interface et inscrit des problèmes ainsi que les solutions acceptables. Le système utilise enfin la modélisation à base de contraintes pour produire les règles définissant les réponses acceptables ainsi que les rétroactions associées. Ce système est très complet, mais est basé sur un modèle que nous n'utilisons pas dans le cadre de notre projet.

Ce sont les systèmes basés sur le traçage de modèle qui sont les plus susceptibles de répondre à nos attentes et ils sont aussi les plus répandus. Le système xPST (Gilbert et al., 2015), par exemple, permet de se connecter à l'interface d'un logiciel existant, à l'aide d'un module d'extension, et de la contrôler de façon bidirectionnelle. Il permet, par l'intermédiaire de scénarios, de définir des séquences d'actions et des rétroactions associées. Le langage de script est relativement facile à utiliser, mais est limité à des interactions simples. Le CMSDK (Blessing et al., 2009) utilise le même système de module que xPST pour se connecter aux interfaces. Par contre, il est basé sur une hiérarchie de types, dont chacun contient des attributs, et une liste de prédicats pour chaque type. Chaque prédicat est en fait une règle de production, qui permet de résoudre une partie d'un problème, auquel il est possible d'attacher un ensemble de messages et de rétroactions. CMSDK offre aussi un outil qui permet de simuler l'activation des différents prédicats en définissant les valeurs des attributs du type associé. L'utilisation de ces deux derniers systèmes se limite, cependant, aux logiciels pour lesquels un module d'extension existe, ce qui n'est actuellement pas le cas pour GeoGebra (Hohenwarter et Fuchs, 2004; Hohenwarter, 2013; International GeoGebra Institute, 2015).

L'outil qui nous a semblé le plus prometteur, et qui est aussi le plus connu, est CTAT (Koedinger et al., 2004; Alevi et al., 2009; Carnegie Mellon University, 2015) qui est basé, en majeure partie, sur la théorie cognitive ACT-R (Anderson et al., 1990; Anderson, 1996; Anderson et Schunn, 2000). Celui-ci est intéressant, car il offre des outils permettant à des personnes ayant des niveaux de compétences diverses en informatique de créer des systèmes tutoriels intelligents. Il est premièrement possible de construire un exemple interactif par une simple démonstration. Pour y arriver, l'utilisateur construit premièrement son interface avec

les objets offerts. Il inscrit ensuite les données initiales et démarre l'enregistreur. Ce dernier crée un graphe dont les noeuds représentent les différents états de l'interface et les arcs, les actions de l'utilisateur. L'utilisateur peut revenir à un état précédent et réaliser une série d'actions différentes pour démontrer une autre solution ou les erreurs possibles de l'élève. Il ne lui reste qu'à ajouter des messages d'aide et de rétroaction et son exemple interactif est prêt à être utilisé. Cette convivialité limite cependant la possibilité de réutiliser la procédure démontrée dans des problèmes similaires, car les actions de l'utilisateur sont codées en dur dans le graphe.

Pour les personnes ayant un minimum de connaissances informatiques et voulant créer un système tutoriel plus flexible, l'outil SimStudent (Li et al., 2015; MacLellan et al., 2015; Matsuda et al., 2015) disponible dans CTAT, est un choix intéressant. Pour l'utiliser, il faut tout d'abord définir un ensemble de fonctions permettant de représenter certaines caractéristiques du domaine. Par la suite, on lui propose quelques solutions pour le problème posé, comme dans le cas de l'exemple interactif. SimStudent tente enfin de résoudre le problème avec les règles qu'il a construites et demande l'approbation de l'utilisateur. S'il est dans l'erreur, l'utilisateur lui fournit la bonne réponse et il met à jour ses règles. Pour y arriver, il utilise principalement des algorithmes d'apprentissage inductif en logique de premier ordre (FOIL). Il permet donc, sans trop d'effort, d'obtenir un système plus générique, mais sa validité sur différents problèmes n'est pas assurée. Toutefois, pour ceux qui maîtrisent le Jess, il est possible d'éditer les règles de production directement dans ce langage, ce qui permet un maximum de souplesse. Enfin, ASTUS (Paquette et al., 2010, 2015; Paquette, 2013) ajoute à cette architecture une hiérarchie de plan abstrait, ce qui lui permet d'interpréter plus facilement les actions de l'élève et d'offrir de l'aide mieux adaptée à la situation. Le principal avantage de ces outils, démontré par des études, concerne leur convivialité qui permet de créer des systèmes tutoriels dans un temps raisonnable. De plus, ils sont accessibles pour des personnes ayant des niveaux de compétence divers. Par contre, dans le cas d'un tuteur de démonstrations, il est impensable de produire toutes les solutions possibles. Ce type de système oblige aussi l'élève à travailler généralement en chaînage avant, ce qui ne lui permet pas d'explorer le problème. Ces systèmes ne sont donc pas compatibles avec nos objectifs.

Étant donné que nous n'avons pas trouvé de système tutoriel en géométrie qui réponde à nos objectifs, nous avons étendu notre recherche. Nous avons donc présenté, dans cette section, nos trouvailles les plus prometteuses dans le cadre de notre projet. Pour chacune des cinq approches analysées, nous avons choisi un système représentatif que nous avons inclus dans le tableau synthèse (tableau 2.1). Parmi les systèmes analysés, c'est Andes (Conati et al., 2002; VanLehn et al., 2005, 2010) qui offre le système d'aide le plus élaboré et complet. En effet, il offre une rétroaction immédiate en plus de fournir différents niveaux d'aide pour indiquer l'erreur ainsi que la prochaine étape. Cependant, il traite exclusivement des équations algè-

briques, donc il ne peut être utilisé dans d'autres domaines. En ce qui concerne les dialogues en langue naturelle, c'est AutoTutor (Graesser et al., 2004; Olney et al., 2010; D'Mello et Graesser, 2013) qui se démarque. C'est son moteur d'analyse qui est sa principale force, mais nous ne comptons pas permettre une entrée en langue naturelle pour l'instant. Par contre, pour ce qui est de sa gestion des dialogues, elle est semblable à celle des autres systèmes tutoriels intelligents. Nous avons aussi analysé deux autres stratégies pour modéliser le domaine, soit la modélisation à base de contraintes et l'approche par exploration de données. Ces deux approches ne peuvent être utilisées dans notre système, car le nombre de solutions distinctes, ainsi que les stratégies de résolution possibles, est trop grand ce qui risque de réduire dramatiquement l'efficacité de celles-ci. Enfin, les outils de création ne sont pas assez flexibles pour nous permettre de construire le système d'aide désiré.

2.5 Principales caractéristiques des systèmes analysés

Pour présenter une synthèse des quinze systèmes étudiés, nous avons construit le tableau 2.1 qui contient les principales caractéristiques de chacun. Ces dernières ont été déterminées afin de faire ressortir les particularités des différents systèmes par rapport aux objectifs du projet actuel. Le tableau comporte trois sections principales séparées par des lignes doubles verticales. Dans la première section, qui ne contient qu'une seule colonne, on retrouve le nom du système analysé qui est représentatif de sa famille. Par conséquent, les données, inscrites sur la ligne correspondante, sont spécifiques au système nommé, mais sont, en général, partagées par les autres systèmes de la même famille. Nous avons placé, dans le haut du tableau, les dix systèmes tutoriels traitant les démonstrations géométriques en ordre alphabétique. Sous la double ligne de séparation, nous avons inscrit, toujours en ordre alphabétique, cinq autres systèmes qui présentaient un intérêt particulier dans le cadre de notre projet. Nous avons donc appliqué, pour ces derniers, le même type d'analyse que pour les systèmes tutoriels en géométrie. Étant donné que nos critères d'analyse sont assez génériques, l'analyse a été possible sans problèmes majeurs.

Au centre du tableau, nous retrouvons quatre types de modèles qui ont été utilisés pour notre analyse, soit le modèle du domaine, de l'apprenant, de stratégie d'aide et de rétroaction. Cette division est apparue naturellement et elle est en fait inspirée d'un découpage similaire proposé par Beck et al. (1996). En effet, en adaptant les définitions, nous avons une correspondance directe pour le modèle de l'apprenant (Student Model), du domaine (Domain Knowledge) et de rétroaction (Communication Model). En ce qui concerne l'«Expert Model», il peut être associé à notre stratégie d'aide s'il permet de proposer des pistes de solution à l'élève à l'aide des stratégies de l'expert. Nous n'avons pas inclus le modèle pédagogique dans notre analyse,

Tableau 2.1 Caractéristiques des principaux systèmes tutoriels recensés

Système	Modélisation du domaine	Modélisation de l'apprenant	Stratégie d'aide	Rétroactions			Exploration	Bases théoriques	Évaluation
				Identification de l'erreur	Explication de l'erreur	Identification de la prochaine étape			
Advanced Geometry Tutor	GRAMY	dernière étape	déterministe	message	non	4 niveaux hiérarchiques	non	non explicite	aucune
ANGLE	graphe DC	éléments construits	chaînage avant	message	2 niveaux hiérarchiques	4 niveaux hiérarchiques	libre	DC	pré/post test
Baghera	géométrie formelle	conceptions	aucune	annotations	non	non	libre	cKc	experts
Cabri-Euclide	LDL	inférences	aucune	message	2 niveaux hiérarchiques	non	libre	didactiques	qualitative
DEFI	questions, inférences	énoncés démontrés	aucune	message	non	non	chaînage arrière, avant	non explicite	qualitative
Géométrix	géométrie formelle	inférences	aucune	message	non	non	chaînage avant	non explicite	aucune
Geometry Explanation Tutor	règles de production	mémoire de travail	règle optimale	couleur	erreurs modélisées	7 niveaux hiérarchiques	chaînage avant	ACT-R	pré/post test
Geometry Tutor	règles de production	mémoire de travail	règle optimale	couleur	erreurs modélisées	5 niveaux hiérarchiques	chaînage avant/arrière	ACT-R	pré/post test biométrie
Mentoniez	HDL	chronologie d'inférences/reconnaissance de plan	aucune	message	non	non	libre	non explicite	qualitative
Turing	graphe inférentiel	actions valides	dernière action valide	message	non	6 niveaux hiérarchiques	libre	didactiques	qualitative
Andes	graphe des solutions	étapes complétées	chaînage avant	couleur, message	3 niveaux hiérarchiques	3 niveaux hiérarchiques	libre avec contraintes	non explicite	pré/post test experts
AutoTutor	scénario	historique de la conversation	Vygotsky	message	1 niveau	4 niveaux hiérarchiques	libre	analyses, pédagogiques	pré/post test test de Turing
CTAT	règles de production	mémoire de travail	règle optimale	message	erreurs modélisées	variable	chaînage avant	ACT-R	analyse de convivialité
Deep Thought	graphe états-actions	état	processus de décision de Markov	action annulée	non	4 niveaux hiérarchiques	chaînage avant/arrière	non explicite	validation informatique
SQL-Tutor	contraintes	réponse soumise	aucune	message	1 niveau	non	libre	CBM	pré/post test

car il correspond à l'étape de la gestion des problèmes qui déborde du cadre de notre projet. Dans notre tableau, le modèle du domaine contient des informations concernant les moyens utilisés pour produire, manuellement ou de façon automatique, stocker et gérer l'ensemble des solutions d'un problème de démonstration. Il s'agit, par conséquent, d'informations statiques, même si certains logiciels les gèrent dynamiquement, car il est possible, en théorie, de générer l'arbre complet des solutions au préalable. En ce qui concerne le modèle de l'apprenant, nous nous intéressons aux structures dynamiques qui permettent de rendre compte de la progression de l'élève lors de la résolution du problème. Nous souhaitons, par exemple, savoir si la chronologie des événements est préservée pour ensuite être utilisée dans le module de stratégie d'aide. Ce dernier permet de choisir la prochaine étape que l'on peut suggérer à un élève qui est bloqué dans son processus de résolution. Dans le tableau, nous indiquons de quelle façon, ou selon quels critères, la prochaine étape est choisie. Il faut cependant savoir que le fait de déterminer la prochaine étape est indépendant de la rétroaction qui permet de l'indiquer à l'élève. Par exemple, un système peut s'appuyer sur une reconnaissance de plan pour analyser les erreurs de l'élève et ne pas offrir de messages d'aide pour la prochaine étape. Enfin, les rétroactions, soit tout ce qui est visible à l'interface, ont été découpées en trois principales catégories qui correspondent à celles décrites par VanLehn (2006). Pour l'identification de l'erreur, nous souhaitons savoir si le système fournit une indication minimale et la façon dont il s'y prend : par message, par changement de couleur, etc. Dans le cas où une erreur serait trouvée, nous indiquons, dans la colonne de l'explication de l'erreur, si le système est en mesure de fournir une explication, que ce soit de façon automatique ou à la demande de l'élève. Dans la dernière colonne de cette section, soit l'identification de la prochaine étape, nous nous intéressons seulement aux mécanismes et messages utilisés pour l'indiquer à l'élève, sans nous soucier du choix de celle-ci qui a déjà été traité dans la colonne concernant la stratégie d'aide. Dans tous les cas de rétroaction, si le système propose une série de messages, nous indiquons le nombre moyen de messages accessibles et leur organisation.

Dans la dernière section du tableau, à l'extrême droite, nous avons inclus des critères qui correspondent plus particulièrement aux objectifs que nous nous sommes fixés et aux méthodes utilisées dans ce projet. Ces colonnes ont pour objectif de démontrer clairement le caractère distinct de notre approche. Un de nos objectifs est de permettre une exploration libre du problème et être en mesure de fournir une aide adaptée à la progression de l'élève. Nous souhaitons savoir si les différents systèmes permettent ce genre d'exploration et quelles en sont les limites. Nous indiquons donc si le système force l'élève à travailler selon un chaînage quelconque ou s'il permet de proposer des conjectures à sa guise. Nous avons aussi pour objectif de proposer différents chemins de solution dans le cas d'un blocage dans le processus de résolution. Cette capacité est analysée dans la colonne concernant la stratégie d'aide

au centre du tableau. Nous voulons aussi construire un système ayant des bases didactiques solides. Nous indiquons donc les bases théoriques sur lesquelles s'appuie la conception de chacun des systèmes analysés. Pour terminer, à cause de notre processus de développement en plusieurs phases, nous avons décidé de valider notre système à l'aide d'analyses qualitatives et nous voulons savoir si cette approche est courante. Nous avons, par conséquent, indiqué le type de processus de validation qui a été utilisé pour chaque système étudié. En résumé, le tableau synthèse que nous avons construit nous a permis de comparer systématiquement les caractéristiques des différents systèmes en les découpant selon quatre modules distincts. De plus, les trois colonnes de droite contiennent des critères spécifiques à notre projet afin de faire ressortir les facettes uniques de notre approche. Le tableau 2.1 a ainsi mis en évidence les limites des systèmes étudiés par rapport aux objectifs de notre projet et a servi à confirmer l'aspect novateur de ce dernier.

2.6 Conclusion

Ce chapitre visait à présenter une synthèse de notre recherche bibliographique concernant les systèmes tutoriels intelligents et plus particulièrement ceux en géométrie. Nous avons premièrement fait un survol de différentes ressources pour l'apprentissage des mathématiques. Il s'agit principalement de ressources qui sont utiles pour un élève autonome, car ils n'offrent pas d'aide interactive intelligente, comme, par exemple, des vidéos, des micromondes, des systèmes de preuve automatiques, etc. Cependant, les micromondes sont intéressants dans le cadre de notre projet, car ils permettent de visualiser une figure dynamique et peuvent servir de base à un système tutoriel. En ce qui concerne les modèles de l'apprenant associés à la gestion du curriculum, ils s'appliquent en fait aux itinéraires de problèmes qui seront traités dans une phase ultérieure du projet.

Dans le reste du chapitre, nous avons présenté dix systèmes tutoriels en géométrie ainsi que cinq approches prometteuses. Un tableau synthèse (tableau 2.1) a permis de regrouper leurs caractéristiques essentielles, relatives à nos objectifs de recherche, afin de démontrer l'aspect innovateur de notre approche. Un des objectifs principaux concernait la possibilité de permettre une exploration libre du problème. Cette possibilité est offerte par près de la moitié des systèmes analysés. Cependant, ces systèmes n'offrent aucune aide à la prochaine étape ou proposent de travailler en chaînage, ce qui ne convient pas toujours à l'état cognitif de l'élève. Le seul système qui se démarque est AutoTutor (Graesser et al., 2004; Olney et al., 2010; D'Mello et Graesser, 2013) qui applique la théorie de Vygotsky pour choisir le prochain sujet à traiter. Malheureusement, il ne modélise que des problèmes ayant une réponse unique, même si elle peut être complexe et élaborée. En ce qui concerne Turing (El-Khoury et al.,

2005; Richard et Fortuny, 2007), l'ancêtre direct du projet actuel, sa stratégie d'aide est issue d'une programmation ad hoc dont nous n'avons pas d'informations, donc il ne nous est pas possible de la réutiliser dans notre projet.

Le second objectif important que nous nous sommes fixé consiste à proposer différentes pistes de solution, à un élève bloqué en cours de résolution, comme peut le faire un enseignant. Notre recherche nous a permis de découvrir qu'aucun système recensé n'offre cette possibilité. En effet, tous les systèmes proposant une aide à la prochaine étape déterminent l'action optimale et l'indiquent à l'élève en affichant une séquence de messages de plus en plus précis. Le dernier message consiste habituellement à décrire explicitement l'action que doit réaliser l'élève pour poursuivre la résolution, sans lui proposer, au préalable, une autre stratégie. Des systèmes comme AutoTutor (Graesser et al., 2004; Olney et al., 2010; D'Mello et Graesser, 2013), Géométrie (Gressier, 2015) ou CTAT (Koedinger et al., 2004; Aleven et al., 2009; Carnegie Mellon University, 2015) permettent d'ajuster le nombre, l'ordre et le contenu des messages à afficher dans différentes situations, ce qui pourrait permettre de proposer diverses pistes de solution. Cependant, pour choisir une autre stratégie à proposer, il est nécessaire d'avoir accès au travail réalisé par l'élève, mais cette information n'est pas disponible au moment de composer les messages. Nous n'avons donc aucune référence, dans le domaine des systèmes tutoriels intelligents, qui présente un modèle de gestion dynamique des messages comme nous avons l'intention de l'implanter.

Même si aucun système recensé ne peut être utilisé ou facilement adapté pour la réalisation de ce projet, il n'en demeure pas moins que certains offrent des caractéristiques intéressantes qui ont influencé notre travail. Par exemple, nous avons déjà mentionné l'intégration d'un micromonde dans notre système. Nous pensons aussi au système d'aide d'Andes (Conati et al., 2002; VanLehn et al., 2005, 2010), qui est une référence dans le domaine, dont nous nous sommes, comme plusieurs, inspiré en ce qui concerne le type et la structure des rétroactions offertes par le système. L'utilisation de la reconnaissance de plan dans Mentoniez (Py, 1994, 1996, 2001) a aussi eu un impact sur la stratégie d'aide que nous avons implantée. D'autres approches, comme l'exploration de données, même si elles n'ont pas été utilisées dans le cadre de notre projet, pourrait être utile lors des prochaines phases de développement. Le système que nous proposons offre donc des caractéristiques innovatrices et d'autres inspirées des leaders dans le domaine. Le chapitre suivant propose une synthèse du système construit et nous y présentons aussi les fondements didactiques sur lesquels son développement s'appuie. La description technique de chaque composant est présentée dans les chapitres subséquents.

CHAPITRE 3 SURVOL DU SYSTÈME ET FONDEMENTS DIDACTIQUES

*«Les tentatives de création de machines pensantes
nous seront d'une grande aide pour découvrir
comment nous pensons nous-mêmes.»*

— Alan Turing

3.1 Introduction

Étant donné que l'apprentissage de la démonstration peut être ardu pour les élèves (Tall, 1991) et que la durée des interventions individualisées des enseignants tend à diminuer, nous avons eu l'idée de proposer un système informatique afin d'épauler ces derniers. Nous avons alors recherché un logiciel qui traite les preuves telles qu'elles sont acceptées au secondaire, qui permet une exploration libre du problème et qui offre un soutien semblable à celui apporté par l'enseignant. Dans le chapitre précédent, nous avons notamment présenté divers systèmes tutoriels intelligents pour l'apprentissage de la démonstration, dont Geometry Tutor (Anderson et al., 1985), Advanced Geometry Tutor (Matsuda et VanLehn, 2005; Matsuda, 2004), Géométrie (Gressier, 2015), Cabri-DEFI (Gras, 1988; Almouloud, 1992; Luengo et Balacheff, 1998), etc. Toutefois, la majorité de ceux-ci exigent des démonstrations formelles et forcent l'élève à travailler dans un ordre déterminé en ignorant ainsi la phase d'exploration du problème. De plus, malgré leur succès et les impacts qu'ils ont sur l'apprentissage, ces systèmes offrent un soutien qui est différent de l'aide apportée par l'enseignant (Matsuda et VanLehn, 2003). Deux logiciels mettent cependant en oeuvre une approche plus naturelle face à la démonstration en permettant une exploration libre du problème, soit Mentoniez (Py, 1994, 1996, 2001) et ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991). Par contre, l'aide proposée par ceux-ci est limitée ou elle ne respecte pas nécessairement l'état cognitif de l'élève. Enfin, aucun des systèmes recensés, toutes catégories confondues, n'offre différentes pistes de solution lors d'un blocage du processus de résolution. En réaction à ces observations, nous avons conçu QED-Tutrix (QEDX), un système tutoriel intelligent basé sur les interactions réelles entre les enseignants et les élèves.

QEDX est ainsi en mesure de guider l'élève, à la manière d'un enseignant, lors de l'apprentissage des démonstrations en géométrie euclidienne. Il est issu d'une collaboration entre des chercheurs en didactique des mathématiques et en informatique. Il se veut un outil interactif qui guide l'élève sans lui imposer de contraintes dans l'ordre de ses actions. Ce dernier

peut donc naviguer librement dans QEDX et inscrire les pas de sa démonstration selon le raisonnement qui lui est propre et s'inspirer, au besoin, des conseils du tuteur pour ultimement obtenir une démonstration complète qui est reconnue par le système. Si ce dernier juge qu'il n'est pas en mesure d'aider adéquatement l'élève, il peut le référer à son enseignant. Notre but étant de proposer un système tutoriel intelligent qui peut interagir avec l'élève à la manière d'un enseignant, tout en rendant plus ergonomique l'apprentissage de la démonstration, nous avons opté pour une approche de conception dans l'usage (Rabardel, 1995), soit une alternance entre des analyses didactiques et des expérimentations, à laquelle nous avons greffé une validation par des experts.

Dans ce chapitre, nous présentons une synthèse du système QEDX axée sur son architecture informatique globale, enrichie de quelques notions didactiques qui ont influencé son développement. Nous nous intéressons d'abord à l'ergonomie du logiciel, tant au niveau de l'interface utilisateur qu'au niveau des notions didactiques qui sous-tendent sa conception. Nous montrons, entre autres, comment l'interface actuelle permet à l'élève de mettre en oeuvre diverses démarches associées au processus de résolution. Dans la section suivante, nous exposons l'architecture logicielle générale du système en quatre couches (figure 3.7), soit le coeur de notre travail et notre contribution au domaine des systèmes tutoriels intelligents. Pour ce faire, nous donnons une description sommaire de chaque couche en plus de décrire leurs interactions. Par la suite, nous présentons brièvement la méthodologie employée dans le cadre du projet. Nous décrivons notamment les deux phases expérimentales qui ont permis de développer la version actuelle de QEDX dans le cadre de l'approche de la conception dans l'usage. Nous résumons aussi les méthodes d'analyse utilisées, essentiellement par les didacticiens, lors des différents tests et expérimentations. Le chapitre se termine par une conclusion générale.

3.2 Aspects ergonomiques de QEDX

Notre recensement des systèmes tutoriels intelligents existants nous a fait découvrir qu'une minorité de systèmes sont construits autour de bases théoriques issues de la didactique des mathématiques et qu'aucun n'offre une aide comparable aux interventions des enseignants en classe. Nous croyons pourtant que les deux vont de pair, car les analyses didactiques peuvent nous permettre d'interpréter les interactions enseignants-élèves et c'est pour cette raison que nous avons choisi d'appuyer la conception de notre système sur des bases didactiques. Ainsi, dans cette section, nous présentons les aspects ergonomiques de la version actuelle de QEDX, soit le système tutoriel intelligent élaboré dans le cadre de nos travaux qui est le résultat du second cycle de sa conception (figure 3.2). Dans un premier temps, nous décrivons les principales théories didactiques qui agissent à titre d'analyse à priori dans la conception dans

l'usage de QEDX, soit la théorie des situations didactiques de Brousseau (1998), la genèse instrumentale de Rabardel (1995) et les espaces de travail géométriques de Kuzniak (2006). Nous présentons ensuite l'interface de notre logiciel, soit la fenêtre principale, qui contient, entre autres, une zone de clavardage avec l'agent-tuteur, et les quatre onglets accessibles en tout temps lors de l'élaboration de la démonstration. Nous nous intéressons notamment aux aspects didactiques de l'interface en plus de décrire les fonctionnalités offertes par celle-ci.

3.2.1 Fondements didactiques

Les objectifs que nous nous sommes fixé sont très ambitieux, car ils impliquent, entre autres, l'analyse des interactions entre des enseignants et des élèves réels afin de produire le système proposé. Or, nous savons que l'efficacité des interventions pédagogiques dépend non seulement du discours complexe de l'enseignant, mais aussi de ses réactions non verbales, du contexte, de ses gestes, etc. Il serait donc utopique de penser qu'une analyse superficielle puisse nous permettre de concevoir le système souhaité. C'est pour cette raison qu'à la base, QEDX s'ancre dans la **Théorie des Situations Didactiques** (TSD) proposée par Brousseau (1998). Celui-ci présente une situation didactique comme une interaction élève-milieu qui est régulée par l'enseignant (flèches 1 et 2 de la figure 3.1). Dans le but d'intégrer un tuteur virtuel au système, le milieu didactique, soit une extension de la notion de milieu de Brousseau, a été proposé. Dans le cadre de la TSD, à tout état d'équilibre du système élève-milieu correspond un ensemble de connaissances et de compétences. Le fait de proposer à l'élève un problème adapté à l'objectif d'apprentissage visé crée un déséquilibre dans le système qui aura pour effet de pousser celui-ci vers un nouvel état d'équilibre lié à la nouvelle connaissance (partiellement) développée par la résolution du problème proposé. Le rôle de l'enseignant consiste donc à offrir des problèmes adaptés et à mettre en place un milieu qui permet au système de tendre vers le nouvel état d'équilibre recherché.

La TSD permet de bien modéliser les interactions didactiques dans une classe ordinaire, mais lorsque l'on intègre un système tutoriel intelligent au milieu, les relations entre l'enseignant et le système élève-milieu se voient modifiées. En effet, le milieu didactique contient un logiciel, dans notre cas QEDX, qui intègre à la fois un milieu virtuel de résolution de problèmes ainsi qu'un agent-tuteur qui joue le rôle d'un enseignant virtuel. La figure 3.1, traduite de Richard et al. (2011), montre cette nouvelle organisation du milieu didactique. L'ajout des relations 6 et 7, qui sont en fait une transposition des relations 1 et 2, sans pour autant éradiquer ces dernières, montre que l'agent-tuteur influence le système élève-milieu virtuel en lui fournissant une aide modélisée selon un contrat didactique réel observé et complémentaire aux interventions de l'enseignant. L'aide proposée dans QEDX est actuellement sous forme

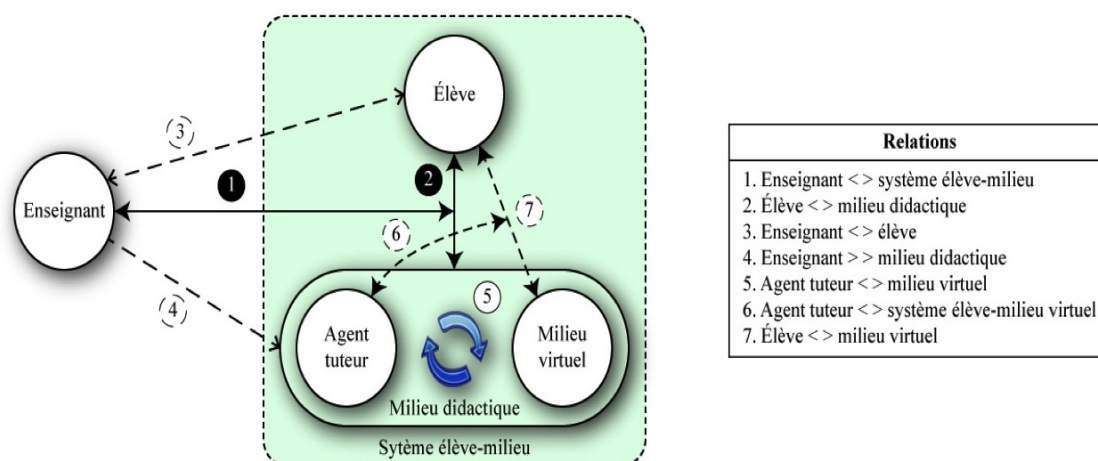


Figure 3.1 Carte des interactions didactiques (traduit de Richard et al., 2011)

discursive, mais nous visons éventuellement à proposer des sous-problèmes, qui pourraient remplacer certains messages discursifs, ainsi que des itinéraires d'apprentissage personnalisés.

La construction d'un système tutoriel intelligent comme QEDX, qui est en mesure de prendre en compte la relation 1 de la figure 3.1 pour produire une relation 6 cohérente, est très ambitieuse et une approche centrée sur le fonctionnement de l'outil produit rarement les résultats escomptés. La théorie instrumentale de Rabardel (1995) distingue l'outil technique de l'instrument au sens où ce dernier n'est pas seulement un objet, mais aussi un ensemble de schèmes d'utilisation. Pour créer un instrument, il faut donc utiliser une approche anthropocentrique, au sens de Rabardel (1995), c'est-à-dire qu'elle place l'utilisateur au centre du processus. Comme nous voulons que QEDX devienne un instrument d'apprentissage, nous devons non seulement créer le logiciel en fonction des apprentissages visés, mais aussi l'adapter en fonction de l'utilisation constatée afin qu'il soit en symbiose avec les objectifs de l'enseignant. L'élaboration de QEDX s'est donc faite selon le paradigme de conception dans l'usage (Rabardel, 1995), illustré à la figure 3.2.

La conception dans l'usage consiste à faire des allers-retours entre des analyses théoriques et des expérimentations pour construire le système par réorganisations successives. Chaque cycle de développement de QEDX, qui peut être perçu comme une micro-recherche, est divisé en quatre phases qui s'apparentent à celles employées dans le cadre de l'ingénierie didactique. Nous adaptons ainsi «la phase 1 des analyses préalables, la phase 2 de la conception et de l'analyse à priori des situations didactiques de l'ingénierie, la phase 3 de l'expérimentation et enfin la phase 4 de l'analyse à posteriori et de l'évaluation» (Artigue, 1996) à la conception d'un système tutoriel intelligent. Toutefois, un retour sur les phases antérieures est indispensable afin de faire évoluer notre système en accord avec les nouvelles informations relevées.

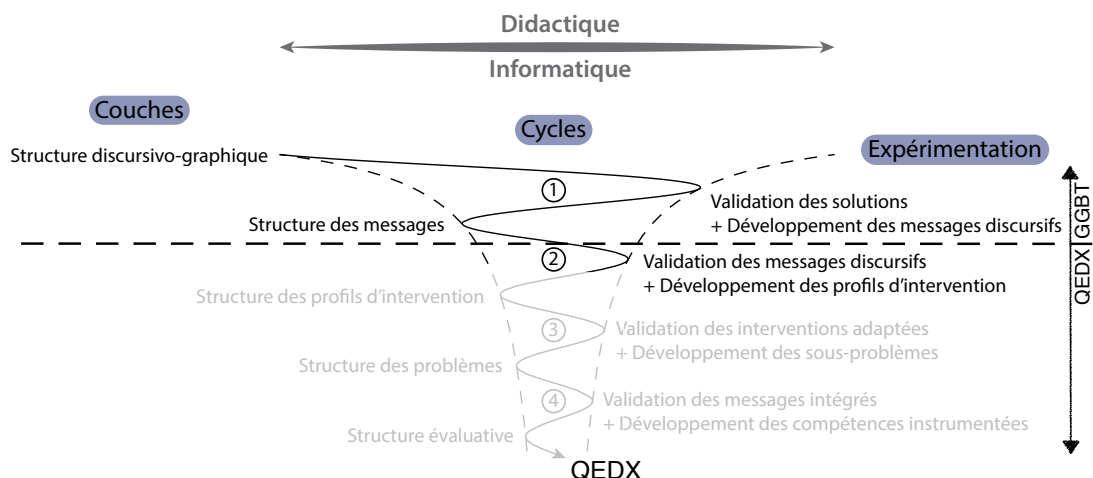


Figure 3.2 Cycles de conception de QEDX (adapté de Richard et al., 2011)

Or, cette stratégie s'inspire de la **Théorie Ancrée** de Glaser et Strauss (1967) qui admet que le chercheur est influencé notamment par ses expériences et que celles-ci viendront transformer le processus même de la recherche. Pour que ses analyses représentent le plus fidèlement possible la réalité observée, il doit alors s'imprégner consciemment de ces influences sans tenter de les supprimer. Ainsi, chaque phase expérimentale, en plus de servir à valider le travail réalisé, est mise à profit afin de recueillir des données pour la conception de la couche suivante. Il est dès lors possible d'intégrer des corrections de bogues, mais aussi de prendre en compte les schèmes d'utilisation constatés à la suite de chaque expérimentation. Le système issu de chaque cycle n'est donc pas un produit fini, mais un instrument en constante évolution.

Dans le cadre de notre projet (figure 3.2), l'objectif du premier cycle de conception consistait à produire un système, nommé à l'époque GeoGebraTUTOR (GGBT), qui est en mesure d'encoder et d'inventorier toutes les solutions acceptables pour différents problèmes de démonstration (structure discursivo-graphique). Il s'agissait alors, au terme d'une analyse théorique, de choisir et d'implanter un certain nombre de problèmes qui proposent à la fois une diversité didactique, mais aussi des défis informatiques. GGBT permettait ainsi de résoudre quatre problèmes à son interface, mais n'offrait aucune structure d'aide. En effet, notre intention était d'étudier des interactions authentiques entre l'enseignant, l'élève et le système, donc une implantation ad hoc d'interventions artificielles aurait affecté nos observations. Le premier cycle se conclut par une expérimentation qui permet, d'une part, de valider les solutions implantées ainsi que l'interface proposée et, d'autre part, de recueillir des informations relatives aux messages discursifs employés par les enseignants pour aider les élèves.

Les données issues de la première expérimentation nous ont permis de corriger les solutions aux problèmes et de réaménager l'interface de GGBT. De plus, elles ont servi d'ancrage au

second cycle de développement où l'on a cherché à proposer une aide mettant en oeuvre une structure de messages discursifs inspirée des interventions des enseignants. Cependant, étant donné la complexité de ces dernières, nous avons restreint notre analyse à un sous-ensemble qui nous semblait pertinent. Un modèle de rétroaction d'un enseignant typique a alors été extrait et implanté dans la seconde itération de notre système, soit QEDX. Nous l'avons renommé afin de dénoter son indépendance par rapport à GeoGebra (Hohenwarter, 2013), un logiciel de géométrie dynamique, car même si nous employons actuellement ce dernier pour gérer la figure dynamique, nous envisageons de le remplacer. QEDX a ensuite fait l'objet d'une expérimentation en classe (voir section 3.4) permettant ainsi de clore le second cycle. Nous avons donc eu l'occasion de valider les messages discursifs proposés par notre système et de colliger des données concernant les différents profils d'intervention des enseignants. Ces dernières seront, par la suite, réinvesties dans les phases ultérieures du projet, comme indiqué, à titre informatif, sur la figure 3.2.

QEDX, qui est le sujet de cette thèse, n'est donc pas un produit final. En fait, il est en constante évolution, car il y a adaptation mutuelle entre les utilisateurs et notre système tutoriel intelligent. Par conséquent, nous ne prétendons pas que ce dernier aura un impact mesurable sur la réussite des élèves dans son état actuel et c'est pour cette raison que nous avons évité d'évaluer sa performance à l'aide de prétests et de post-tests. Toutefois, la conception de notre système est basée sur des observations d'interventions d'enseignants réels et celles-ci ont servi à élaborer des spécifications en collaboration avec les didacticiens de l'équipe. Dans ce contexte, nous avons opté pour une validation par les experts, soit les didacticiens, car ils ont la compétence requise pour analyser les échanges entre les élèves et QEDX. Nous nous assurons ainsi que notre système est valide du point de vue didactique et qu'il peut prétendre offrir un soutien pertinent (relation 6 de la figure 3.1) par rapport aux interactions qui ont été observées entre les élèves et les enseignants, sans pour autant remplacer ces derniers.

Finalement, nous avons tenté, en concevant QEDX, de créer un **Espace de Travail Géométrique** (ETG), soit un environnement propice à l'exercice réussi de la géométrie. Premièrement, un ETG est un univers organisé dans lequel s'effectue le travail du géomètre (novice) et, selon Kuzniak (2006), il intervient naturellement dans l'analyse de l'interaction sujet-milieu lors de la résolution de problèmes de géométrie. Le concept d'ETG permet de préciser la relation 2 (figure 3.1) de la TSD. Par exemple, Kuzniak (2011) utilise trois genèses, soit la genèse instrumentale, la genèse vidéo-figurale et la genèse discursivo-graphique, pour préciser cette relation. Dans le cadre de notre travail, nous avons analysé QEDX en cherchant à y repérer la possibilité de recourir à l'une ou l'autre de trois démarches, soit celles définies par Coutat et Richard (2011). Il y a premièrement la démarche de découverte qui s'articule autour de l'exploration des propriétés des objets géométriques et l'intégration des figures

dynamiques tend à bonifier celle-ci. Ensuite, la démarche de validation intervient lorsque l'élève tente de consolider les propriétés observées en un système formalisé et c'est alors que de nouvelles connaissances peuvent émerger. Quant à la démarche de modélisation, elle permet à l'élève de traduire son raisonnement opéré sur la figure géométrique en un raisonnement qui respecte le modèle déductif. Ainsi, elle s'accomplit principalement lors de la rédaction de la démonstration. Ces trois démarches sont, comme les genèses, des témoins du travail du géomètre au sein de l'ETG. Nous verrons, dans la section suivante, comment l'interface de QEDX permet de résoudre des problèmes par l'entremise de ces trois démarches.

Pour mériter le statut d'ETG, QEDX doit respecter deux conditions (Tessier-Baillargeon et al., 2014), soit d'être organisé de façon à permettre à l'élève de s'approprier le problème posé puis de clore sa résolution et, de plus, amener l'élève à travailler conformément aux attentes de l'institution scolaire. La première condition est considérée à priori par le fait que QEDX a été conçu à l'aide d'une démarche évolutive illustrée à la figure 3.2 et que chaque itération de celui-ci est influencée par les observations faites dans des situations authentiques de résolution de problèmes en classe. De plus, il a été démontré à postériori que son interface permet de réaliser les démarches de découverte, de validation et de modélisation (voir chapitre 7). En ce qui concerne la deuxième condition, il faut savoir que les graphes de démonstration pour les cinq problèmes inclus dans QEDX (annexe A) ont été sélectionnés et composés par les didacticiens de l'équipe qui connaissent le système scolaire québécois. Notre logiciel ne travaille donc pas en géométrie purement formelle, même s'il s'en approche, mais peut se conformer aux démarches ciblées par des enseignants québécois et peut aussi être adapté en fonction des approches adoptées par d'autres enseignants d'ici et d'ailleurs.

3.2.2 Interface utilisateur

QEDX, soit la version actuelle de notre système qui est exposée dans cette thèse, est le fruit du second cycle de conception (figure 3.2). Il fut élaboré à partir des observations réalisées lors de l'expérimentation qui a permis de clore le premier cycle (Tessier-Baillargeon et al., 2011). Dans son état courant, il s'agit d'un système tutoriel intelligent qui assiste l'élève lors de la résolution de problèmes de preuve en géométrie en produisant essentiellement des messages discursifs. Concrètement, QEDX permet de résoudre cinq problèmes de démonstration par le biais de son interface qui est illustrée à la figure 3.3. Il se compose d'une fenêtre principale qui est visible en tout temps et de quatre onglets qui permettent d'explorer le problème et de produire une preuve. Dans les paragraphes qui suivent, nous décrivons en détail l'interface et nous montrons comment chaque onglet permet à l'élève de réaliser son travail de géomètre confirmant ainsi le statut d'ETG à priori de QEDX.

Fenêtre principale

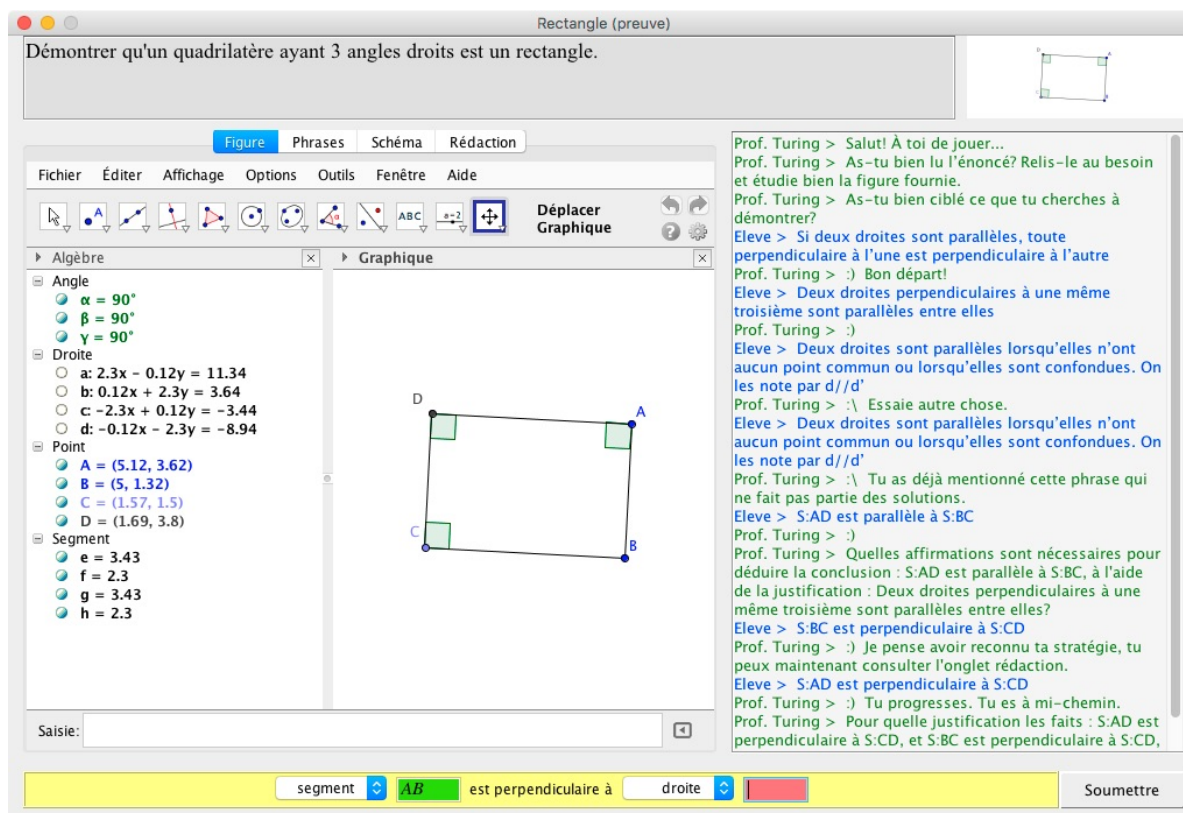


Figure 3.3 Fenêtre principale

La fenêtre principale est illustrée à la figure 3.3. Elle s'affiche à la suite du choix, par l'élève, du problème à résoudre parmi les cinq qui sont offerts dans la version actuelle de QEDX (annexe A). Dans le haut de la fenêtre, l'énoncé du problème et la figure statique rattachée sont affichés en permanence et ne peuvent pas être modifiés. Il y a ensuite la zone de gauche en dessous de l'énoncé qui permet d'afficher le contenu de l'onglet choisi par l'élève. Lors de l'amorçage de QEDX, l'onglet *Figure* est actif et celui-ci affiche la figure dynamique associée au problème. Le contenu des onglets est disponible en tout temps à la demande de l'élève, ce qui est unique, car nous n'imposons pas de contraintes ni de séquence prédéterminée à l'égard de son processus de résolution. Il peut donc naviguer à sa guise et le système est en mesure de lui proposer de l'aide tout au long de sa démarche par l'entremise de la zone de clavardage simulé.

La ligne de saisie, au bas de la fenêtre, et la zone de clavardage, à droite des onglets, sont reliées, car les phrases soumises par la ligne de saisie seront affichées dans la zone de clavardage à titre de contribution de l'élève (bleu) dans la conversation avec le tuteur (vert).

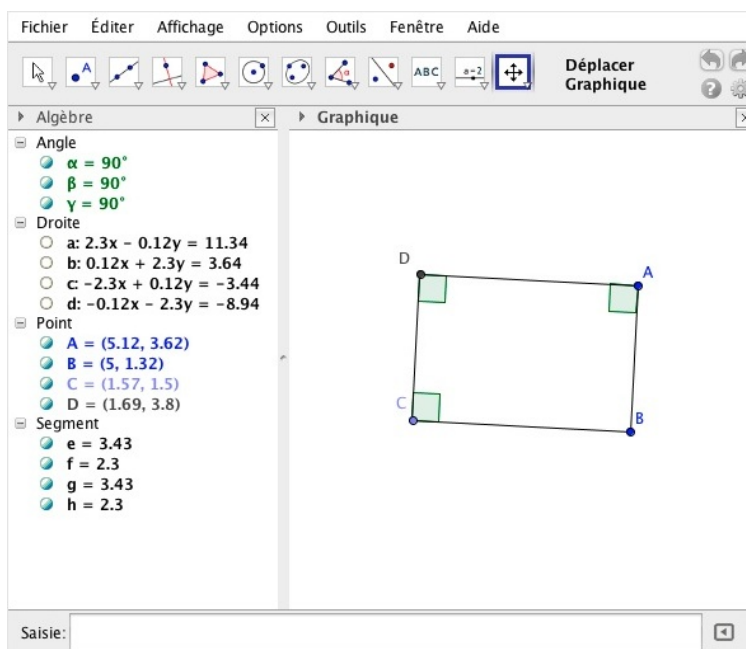
Les élèves utilisent l'onglet *Phrases* pour choisir les énoncés qu'ils devront ensuite compléter, si nécessaire, dans la ligne de saisie. Pour les justifications, il n'y a habituellement aucun élément à ajouter à l'énoncé choisi, donc il peut être soumis directement au tuteur. Par contre, en ce qui concerne les résultats intermédiaires, l'élève doit compléter les trous et une rétroaction sémaphorique (rouge : invalide, vert : valide) lui permet de s'assurer de la validité mathématique de chaque argument inscrit. Il s'agit cependant d'une simple aide syntaxique, car nous n'avons actuellement pas accès aux objets de l'onglet *Figure* pour vérifier l'existence des arguments dans la figure dynamique.

Nous retrouvons finalement, dans la zone de clavardage simulé affichée à droite et accessible en tout temps, la discussion entre Prof Turing (vert), qui est l'agent-tuteur développé dans le cadre de nos travaux, et l'élève (bleu). Prof Turing, après avoir inscrit un message de bienvenue, est à l'écoute des actions (strictement discursives) de l'élève. Lorsque ce dernier soumet un énoncé, le système fournit une rétroaction instantanée, sous forme d'émoticônes et de messages courts, qui indique si cet énoncé fait partie d'une solution connue et qui fait état de l'avancement de la démonstration considérée comme la plus complète par le tuteur. De plus, si l'élève n'est pas en mesure de progresser dans sa démonstration, Prof Turing peut lui proposer des pistes de solution plus ou moins précises à des intervalles prédéfinis pour tenter de le relancer. Il est aussi en mesure de détecter des répétitions dans le discours de l'élève ainsi que l'utilisation de propriétés réciproques. Cet agent-tuteur a été conçu à la suite de l'analyse des interactions observées entre les élèves et leurs enseignants lors de l'expérimentation qui a clos le premier cycle de conception et il constitue la principale contribution de cette recherche.

Onglet *Figure*

L'onglet *Figure*, illustré à la figure 3.4, est l'onglet initial affiché lors du chargement d'un problème. À l'heure actuelle, il contient la fenêtre principale de GeoGebra (Hohenwarter, 2013) qui a été importée dans QEDX afin de permettre à l'élève d'explorer la figure dynamique associée au problème. GeoGebra pourrait toutefois être facilement remplacé par un autre logiciel de géométrie dynamique. En effet, en ce moment seul un lien minimal existe entre celui-ci et QEDX, car son interface logicielle est inadaptée à nos besoins. Ainsi, faute de communication adéquate entre les deux systèmes, le travail fait sur la figure ne peut pas être reconnu par notre agent-tuteur. C'est pour cette raison que nous fournissons, dans l'onglet *Figure*, une figure complète, c'est-à-dire qu'elle contient tous les éléments figuratifs nécessaires à la rédaction des différentes démonstrations acceptées pour chaque problème.

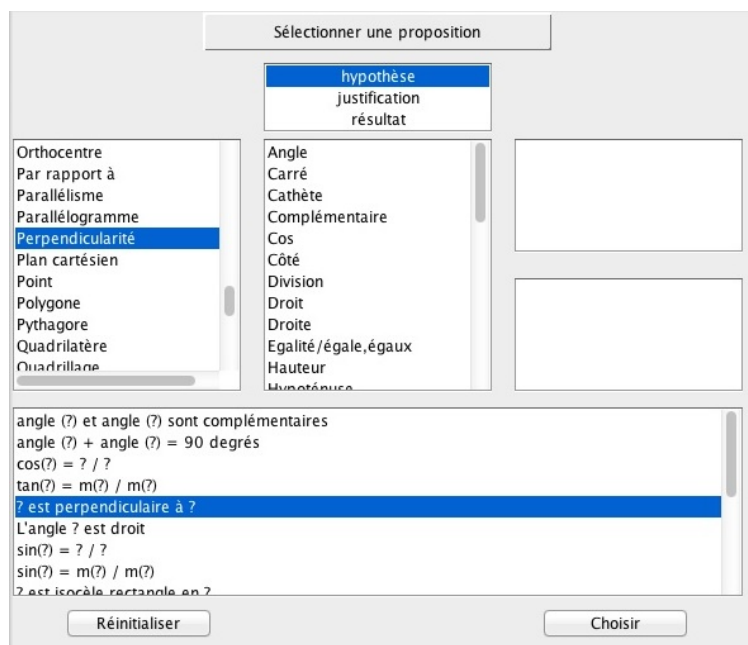
Malgré l'absence de tutorat relatif aux actions réalisées à l'interface du logiciel de géométrie dynamique importé dans QEDX, soit GeoGebra, il est malgré tout permis à l'élève d'employer

Figure 3.4 Onglet *Figure*

la figure dynamique affichée pour déduire et mettre à l'épreuve différentes conjectures. En effet, il lui est possible de faire bouger la figure, d'interroger les oracles et d'utiliser divers outils offerts dans GeoGebra. Cet onglet est donc une première étape dans la composition de la démonstration et correspond généralement à la démarche de découverte de l'élève, selon la nomenclature des ETG.

Onglet *Phrases*

Parallèlement à l'examen du libellé du problème et à l'étude de la figure dynamique, l'élève peut décrire ses observations et sa stratégie à QEDX. Pour ce faire, l'onglet *Phrases* (figure 3.5) est mis à sa disposition et contient un répertoire d'énoncés classé par thèmes. Concrètement, notre système inclut actuellement environ 240 énoncés de géométrie euclidienne qui ont chacun été indexés par un sous-ensemble de thèmes sélectionnés parmi les quelque 120 thèmes catalogués. Nous avons choisi de proposer un répertoire d'énoncés au lieu d'un module d'analyse de la langue naturelle, car QEDX s'adresse à des élèves qui font leurs premiers pas en géométrie, donc ils ne maîtrisent pas le langage associé à ce domaine. Il est ainsi difficile de créer un module qui est en mesure de comprendre ce que l'élève veut dire si ce dernier ne sait pas vraiment comment le dire. Cependant, même si ce choix simplifie la programmation, il ne réduit en rien le travail de l'élève, car ce dernier doit apprendre à retrouver efficacement l'énoncé désiré.

Figure 3.5 Onglet *Phrases*

À l'interface de l'onglet *Phrases*, il faut premièrement choisir si l'on désire écrire une hypothèse, un résultat ou une justification. Il est ensuite possible de sélectionner successivement jusqu'à quatre thèmes dans les fenêtres centrales. Chaque choix permet de filtrer les phrases affichées dans la fenêtre du bas, c'est-à-dire que seules celles qui sont caractérisées par tous les thèmes sélectionnés seront listées. De plus, uniquement les thèmes exploités par ces dernières apparaîtront dans la fenêtre suivante de sélection des thèmes. En faisant des choix judicieux, il est possible de réduire le nombre de phrases affichées de façon significative pour ainsi atteindre rapidement l'énoncé convoité. À tout moment, si l'élève trouve la phrase qu'il désire communiquer à Prof Turing, il la sélectionne et presse le bouton «Choisir» ce qui affiche la phrase dans la ligne de saisie de la fenêtre principale. Il peut alors compléter les trous, si nécessaire, avant de la soumettre au système. Il s'agit donc de l'onglet qui permet une interaction discursive avec QEDX et ce sont les phrases soumises, issues de ce répertoire, qui sont analysées par le tuteur.

De cette façon, l'élève peut employer l'onglet *Phrases* pour entrer les étapes de sa démonstration, mais il peut aussi l'exploiter pour enrichir ses connaissances en géométrie. En effet, lorsque celui-ci maîtrise l'utilisation des thèmes, il peut être en mesure de trouver une justification correctement écrite qui correspond à la situation observée. Il peut ainsi retrouver une propriété qu'il aurait oubliée ou en découvrir une nouvelle qui n'aurait pas encore été enseignée. Il est même possible de soumettre différentes justifications, concurremment à la

vérification de diverses conjectures sur la figure dynamique, dans le but de valider un plan de démonstration auprès du tuteur. Il en découle donc une véritable démarche de validation qui s'effectue principalement au sein de cet onglet.

Onglet *Schéma*

Le troisième onglet, prévu dans QEDX, consiste en une représentation graphique de la démonstration sous forme d'un arbre. Nous avons réservé de l'espace pour celle-ci, car nous croyons qu'il serait avantageux de proposer différentes représentations de la démonstration pour favoriser un meilleur apprentissage (Wong et al., 2011; Rau et al., 2012; Rau, 2013). Nous envisageons de nous inspirer de celles offertes, entre autres, dans le système MR Geo (Wong et al., 2011), car celui-ci a permis d'améliorer l'apprentissage des élèves selon ses auteurs. Malgré les bénéfices anticipés à la suite de l'ajout d'une représentation graphique de la preuve, l'onglet *Schéma* n'a pas été implanté faute de temps et n'est donc pas accessible dans la version actuelle de QEDX. Il fera toutefois l'objet d'une phase de conception ultérieure.

Onglet *Rédaction*

Le texte de rédaction est susceptible de changer au fur et à mesure de tes actions. Une fois ta démonstration complétée, tu peux cliquer sur celle-ci pour voir les codes de couleur des hypothèses (turquoise), des résultats (jaune), des justifications (vert) et de la conclusion (violet) apparaître.

Comme S:AD est perpendiculaire à S:CD , et S:BC est perpendiculaire à S:CD , alors, selon la propriété qui dit «Deux droites perpendiculaires à une même troisième sont parallèles entre elles» , S:AD est parallèle à S:BC .

Comme _____ , et S:AD est parallèle à S:BC , alors, selon la propriété qui dit «Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre» , _____ .

Comme _____ , S:AD est perpendiculaire à S:CD , S:BC est perpendiculaire à S:CD , et _____ , alors, selon la propriété qui dit «_____» , _____ .

Code chromatique

Hypothèse
Résultat intermédiaire
Conclusion
Justification

Figure 3.6 Onglet *Rédaction*

Le dernier onglet, soit l'onglet *Rédaction* (figure 3.6), permet à l'élève de visualiser la progression de son travail en lui présentant sa preuve sous forme de texte. Au début du problème, l'onglet est accessible, mais aucune démonstration n'est inscrite, car QEDX n'a pas encore

reconnu la stratégie de l'élève. Lorsqu'une preuve est complétée à 40%, elle est identifiée comme stratégie dominante et est affichée. Prof Turing indique alors, dans la zone de clavardage, qu'une démonstration est proposée dans l'onglet *Rédaction*. Par la suite, si une nouvelle stratégie dominante est détectée, l'onglet *Rédaction* affiche la preuve correspondante. Une démonstration est composée d'un certain nombre de phrases, soit des *inférences*, qui contiennent chacune un ensemble d'antécédents qui sont amalgamés à l'aide de la justification pour générer un conséquent. Dans cet onglet, nous avons identifié ces différents éléments au moyen d'un code chromatique. De plus, les inférences sont inscrites selon une séquence qui correspond au *chaînage avant*, c'est-à-dire que nous partons des hypothèses pour progresser jusqu'à la conclusion, et elles contiennent des trous pour indiquer les énoncés requis afin de compléter la démonstration. Toutefois, l'élève ne peut pas interagir directement avec la preuve inscrite et doit donc utiliser l'onglet *Phrases* pour soumettre les énoncés manquants.

L'onglet *Rédaction* a été ajouté à la suite de la première phase expérimentale (voir section 3.4), car nous avons observé que les élèves étaient déstabilisés par la structure peu orthodoxe de QEDX. En effet, étant donné qu'il est possible d'entrer les énoncés dans un ordre quelconque, il peut être difficile d'identifier les éléments manquants pour compléter sa démonstration. Nous avons donc intégré cet onglet non pas en tant qu'objectif de la résolution, comme dans Mentoniez (Py, 1994, 1996, 2001), mais bien en guise de moyen que les élèves peuvent employer pour juger de leur progression et pour apprendre la structure des preuves. L'utilisation de phrases à trous et du code chromatique ont ainsi permis de faire de la rédaction un véritable outil d'apprentissage. Enfin, cet onglet permet de réaliser la démarche de modélisation en proposant de faire une synthèse, par la rédaction d'une preuve complète, des découvertes et des validations réalisées au cours de la résolution du problème.

Dans cette section, nous avons d'abord présenté les aspects didactiques qui ont mené à l'élaboration de la version actuelle de QEDX. Nous avons montré comment notre système tutoriel intelligent est, par sa conception dans l'usage, un ETG à priori. Nous avons aussi expliqué comment l'agent-tuteur s'intègre dans la TSD de Brousseau par l'ajout de nouvelles relations dans le système sujet-milieu. Nous avons ensuite présenté les différents éléments de l'interface de notre système qui comprend une fenêtre principale et quatre onglets. Ces derniers permettent aux élèves de résoudre le problème posé en utilisant des démarches de découverte, de validation et de modélisation, ce qui confère à QEDX le statut d'ETG. Notre système propose donc une interface apte à soutenir un élève lors de la production d'une preuve associée à la résolution d'un problème de géométrie. La section suivante expose la structure interne actuelle de QEDX, soit un système comprenant quatre couches logicielles qui, par leurs interactions, font émerger l'intelligence de Prof Turing.

3.3 Architecture logicielle de QEDX

Maintenant que les bases didactiques qui sous-tendent l'élaboration de la version actuelle de notre système, soit QEDX, ont été clairement définies et que l'interface de celui-ci a été décrite à la section précédente, nous présentons la structure logicielle qui forme l'intelligence de Prof Turing et qui est le coeur de notre contribution au domaine des systèmes tutoriels intelligents en géométrie. Puisque la conception de notre système s'appuie sur une modélisation, fondée notamment sur des analyses didactiques, des interventions d'enseignants observés lors de la résolution, par les élèves, de problèmes en géométrie, la collaboration entre informaticiens et didacticiens est essentielle. Par conséquent, les structures et les algorithmes proposés ont été grandement influencés par ces derniers, car ce sont eux qui ont essentiellement identifié les interactions souhaitées et observées à l'interface de QEDX. De plus, comme il fallait que le logiciel puisse s'adapter à différents types d'enseignements, nous avons opté pour une approche guidée par les données, ce qui laisse un maximum de contrôle aux concepteurs de contenu, soit les didacticiens et les enseignants.

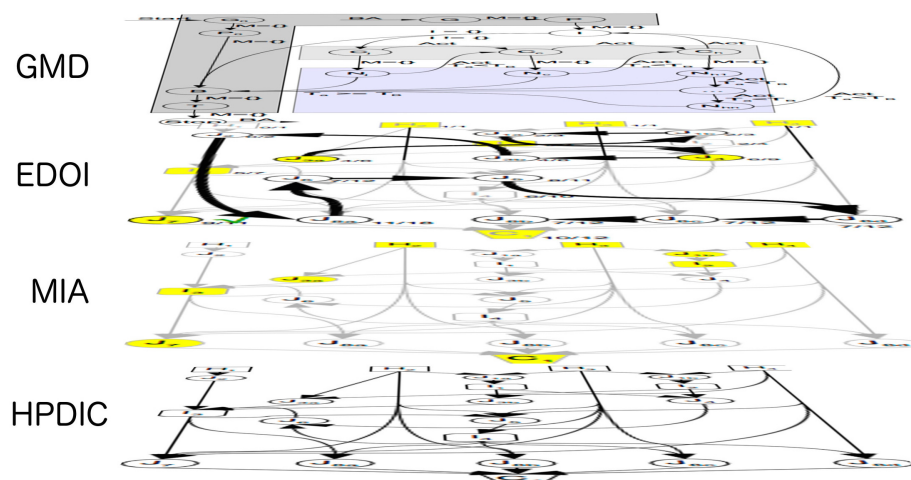


Figure 3.7 Architecture logicielle de QEDX

L'architecture logicielle de QEDX est construite autour du concept d'inférence, même si aucun ordre particulier n'est imposé à l'élève lors de l'écriture des énoncés. Elle est composée de quatre couches distinctes (figure 3.7) qui interagissent afin de fournir un soutien à la démonstration. Sommairement, la couche HPDIC permet de modéliser l'ensemble des solutions à un problème et celles-ci sont annotées par le MIA afin de consigner la chronologie des actions de l'élève. Ces informations sont ensuite interprétées par l'EDOI qui planifie une séquence de pistes de résolution, qui est exploitée par le GMD pour générer les messages d'aide discursifs de QEDX. Dans les paragraphes qui suivent, nous présentons une brève description

de chaque couche et de leurs interactions dans le but d'illustrer le fonctionnement général de notre système tutoriel intelligent.

3.3.1 Graphe HPDIC

La couche de base de QEDX sert à modéliser l'ensemble des démonstrations acceptables pour un problème donné. Nous l'avons nommée «graphe HPDIC», car il s'agit d'un graphe qui contient des **H**ypothèses, **P**ropriétés, **D**éfinitions, résultats **I**ntermédiaires et **C**onclusion. Celui-ci est unique pour chaque problème et est construit à partir des inférences acceptables, déterminées par les didacticiens en fonction du problème à résoudre et du contexte scolaire, qui sont inscrites dans un fichier texte. Il y a donc un travail préalable à faire de la part de ces derniers pour être en mesure d'ajouter un problème au logiciel. Le graphe HPDIC est une structure statique qui est mise en mémoire lors du choix du problème.

Pour construire le graphe HPDIC, nous devons premièrement créer une structure en arbre pour chaque inférence inscrite dans le fichier texte. Dans cet arbre, la justification constitue le noeud central auquel s'arriment les antécédents, en tant que parents, et le conséquent, en tant qu'unique enfant. À partir de l'ensemble des arbres produits, il suffit de fusionner les noeuds comportant des résultats intermédiaires (antécédents et conséquents) identiques afin de produire le graphe. Celui-ci relie donc les hypothèses à la conclusion par une alternance de justifications et de résultats intermédiaires. Le graphe HPDIC obtenu est véritablement un graphe, car il peut comporter des cycles dans le cas où des inférences symétriques, dans lesquelles les antécédents et le conséquent sont intervertis, seraient utilisées dans des démonstrations distinctes. Par exemple, dans le cas du problème du rectangle (annexe A.1), une stratégie de résolution consiste à montrer que la figure est un parallélogramme pour déduire la valeur de l'angle ABC . De façon symétrique, il est aussi possible de calculer la valeur de l'angle ABC afin de déterminer que la figure est un parallélogramme. Ce sont ces deux inférences légitimes qui forment un cycle dans le graphe. À partir du graphe HPDIC, nous sommes en mesure d'énumérer toutes les solutions possibles à l'aide d'un algorithme que nous avons conçu et qui fonctionne en *chaînage arrière*, c'est-à-dire qu'il part de la conclusion pour remonter jusqu'aux hypothèses.

Le graphe HPDIC constitue véritablement la base de notre système, car les informations qu'il contient sont exploitées par les couches supérieures. En effet, le MIA annote les noeuds pour rendre compte du cheminement de l'élève alors que l'EDOI l'utilise pour trouver la solution la plus avancée et pour choisir les inférences pour lesquelles Prof Turing fournira de l'aide. De plus, il est employé directement pour produire le texte de la fenêtre de rédaction. Sa structure a aussi été conçue dans le but de pouvoir y intégrer les graphes de construction et de pouvoir

l'utiliser comme représentation des preuves dans l'onglet *Schéma*. Ainsi, lorsque QEDX sera complété, le graphe HPDIC sera la structure qui unifiera les différentes représentations de la preuve et qui permettra d'offrir une aide contextualisée.

3.3.2 Modèle Itératif de l'Apprenant (MIA)

La couche suivante, soit le MIA, sert à modéliser le cheminement de l'élève lors de l'élaboration de sa démonstration. Les systèmes tutoriels classiques en géométrie utilisent habituellement une progression en chaînage avant ou arrière pour prévoir la prochaine action souhaitable de l'élève et limitent l'aide à cette unique action. Pour y arriver, seul l'ensemble des noeuds activés, sans égard à l'ordre d'activation, est requis. Certains systèmes utilisent, en plus, le dernier noeud activé pour offrir une aide plus personnalisée. En créant QEDX, notre but était de proposer, lors d'un blocage de l'élève, différentes pistes de solution, comme le ferait typiquement un enseignant. Nous avons donc proposé le MIA qui permet de rendre compte de l'avancement de l'élève, mais aussi de la chronologie de ses actions, pour être en mesure de proposer une aide qui a pour objectif de respecter son état cognitif, donc mieux adaptée à sa démarche de résolution.

Le graphe HPDIC étant statique, il nous fallait définir une structure pour enregistrer la progression de l'élève dans l'élaboration de sa démonstration. En raison d'une communication actuellement déficiente entre le logiciel de géométrie dynamique, soit GeoGebra (Hohenwarter, 2013), et QEDX, nous avons décidé, pour l'instant, de ne traiter que les actions discursives de l'élève, à savoir les énoncés soumis qui sont issus de l'onglet *Phrases*. Dans ce contexte, le MIA consiste à ajouter des informations concernant les actions discursives de l'élève à chaque noeud du graphe HPDIC. Ainsi, pour chaque phrase inscrite qui se trouve dans le graphe HPDIC (action valide), nous indiquons, dans tous les noeuds la contenant, une valeur qui correspond au moment de l'activation. Dans la couche MIA de la figure 3.7, les noeuds activés ont une coloration jaune. Ces informations sont constamment mises à jour lors de l'élaboration, par l'élève, d'une démonstration. Elles sont utilisées dans les couches supérieures pour générer des messages adaptés et pour proposer des pistes de solution relativement aux actions valides récentes.

3.3.3 Évaluation des Démonstrations / Ordonnancement des Inférences (EDOI)

Pour aider l'élève, QEDX génère des messages instantanés et une série d'indices par l'entremise de la couche GMD. Ceux-ci ont pour but de relancer le processus de résolution lors d'un blocage sans pour autant donner directement les réponses. Pour produire ces messages, nous devons préalablement déterminer la démonstration la plus avancée parmi toutes celles qui

sont possibles et ordonnancer les inférences selon un ordre logique qui respecte l'état cognitif de l'élève. Dans le cadre de notre système, c'est la couche EDOI qui met en oeuvre ces deux opérations. Elle se charge, d'une part, d'estimer l'achèvement des différentes démonstrations, mais, en fait, seule celle qui est la plus avancée nous intéresse, car elle permet d'évaluer la progression de l'élève. Ainsi, cette dernière est utilisée pour générer le message instantané et proposer une preuve dans l'onglet *Rédaction*. D'autre part, l'algorithme d'ordonnancement emploie cette même démonstration, qui est considérée comme le plan de résolution courant de l'élève, pour privilégier les inférences qui font partie de celle-ci. Il combine cette information avec les données du MIA afin d'obtenir une liste ordonnée d'inférences qui est mise à jour après chaque action valide de l'élève et qui est exploitée par le GMD.

Dans QEDX, nous définissons la démonstration la plus avancée comme étant celle dont le ratio du nombre de phrases énoncées par l'élève par rapport au nombre de phrases nécessaires pour la rédiger en entier est le plus élevé. En présence d'un graphe HPDIC, il s'agit de calculer le pourcentage de noeuds activés pour chaque solution. Cette procédure demande cependant de calculer le ratio pour chaque démonstration après chaque action valide de l'élève, ce qui peut être long pour des problèmes possédant des millions de solutions. Pour éviter ces calculs onéreux, nous avons conçu un algorithme qui permet de propager une approximation du pourcentage de noeuds activés, mais celui-ci fonctionne uniquement sur des arbres HPDIC. Pour ce qui est des graphes HPDIC, nous créons, au préalable, un ensemble d'arbres HPDIC qui contiennent collectivement toutes les démonstrations du graphe d'origine. On peut alors, par traçage arrière, retrouver la démonstration la plus avancée de façon presque immédiate. Cette dernière sert notamment à ordonner les inférences dans l'EDOI et à produire le message instantané dans le GMD.

L'algorithme d'ordonnancement des inférences entre ensuite en jeu pour indiquer au GMD dans quel ordre les indices devraient être proposés à l'élève en état de blocage. À l'aide des données du MIA, il est possible de savoir sur quelle inférence l'élève a le plus récemment travaillé afin de l'aider sur celle-ci en priorité. En fait, nous ordonnons toutes les inférences du graphe HPDIC selon un ordre chronologique inverse, soit de la plus récente à la plus ancienne. Dans le cas des inférences n'ayant pas encore été traitées par l'élève, elles se retrouvent dans un ordre quelconque en queue de liste, car il s'agit de nouvelles pistes qui n'ont pas encore été explorées par ce dernier. Lorsque la démonstration la plus avancée dépasse un certain seuil d'achèvement, les inférences faisant partie de celle-ci sont traitées en priorité, même si d'autres ont été travaillées plus récemment, car nous voulons inciter l'élève à compléter cette démonstration. Pour y arriver, nous ordonnons premièrement les inférences de la démonstration la plus avancée. Nous y concaténons ensuite, en ordre chronologique inverse, le reste des inférences du graphe. Une fois l'ordonnancement complété, la liste ordonnée est fournie

au GMD, qui l'utilise pour générer les messages qui permettront possiblement de relancer le processus de résolution d'un élève bloqué.

3.3.4 Génération des Messages Discursifs (GMD)

Un des objectifs de QEDX consiste à offrir à l'élève une aide inspirée des interventions d'enseignants réels. Afin de l'atteindre, nous avons ajouté à notre système la couche GMD, qui génère les messages discursifs de Prof Turing visant à relancer le processus de résolution de l'élève. La séquence et la teneur des messages à afficher ont été déterminées à la suite de l'analyse des interactions entre les élèves et leurs enseignants, qui furent observées lors de l'expérimentation qui a permis de clore le premier cycle de conception (figure 3.2). Nous avons alors décelé une certaine structure tant au niveau de l'ordre des messages que de leur contenu et nous l'avons exploitée lors de la programmation de ceux-ci au sein de QEDX. La couche GMD, utilisant notamment la liste d'inférences ordonnée, complète donc l'intelligence de notre agent-tuteur. Elle est d'ailleurs une des principales contributions de notre recherche, car elle est en mesure de proposer des messages d'aide, selon un horaire préétabli, comme le ferait un enseignant typique. L'élève ne peut toutefois pas forcer QEDX à résoudre le problème à sa place, car il lui est impossible de solliciter directement des indices.

Premièrement, la couche GMD génère les rétroactions instantanées qui modélisent certaines réactions spontanées de l'enseignant. Elles apparaissent dans la fenêtre de clavardage sous forme d'émoticônes et de messages courts en réponse à la soumission d'un énoncé. Ces rétroactions sont simples à produire, car elles ne dépendent que de la pertinence de l'énoncé soumis et de la progression de l'élève. D'une part, la pertinence est évaluée en utilisant les données de la couche HPDIC ainsi qu'une banque d'erreurs modélisées. La progression est, d'autre part, calculée sur la solution la plus avancée, extraite par la couche EDOI, et permet d'influencer la rétroaction dans le cas d'un énoncé valide, à savoir s'il est contenu dans un des noeuds du graphe HPDIC.

La couche GMD met également en oeuvre une machine à états finis qui comporte tous les états nécessaires afin de reproduire les principales stratégies employées par les enseignants observés pour relancer un élève bloqué dans son processus de résolution. À chaque état est associé un ensemble de messages qui pourraient être affichés dans la fenêtre de clavardage. Cependant, puisque tous les états ne sont pas utilisés dans toutes les situations, il est possible d'en sauter certains en ne leur attribuant aucun message. Comme QEDX a été conçu selon une architecture guidée par les données, les messages pour les différents états sont définis par les didacticiens ou les enseignants dans un fichier texte. En plus des messages statiques, il est possible de composer des messages dynamiques qui intègrent certains énoncés inscrits

par l'élève. Différents niveaux de messages peuvent même être programmés, allant des plus génériques aux plus précis. Les concepteurs de contenu ont donc un contrôle déterminant sur les interventions de Prof Turing.

Pour générer les messages, la machine, à la suite d'une action valide de l'élève, récupère la liste ordonnée d'inférences produite par l'EDOI et se place dans un état initial. Elle parcourt alors séquentiellement les divers états et affiche, dans l'ordre prévu, les messages contenus dans chacun, tout en respectant les délais établis. Les premiers états renferment des messages génériques qui s'appliquent à l'ensemble du problème tandis que les suivants sont associés aux éléments d'une inférence particulière qui provient de la liste de l'EDOI. Notre machine parcourt, dès lors, tous les messages d'aide par rapport à une inférence avant de passer à la suivante, soit une autre piste de résolution. Elle continue de traiter les inférences jusqu'à ce que la liste de l'EDOI soit vide ou qu'un délai maximal sans action valide soit atteint. À ce moment, la machine se place dans un état final qui indique l'action que l'élève devrait exécuter ou qui l'invite à consulter son enseignant. Tous les messages de QEDX sont ainsi générés par la couche GMD, alors que les autres couches (EDOI, MIA et HPDIC), de même que les textes créés par les didacticiens, servent à déterminer leur séquence et leur contenu. Cette dernière couche permet donc de reproduire un sous-ensemble appréciable des interventions d'un enseignant typique lors de la résolution, par l'élève, d'un problème de géométrie.

Dans cette section, nous avons proposé une synthèse des quatre couches, relativement autonomes, constituant QEDX (figure 3.7). La couche de base contient le graphe HPDIC qui rassemble toutes les démonstrations acceptables pour un problème donné et qui est exploité par les différents composants de notre système. Au-dessus s'ajoute la couche MIA qui permet de rendre compte du cheminement de l'élève dans son processus de résolution en consignait la chronologie de ses actions discursives. À partir de cette dernière, la couche EDOl détermine la démonstration la plus avancée et ordonne les inférences pour établir la stratégie d'aide à adopter. Enfin, la couche GMD génère une rétroaction instantanée et, si nécessaire, un message court à la suite de chaque action discursive de l'élève afin de l'informer de sa progression. De plus, à partir de la liste ordonnée des inférences, elle produit, à la manière d'un enseignant, les indices proposés par Prof Turing dans le but de soutenir le processus de résolution de l'élève. Dans la section suivante, nous décrivons les expérimentations ainsi que la méthodologie qui ont permis de concevoir et de valider la version actuelle de QEDX.

3.4 Méthodologie

Dans le contexte de la conception dans l'usage (figure 3.2), il n'est pas logique d'utiliser des prétests et des post-tests pour évaluer QEDX, car ce dernier est en constante évolution

afin de l'adapter aux schèmes d'utilisation. Ainsi, les phases expérimentales ont pour but à la fois la validation du système présenté aux élèves, mais aussi la cueillette de données qui seront utilisées au cours des étapes subséquentes de conception. Dans le cadre de notre projet, nous avons réalisé deux expérimentations qui ont permis de développer la version actuelle de QEDX. Dans cette section, nous exposons d'abord les démarches engagées lors de la validation fonctionnelle du système. Nous expliquons comment les tests unitaires, les simulations et l'examen effectué par un expert ont permis notamment de s'assurer que les messages discursifs de Prof Turing sont conformes aux spécifications établies. Par la suite, nous décrivons les deux expérimentations en classe mises en oeuvre pour façonner QEDX. Nous présentons les buts et le contexte de chacune en plus de proposer un portrait sommaire de la méthodologie adoptée par les didacticiens de l'équipe afin d'extraire, des informations recueillies, les données pertinentes à l'élaboration et à la validation de notre système.

Lors de l'implantation de QEDX, nous avons initialement eu recours au développement dirigé par les tests afin de nous assurer de la qualité des classes et des algorithmes programmés. Nous avons alors créé et entretenu 81 séries de tests qui ont permis de valider les structures de données de base, comme l'encodage des énoncés et de leurs arguments. Toutefois, nous avons abandonné cette pratique en cours de projet, car l'effort requis pour vérifier les classes complexes nous a semblé démesuré par rapport aux gains potentiels reliés à l'usage d'une telle approche. Étant donné que nous devons tout de même tester les différentes fonctionnalités implantées dans notre système, nous avons donc décidé d'exécuter systématiquement, à son interface, des simulations manuelles de divers scénarios d'utilisation. Celles-ci, en plus de nous permettre de détecter des erreurs de programmation, sont à l'origine de plusieurs exemples présentés dans cette thèse. Néanmoins, nous savions pertinemment que ces simulations étaient loin de permettre une vérification exhaustive de QEDX. Par conséquent, à la suite de la programmation des messages de Prof Turing, nous avons demandé à un expert en didactique indépendant de mettre à l'épreuve notre système. Son mandat consistait à produire des simulations de résolution, à la manière d'un élève, afin de vérifier si le contenu et la séquence des messages générés correspondaient à ceux qui étaient prévus. Il a ainsi exécuté une dizaine de simulations pour chacun des quatre problèmes fonctionnels à l'époque, ce qui a nécessité l'écriture d'un peu plus de 900 énoncés. Un rapport technique décrivant les erreurs détectées et proposant des améliorations à apporter à QEDX a enfin été produit et a servi à corriger notre système préalablement à la seconde expérimentation en classe.

Les tests fonctionnels et les simulations témoignent d'une programmation conforme aux spécifications, mais ils ne nous informent pas sur les schèmes d'utilisation exploités par les élèves en situation réelle de résolution de problème. Afin de compléter l'évaluation de QEDX, mais aussi de le faire évoluer, nous avons réalisé deux expérimentations en classe dont un résumé

du contexte de chacune est disponible dans le tableau 3.1. La première phase expérimentale avait pour but de valider les solutions aux problèmes proposés, encodées sous forme de graphes HPDIC, et de recueillir des données concernant les interactions entre les élèves et leurs enseignants pour être en mesure d'intégrer à notre système diverses rétroactions sous forme de messages discursifs. Il faut savoir que lors de cette expérimentation, GGBT ne fournissait que des valeurs numériques correspondant au pourcentage d'achèvement de chacune des solutions. Ainsi, l'élève avait une idée de sa progression, mais il lui était difficile de déterminer les énoncés manquants pour compléter une démonstration. Nous avons donc choisi des groupes d'élèves doués de l'École d'Éducation Internationale de Laval pour maximiser nos chances d'obtenir des données intéressantes avec ce logiciel rudimentaire. Nous avons aussi eu l'occasion de travailler avec deux enseignants préconisant des approches distinctes, ce qui nous a permis d'identifier des stratégies communes d'intervention. À la suite de l'implantation des messages discursifs de Prof Turing, nous avons eu recours à une seconde expérimentation afin de les valider et de confirmer le statut d'ETG de QEDX. Nous avons, de plus, récolté des données concernant les différents profils d'interventions pour intégrer ultérieurement à notre système une aide adaptée à la compétence de l'élève. La deuxième expérimentation s'est alors déroulée à l'Académie Ste-Thérèse, dans des classes ordinaires, car nous avions, à ce moment, une version plus évoluée de QEDX. Ce choix nous a ainsi permis de tester l'efficacité de notre système tutoriel intelligent auprès d'élèves ayant des niveaux de compétence variés.

Tableau 3.1 Caractéristiques des deux phases expérimentales

	Phase I (GGBT) décembre 2010	Phase II (QEDX) février 2013
Objectifs	Validation des solutions Développement des messages discursifs	Validation des messages discursifs Développement des profils d'intervention
Contexte	4 problèmes (annexe A : A.1, A.2, A.3, A.4) Capture d'écran (ScreenFlow) : 4 équipes / groupe Fichiers journaux : toutes les équipes Dictaphone et interview pour chaque enseignant 4 groupes - 2 enseignants 4 ^e et 5 ^e secondaire (15-16 ans) 4 périodes × 75 min / groupe environ 54 équipes de 2 élèves	3 groupes - 2 enseignants 4 ^e secondaire (15-16 ans) 2 périodes × 60 min / groupe environ 30 équipes de 2 élèves

Lors des deux expérimentations, les élèves de quatrième ou cinquième secondaire travaillaient en duos. À la suite d'une séance préalable où l'on présentait le concept de démonstration ainsi que le fonctionnement de notre système, les élèves étaient invités à résoudre quatre problèmes. La conversation entre Prof Turing et les élèves a été sauvegardée dans des fichiers journaux

qui furent récupérés par l'expérimentateur à la fin de chaque séance. Nous avons aussi utilisé le logiciel ScreenFlow (figure 3.8), qui permet d'enregistrer, en plus des actions à l'écran, les visages et les discussions des coéquipiers, dans le but de rendre compte du travail effectué par quatre équipes choisies par l'enseignant dans chaque classe. Enfin, pour être en mesure d'analyser les interactions entre les élèves et leurs enseignants, ces derniers portaient chacun un dictaphone qui enregistrerait tous leurs échanges verbaux.

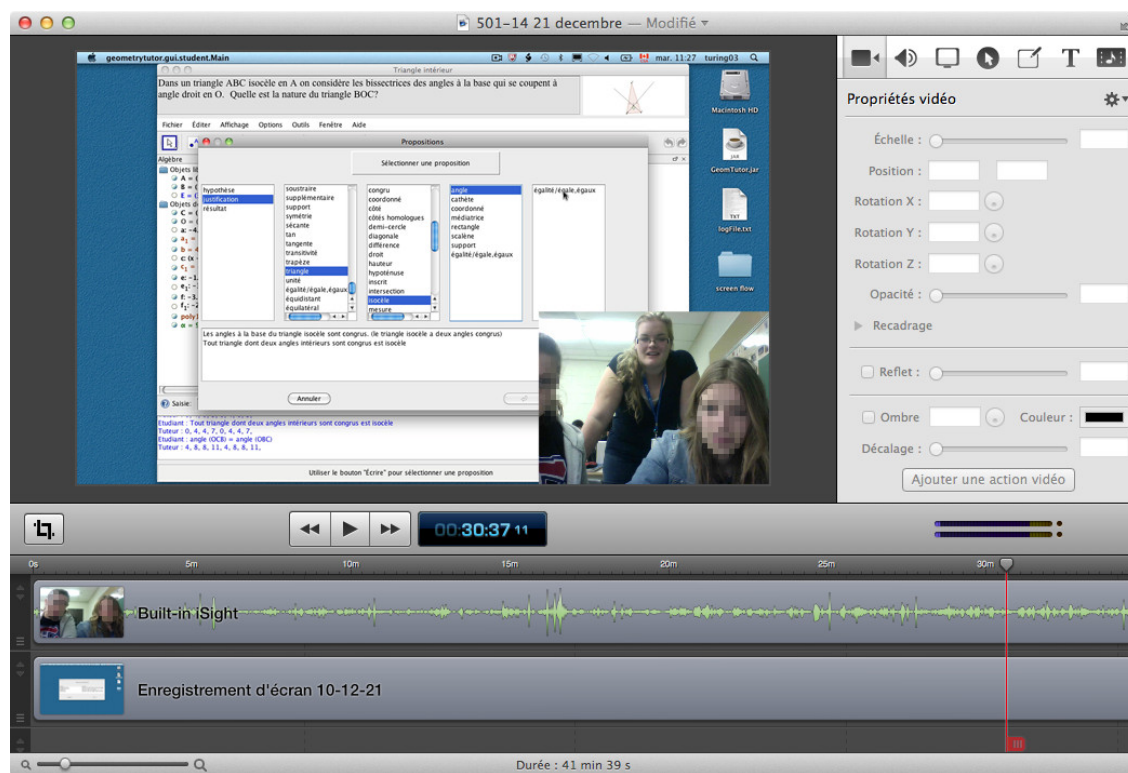


Figure 3.8 Interface du logiciel ScreenFlow

L'analyse des données issues des expérimentations a été principalement réalisée par les experts en didactique de notre équipe. Une description détaillée de la méthodologie qu'ils ont employée est proposée dans Tessier-Baillargeon (2016), alors qu'ici nous en dressons un bref portrait à titre informatif. Afin de vérifier si QEDX permet à un élève de travailler en accord avec les principes didactiques mis en oeuvre à chaque étape de son élaboration, nous avons choisi de l'analyser à l'aide de méthodes qualitatives relevant des sciences humaines. Pour ce faire, les didacticiens ont d'abord produit les verbatims de toutes les vidéos auxquels ils ont combiné l'information contenue dans les fichiers journaux associés. Ils ont ensuite annoté les fichiers obtenus en mettant en application des analyses qualitatives issues du cadre rigoureux exposé par Paillé et Mucchielli (2012). Ceux-ci décrivent, entre autres, différentes techniques d'analyse de documents, allant de l'extraction de mots clés aux catégories conceptualisantes.

Ces dernières consistent à associer à une séquence d'actions un concept établi à la suite d'une véritable réflexion de la part du chercheur. Elles ont ainsi été utilisées afin d'extraire les stratégies d'interventions observées lors de la première expérimentation et pour déterminer les démarches mises en oeuvre par les élèves lors de la seconde. De plus, au moment de l'analyse, les didacticiens ne se sont pas limités aux concepts déterminés à priori, mais ils se sont adaptés afin que de nouveaux schèmes d'utilisation, qui n'étaient pas prévus initialement, puissent émerger. Ce sont donc ces analyses qui ont contribué à donner naissance à la version actuelle de QEDX et qui ont permis de confirmer son statut d'ETG.

L'objectif de cette section était de présenter une description sommaire de la méthodologie utilisée pour valider QEDX. D'une part, afin de nous assurer de la qualité du logiciel réalisé, nous avons d'abord employé une approche de développement dirigé par les tests qui a éventuellement été remplacée par des tests fonctionnels de haut niveau, soit l'exécution manuelle de simulations à l'interface. De plus, pour compléter la vérification fonctionnelle du système, nous avons mandaté un expert en didactique afin qu'il exécute des scénarios de résolution inspirés du travail des élèves et qu'il produise un rapport technique relatant ses conclusions. Dans le but de vérifier si notre système autorise la mise en oeuvre du contrat didactique, les tests fonctionnels ne sont toutefois pas suffisants. Par conséquent, nous avons eu recours à deux expérimentations en classe qui ont permis à la fois de valider le système et de récolter des informations utiles pour les phases de conception ultérieures. Ainsi, les données issues de la première expérimentation ont servi à vérifier les solutions implantées et à élaborer les messages discursifs de QEDX, qui a ensuite été validé lors de la seconde. Diverses données ont été recueillies au cours de chaque expérimentation, mais ce sont principalement les enregistrements vidéos et les fichiers journaux qui ont été étudiés. Comme nous avons choisi d'employer des méthodes d'analyses qualitatives, ce sont essentiellement les didacticiens de l'équipe qui ont effectué les analyses. Ils ont utilisé, entre autres, les catégories conceptualisantes afin d'extraire les schèmes d'utilisations prévus et inattendus apparaissant dans les verbatims des vidéos. Cette méthodologie, qui peut paraître surprenante dans le cadre d'une thèse en informatique, est toutefois cohérente avec notre démarche de conception multidisciplinaire.

3.5 Conclusion

Dans ce chapitre, nous avons présenté une vue d'ensemble de QEDX en nous intéressant plus particulièrement à ses aspects didactiques et à son architecture informatique globale. Il faut premièrement rappeler que l'objectif de notre projet était de proposer un système tutoriel intelligent qui soutient l'élève dans une démarche d'exploration, plutôt que de rédaction, dans le cadre de l'élaboration d'une preuve en géométrie. Pour l'atteindre, nous avons employé

une approche multidisciplinaire, donc chaque décision concernant l'architecture du système a été prise à la suite d'analyses didactiques auxquelles se sont greffées des considérations informatiques. Par conséquent, nous nous sommes appuyé sur des bases didactiques, comme la TSD et les ETG, lors de notre analyse à priori. Ainsi, nous avons montré que QEDX est en fait un ETG en se basant sur l'étude de l'ergonomie de son interface alors qu'une confirmation à posteriori est présentée au chapitre 7. De plus, il faut se rappeler que notre système est en constante évolution, car nous avons appliqué le paradigme de la conception dans l'usage. Dans celui-ci, chaque expérimentation a pour objectif la cueillette de données pour l'élaboration des nouvelles fonctionnalités, mais aussi la vérification des schèmes d'utilisation dans le but de les évaluer et de les corriger afin qu'ils soutiennent la mise en oeuvre du contrat didactique. Les données récoltées au cours des deux expérimentations réalisées ont fait notamment l'objet d'analyses didactiques rigoureuses pour produire des résultats qui sont exposés dans les chapitres concernés. Cette stratégie de conception rend donc QEDX unique, car il y a adaptation mutuelle entre celui-ci et l'élève dans le but d'obtenir un éventuel état d'équilibre, ce qui n'est pas le cas pour les systèmes tutoriels intelligents traditionnels.

En ce qui concerne le volet informatique, notre système se distingue premièrement par son interface qui permet à l'élève de naviguer librement entre les différents onglets tout au long de la construction de sa démonstration. Cette caractéristique a permis de proposer la rédaction non seulement en tant que fin, comme plusieurs le pensent, mais plutôt en guise d'outil qui peut être exploité lors de la démarche de modélisation. Du côté de son architecture, le système actuel comporte quatre couches distinctes et modulaires, soit le graphe HPDIC, le MIA, l'EDOI et le GMD, qui sont décrites en détail dans les trois chapitres subséquents. L'inférence est la structure de base de QEDX et sert à construire le graphe HPDIC qui représente l'ensemble des démonstrations pour un problème donné. En utilisant ce graphe, les autres couches sont en mesure de modéliser l'apprenant et de produire les messages discursifs de Prof Turing. Dans le but de donner aux didacticiens un contrôle important sur ces messages et de leur permettre de produire du contenu, nous avons employé une approche centrée sur les données. De plus, comme QEDX ne produit pas seulement des messages pour la prochaine étape idéale, mais offre des pistes de solution qui dépendent du travail de l'élève, il a notamment été nécessaire de créer le MIA, qui contient un véritable historique des actions de l'élève. Les messages proposés par Prof Turing, modélisés à partir d'observations du travail de l'enseignant, ainsi que les algorithmes requis pour produire l'intelligence artificielle de QEDX sont les principaux apports de notre recherche et ils constituent les éléments qui nous démarquent des systèmes tutoriels intelligents classiques pour la géométrie.

CHAPITRE 4 REPRÉSENTATION DES DÉMONSTRATIONS (HPDIC) ET MODÉLISATION DE L'APPRENANT (MIA)

«Les mathématiques consistent à prouver des choses évidentes par des moyens complexes.»

— George Polya

4.1 Introduction

Dans le chapitre 3, nous avons présenté une description sommaire de QED-Tutrix (QEDX), soit le système tutoriel intelligent conçu dans le cadre de notre recherche, ainsi que les bases didactiques sur lesquelles notre travail s'est appuyé. Comme notre mise en oeuvre utilise le paradigme de conception dans l'usage, QEDX est en constante évolution et chacune des deux expérimentations effectuées a permis à la fois de valider le travail réalisé, mais aussi de recueillir des données pour les phases de développement subséquentes. La version actuelle de QEDX est en mesure de proposer une aide discursive qui s'adapte aux actions de l'élève pour cinq problèmes qui ont été choisis de façon à représenter la plupart des aspects didactiques qui doivent être pris en compte par le système.

QEDX comporte quatre couches principales (HPDIC, MIA, EDOI et GMD) qui permettent à la fois de gérer la multitude de démonstrations possibles pour chaque problème, mais aussi l'aide discursive proposée à l'élève. La couche HPDIC contient le graphe de toutes les solutions possibles et est la base de QEDX. On y superpose ensuite la couche MIA qui contient le modèle de l'apprenant. Nous planifions ensuite une séquence d'interventions qui permettra de faire cheminer l'élève à l'intérieur d'un problème dans la couche EDOI. Finalement, la couche GMD contient le moteur qui permet de générer les messages discursifs selon la séquence d'interventions planifiée.

Dans ce chapitre, nous décrivons, de manière formelle, les deux couches formant la base de QEDX, soit les graphes HPDIC et le MIA. Les descriptions détaillées des couches EDOI et GMD sont présentées dans les chapitres 5 et 6 respectivement. Dans un premier temps, nous traitons des formes les plus courantes que pourrait prendre une démonstration en géométrie et nous faisons ressortir les difficultés de leur utilisation dans un système tutoriel intelligent. Nous faisons aussi un tour d'horizon de différentes stratégies de modélisation de l'apprenant et nous montrons que celles-ci s'appliquent difficilement dans le cadre de la mise en oeuvre de QEDX. Les descriptions formelles des graphes HPDIC et du MIA suivent. Les

solutions associées à un problème spécifique, tel qu'elles ont été implantées dans QEDX, sont utilisées pour illustrer chacune des définitions formelles. Nous présentons ensuite les statistiques concernant les problèmes implantés. Nous montrons enfin que l'utilisation des graphes HPDIC est avantageuse, car ils permettent de représenter l'ensemble des solutions à un problème et de produire des solutions sous différentes formes. Les choix concernant le MIA et son intégration aux graphes HPDIC sont aussi expliqués.

4.2 Problématique et enjeux

Dans cette section, nous présentons premièrement le problème du rectangle qui a été implanté dans QEDX et qui est utilisé pour initier les élèves à l'utilisation du logiciel. Nous proposons aussi un encodage des différents éléments faisant partie des solutions acceptables dans le but d'alléger le texte, car le problème du rectangle sera utilisé tout au long du chapitre pour illustrer les différents concepts. Nous proposons ensuite une solution possible à ce problème et nous la présentons en utilisant les représentations les plus courantes. Nous verrons que ces représentations permettent de rédiger adéquatement une preuve, mais sont inadéquates lorsqu'il s'agit de montrer l'ensemble des démonstrations acceptables qui permettent de résoudre le problème. Dans un second temps, nous présentons les différents modèles de l'apprenant utilisés dans les logiciels d'aide à la démonstration en géométrie. Nous verrons que l'application de ceux-ci au projet actuel est assez limitée, ce qui nous a contraint à proposer un nouveau modèle mieux adapté.

4.2.1 Problème du rectangle et classification des énoncés

Le problème du rectangle, présenté à la figure 4.1, est un problème typique qui pourrait être proposé aux élèves de quatrième secondaire dans le cadre du volet géométrie du programme québécois de mathématique au secondaire (Ministère de l'Éducation, du Loisir et du Sport, 2009). Ce problème exige, de la part de l'élève, d'exercer sa compétence en démonstration, car il ne fait pas intervenir de valeurs numériques.

Démontrer qu'un quadrilatère ayant trois angles droits est un rectangle.

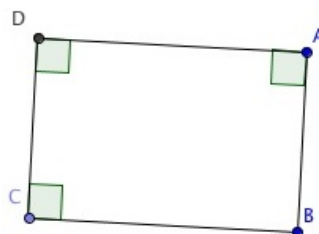


Figure 4.1 Problème du rectangle

Pour résoudre ce problème, un élève aura besoin de trouver un certain nombre d'hypothèses ainsi que la conclusion. De plus, il est fort probable que sa démonstration comporte plus d'une étape, donc des résultats intermédiaires apparaîtront. L'ensemble de ces énoncés utiles à la résolution du problème du rectangle, tels qu'identifiés par les didacticiens, est présenté dans le tableau 4.1, ainsi que l'encodage qui sera utilisé dans la suite du chapitre. Nous utilisons le terme *résultats intermédiaires* pour les représenter indépendamment de leur emplacement à l'intérieur de la démonstration.

Tableau 4.1 Hypothèses, résultats Intermédiaires et Conclusion (HIC) permettant de résoudre le problème du rectangle

Code	Énoncé de résultat
H_1	ABCD est un quadrilatère
H_2	BC est perpendiculaire à CD
H_3	AD est perpendiculaire à CD
H_4	AB est perpendiculaire à AD
I_1	AD est parallèle à BC
I_2	AB est parallèle à CD
I_3	AB est perpendiculaire à BC
I_4	ABCD est un parallélogramme
C_1	ABCD est un rectangle

Pour créer des liens entre les différents résultats intermédiaires, nous utilisons des propriétés et des définitions issues de la géométrie euclidienne, que nous regroupons sous l'appellation de *justifications*. L'ensemble des justifications retenues pour résoudre le problème du rectangle est présenté dans le tableau 4.2. Dans ce tableau, nous avons ajouté des identificateurs pour chaque justification, afin de les retrouver dans le graphe HPDIC qui sera construit, selon la procédure décrite plus loin.

4.2.2 Principaux styles de preuve

Une solution possible au problème du rectangle, en utilisant les énoncés de la section précédente, est présentée à la figure 4.2 en utilisant différentes représentations. La preuve sous forme de texte écrit (4.2a) est la forme que plusieurs enseignants et logiciels, comme Mento-niezh (Py, 1994, 1996, 2001), préconisent comme résultat final de l'élaboration d'une preuve. Le texte est structuré, ce qui le rend facile à lire, et la démonstration est ainsi plus facile à suivre. Cependant, dans le cas où nous aurions besoin de représenter un ensemble de preuves, il est nécessaire d'avoir un texte pour chacune, ce qui est difficile à produire et à traiter pour des problèmes admettant des centaines voire des milliers de preuves différentes.

Tableau 4.2 Propriétés et Définitions (PD) permettant de résoudre le problème du rectangle

Code	Énoncé de justification
J_1	Deux droites perpendiculaires à une même troisième sont parallèles entre elles
J_2	La somme S des mesures des angles intérieurs d'un polygone à n côtés est donnée par la formule : $S = (n - 2) \times 180^\circ$
J_3	Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre
J_4	Un parallélogramme est un quadrilatère dont les côtés opposés sont parallèles
J_5	Un quadrilatère dont les angles opposés sont congrus est un parallélogramme
J_6	Les angles opposés d'un parallélogramme sont congrus
J_7	Un rectangle est un quadrilatère ayant quatre angles droits
J_8	Un parallélogramme ayant un angle droit est un rectangle

On remarque que le texte a une structure fixe à l'intérieur de chaque paragraphe, que l'on nomme *inférence*. Le tableau 4.3 montre la décomposition de la deuxième inférence de la preuve du rectangle proposée en ses différents éléments. Chaque inférence contient une liste d'antécédents, soit des énoncés qui représentent des résultats intermédiaires, une justification et un conséquent, qui est, comme les antécédents, un résultat intermédiaire. Un conséquent peut être utilisé comme antécédent d'une inférence subséquente et c'est cet enchaînement qui permet de construire une démonstration.

Tableau 4.3 Éléments d'une inférence

Antécédents	BC est perpendiculaire à CD
	AB est parallèle à CD
Justification	Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre
Conséquent	AB est perpendiculaire à BC

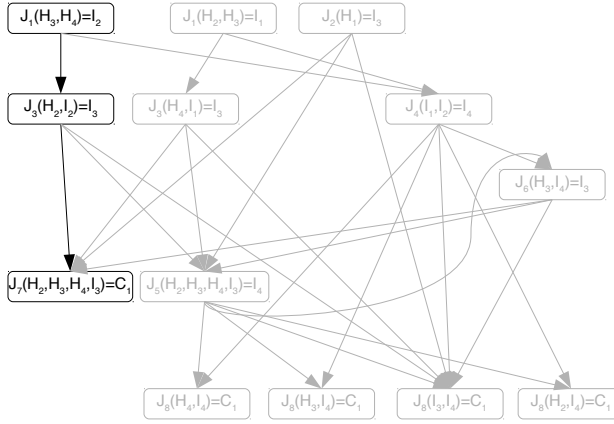
Une deuxième forme de représentation commune est la preuve en 2 colonnes, telle que présentée à la figure 4.2b. Chaque ligne de ce type de preuve représente habituellement une inférence. Certains systèmes tutoriels intelligents (Matsuda et VanLehn, 2005; Aleven et al., 2001b; Wong et al., 2011) utilisent cette représentation. Celle-ci est plus compacte que la preuve textuelle, mais elle perd de la clarté, car les liens entre les différents résultats sont moins apparents. Elle présente aussi la même lacune que la preuve textuelle, soit qu'il est laborieux de représenter toutes les preuves possibles pour un problème de manière compacte.

- Comme **AB est perpendiculaire à AD**, et **AD est perpendiculaire à CD**, alors, selon la propriété qui dit «**Deux droites perpendiculaires à une même troisième sont parallèles entre elles**», **AB est parallèle à CD**.
- Comme **BC est perpendiculaire à CD**, et **AB est parallèle à CD**, alors, selon la propriété qui dit «**Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre**», **AB est perpendiculaire à BC**.
- Comme **AB est perpendiculaire à AD**, **AD est perpendiculaire à CD**, **BC est perpendiculaire à CD**, et **AB est perpendiculaire à BC**, alors, selon la propriété qui dit «**Un rectangle est un quadrilatère ayant quatre angles droits**», **ABCD est un rectangle**.

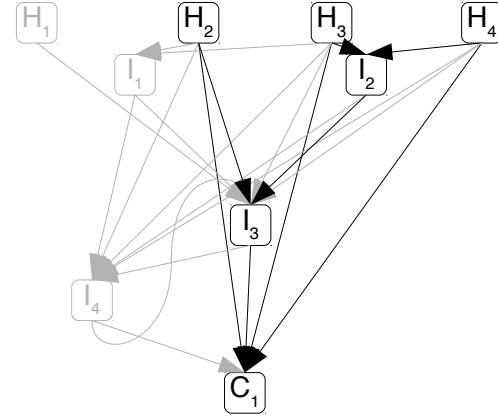
Affirmation	Justification
$AB \perp AD$	Par hypothèse
$AD \perp CD$	Par hypothèse
$AB \parallel CD$	Deux droites perpendiculaires à une même troisième sont parallèles entre elles
$BC \perp CD$	Par hypothèse
$AB \perp BC$	Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre
ABCD est un rectangle	Un rectangle est un quadrilatère ayant quatre angles droits

(a) Rédaction

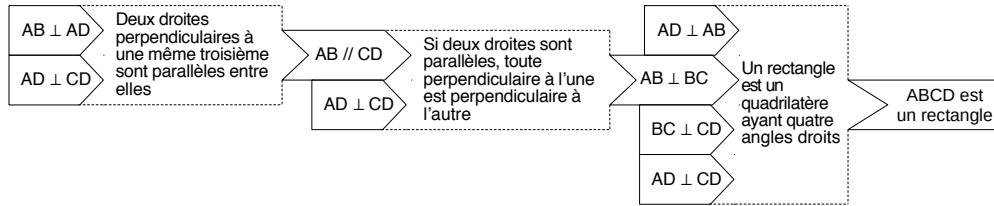
(b) Deux colonnes



(c) Graphe inférentiel



(d) Graphe de résultats



(e) Casse-tête

Figure 4.2 Représentations usuelles de la démonstration

Une autre stratégie pour rédiger une preuve consiste à la représenter sous forme de graphe. Ceux-ci sont souvent utilisés pour planifier la preuve avant de produire un texte final, mais il arrive que le graphe soit le résultat attendu. Le graphe inférentiel (figure 4.2c) contient une inférence par noeud et chaque flèche relie deux inférences dans lesquelles le conséquent de l'une est utilisé comme antécédent dans l'autre. Dans ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991), dans Geometry Tutor (Anderson et al., 1985; Ritter et al., 2010) et dans les travaux de Tadinada (2011), le graphe de résultats (figure 4.2d), à l'intérieur duquel chaque noeud représente un résultat intermédiaire, a été utilisé. Dans ce cas, l'ensemble des inférences permettant de produire ce résultat est implicite dans le noeud. Les flèches ont, en gros, la même signification que celles du graphe inférentiel. Pour les deux graphes, nous avons présenté

la solution proposée en noir. De plus, nous avons ajouté, en gris, l'ensemble des solutions admissibles pour le problème du rectangle. Il est donc possible de représenter l'ensemble des solutions de manière compacte. Par contre, il est difficile d'identifier précisément à quelle inférence appartient chaque flèche, comme dans le cas de I_3 , car chaque inférence permettant de produire un résultat utilise un sous-ensemble d'antécédents unique, mais non exclusif. Le traitement adéquat de ce type de graphe pour représenter plus d'une solution à la fois est donc complexe et délicat.

Des logiciels comme Géométrix (Gressier, 2015) et Mentoniez (Py, 1994, 1996, 2001) proposent une structure en tableau pour l'élaboration de la preuve. Dans ceux-ci, on retrouve habituellement une colonne contenant les hypothèses, une deuxième contenant les éléments prouvés et une dernière contenant les résultats qui restent à prouver. Au départ, la conclusion du problème se trouve dans la dernière colonne et le but de l'élève est de trouver une stratégie pour l'amener dans la colonne des résultats prouvés. Cette représentation est intéressante pour l'élaboration d'une stratégie de preuve, mais ne constitue pas une preuve, car elle est en fait une preuve en deux colonnes dans laquelle on aurait omis les justifications. Ces dernières sont tout de même exigées par le logiciel lorsque l'élève veut prouver un énoncé.

La majorité des systèmes cités jusqu'à présent, peu importe le style de preuve exigé, utilisent un système formel pour produire les différentes solutions possibles pour le problème posé. On y implante les contraintes et les règles (inférences) acceptables en géométrie euclidienne et le système est en mesure de combiner les inférences pour produire une solution, si elle existe dans la réalité représentée. En théorie, ces systèmes permettent de produire l'ensemble des solutions possibles pour un problème donné en ne fournissant que les hypothèses et la conclusion. Par contre, il est difficile d'ajouter des exceptions ou de changer le niveau de granularité sans mettre en péril la cohérence même du système d'inférences. Nous voulons, par ailleurs, que QEDX puisse être facilement adapté à différents contextes d'enseignement et que les enseignants puissent ajuster le niveau d'exigence du système. Par exemple, il peut s'avérer utile de traiter et d'accepter une démonstration menant à une contradiction ou acceptable de justifier un résultat en utilisant une observation sur la figure. Certains enseignants acceptent même des preuves figurales (Richard, 2004a), soit des preuves construites uniquement par des manipulations pertinentes de la figure. Il n'est toutefois pas exclu de proposer un moteur formel pour que QEDX puisse éventuellement accepter des solutions qui n'auraient pas été produites par l'expert.

Enfin, pour initier les élèves à la démonstration, la preuve sous forme de casse-tête (figure 4.2e) peut être proposée comme activité en classe. Ce type de représentation a pour but de bien montrer les liens entre les différents éléments au sein d'une inférence et ceux entre ces

dernières. Malgré son potentiel didactique, cette représentation n'a pas été observée dans les logiciels analysés. Elle est, en fait, équivalente au graphe de résultats. Ce dernier est par contre plus compact et permet de représenter un ensemble de solutions plus facilement.

Les différentes représentations des preuves présentées dans cette section ont des lacunes qui rendent leur utilisation difficile au sein d'un logiciel comme QEDX. Soit il est difficile de représenter l'ensemble des solutions d'un problème, soit le système est trop rigide pour l'adapter facilement à différents contextes d'enseignement. Or, notre but est de proposer une structure qui pourrait permettre de produire la majorité des formes de preuves présentées. Elle devra aussi permettre de modéliser l'apprenant pour lui proposer des pistes de solution utiles. L'approche que nous avons élaborée est présentée en détail à la section 4.3.

4.2.3 Revue sommaire des modèles de l'apprenant

Pour être en mesure d'aider l'élève lors de la résolution d'un problème de démonstration, il faut, en plus d'une structure de données qui permet de représenter les différentes solutions acceptables, un modèle de l'apprenant qui permet de rendre compte de sa progression. En effet, nous ne voulons pas contraindre l'élève à produire la démonstration optimale, mais bien lui offrir différentes pistes, comme le ferait un enseignant. Dans les articles consultés, cinq principales stratégies ont été proposées.

Une première stratégie consiste à utiliser un logiciel de création de systèmes tutoriels, par exemple CTAT (Aleven et al., 2009; Koedinger et al., 2004; Carnegie Mellon University, 2015) ou ASTUS (Paquette et al., 2012, 2015; Paquette, 2013). Ceux-ci doivent être programmés à l'aide de règles de productions formelles reliées au domaine. Pour un problème donné, on indique ensuite les règles de production que l'élève doit utiliser pour le résoudre. Lorsque le système détecte que l'élève n'applique pas une règle prédéfinie, il lui envoie un message et peut lui fournir des indices. Comme les règles sont génériques, les messages associés n'ont pas à être créés pour chaque situation, car des règles pour produire les messages sont utilisées. Cependant, l'utilisation d'un tel logiciel pour la création de QEDX posait deux problèmes majeurs. Premièrement, le système de règles de production formelles nous empêchait d'avoir la flexibilité nécessaire pour adapter QEDX aux différents styles d'enseignement. De plus, le logiciel s'attend à ce que les actions de l'élève soient exécutées dans un ordre précis, comme pour résoudre un problème de physique ou de calcul. Dans le cadre de l'élaboration d'une démonstration, nous n'accordons pas d'importance à l'ordre d'écriture des énoncés, ce qui rend l'utilisation de tels logiciels difficile pour produire les rétroactions voulues.

En ce qui concerne les systèmes tutoriels en géométrie, par exemple Geometry Tutor (Anderson et al., 1985; Ritter et al., 2010), Advanced Geometry Tutor (Matsuda et VanLehn,

2005; Matsuda, 2004), Géométrie (Gressier, 2015), ainsi qu'Andes (Conati et al., 2002; Van-Lehn et al., 2005, 2010), un système tutoriel en physique, ils utilisent le chaînage avant ou arrière pour guider l'élève. Dans le chaînage avant, on part des hypothèses pour en déduire la conclusion alors que le chaînage arrière consiste à parcourir la solution à rebours. Même si le logiciel accepte plusieurs solutions au problème posé, il s'attend à ce que l'élève produise les énoncés dans un ordre déterminé. Il peut même refuser un énoncé valide s'il n'est pas proposé au bon moment. Cette structure est relativement facile à programmer, car le choix d'une solution par l'élève se fait à quelques moments clés et il lui est impossible de changer de stratégie s'il se retrouve bloqué. Le logiciel n'a donc qu'à se rappeler les énoncés produits par l'élève, sans égard à la chronologie de ceux-ci, pour lui proposer la prochaine étape. De plus, le logiciel lui fournit de l'aide seulement pour l'étape qu'il juge la plus pertinente dans la situation. Ce genre de stratégie d'aide est efficace en ce qui concerne sa programmation, mais ne permet pas à l'élève d'explorer librement le problème, comme il est prévu dans QEDX.

Dans le cas de Mentoniez (Py, 1994, 1996, 2001), le système fournit une rétroaction pour les inférences complètes soumises par l'élève et n'offre pas de support pour proposer l'étape suivante. L'inférence soumise est comparée à celles qui sont utiles pour résoudre le problème et l'on fournit un message pour indiquer quelles sont les erreurs probables par rapport à l'inférence apparemment ciblée par l'élève. Pour proposer des correctifs plus adaptés aux situations, une reconnaissance de plan a été ajoutée pour identifier la solution sur laquelle l'élève travaille probablement au moment où l'erreur s'est produite. Même si Mentoniez implante l'exploration du problème correctement, son système d'aide ne ressemble pas à ce que nous voulons implanter dans QEDX, car nous voulons proposer un cheminement utile pour relancer l'élève dans la résolution.

Dans le système ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991), au lieu de modéliser le processus de résolution de l'apprenant, les auteurs ont tenté de modéliser la stratégie de résolution de l'expert. Ils ont remarqué que les experts ont en tête des figures remarquables qui permettent de synthétiser un ensemble de connaissances par rapport au problème et ont nommé ce concept «Diagram Configuration (DC)» (Koedinger et Anderson, 1990). Le logiciel a donc pour but de faire apprendre les différents DC identifiés aux élèves. L'exploration du problème est laissée libre, mais si l'élève demande un indice, l'aide est dirigée vers l'identification et l'utilisation des DC en chaînage avant. Dans le cadre de QEDX, nous ne tentons pas de montrer aux élèves comment raisonner comme des experts, car nous voulons laisser le soin aux élèves d'élaborer leurs propres stratégies, comme le fait un enseignant.

Une approche prometteuse, qui a été appliquée à plusieurs domaines, consiste à utiliser différentes stratégies d'exploration de données. Deep Thought (Barnes et Stamper, 2010;

Mostafavi et al., 2011; Stamper, 2010) est le premier et est actuellement le seul système à utiliser une telle approche pour l'apprentissage des preuves, mais il se limite aux preuves en logique de premier ordre. Dans celui-ci, la solution d'un élève consiste en une série d'états dont chacun contient l'ensemble des énoncés valides inscrits par l'élève depuis le début du problème jusqu'au moment correspondant à cet état. Deux états consécutifs sont reliés par une transition sur laquelle on indique l'énoncé qui permet de passer d'un état à l'autre. Toutes les solutions sont combinées en un seul graphe sur lequel on calcule différentes statistiques concernant l'utilisation des états et des transitions afin d'offrir une aide adaptée au cheminement de l'élève. Cette approche est intéressante, mais est difficilement applicable au projet actuel, car nous n'avons pas de banques de solutions pour les problèmes proposés. Cependant, la génération de fichiers journaux, par QEDX, permettra de créer une telle banque de données. Celle-ci pourra être utilisée pour améliorer le logiciel ou pour alimenter d'autres projets de recherche.

Les différentes modélisations de l'apprenant que nous avons recensées ne peuvent donc pas être utilisées directement dans QEDX. Soit elles sont trop contraignantes, soit elles nécessitent des données que nous n'avons pas. De plus, il faut que la modélisation que nous allons proposer s'intègre à la structure de donnée permettant de représenter l'ensemble des solutions pour un problème donné. Dans la section suivante, nous décrivons les graphes HPDIC et le MIA, qui se veulent une solution pour permettre à QEDX de modéliser les interventions d'un enseignant.

4.3 Définitions formelles du HPDIC et du MIA

Comme notre objectif est de proposer un système qui est en mesure d'aider l'élève dans une démarche d'exploration en lui proposant notamment différentes pistes de solution dans l'éventualité d'un blocage et qu'aucun système recensé n'a cette capacité, nous avons développé QEDX. Cette section présente les structures de base de QEDX, soit les graphes HPDIC, qui permettent de modéliser l'ensemble des solutions d'un problème, et le MIA, qui modélise la chronologie des états cognitifs de l'élève. La description formelle est présentée et nous utilisons les données provenant du problème du rectangle pour illustrer chacun des concepts.

4.3.1 Graphe HPDIC

Un graphe HPDIC (**H**ypothèse, **P**ropriété, **D**éfinition, résultat **I**ntermédiaire et **C**onclusion) est une structure qui contient l'ensemble des inférences valides pour un problème ainsi que les liens qui existent entre elles. Pour le construire, il suffit de transformer chaque inférence sous forme d'arbre, comme on peut le voir à la figure 4.3, qui correspond à la deuxième inférence

de la solution proposée à la figure 4.2. Par la suite, on rassemble les noeuds contenant des résultats intermédiaires identiques en un seul noeud, tout en préservant leurs liens, pour obtenir le graphe HPDIC. En ce qui concerne les noeuds de justification, ceux-ci forment le pivot de chaque inférence et sont donc considérés comme uniques même s'ils contiennent des justifications identiques. Par conséquent, nous ne fusionnons pas les noeuds de justification semblables. Le graphe comprend donc autant de noeuds de justification qu'il y a d'inférences valides pour le problème. Il suffit ensuite de parcourir le graphe pour retrouver l'ensemble des démonstrations possibles.

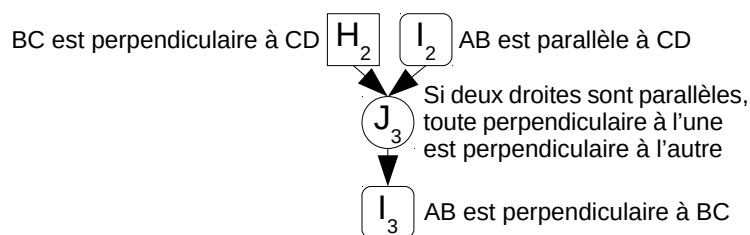


Figure 4.3 Transformation d'une inférence en arbre HPDIC

À titre d'exemple, la figure 4.4 représente le graphe HPDIC pour le problème du rectangle. Dans celui-ci, on voit que I_3 peut être produit par quatre inférences distinctes (flèches entrantes) et qu'il est utilisé comme antécédent dans trois inférences (flèches sortantes). Nous avons utilisé des couleurs distinctes pour les différentes composantes du graphe afin de bien montrer sa structure. On peut voir les hypothèses au sommet et la conclusion à la base. De plus, on remarque l'alternance entre les résultats intermédiaires et les justifications, ce qui respecte la structure de l'inférence. Les noeuds colorés en rouge permettent de produire la démonstration présentée à la figure 4.2.

Le graphe HPDIC est semblable aux réseaux déductifs de Tanguay (2005, 2007) en ce qui concerne leur structure. En effet, dans les deux cas, tous les noeuds, y compris les justifications, sont présents et visibles, ce qui n'est pas le cas dans les graphes de résultats (figure 4.2d). L'ajout des noeuds de justification permet de bien voir la structure des inférences et ainsi de clarifier la signification des liens entre les noeuds. Les flèches qui entrent dans un noeud de justification correspondent aux antécédents de l'inférence et ces derniers sont tous nécessaires pour obtenir une inférence complète. Par contre, si plusieurs flèches entrent dans un noeud de résultat intermédiaire, cela signifie que plusieurs inférences permettent d'obtenir ce résultat, mais on n'en choisira qu'une seule lors de la construction d'une solution.

Malgré cette structure semblable, l'objectif qui a permis d'obtenir ces graphes est différent. Dans le cas de Tanguay, l'objectif était de proposer un outil pour permettre aux élèves de

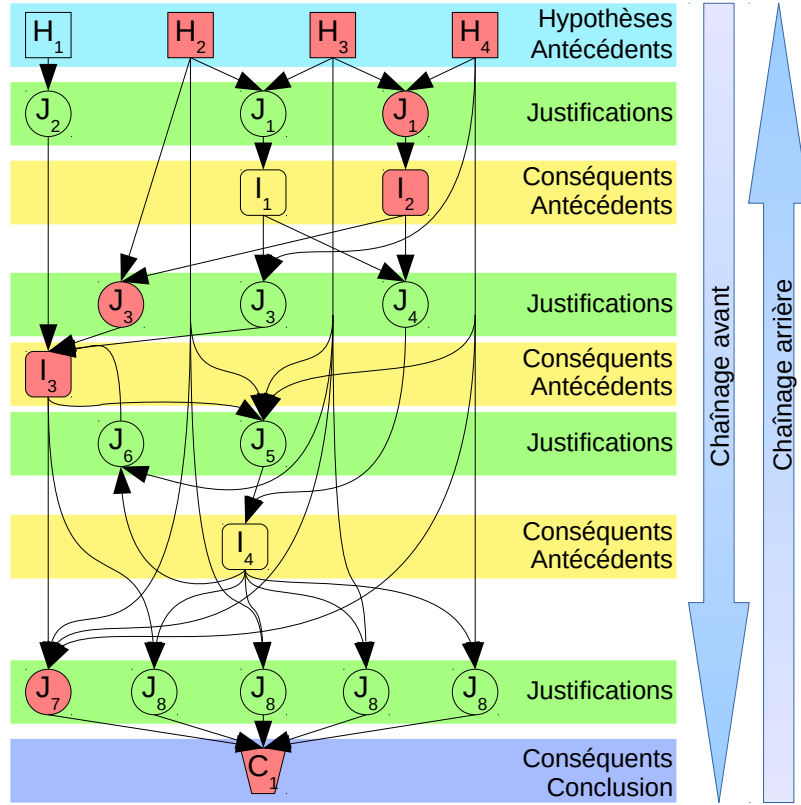


Figure 4.4 Graphe HPDIC pour le problème du rectangle

construire leur démonstration. Il s'agit donc d'un mécanisme d'organisation de la pensée avant de passer à la rédaction de la preuve proprement dite. Dans notre cas, le graphe devait à la fois permettre de représenter l'ensemble des preuves d'un problème et fournir une structure pour modéliser l'apprenant. Il est rassurant de savoir qu'un graphe semblable au graphe HPDIC est déjà utilisé dans un contexte d'enseignement, car cela confirme sa valeur pédagogique en plus d'être une structure informatique essentielle au développement de QEDX.

Définition formelle du graphe

De manière formelle, nous définissons premièrement l'ensemble des énoncés valides en géométrie euclidienne :

$$\mathcal{R} : \text{Ensemble des énoncés de résultats} \quad (4.1)$$

(Hypothèses, résultats Intermédiaires, Conclusions)

$$\mathcal{E} : \text{Ensemble des énoncés de justification} \quad (4.2)$$

(Propriétés, Définitions)

Dans la version actuelle de QEDX, il y a 74 énoncés de résultats intermédiaires et 170 justifications qui sont issus du volet géométrie du programme du Ministère de l'Éducation concernant l'enseignement des mathématiques au secondaire (Ministère de l'Éducation, du Loisir et du Sport, 2009). En combinant ceux-ci, il est possible de construire un ensemble infini d'inférences. Cependant, seul un sous-ensemble de toutes les inférences possibles nous permettra de résoudre un problème ciblé :

$$\mathbf{I} \subseteq \{\langle \mathcal{A}, J, C \rangle \mid \mathcal{A} \subseteq \mathcal{R}, J \in \mathcal{E}, C \in \mathcal{R}\} \quad (4.3)$$

Pour chacun des tuples, $\mathcal{A}$ représente l'ensemble des antécédents, J , la justification et C , le conséquent d'une inférence donnée. Dans le cas du problème du rectangle, les didacticiens ont cerné 13 inférences qui sont inscrites ci-dessous en utilisant l'encodage des tableaux 4.1 et 4.2. De plus, entre parenthèses, nous avons donné un nom à chacune afin de les identifier dans la suite du chapitre.

$\langle \{H_2, H_3\}, J_1, I_1 \rangle$	(λ_{1a})	$\langle \{H_3, I_4\}, J_6, I_3 \rangle$	(λ_6)
$\langle \{H_3, H_4\}, J_1, I_2 \rangle$	(λ_{1b})	$\langle \{H_2, H_3, H_4, I_3\}, J_7, C_1 \rangle$	(λ_7)
$\langle \{H_1\}, J_2, I_3 \rangle$	(λ_2)	$\langle \{I_3, I_4\}, J_8, C_1 \rangle$	(λ_{8a})
$\langle \{H_2, I_2\}, J_3, I_3 \rangle$	(λ_{3a})	$\langle \{H_2, I_4\}, J_8, C_1 \rangle$	(λ_{8b})
$\langle \{H_4, I_1\}, J_3, I_3 \rangle$	(λ_{3b})	$\langle \{H_3, I_4\}, J_8, C_1 \rangle$	(λ_{8c})
$\langle \{I_1, I_2\}, J_4, I_4 \rangle$	(λ_4)	$\langle \{H_4, I_4\}, J_8, C_1 \rangle$	(λ_{8d})
$\langle \{H_2, H_3, H_4, I_3\}, J_5, I_4 \rangle$	(λ_5)		

En transformant l'arbre de l'inférence de la figure 4.3 sous forme de tuple, nous obtenons $\langle \{H_2, I_2\}, J_3, I_3 \rangle$, ce qui correspond à l'inférence λ_{3a} de notre liste. Il est donc possible de déduire que la démonstration proposée à la figure 4.2 utilise respectivement les inférences λ_{1b} , λ_{3a} et λ_7 .

L'identification de l'ensemble $\mathbf{I}$ est cruciale pour le bon fonctionnement du système, car toutes les autres structures utilisent ces inférences. Cependant, aucune contrainte formelle en ce qui concerne leur cohérence n'est imposée, ce qui permet d'ajuster les solutions selon les besoins des enseignants. Par exemple, certains enseignants peuvent accepter un énoncé qui découle d'une expérimentation avec la figure. D'autres, selon le stade de l'apprentissage, pourraient permettre des démonstrations pouvant contenir une certaine part d'argumentation ou des raccourcis inférentiels. Le travail préalable à l'utilisation du logiciel en classe consiste donc à fournir un ensemble d'inférences valides pour le problème et adaptées au style de preuve attendu de la part des élèves.

À partir de l'ensemble $\mathbf{I}$, il nous est possible de retrouver l'ensemble des énoncés de résultats intermédiaires (R) ainsi que l'ensemble des énoncés de justification (E) utilisés dans le cadre du problème posé. Ils forment des sous-ensembles de $\mathcal{R}$ et $\mathcal{E}$ respectivement.

$$R = \bigcup \{ \mathcal{A} \mid \langle \mathcal{A}, J, C \rangle \in \mathbf{I} \} \cup \{ C \mid \langle \mathcal{A}, J, C \rangle \in \mathbf{I} \} \quad (4.4)$$

$$E = \{ J \mid \langle \mathcal{A}, J, C \rangle \in \mathbf{I} \} \quad (4.5)$$

Les tableaux 4.1 et 4.2 contiennent les énoncés qui correspondent à chacun de ces deux ensembles pour le problème du rectangle.

Parmi les résultats intermédiaires, certains jouent un rôle particulier à l'intérieur d'un problème. En effet, en considérant l'ensemble $\mathbf{I}$, on peut trouver des résultats intermédiaires qui ne sont le conséquent d'aucune inférence et que nous nommons les hypothèses du problème (ensemble *Init*). Nous appelons *conclusion* (ensemble *Final*), l'ensemble des résultats intermédiaires n'apparaissant comme antécédent d'aucune inférence.

$$Init = \{ a \mid a \in R \wedge \neg \exists \langle \mathcal{A}, J, a \rangle \in \mathbf{I} \} \quad (4.6)$$

$$Final = \{ c \mid c \in R \wedge (\neg \exists \langle \mathcal{A}, J, C \rangle \in \mathbf{I} \mid c \in \mathcal{A}) \} \quad (4.7)$$

À l'intérieur du tableau 4.1, le code des hypothèses débute par H , alors que le code des conclusions débute par C . Parmi les 9 résultats intermédiaires utiles pour résoudre le problème du rectangle, 4 sont des hypothèses et nous avons 1 seule conclusion possible.

L'ensemble $\mathbf{S}$ permet finalement de décrire la structure du graphe HPDIC en définissant un ensemble de couples d'inférences qui sont liés. Un lien est créé lorsqu'un conséquent d'une première inférence est utilisé comme antécédent pour une autre. C'est cette structure qui va nous permettre de trouver l'ensemble des démonstrations possibles pour un problème.

$$\mathbf{S} = \{ \langle \langle \mathcal{A}_1, J_1, C_1 \rangle, \langle \mathcal{A}_2, J_2, C_2 \rangle \rangle \mid C_1 \in \mathcal{A}_2 \wedge \langle \mathcal{A}_1, J_1, C_1 \rangle \in \mathbf{I} \wedge \langle \mathcal{A}_2, J_2, C_2 \rangle \in \mathbf{I} \} \quad (4.8)$$

Dans le cas du problème du rectangle, on trouve l'ensemble suivant :

$$\begin{aligned} \mathbf{S} = \{ & \langle \lambda_{1a}, \lambda_{3b} \rangle, \langle \lambda_{1a}, \lambda_4 \rangle, \langle \lambda_{1b}, \lambda_{3a} \rangle, \langle \lambda_{1b}, \lambda_4 \rangle, \langle \lambda_2, \lambda_5 \rangle, \langle \lambda_2, \lambda_7 \rangle, \langle \lambda_2, \lambda_{8a} \rangle, \langle \lambda_{3a}, \lambda_5 \rangle, \langle \lambda_{3a}, \lambda_7 \rangle, \\ & \langle \lambda_{3a}, \lambda_{8a} \rangle, \langle \lambda_{3b}, \lambda_5 \rangle, \langle \lambda_{3b}, \lambda_7 \rangle, \langle \lambda_{3b}, \lambda_{8a} \rangle, \langle \lambda_4, \lambda_6 \rangle, \langle \lambda_4, \lambda_{8a} \rangle, \langle \lambda_4, \lambda_{8b} \rangle, \langle \lambda_4, \lambda_{8c} \rangle, \langle \lambda_4, \lambda_{8d} \rangle, \\ & \langle \lambda_5, \lambda_6 \rangle, \langle \lambda_5, \lambda_{8a} \rangle, \langle \lambda_5, \lambda_{8b} \rangle, \langle \lambda_5, \lambda_{8c} \rangle, \langle \lambda_5, \lambda_{8d} \rangle, \langle \lambda_6, \lambda_5 \rangle, \langle \lambda_6, \lambda_7 \rangle, \langle \lambda_6, \lambda_{8a} \rangle \} \end{aligned}$$

Sachant que Sol est un ensemble de solutions, telles que calculées en utilisant l'algorithme décrit ci-après, le graphe $HPDIC$ correspondant à un problème spécifique peut alors être défini de la manière suivante :

$$HPDIC = \langle \mathbf{I}, \mathbf{S}, R, E, Init, Final, Sol \rangle \quad (4.9)$$

Extraction des solutions

Afin de compléter la définition du graphe HPDIC, il ne nous reste qu'à proposer l'algorithme qui permet de calculer Sol , qui représente l'ensemble des solutions contenues dans le graphe. Pour les obtenir, on parcourt le graphe en chaînage arrière, soit de la conclusion vers les hypothèses. Une solution particulière est produite en choisissant, pour chaque résultat intermédiaire visité, une seule inférence (noeud de justification) permettant de l'engendrer. Par contre, pour chaque justification, tous les antécédents sont nécessaires pour obtenir une inférence complète et devront donc être visités. Afin d'obtenir l'ensemble des solutions, nous dupliquons les chemins, lors de la visite d'un noeud de résultat intermédiaire. Nous créons autant de duplicatas qu'il y a d'inférences qui permettent de produire celui-ci et nous traitons ensuite individuellement chacun des chemins.

Afin de présenter l'algorithme permettant d'énumérer les solutions de manière formelle, nous avons besoin de la notion de **Parents**. Un parent est en fait une inférence dont le conséquent correspond au résultat intermédiaire r .

$$\mathbf{Parents}(r) = \{ \langle \mathcal{A}, J, r \rangle \mid \langle \mathcal{A}, J, r \rangle \in \mathbf{I} \} \quad (4.10)$$

L'algorithme 4.1 permet de calculer l'ensemble des solutions et produit l'élément Sol de la définition du graphe HPDIC. Dans celui-ci, nous utilisons deux ensembles principaux, soit Sol qui contient des solutions complètes et CH qui contient celles partielles que nous devons traiter. Une solution est dite partielle si au moins un résultat intermédiaire en faisant partie n'est pas prouvé. Ce cas se produit lorsqu'un résultat n'est pas dans l'ensemble $Init$ et qu'il n'est le conséquent d'aucune inférence incluse dans la solution. Le rôle de la fonction **EXPAND** est de détecter les solutions partielles et de leur ajouter une inférence en espérant la compléter. Si plusieurs inférences (**Parents**) produisent le résultat intermédiaire ciblé, nous créons autant de copies du chemin que nécessaire pour que chaque inférence soit ajoutée à un chemin distinct. La fonction **EXTRAITSOLUTIONS** fournit les solutions partielles à **EXPAND** jusqu'à ce que toutes les solutions soient complètes. On initialise l'algorithme en créant un chemin partiel pour chaque inférence ayant une conclusion ($Final$) comme conséquent.

```

fonction EXTRAITSOLUTIONS
   $Sol \leftarrow \{\}$ 
   $CH \leftarrow \{\langle \mathcal{A}, J, c \rangle \mid \langle \mathcal{A}, J, c \rangle \in \mathbf{I} \wedge c \in Final\}$ 
  tant que  $CH \neq \emptyset$  faire
    POP(CH, Chemin)
     $EXP \leftarrow EXPAND(Chemin)$ 
    si  $EXP \neq \emptyset$  alors
       $CH \leftarrow CH \cup EXP$ 
    sinon
       $Sol \leftarrow Sol \cup Chemin$ 
    fin si
  fin tant que
  retourne  $Sol$ 
fin fonction

fonction EXPAND(Chemin)
  Résultat  $\leftarrow \{\}$ 
  pour tout  $\langle \mathcal{A}, J, C \rangle \in Chemin$  faire
    pour tout  $a \in \mathcal{A}$  faire
      si  $a \notin Init \wedge \neg \exists \langle \mathcal{A}', J', a \rangle \in Chemin$  alors
        pour tout  $p \in Parents(a)$  faire
          Résultat  $\leftarrow Résultat \cup \{Chemin \cup p\}$ 
        fin pour
      retourne Résultat
    fin si
  fin pour
  retourne  $\emptyset$ 
fin fonction

```

Algorithme 4.1 Énumération de toutes les solutions du graphe HPDIC

Dans le cas où le graphe HPDIC ne contiendrait pas de boucles, soit un arbre HPDIC, l'ensemble des solutions trouvé est valide. Une solution est dite invalide s'il est impossible d'en prouver la conclusion à l'aide des inférences faisant partie de celle-ci. Pour qu'un conséquent soit considéré comme étant prouvé, il faut que l'ensemble des antécédents faisant partie de l'inférence soit prouvé. Un antécédent est considéré comme étant prouvé s'il fait partie des hypothèses ou s'il est un conséquent prouvé. On peut donc vérifier la validité d'une solution en utilisant le chaînage avant. L'algorithme 4.2 permet de valider les solutions en vérifiant si la conclusion peut être prouvée. Dans celui-ci, la fonction ESTVALIDE tente de prouver un maximum d'inférences faisant partie de la solution reçue en paramètre. Pour ce faire, tant que l'on est en mesure de prouver de nouvelles inférences, on vérifie l'ensemble de celles qui ne sont pas encore prouvées. Par ailleurs, la fonction PROUVABLE sert à vérifier si une inférence peut être prouvée à partir des inférences déjà prouvées et des hypothèses du problème. Lorsqu'une itération ne permet pas de prouver de nouvelles inférences, on vérifie si la conclusion fait partie de l'ensemble des inférences prouvées, ce qui permet de savoir si la solution est valide.

```

fonction ESTVALIDE(Chemin)
  Prouvé  $\leftarrow \{\}$ 
  répéter
    ProuvéAjouté  $\leftarrow Faux$ 
    pour tout Inf  $\in$  Chemin faire
      si Inf  $\notin$  Prouvé  $\wedge$  PROUVABLE(Inf, Prouvé) alors
        Prouvé  $\leftarrow$  Prouvé  $\cup$  Inf
        ProuvéAjouté  $\leftarrow Vrai$ 
      fin si
    fin pour
  jusqu'à  $\neg$ ProuvéAjouté
  retourne  $\exists\{\langle \mathcal{A}, J, c \rangle \mid \langle \mathcal{A}, J, c \rangle \in \text{Prouvé} \wedge c \in Final\}$ 
fin fonction

fonction PROUVABLE( $\langle \mathcal{A}, J, C \rangle$ , Prouvé)
  pour tout  $a \in \mathcal{A}$  faire
    si  $a \notin Init \wedge \neg \exists \langle \mathcal{A}', J', a \rangle \in \text{Prouvé}$  alors
      retourne Faux
    fin si
  fin pour
  retourne Vrai
fin fonction

```

Algorithme 4.2 Validation d'une solution afin de s'assurer qu'elle est complète

En appliquant l'algorithme d'énumération des chemins au graphe HPDIC du problème du rectangle, nous obtenons l'ensemble des solutions présentées dans le tableau 4.4. Chaque ligne du tableau correspond à une solution identifiée par l'algorithme 4.1. Chaque colonne indique la présence d'une inférence, identifiée par son code unique, dans la solution.

Dans cet exemple, l'algorithme a trouvé 28 chemins différents, dont 23 valides (code débutant par S) et 5 invalides (code débutant par L). En effet, le graphe HPDIC de la figure 4.4 comporte une boucle qui contient 2 inférences, soit λ_5 et λ_6 . Lorsque l'algorithme 4.1 énumère les chemins, il trouve des solutions dans lesquelles J_5 permet de justifier un antécédent utilisé par J_6 , en l'occurrence I_4 , et vice versa dans le cas de I_3 . Il s'arrête donc, car tous les antécédents sont soit des conséquents d'autres inférences ou des hypothèses. Cependant, si nous parcourons le chemin en chaînage avant, à l'aide de l'algorithme 4.2, nous ne sommes pas en mesure de justifier I_3 ou I_4 uniquement à l'aide d'hypothèses ou d'inférences prouvées, car λ_5 et λ_6 se justifient mutuellement. Dans ce cas, il est impossible de produire une démonstration valide. On peut donc voir, dans le tableau 4.4, que toutes les solutions invalides contiennent à la fois λ_5 et λ_6 .

Nous avons observé, lors des deux expérimentations en classe (voir chapitre 3), que les élèves utilisent généralement les justifications afin de composer leur stratégie de résolution. Ils ne travaillent donc pas sur l'ensemble des solutions, mais ciblent plutôt un sous-ensemble en

Tableau 4.4 Ensemble des solutions au problème du rectangle

Sol	λ_{1a}	λ_{1b}	λ_2	λ_{3a}	λ_{3b}	λ_4	λ_5	λ_6	λ_7	λ_{8a}	λ_{8b}	λ_{8c}	λ_{8d}
S01	X	X	X			X				X			
S02	X	X		X		X				X			
S03	X	X			X	X				X			
S04	X				X		X			X			
S05	X				X		X				X		
S06	X				X		X					X	
S07	X				X		X						X
S08		X		X			X			X			
S09		X		X			X				X		
S10		X		X			X					X	
S11		X		X			X						X
S12	X				X				X				
S13		X		X					X				
S14	X	X				X		X	X				
S15	X	X				X		X		X			
S16	X	X				X					X		
S17	X	X				X						X	
S18	X	X				X							X
S19			X				X			X			
S20			X				X				X		
S21			X				X					X	
S22			X				X						X
S23			X						X				
L01							X	X	X				
L02							X	X		X			
L03							X	X			X		
L04							X	X				X	
L05							X	X					X

fonction des concepts qu'ils maîtrisent et qu'ils croient utiles à la résolution. À la manière des élèves, nous avons proposé de regrouper les solutions en différentes stratégies afin d'en isoler un sous-ensemble minimal qui sera traité par les couches supérieures de QEDX. Cette optimisation peut être utile, entre autres, pour traiter des problèmes admettant des millions de solutions. Par conséquent, dans le cadre de nos travaux, une stratégie regroupe toutes les solutions utilisant un même ensemble d'énoncés de justification sans égard aux énoncés de résultats intermédiaires. Par exemple, les solutions $S12$ et $S13$ utilisent toutes les deux les justifications J_1 , J_3 et J_7 , donc elles correspondent à une même stratégie. Dans le tableau 4.4, les séparations horizontales servent à regrouper les solutions en 9 stratégies distinctes.

4.3.2 Modèle Itératif de l'Apprenant (MIA)

Le graphe HPDIC contient donc les informations nécessaires pour produire l'ensemble des solutions pour un problème donné. À partir de celui-ci, nous avons développé le Modèle Itératif

de l'Apprenant (MIA), qui permet de suivre le cheminement de l'élève lors de l'élaboration de sa preuve. Son rôle consiste à fournir des informations aux autres couches de QEDX dans le but d'offrir une aide personnalisée. Avant de donner la définition formelle du MIA, il est important de définir le concept d'ensemble des noeuds d'un graphe :

$$\mathbf{ND}(G) = \{n \mid \langle \mathcal{A}, J, C \rangle \in G \wedge (n \in \mathcal{A} \vee n = J \vee n = C)\} \quad (4.11)$$

Dans cette équation, G représente un graphe dont la structure est définie à l'aide d'un ensemble d'inférences, comme c'est le cas pour le graphe HPDIC ou pour une solution particulière issue de ce dernier. Les noeuds faisant partie de cet ensemble contiennent tous des énoncés distincts. En effet, chaque noeud de résultat intermédiaire est unique. Cependant, différents noeuds de justification peuvent représenter un énoncé commun, comme c'est le cas pour J_3 qui est utilisé dans les inférences λ_{3a} et λ_{3b} . Dans ce cas, une seule occurrence de cet énoncé fait partie de l'ensemble, car il est impossible de déterminer, dans la version actuelle de QEDX, le noeud précis visé par l'élève lors de l'écriture de l'énoncé.

Pour être en mesure de faire fonctionner le système d'aide de QEDX, le MIA proposé comporte deux structures, soit un historique des actions de l'élève (**TA**) et une estimation de l'achèvement de chacune des solutions possibles (**PA**).

$$\mathbf{MIA}(HPDIC) = \langle \mathbf{TA}(HPDIC), \mathbf{PA}(HPDIC) \rangle \quad (4.12)$$

Dans le but d'implanter l'historique des actions, nous avons décoré chaque noeud du graphe HPDIC afin d'y ajouter le temps d'activation de l'énoncé correspondant. Un noeud est dit activé lorsque l'élève a soumis l'énoncé correspondant à l'interface de QEDX.

$$\begin{aligned} \mathbf{TA}(HPDIC) &= \{\langle n, t \rangle \mid n \in \mathbf{ND}(HPDIC) \wedge t \in \{-1\} \cup [1.. \infty[\} \\ &\quad \neg \exists \langle n, t \rangle, \langle n, t' \rangle \mid \langle n, t \rangle \in \mathbf{TA}(HPDIC) \wedge \langle n, t' \rangle \in \mathbf{TA}(HPDIC) \wedge t < t' \} \end{aligned} \quad (4.13)$$

Nous utilisons un compteur entier croissant, remis à zéro à chaque démarrage de QEDX, comme valeur de temps d'activation. Le noeud ayant la valeur maximale contient donc la dernière phrase valide entrée par l'élève, alors que les noeuds contenant la valeur -1 n'ont pas encore été activés. Si l'élève entre plusieurs fois le même énoncé, seul le temps d'activation le plus récent sera enregistré. En effet, nous considérons que l'élève utilise cette répétition pour guider le tuteur et l'aider à reconnaître son état cognitif actuel.

Certains messages de QEDX prennent en compte, en plus de l'historique des actions, la progression globale de l'élève vers une solution complète. Pour chaque solution possible ($s \in Sol$) du graphe HPDIC, nous calculons le pourcentage d'avancement de celle-ci :

$$\mathbf{NDACT}(s) = \{n \mid n \in \mathbf{ND}(s) \wedge n \in \{n' \mid \langle n', t' \rangle \in \mathbf{TA}(HPDIC) \wedge t' > 0\}\} \quad (4.14)$$

$$p(s) = 100 * \mathbf{COMPTE}(\mathbf{NDACT}(s)) / \mathbf{COMPTE}(\mathbf{ND}(s)) \quad (4.15)$$

Nous calculons donc le rapport entre le nombre de phrases activées ($\mathbf{NDACT}(s)$) par l'élève et le nombre total de phrases distinctes ($\mathbf{ND}(s)$) nécessaires pour produire la solution complète. Par exemple, considérons la solution suivante au problème du rectangle (reprise de la figure 4.2) :

- Comme **AB est perpendiculaire à AD**, et **AD est perpendiculaire à CD**, alors, selon la propriété qui dit «**Deux droites perpendiculaires à une même troisième sont parallèles entre elles**», **AB est parallèle à CD**.
- Comme **BC est perpendiculaire à CD**, et **AB est parallèle à CD**, alors, selon la propriété qui dit «**Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre**», **AB est perpendiculaire à BC**.
- Comme **AB est perpendiculaire à AD**, **AD est perpendiculaire à CD**, **BC est perpendiculaire à CD**, et **AB est perpendiculaire à BC**, alors, selon la propriété qui dit «**Un rectangle est un quadrilatère ayant quatre angles droits**», **ABCD est un rectangle**.

Dans celle-ci, nous comptons un total de 14 éléments de phrases à compléter, identifiés en caractères gras, soit 4 pour la première inférence, 4 pour la seconde et 6 pour la dernière. Cependant, on s'aperçoit que certains énoncés sont utilisés à plus d'un endroit. Il suffit donc, en réalité, de produire un total de 9 énoncés distincts pour compléter cette solution et c'est cette dernière valeur qui est utilisée dans notre calcul.

Pour chacune des solutions ($s \in Sol$) du graphe HPDIC, nous gardons en mémoire son pourcentage d'achèvement :

$$\mathbf{PA}(HPDIC) = \{\langle s, p(s) \rangle \mid s \in Sol \wedge p(s) \in [0..100]\} \quad (4.16)$$

Nous nous servons de cette métrique afin d'identifier la solution la plus susceptible de correspondre au plan actuel de résolution de l'élève. Par conséquent, QEDX utilise l'information sur l'historique d'activation des noeuds ainsi que la solution la plus avancée pour évaluer l'état cognitif de l'élève et lui proposer une aide appropriée.

Dans cette section, nous avons présenté la description formelle de QEDX pour ce qui concerne les graphes HPDIC et le MIA. Les graphes HPDIC sont construits à l'aide des inférences permettant de résoudre un problème. Ils contiennent l'ensemble des solutions acceptables selon les inférences choisies par l'enseignant, sans imposer de contraintes formelles sur la structure de ces inférences. Dans le cas où le graphe contiendrait une boucle, nous avons présenté un algorithme en chaînage avant qui permet de détecter les solutions invalides. Pour sa part, le MIA permet de conserver un historique du travail de l'élève dans le but d'évaluer son état cognitif et de lui fournir une aide adaptée à son cheminement de résolution. Il contient l'historique d'activation des noeuds ainsi que le pourcentage d'achèvement de chacune des solutions possibles. Nous présentons, dans la section suivante, des statistiques concernant les graphes HPDIC des cinq problèmes implantés dans QEDX.

4.4 Résultats relatifs aux graphes et simulation du MIA

Pour illustrer le modèle formel décrit à la section précédente, nous avons utilisé le problème du rectangle. Il s'agit d'un problème simple, mais riche, qui permet de bien mettre en lumière les aspects techniques du fonctionnement interne de notre système. Par contre, pour être en mesure de tester QEDX dans un contexte scolaire, un total de cinq problèmes, en incluant le problème du rectangle, ont été analysés par des didacticiens et implantés. Dans cette section, nous présentons les statistiques concernant les graphes de ces cinq problèmes. Nous verrons aussi de quelle façon le MIA est utilisé pour permettre de tenter d'identifier l'état cognitif de l'élève qui résout le problème proposé par QEDX.

4.4.1 Statistiques des graphes HPDIC implantés

Les graphes HPDIC, pouvant être difficiles à produire, ont été vérifiés à l'aide de différentes méthodes. Premièrement, il faut mentionner qu'aucune vérification formelle de la structure des inférences n'a été faite de façon automatique, car nous avons conçu un logiciel qui peut s'adapter aux différents niveaux d'exigence des enseignants. Par contre, la structure des graphes a été validée à l'aide d'un algorithme qui permet de vérifier si chaque inférence fait partie d'au moins une solution valide. De plus, lors de la première expérimentation en classe (voir section 3.4), un des objectifs était justement de vérifier la validité des graphes. Les données recueillies nous ont permis de constater qu'un nombre limité de nouvelles démonstrations ont été proposées par les élèves par rapport à celles implantées dans le logiciel. Toutefois, plusieurs énoncés équivalents étaient manquants. Par exemple, on pouvait entrer «AB est perpendiculaire à BC», mais le système refusait «L'angle ABC est droit», qui est pourtant équivalent. Des erreurs à l'intérieur de quelques inférences ont aussi été détectées.

Néanmoins, toutes les omissions et erreurs ont d'emblée été corrigées. Enfin, lors de la saisie des messages du tuteur pour chaque inférence, une nouvelle vérification, plus formelle, a été réalisée par rapport à la structure de chaque inférence. À la suite de ces différentes vérifications, nous pouvons affirmer que les graphes sont valides et ils ont été utilisés lors d'une seconde expérimentation. Ici, nous disons que les graphes sont valides, car ils permettent de produire des solutions acceptables pour un niveau d'exigence visé. Cependant, pour un même problème, des enseignants différents peuvent proposer des graphes HPDIC différents qui seront chacun valide dans son contexte d'enseignement.

Le tableau 4.5 présente les statistiques concernant les graphes HPDIC des cinq problèmes implantés à l'intérieur de QEDX (annexe A). Dans ce tableau, nous indiquons le nombre d'hypothèses ($Init$), de résultats intermédiaires ($R \setminus \{Init \cup Final\}$) et de conclusion ($Final$) dans chacun des graphes. Dans la colonne **I**, nous avons indiqué le nombre total d'inférences. Le nombre d'énoncés de justification distincts est indiqué dans la colonne E . Il faut se rappeler que chaque noeud de justification, en considérant ses liens, représente une inférence, mais que plusieurs inférences peuvent utiliser un même énoncé de justification. Nous pouvons donc retrouver plusieurs noeuds de justification contenant des énoncés identiques, ce qui n'est pas le cas pour les noeuds H, I et C. Nous indiquons enfin le nombre total de noeuds dans chaque graphe. Le problème du rectangle présenté dans ce chapitre apparaît sur la ligne ayant comme identifiant de problème (PID) le sigle RT.

Tableau 4.5 Données concernant la structure des cinq graphes HPDIC implantés

PID	$Init$	$R \setminus \{Init \cup Final\}$	$Final$	I	E	Total
RT	4	4	1	13	8	22
TT	5	16	1	31	13	53
PT	3	34	1	76	23	114
PA	51	122	1	214	31	388
TS	14	47	1	116	44	178

L'énumération de l'ensemble des solutions, pour chacun des problèmes, à l'aide de notre algorithme, nous a permis de produire les données présentées dans le tableau 4.6. On y retrouve le nombre de solutions valides (Valide), mais aussi le pourcentage de solutions valides (% Valide). En effet, dans le cas d'un graphe, l'algorithme d'énumération des solutions produit des solutions invalides, ce qui peut avoir un impact non négligeable sur le temps de calcul des solutions. Nous indiquons ensuite le nombre d'inférences (**I**) minimum, maximum et moyen nécessaire pour produire une solution. Le nombre minimum, maximum et moyen d'énoncés distincts ($E \cup R$), donc de phrases à inscrire à l'interface de QEDX pour produire une solution, est aussi inclus. Il s'agit toutefois de valeurs théoriques qui ne reflètent pas nécessairement

les longueurs moyennes des solutions produites par les élèves. Ces dernières données ne sont cependant pas disponibles actuellement à cause d'erreurs dans les graphes HPDIC utilisés lors des expérimentations. Enfin, le nombre de stratégies (Strat), soit le nombre de groupes de solutions utilisant un même ensemble d'énoncés de justification, y est inscrit.

Tableau 4.6 Statistiques associées aux solutions des cinq problèmes implantés

PID	Solution			I			$E \cup R$			Strat
	Valide	% Valide	Unique	min	max	moy	min	max	moy	
RT	23	82,14	11	2	5	3,87	8	13	10,64	9
TT	244	52,14	74	9	15	13,26	20	31	25,61	14
PT	1056	100	531	5	17	11,37	10	29	19,92	176
PA	68722	100	35584	5	47	35,76	13	88	65,86	772
TS	5504194	50,77	726615	4	35	26,71	10	75	52,12	12634

Étant donné que QEDX n'impose aucune structure d'inférence lors de l'écriture des énoncés, on peut représenter chaque solution comme un simple ensemble d'énoncés. Dans ce contexte, les solutions valides peuvent être regroupées selon les énoncés utilisés, à la manière des stratégies. Cependant, dans ce cas-ci, nous ne nous restreignons pas aux justifications, mais nous prenons plutôt en compte tous les énoncés nécessaires à la production de la preuve. Le nombre de groupes contenant des énoncés distincts est noté dans le tableau 4.6 comme étant le nombre de solutions uniques. Afin d'illustrer ce concept, nous présentons, dans le tableau 4.7, les 11 solutions uniques, pour QEDX, associées au problème du rectangle. Dans ce tableau, nous remarquons, par exemple, que les solutions S20 et S21 sont identiques. Pourtant, en consultant le tableau 4.4, on note qu'elles utilisent une inférence différente pour produire la conclusion. En fait, pour λ_{8b} de S20, nous utilisons, entre autres, H_2 comme antécédent alors que H_3 est utilisé pour λ_{8c} de S21. Les deux solutions utilisent de plus le résultat de λ_5 , qui a comme antécédents H_2 , H_3 , H_4 et I_2 . Étant donné que H_2 et H_3 sont nécessaires dans les deux solutions, il n'est pas possible de déterminer sur quelle solution l'élève travaille précisément, à moins d'avoir une information supplémentaire concernant la structure de la démonstration.

En analysant les données du tableau 4.6, nous remarquons que les problèmes PA et PT produisent des arbres HPDIC alors que les trois autres forment des graphes. En effet, PA et PT ne produisent aucune solution invalide, donc aucun cycle n'est présent dans le graphe HPDIC. Un cycle produit obligatoirement des solutions invalides qui sont détectées par la suite. La présence d'un graphe ou d'un arbre ne nous donne aucun indice sur la complexité relative du problème, mais nous permet par contre d'éviter de devoir détecter les solutions invalides dans le second cas.

Tableau 4.7 Solutions uniques pour le problème du rectangle

Solution	H_1	H_2	H_3	H_4	I_1	I_2	I_3	I_4	C_1	J_1	J_2	J_3	J_4	J_5	J_6	J_7	J_8
S01	X	X	X	X	X	X	X	X	X	X	X		X				X
S02, S03		X	X	X	X	X	X	X	X	X		X	X				X
S04, S05, S06, S07		X	X	X	X		X	X	X	X		X		X			X
S08, S09, S10, S11		X	X	X		X	X	X	X	X		X		X			X
S12		X	X	X	X		X		X	X		X				X	
S13		X	X	X		X	X		X	X		X				X	
S14		X	X	X	X	X	X	X	X	X			X		X	X	
S15		X	X	X	X	X	X	X	X	X			X		X		X
S16, S17, S18		X	X	X	X	X		X	X	X			X				X
S19, S20, S21, S22	X	X	X	X			X	X	X		X			X			X
S23	X	X	X	X			X		X		X					X	

Pour évaluer la complexité relative d'un problème, la statistique qui nous semble la plus intéressante est le nombre d'énoncés distincts de justification (E). D'une part, elle est facile et rapide à calculer, car un simple parcours de la liste des inférences est nécessaire pour l'obtenir. D'autre part, il existe une forte corrélation ($r = 0,9972$) entre cette valeur et le logarithme du nombre de stratégies possibles pour résoudre un problème, tel qu'observé pour nos cinq problèmes. Intuitivement, ce résultat était attendu. En effet, généralement un nombre plus grand d'énoncés de justifications distincts, pour un problème donné, ne signifie pas que les solutions sont proportionnellement plus longues. Le nombre d'énoncés est plutôt relié à la variété des démarches admissibles pour résoudre le problème. Dans ce cas, l'ensemble des stratégies est en fait une combinaison de ces différentes démarches. Il en résulte donc qu'un plus grand nombre d'énoncés de justifications distincts dénote la richesse du problème et produit, par conséquent, un nombre de stratégies plus élevé, selon une relation exponentielle.

Intuitivement, nous pourrions aussi penser que le nombre de solutions ou de solutions uniques dépend directement du nombre d'inférences, ce qui n'est pas le cas, comme le montrent les données du graphe PA. Celui-ci comporte un nombre d'inférences, un nombre total de noeuds et un nombre d'inférences moyen par solution qui sont plus élevés que dans les autres problèmes, mais arrive au second rang en ce qui concerne le nombre de solutions et de stratégies. Une corrélation forte ($r = 0,9891$) existe cependant entre le nombre d'énoncés distincts de justification et le logarithme du nombre de solutions valides, mais elle est difficile à expliquer. En fait, nous n'avons pas été en mesure de proposer une explication satisfaisante autre que le fait que l'ensemble E permet d'estimer la complexité du problème à résoudre. Cette complexité dénote à la fois la difficulté du problème, mais aussi la complexité du graphe HPDIC correspondant. Les cinq problèmes choisis par les didacticiens avaient pour but de démontrer cette diversité et de tester QEDX dans différents contextes. Afin de fournir une interprétation plus étoffée de la relation, il sera donc nécessaire d'avoir un échantillon plus important de problèmes.

4.4.2 Validation du MIA

Le MIA, tel qu'il est défini dans ce chapitre, a été implanté avec succès dans QEDX. Cependant, il faut rappeler que la définition présentée ne sert qu'à annoter les noeuds du graphe HPDIC. Ces informations sont par la suite utilisées par la couche EDOI afin d'être en mesure de produire des messages d'aide personnalisés. Étant donné que la présentation d'un simple historique d'annotation des noeuds n'est pas d'un très grand intérêt, nous allons montrer comment le MIA peut être utilisé par la couche EDOI afin de tenter d'estimer l'état cognitif de l'élève. Dans le tableau 4.8, nous avons simulé un parcours possible pour la résolution du problème du rectangle. Pour chaque pas de la solution, nous indiquons le pourcentage d'achèvement de chacun des groupes de solutions. La ou les solutions dominantes, soit celles ayant le plus haut pourcentage d'achèvement après chaque pas, sont indiquées par un fond gris. Les données du tableau 4.8 sont celles générées par le MIA tel que défini dans ce chapitre.

Tableau 4.8 Pourcentage d'avancement pour chaque solution à chaque étape d'une résolution simulée

Temps Solution	1 C_1	2 H_2	3 H_4	4 J_1	5 J_8	6 J_5	7 H_2	8 H_3	9 I_1	10 I_3	11 J_3	12 I_4
S1	8	15	23	31	38	38	38	46	54	62	62	69
S2, S3	8	17	25	33	42	42	42	50	58	67	75	83
S4, S5, S6, S7	9	18	27	36	45	55	55	64	73	82	91	100
S8, S9, S10, S11	9	18	27	36	45	55	55	64	64	73	82	91
S12	11	22	33	44	44	44	44	56	67	78	89	89
S13	11	22	33	44	44	44	44	56	56	67	78	78
S14	8	17	25	33	33	33	33	42	50	58	58	67
S15	8	17	25	33	42	42	42	50	58	67	67	75
S16, S17, S18	10	20	30	40	50	50	50	60	70	70	70	80
S19, S20, S21, S22	10	20	30	30	40	50	50	60	60	70	70	80
S23	12	25	38	38	38	38	38	50	50	62	62	62

À l'aide des informations contenues dans ce tableau, il est possible d'évaluer l'état cognitif de l'élève fictif et de prévoir les actions de QEDX. Avant que l'élève n'écrive sa première phrase, nous n'avons aucune information concernant sa stratégie. Nous proposons donc des messages et des stratégies génériques, qui suggèrent, par exemple, de relire l'énoncé du problème.

Dès qu'une première phrase est reconnue par le système, le MIA contient de l'information qui permet de mieux guider l'élève. Dans notre simulation, pour les temps 1, 2 et 3, l'élève inscrit des énoncés nécessaires à toutes les solutions, donc celles qui sont les plus courtes seront privilégiées, car leur pourcentage d'achèvement augmente plus rapidement. Au temps 3, pour aider un élève qui serait bloqué, nous allons regarder les inférences qui contiennent

H_4 et qui font partie de S23, donc λ_7 . Nous allons alors chercher à lui faire compléter λ_7 pour ensuite lui proposer de travailler sur λ_2 , soit la seule autre inférence de S23, ce qui lui permettrait d'achever cette solution. Si ces essais demeurent infructueux, nous allons lui proposer de travailler sur une autre inférence utilisant H_4 , car c'est la dernière phrase soumise. Un choix judicieux serait de sélectionner λ_5 , car c'est la seule autre inférence, à part λ_7 , qui fait intervenir H_2 , en plus de H_4 , qui a été écrite récemment par l'élève.

Par la suite, en écrivant J_1 et J_8 , l'élève précise sa stratégie. Cependant, plus de la moitié des groupes de solutions utilisent ces deux justifications, donc le système aura tendance, encore une fois, à privilégier la solution la plus courte. Le fait d'écrire ensuite J_5 permet de réduire le nombre de solutions probables à deux. Étant donné que la seule différence entre ces deux groupes est le fait d'utiliser I_1 ou I_2 et que l'élève ne nous a fourni aucun de ceux-ci, QEDX va choisir aléatoirement une solution et l'utiliser comme solution dominante. En ce qui concerne l'ordre de traitement des inférences au temps 6, nous allons aider l'élève à compléter λ_5 puis λ_{8d} étant donné que H_4 a été proposé plus récemment par rapport à H_2 .

Au temps 7, l'élève n'entre pas une nouvelle donnée, mais répète plutôt H_2 . Nous considérons donc que celui-ci veut indiquer à QEDX qu'il veut reprendre son raisonnement en privilégiant cette hypothèse. Dans cette situation, on propose d'abord de travailler sur λ_5 , qui contient les deux phrases les plus récentes. Par la suite, nous allons privilégier λ_{8b} , car elle utilise H_2 , au lieu de choisir λ_{8d} . Nous pouvons ensuite proposer de travailler sur λ_{1a} pour obtenir I_1 et c'est ce que va choisir l'élève. Il termine ensuite sa solution sans changer de stratégie.

La simulation montre que les raisonnements des élèves ne sont pas linéaires. Par conséquent, le MIA est un outil essentiel pour situer l'état cognitif de ceux-ci et ainsi permettre au système tutoriel de proposer des indices pertinents. Cependant, il est important de rappeler que le MIA ne fournit que des données brutes qui sont interprétées, dans cette simulation, par le didacticien. Par contre, le choix de conserver la chronologie des énoncés s'est avéré déterminant afin de fournir une aide pertinente. En effet, en ne notant que l'ensemble des énoncés inscrits par l'élève sans égards à l'ordre d'entrée, comme c'est le cas dans d'autres systèmes (Anderson et al., 1985; Koedinger et Anderson, 1993), les didacticiens ne proposaient que de l'aide en chaînage avant ou arrière. N'ayant pas accès au cheminement de l'élève, ils ne sont pas en mesure d'offrir un soutien respectant l'état cognitif de ce dernier. L'ajout de la chronologie a donc résolu ce problème, car le cheminement de l'élève est alors accessible. Elle a aussi permis de proposer d'autres pistes de solution cohérentes avec la stratégie de l'élève. Les données contenues dans le MIA sont donc essentielles pour la production des messages par QEDX. Elles sont par la suite analysées par la couche EDOI, qui a pour objectif de modéliser les décisions du didacticien.

Cette section était divisée en deux principales parties, soit les résultats concernant les graphes HPDIC et une simulation de l'utilisation du MIA. Concernant les graphes HPDIC, nous avons premièrement décrit la façon dont nous les avons testés et validés, car ils n'utilisent pas une approche formelle classique. Nous avons ensuite présenté les statistiques concernant les cinq problèmes implantés dans QEDX et utilisés en classe. Nous avons observé qu'il semble y avoir une relation entre le nombre d'énoncés de justification distincts et le nombre de stratégies par problème. En ce qui concerne le nombre de solutions, aucune relation probante n'a été découverte, mais des pistes ont été proposées. Un nombre plus important de graphes seront cependant nécessaires pour valider ces relations. Nous avons ensuite montré comment les données du MIA peuvent être utilisées pour produire une aide adaptée à l'état cognitif de l'élève. Nous avons proposé un cheminement simulé pour montrer comment QEDX devrait agir dans différentes situations étant donné les données contenues dans le MIA. Dans la section suivante, nous présentons une discussion générale des graphes HPDIC et du MIA et nous montrons comment ceux-ci sont avantageux pour la conception de QEDX.

4.5 Discussion

Maintenant que nous avons défini formellement les graphes HPDIC et le MIA et que nous avons présenté quelques résultats les concernant, il est maintenant temps de prendre un peu de recul et de voir comment notre modèle se compare aux autres formes de représentation. Plus particulièrement, dans cette section, nous présentons les avantages, mais aussi les contraintes des graphes HPDIC. Ils sont bien adaptés au travail que l'on demande aux élèves, mais ils nécessitent un effort important pour leur création. Nous discuterons aussi du MIA, qui permet de proposer une aide qui respecte l'état cognitif de l'élève. Cependant, pour des problèmes complexes, nous verrons comment celui-ci est limité.

Notre but en produisant QEDX était de permettre à l'élève d'explorer le problème sans le contraindre à la solution idéale selon les experts, comme le fait, par exemple, Geometry Tutor (Anderson et al., 1985; Ritter et al., 2010) ou ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991). Il ne suffisait donc pas de proposer la solution idéale par rapport à laquelle on compare le cheminement de l'élève. De plus, on voulait éviter de forcer un chaînage avant ou arrière, car nous croyons que lors de l'élaboration d'une preuve, le cheminement est rarement linéaire, comme le suppose une lecture de la preuve finale. Le graphe HPDIC que nous proposons contient l'ensemble des solutions acceptables pour un problème en occupant un minimum d'espace mémoire. En effet, un graphe contenant quelques centaines de noeuds peut permettre de produire les milliers de solutions acceptables. De plus, il est construit à partir des inférences, ce qui est justement l'élément de base pour la construction

des démonstrations en géométrie. Il permet aussi de sauvegarder les informations nécessaires au MIA et permet à QEDX de suivre le cheminement de l'élève. Enfin, il faut rappeler que notre graphe ressemble aux réseaux déductifs de Tanguay (2005, 2007). Étant donné que les graphes HPDIC ont été développés alors que nous ignorions l'existence des réseaux déductifs, il est sécurisant de constater que nous avons conçu une structure similaire à celle qui est enseignée dans certaines écoles.

Le graphe HPDIC, en plus de contenir l'ensemble des solutions d'un problème dans un espace mémoire restreint, contient toute l'information nécessaire pour produire les preuves sous les différentes formes, telles que présentées dans la figure 4.2. Pour la rédaction, nous remarquons que chaque phrase est une inférence complète qui contient les antécédents, la justification et le conséquent. Pour produire la rédaction à partir du graphe HPDIC, il suffit de produire la phrase correspondant à chaque noeud de justification (inférence) faisant partie de la solution. Pour produire la preuve en 2 colonnes, il suffit, pour chaque noeud de résultat intermédiaire faisant partie de la solution, d'inscrire le résultat à gauche et la justification parente à droite. Dans le cas du graphe inférentiel, il faut tout simplement créer un noeud pour chaque noeud de justification du graphe HPDIC et les relier lorsque le conséquent de l'un devient l'antécédent d'un autre. Un travail similaire est possible pour produire le graphe de résultats, mais pour ce dernier, on ne présente que les résultats intermédiaires. Quant à la preuve en casse-tête, elle est en fait une réorganisation spatiale de la rédaction et peut aussi être produite par les données du graphe HPDIC. Enfin, les réseaux déductifs sont équivalant aux graphes HPDIC, mais sont présentés d'une façon différente graphiquement. Les graphes HPDIC sont donc très polyvalents et cette caractéristique pourra permettre de proposer la preuve sous différentes formes simultanément dans QEDX.

Malgré ces avantages, le graphe HPDIC comporte certaines limites. Étant donné que nous ne travaillons pas avec des preuves formelles pour respecter les différents styles d'enseignement, il est impossible de produire la liste des inférences acceptables automatiquement. Pour chaque problème, il faut produire manuellement l'ensemble de celles-ci. Ce procédé est long et fastidieux, donc il est très facile de faire des erreurs. Malgré l'algorithme qui permet de trouver les noeuds inutilisés, plusieurs erreurs se sont faufilees et ont été trouvées lors de l'expérimentation en classe. De plus, QEDX ne permet pas de produire une preuve qui n'est pas contenue dans le graphe. Il faut donc que l'enseignant ou le didacticien qui produit le graphe soit minutieux et, de plus, qu'il s'assure qu'il a fait la liste exhaustive des preuves acceptables. Pour que QEDX puisse être utilisé à grande échelle, un travail important devra être fait pour rendre plus conviviale la production des graphes HPDIC. L'ajout d'un moteur de déduction formel peut aussi être envisagé pour permettre d'accepter des preuves qui ne feraient pas partie du graphe.

Dans une approche classique, on demande de proposer des inférences complètes, ce qui permet d'identifier la solution précise parmi toutes celles qui sont valides. Par contre, dans le cadre de QEDX, les didacticiens ont délibérément choisi de ne pas imposer de structure ou d'ordre lors de l'écriture des énoncés, dans le but de permettre une exploration plus libre du problème. Cette apparente absence de structure a, au départ, déstabilisé les élèves habitués à une production plus classique des démonstrations. Ils ont cependant découvert que QEDX leur permettait facilement de tester différentes hypothèses dans le but de construire leur preuve. Par contre, au moment de compléter une preuve, il est difficile de savoir quels éléments sont manquants. Dans un onglet de QEDX, nous proposons donc une preuve à compléter ce qui permet de voir les éléments manquants. La preuve qui est affichée est celle qui a le pourcentage d'achèvement le plus élevé à partir d'un certain seuil. En dessous du seuil, nous n'avons pas suffisamment d'informations pour choisir une démonstration qui représente bien la stratégie de l'élève.

Le problème qui se pose est que l'absence de structure ne permet pas de déterminer la démonstration exacte que l'élève désire produire. Lors de la présentation des résultats pour les 5 graphes HPDIC (section 4.4.1), nous avons défini le concept de solutions uniques, soit le nombre de groupes de solutions utilisant des énoncés identiques. À chaque solution unique correspond un certain nombre de solutions valides. L'absence de structure nous fait donc perdre de l'information, mais permet de simplifier le travail de l'élève en lui imposant moins de contraintes. De plus, la démonstration proposée est habituellement assez proche de celle imaginée par l'élève, car elle utilise les mêmes énoncés. Cette perte de précision dans la preuve proposée est donc largement compensée par la possibilité d'explorer librement le problème. De plus, pour nous aider à retrouver une partie de la structure, nous voulons ajouter la possibilité de produire un schéma de démonstration à l'interface de QEDX. Chaque élément ajouté au schéma permettra de cibler une solution précise sans pour autant contraindre l'élève.

Que l'on parle de solutions valides ou uniques, il faut se rendre à l'évidence que leur nombre augmente rapidement pour des problèmes complexes. Dans la définition du MIA, nous avons cependant besoin de stocker le pourcentage d'achèvement de chacune des solutions valides pour retrouver celle qui a le pourcentage le plus élevé. Il faut recalculer le pourcentage après chaque phrase inscrite par l'élève. Le temps nécessaire pour calculer ceux-ci est raisonnable si l'ensemble des solutions est déjà en mémoire. Cependant, dans le problème TS, il est impossible de garder toutes les solutions en mémoire, donc il est nécessaire de les retrouver à l'aide de l'algorithme d'énumération des solutions et de les valider ensuite. Cette opération prend plusieurs minutes, ce qui n'est pas une stratégie viable si nous voulons que QEDX puisse proposer de l'aide interactive.

Nous avons tenté d'utiliser le concept de stratégies pour regrouper les solutions selon les énoncés de justifications utilisées à l'intérieur de celles-ci. Ce concept, en plus de réduire l'espace des solutions à analyser, s'inscrit dans une démarche de résolution de problème. En effet, nous avons remarqué que les enseignants, lors des expérimentations, avaient tendance à insister d'abord sur les justifications avant de proposer de travailler sur les résultats intermédiaires. Même si cette approche semblait prometteuse, nous nous sommes aperçus que nous n'étions pas en mesure de détecter correctement la solution la plus avancée, car nous n'avions aucune information par rapport aux noeuds intermédiaires dans les stratégies. En ajoutant les noeuds intermédiaires, nous retrouvons les solutions uniques qui sont en trop grand nombre pour être stockées en mémoire. L'explosion combinatoire du nombre de solutions est donc un problème important, qui a été résolu dans Mentoniez (Py, 1994, 1996, 2001) en limitant la longueur des solutions acceptables. Dans notre cas, nous avons développé une heuristique qui consiste à propager des statistiques concernant les solutions partielles contenues à l'intérieur d'un arbre HPDIC, ce qui permet d'estimer efficacement la solution la plus avancée. Un algorithme a aussi été conçu afin de transformer un graphe HPDIC en un ensemble d'arbres pour être en mesure d'appliquer l'heuristique proposée (voir chapitre 5 pour une analyse détaillée).

En ce qui concerne le MIA, nous pouvons affirmer qu'il permet véritablement de proposer une aide qui respecte l'état cognitif de l'élève. En effet, nous avons la capacité, à l'aide de la solution la plus avancée, de proposer des indices qui permettront de compléter celle-ci, donc de respecter le cheminement de l'élève. De plus, en utilisant l'historique des actions, il est possible de proposer d'autres pistes de solution si l'élève n'est plus en mesure de cheminer dans la solution courante. Le choix des indices suit l'ordre chronologique inverse des actions de l'élève dans le but de lui rappeler ses tentatives récentes et prometteuses. Les systèmes tutoriels, comme Geometry Tutor (Anderson et al., 1985; Ritter et al., 2010), ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991) et même Mentoniez (Py, 1994, 1996, 2001), proposent habituellement une série d'indices qui ciblent une étape précise et donnent des indices allant des plus vagues aux plus précises. Les concepteurs de ces logiciels supposent donc que l'élève prendra le chemin optimal et le poussent vers celui-ci sans lui proposer d'autres chemins. Dans notre approche, nous acceptons que l'élève puisse changer de stratégie en cours de résolution et c'est pour cette raison que QEDX propose différentes pistes de solution avant d'arriver à un indice précis. Le choix de l'ordre des indices est discuté dans les chapitres 5 et 6. Pour l'instant, il faut se rappeler que les informations contenues dans le MIA sont suffisantes pour la production des messages de QEDX, mais qu'il ne permet pas d'adapter la personnalité du tuteur à différents types d'enseignement.

En résumé, le principal avantage des graphes HPDIC est la possibilité de stocker l'ensemble des solutions relatives à un problème dans un espace mémoire restreint. En utilisant les

inférences comme élément de base, la structure obtenue permet d'unifier l'ensemble des représentations des solutions. En effet, à partir du graphe HPDIC, il est possible de produire une démonstration sous différentes formes et QEDX pourrait permettre de les proposer à son interface. Le plus grand obstacle à l'utilisation de QEDX est par contre la production des graphes qui ne possèdent pas de contraintes formelles. En ce qui concerne le MIA, celui-ci contient des informations qui permettent un système d'aide innovateur qui autorise une véritable exploration du problème. Cependant, la nécessité d'énumérer l'ensemble des solutions peut être problématique pour des problèmes complexes, comme nous avons pu le voir avec le problème TS. Une solution pour résoudre cette explosion combinatoire a été proposée dans le chapitre 5 et a été implantée dans QEDX.

4.6 Conclusion

Le système QEDX comporte quatre couches (HPDIC, MIA, EDOI et GMD) pour permettre de proposer une aide adaptée au cheminement de l'élève lors de la résolution d'un problème en géométrie euclidienne. Ce chapitre avait pour but de proposer une définition formelle des structures de base de QEDX, soit les couches HPDIC et MIA. En ce qui concerne les graphes HPDIC, nous avons premièrement illustré différents styles de preuves pour démontrer que ceux-ci sont inadéquats pour représenter efficacement l'ensemble des solutions à un problème donné. Le graphe HPDIC a donc été développé en utilisant l'inférence comme unité de base. En effet, dans la définition formelle, les deux éléments qui définissent réellement le graphe sont **I**, soit l'ensemble des inférences et **S**, soit les liens entre ces dernières. Un algorithme pour énumérer l'ensemble des solutions et pour détecter les solutions valides a été proposé. Les statistiques concernant les graphes HPDIC pour les cinq problèmes implantés dans QEDX ont été présentées. Il semble y avoir un lien entre le nombre de stratégies et le nombre d'énoncés distincts de justification, mais celui-ci devra être confirmé. Cependant, nous n'avons pas été en mesure d'estimer le nombre de solutions avec les données recueillies, ce qui signifie que la complexité des problèmes est plus riche que nous pouvions imaginer. Le principal avantage des graphes HPDIC est le fait de pouvoir représenter l'ensemble des preuves et de contenir toute l'information nécessaire pour produire les différents styles de preuves. Le fait qu'ils ne contiennent aucune contrainte formelle permet de produire des preuves qui correspondent aux exigences des enseignants, mais les rendent difficiles à produire.

Pour représenter la chronologie des états cognitifs de l'élève, nous avons proposé le MIA. Il a été développé, car nous voulions un système qui permet de proposer différentes pistes de solution au lieu de forcer l'élève à produire la meilleure solution selon les experts. Les systèmes tutoriels actuellement offerts contraignent habituellement l'élève à travailler en chaîne avant

et ils ciblent seulement la prochaine étape identifiée. Pour parvenir à notre fin, plusieurs avenues ont été analysées, par exemple l'approche par exploration des données qui se base sur un corpus de travaux d'élèves pour produire des messages d'aide. Nous n'avons toutefois pas les données nécessaires pour l'utilisation d'une telle approche, mais l'utilisation de QEDX pourrait permettre de les recueillir. Le MIA, qui intègre la chronologie des actions de l'élève, a donc été développé. Formellement, il contient le temps d'activation de chacun des noeuds et le pourcentage d'avancement de chacune des solutions issues du graphe HPDIC. Nous avons ensuite montré comment le MIA pourrait être utilisé pour aider un élève fictif. Les données contenues dans le MIA nous semblent donc suffisantes pour atteindre notre objectif. Cependant, une limite existe en ce qui concerne le nombre de solutions qui peut être géré simultanément dans QEDX, ce qui peut limiter son utilisation pour des problèmes complexes. Malgré cette limite, qui a été adressée dans le chapitre 5, le MIA constitue un outil essentiel au fonctionnement de QEDX pour aider l'élève à explorer le problème.

Malgré la puissance et la polyvalence du graphe HPDIC et du MIA, plusieurs améliorations peuvent être envisagées. Pour que QEDX puisse être utilisé à plus grande échelle, il faudrait premièrement proposer une interface pour produire les graphes HPDIC. En effet, actuellement ils sont décrits et encodés dans des fichiers textes, donc une interface graphique serait bénéfique. On peut aussi envisager de proposer différentes représentations des preuves à l'interface de QEDX, car le graphe HPDIC nous le permet. En fait, l'ajout d'un onglet permettant de construire la preuve sous forme graphique est déjà planifié. En ce qui concerne le MIA, des données concernant les connaissances des élèves pourraient être ajoutées. Celles-ci pourraient permettre de choisir, parmi plusieurs stratégies, celle qu'il est le plus susceptible de privilégier. QEDX pourrait alors décider de le faire travailler au sein des connaissances qu'il maîtrise ou de lui proposer une autre avenue. Enfin, même si nous avons en partie réglé le problème dans la version actuelle de QEDX, du travail reste à faire pour limiter l'explosion combinatoire du nombre de solutions qui peut rendre le MIA inutilisable.

CHAPITRE 5 ÉVALUATION DES DÉMONSTRATIONS ET ORDONNANCEMENT DES INFÉRENCES (EDOÏ)

«... se mettre le nez dans l'enfer du mal.»

— Jean Perron

5.1 Introduction

Dans le cadre de nos recherches, nous avons développé un logiciel pour aider les élèves à produire des preuves en géométrie euclidienne, que nous avons nommé QED-Tutrix (QEDX). Un aperçu du système ainsi qu'une description de ses fondements didactiques ont été présentés dans le chapitre 3. QEDX, ayant pour objectif d'imiter le travail de l'enseignant, doit être en mesure de gérer l'ensemble des preuves acceptables et de proposer des pistes de solution utiles à un élève qui subit un blocage en cours de résolution. Pour un problème simple, le nombre de solutions est limité, mais pour des problèmes plus complexes, il est difficile de gérer les millions de solutions possibles.

Notre système comporte quatre couches principales (HPDIC, MIA, EDOÏ et GMD) dont l'unité de base est l'inférence. Cette dernière est équivalente à un pas dans une preuve en géométrie. En effet, chaque inférence comporte un certain nombre d'antécédents (hypothèses), une justification (propriété ou définition) et un conséquent (conclusion). En combinant les inférences valides pour un problème, il est possible de constituer un ensemble de preuves acceptables. Pour ce faire, il faut savoir que le conséquent d'une inférence peut être utilisé comme antécédent dans d'autres inférences.

La couche HPDIC de notre système représente en fait l'ensemble des solutions acceptables sous forme de graphe et elle est décrite de manière formelle dans le chapitre 4. Le graphe HPDIC est produit à l'aide d'une liste d'inférences fournie par l'enseignant ou le didacticien et les inférences n'ont pas à être strictement formelles. L'avantage du graphe HPDIC est qu'il est en mesure de représenter les millions de solutions acceptables dans un espace mémoire restreint. Cependant, des raisonnements en boucle peuvent exister à l'intérieur de celui-ci et c'est pour cette raison que nous parlons d'un graphe HPDIC et non d'un arbre. La présence de boucle rend plus onéreuse l'énumération des solutions, car il est nécessaire de vérifier la validité de chaque solution produite par notre algorithme.

La seconde couche de QEDX est le MIA. Elle a pour mandat de conserver l'historique des actions de l'élève et d'évaluer sa progression dans la résolution d'un problème donné. Le MIA

est décrit en détail dans le chapitre 4. Pour comprendre le présent chapitre, il faut savoir que le MIA garde une liste des énoncés entrés par l'élève ainsi que le moment d'écriture le plus récent de chaque énoncé. Cette liste est utilisée par les couches supérieures de QEDX pour proposer des messages cohérents avec l'état cognitif supposé de l'élève. Le MIA conserve aussi une métrique qui permet de juger de la progression de l'élève dans la résolution du problème. Cette métrique doit être calculée sur l'ensemble des solutions, ce qui s'avère impraticable dans un problème comportant des millions de solutions.

Dans ce chapitre, nous présentons la couche EDOI, soit celle qui réalise l'Évaluation des Démonstrations et l'Ordonnancement des Inférences. Premièrement, le module d'évaluation des démonstrations permet de calculer la métrique de progression de l'élève qui est contenue dans le MIA. Il permet aussi d'identifier la solution qui est la plus complète. En théorie, le calcul est simple, mais devient impraticable pour des problèmes comportant des millions de solutions ainsi que des boucles dans le graphe HPDIC. Notre but est donc de proposer une solution au problème de l'explosion combinatoire des solutions en utilisant des stratégies à différents niveaux. Nous proposons aussi une métrique originale pour évaluer la progression de l'élève qui peut se calculer facilement et rapidement en parcourant un arbre HPDIC.

Le deuxième objectif de ce chapitre est de présenter le module d'ordonnancement des inférences, qui vise à produire une séquence de pistes de résolution adaptée à l'état cognitif de l'élève. Ce module utilise les données du MIA pour construire une liste ordonnée d'inférences qui est mise à jour à la suite de chaque action de l'élève. L'ordre est déterminé selon la progression des états cognitifs présumés de l'élève et est utilisé par la couche GMD (voir chapitre 6) pour produire les messages du système tutoriel. Notre but est donc de proposer un algorithme qui permet d'ordonner les inférences dans différentes situations. Un modèle d'ordonnancement a été proposé par des didacticiens à la suite de l'observation du travail des enseignants en classe.

Dans ce chapitre, nous présentons d'abord une brève revue de la littérature concernant les systèmes tutoriels en géométrie existants. Nous nous intéressons plus particulièrement aux stratégies qui ont été utilisées pour limiter le nombre de solutions et au choix de la prochaine étape proposée à l'élève en cas de blocage. La section suivante propose différentes stratégies que nous avons utilisées pour circonscrire et gérer l'explosion combinatoire du nombre de solutions pour les problèmes complexes dans le but de ne pas restreindre les solutions acceptables. Pour illustrer notre propos, nous utilisons le graphe du problème du rectangle (figure 5.2). À partir des résultats obtenus, nous proposons une méthode originale, applicable aux arbres HPDIC, pour évaluer la progression de l'élève. Nous décrivons aussi l'algorithme qui permet de retrouver la solution la plus avancée.

La dernière section théorique présente l'algorithme d'ordonnancement des inférences. Nous montrons, à l'aide d'exemples, comment l'algorithme se comporte selon que l'on a reconnu la stratégie de l'élève ou non. Nous présentons aussi une justification intuitive de nos choix. Par la suite, nous faisons part des résultats obtenus par l'utilisation de nos stratégies de gestion des solutions pour les cinq problèmes implantés dans QEDX. Ces stratégies ont permis de rendre possible la résolution, par les élèves, de tous les problèmes implantés à l'interface de QEDX. Nous proposons, de plus, un exemple typique de résolution qui permet d'illustrer les écarts observés, lors de simulations, entre les différentes méthodes d'évaluation de la progression de l'élève. Nous terminons par une discussion des avantages, mais aussi des limites de nos solutions et de notre stratégie d'ordonnancement avant de passer à la conclusion.

5.2 Retour sur les principales stratégies d'aide recensées

Notre objectif, lié à l'élaboration de QEDX, est de produire un système qui assiste l'enseignant dans sa tâche, sans pour autant nous soucier, pour l'instant, de l'effet que celui-ci aura sur la note de l'élève. Cependant, l'analyse des résultats de l'élève, à l'aide de prétests et de post-tests, correspond à l'approche traditionnelle. Il est donc difficile de comparer les performances de QEDX avec celles des autres systèmes tutoriels en géométrie. Malgré cette différence, tous les tuteurs en géométrie ont des points en commun, car ils traitent du même contenu mathématique. Dans le cadre de notre revue de littérature, nous allons donc nous concentrer sur la gestion des solutions et sur le choix de la prochaine étape à proposer à l'élève lors d'un blocage. Nous verrons de quelle façon ces deux sujets ont été traités, pour chacun des principaux systèmes, et nous ferons une critique de leurs décisions. Nous ne nous attardons pas sur les messages particuliers proposés, mais bien sur la stratégie de résolution privilégiée par les différents logiciels.

Plusieurs systèmes utilisent des démonstrateurs automatiques pour analyser les inférences de manière locale (DEFI (Gras, 1988; Almouloud, 1992), Cabri-Euclide (Luengo, 2005, 1997)) ou la preuve complète fournie par l'élève (Baghera (Webber et al., 2001; Balacheff et al., 2003)). Dans les deux cas, il n'est pas nécessaire d'énumérer toutes les preuves, car le travail se fait dans un contexte local. En effet, même dans le cas d'une preuve complète, le moteur de déduction valide chaque pas individuellement pour vérifier si la preuve est correcte. Il se concentre donc sur la preuve proposée et n'a pas besoin de s'attarder aux autres preuves. En travaillant de manière locale ou sur une preuve en particulier, on évite donc le problème de l'explosion combinatoire lors de l'énumération des preuves. Cependant, ces systèmes n'offrent habituellement pas d'aide lorsque l'élève se trouve bloqué lors de la résolution, car ils ne sont pas en mesure de deviner la stratégie probable de ce dernier. Dans le cadre de QEDX, nous

voulons proposer une aide qui tient compte de la stratégie de l'élève. Il nous faut donc être en mesure de comparer les différentes solutions.

En ce qui concerne les systèmes qui offrent véritablement une aide intelligente pour des problèmes en géométrie, on note, entre autres, le Geometry Tutor (Anderson et al., 1985; Ritter et al., 2010), l'Advanced Geometry Tutor (Matsuda et VanLehn, 2005; Matsuda, 2004), Géométrix (Gressier, 2015), etc. Ces systèmes sont, en général, en mesure de proposer une stratégie à un élève qui serait bloqué lors de la résolution d'un problème. Cependant, une étude réalisée par Matsuda et VanLehn (2003) montre que ces systèmes proposent rarement une stratégie d'aide qui correspond à celle utilisée par les enseignants. De plus, ils ne permettent pas une exploration libre du problème et forcent l'élève à travailler des hypothèses vers la conclusion (chaînage avant) ou dans l'ordre inverse (chaînage arrière). Les auteurs de ces logiciels s'attendent donc à ce que les élèves soient en mesure de rédiger leur preuve sans avoir eu l'occasion d'explorer le problème, ce qui ne respecte pas la stratégie typique de résolution. Par contre, l'utilisation du chaînage avant ou arrière permet d'éviter l'énumération de toutes les preuves. En effet, la stratégie de l'élève est déterminée par les choix qu'il fait lorsqu'il atteint un embranchement dans l'arbre des solutions et il ne peut plus changer de stratégie par la suite. Par conséquent, il n'est pas nécessaire de prédire la stratégie utilisée, car elle est en fait construite au cours de la résolution. Pour aider l'élève bloqué, on lui propose habituellement de travailler sur le noeud qui mènera à la solution la plus courte sans lui proposer d'autres stratégies. Enfin, on propose des problèmes relativement simples pour aider à limiter le nombre de solutions possibles. Ces choix permettent donc d'éviter les problèmes d'explosion combinatoire et forcent l'élève à produire une solution optimale. Cependant, dans le cadre de QEDX, nous ne voulons pas imposer d'ordre de résolution, car nous voulons que l'élève puisse vraiment explorer diverses stratégies avant de rédiger sa preuve finale.

Le logiciel ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991) permet à l'élève d'explorer le problème en acceptant des pas de preuve qui ne sont pas strictement en chaînage avant ou arrière. Cependant, lorsque de l'aide est demandée, le logiciel fonctionne en chaînage avant et détermine un seul noeud pour lequel il fournit des indices. Il permet donc à l'élève d'explorer différentes pistes, mais ne l'aide que pour la stratégie optimale déterminée. Pour limiter le nombre de solutions, les auteurs ont choisi des problèmes simples et ont introduit le concept de «Diagram Configuration (DC)» (Koedinger et Anderson, 1990) qui correspond aux étapes clés d'une démonstration telles que déterminées par les experts. Dans ANGLE, une solution est donc un ensemble de DC que l'on relie ensuite avec les déductions nécessaires. La stratégie d'aide en chaînage avant, combinée à l'utilisation des DC, permet donc de limiter le nombre de solutions à analyser pour être en mesure de guider l'élève. Dans QEDX, nous voulons éviter une aide en chaînage avant et l'utilisation de structures utilisées par les

experts, car nous voulons que les élèves construisent leurs propres structures. Cependant, l'utilisation d'étapes clés peut être utile pour limiter l'espace des solutions à analyser.

En permettant l'exploration du problème et en ayant recours à la reconnaissance de plan (ou de stratégie), les concepteurs du logiciel Mentoniez (Py, 1996, 2001) ont fait face au même problème que QEDX. En effet, même s'ils ne proposent pas une aide pour indiquer la prochaine étape que l'élève devrait réaliser pour faire avancer sa preuve, ils identifient la stratégie de l'élève pour fournir des messages d'aide plus précis. Étant donné que l'élève peut explorer le problème dans un ordre quelconque, il est nécessaire d'analyser chacune des preuves possibles pour déterminer la stratégie la plus probable. Pour limiter le nombre de solutions à traiter, un travail préalable a été introduit pour extraire un nombre restreint de solutions parmi toutes celles qui sont possibles. On ne conserve donc que les solutions les plus courtes qui comprennent un nombre maximal d'inférences déterminées. Mentoniez est donc en mesure de reconnaître les solutions les plus courtes et refusera les solutions trop longues. Or, il n'y a aucune raison didactique de refuser des solutions arbitrairement trop longues. La solution de Mentoniez pour réduire le nombre de solutions n'a donc pas été retenue dans le cadre de l'élaboration de QEDX.

L'utilisation de différentes stratégies d'exploration de données a aussi été envisagée. Dans ce domaine, Deep Thought (Barnes et Stamper, 2010; Stamper, 2010) est le premier, et actuellement le seul, système à utiliser l'exploration des données afin de proposer une aide intelligente pour la résolution de problèmes de preuves mathématiques. Dans Deep Thought, les différentes solutions possibles pour un problème sont extraites d'un corpus de travaux d'élèves. En fait, il construit un diagramme d'états dans lequel chaque état représente un moment de la résolution et contient l'ensemble des énoncés qui ont été produits pour y arriver. Chaque nouvel énoncé inscrit permet de créer une transition vers un nouvel état. On associe, de plus, des statistiques, issues de l'analyse du corpus de travaux, à chacun des états et des transitions qui indiquent, par exemple, la probabilité de compléter la démonstration ou de se diriger vers un chemin sans issues. La stratégie d'aide de Deep Thought consiste donc à déterminer l'état dans lequel se trouve l'élève et à lui proposer d'inscrire l'énoncé qui correspond à la transition maximisant ses chances de réussite. Cette approche est intéressante, car elle se base sur des travaux réels et permet de modéliser les erreurs courantes des élèves. Cependant, elle nécessite la cueillette et l'analyse d'une grande quantité de travaux d'élèves que nous n'avons pas à notre disposition. De plus, dans le contexte d'une exploration libre du problème, le nombre d'états risque d'augmenter rapidement, car les énoncés valides inscrits ne font pas nécessairement partie de la solution visée, ce qui produit des parcours de résolution différents même s'il en résulte une solution unique. Par contre, dans la suite des recherches, une telle approche pourrait être testée en utilisant les fichiers journaux générés par QEDX.

La revue des systèmes tutoriels existants nous a fait découvrir différentes méthodes pour limiter le nombre de solutions à traiter. Contraindre l'élève dans une progression en chaînage avant ou arrière et proposer des problèmes simples sont des stratégies répandues. Toutefois, elles limitent fortement le choix des problèmes et elles n'en permettent pas l'exploration. On peut aussi restreindre le nombre de solutions selon la longueur de celles-ci, mais on risque de refuser arbitrairement des solutions valides. Aucune des solutions présentées, utilisées par les autres systèmes pour limiter l'explosion combinatoire des solutions, ne nous a semblé acceptable dans le cadre de l'élaboration de QEDX. En ce qui concerne la stratégie que l'on propose à un élève bloqué lors de la résolution d'un problème, on se restreint habituellement à la prochaine étape optimale. Si cette dernière ne correspond pas à la démarche de l'élève, l'aide risque de ne pas être pertinente. Dans le but de permettre d'explorer différentes stratégies, nous ne voulons pas limiter l'aide seulement au chemin optimal, mais proposer différentes stratégies. Dans les sections suivantes, nous décrivons les méthodes utilisées pour limiter l'explosion combinatoire et gérer l'ensemble des solutions. Nous exposons aussi la solution proposée dans le but de fournir une aide qui offre à l'élève différentes pistes de solution pour un épisode de blocage.

5.3 Contrôle de l'explosion combinatoire des solutions

À la suite de l'analyse des différentes stratégies proposées dans la littérature pour gérer l'ensemble des solutions pour un problème complexe, il est clair que nous devons mettre au point une nouvelle approche dans le cadre de l'élaboration de QEDX. Dans cette section, nous présentons la stratégie qui a été établie et implantée. Elle s'applique à plusieurs niveaux qui s'étendent de la nomenclature des éléments de géométrie jusqu'à la transformation des graphes HPDIC en une forêt d'arbres équivalents. La structure de cette section suit cet ordre hiérarchique. Pour chaque élément de la solution proposée, nous justifions nos choix et nous montrons comment ils permettent de contrôler l'explosion combinatoire du nombre de solutions. La décomposition en arbres est utilisée pour l'évaluation des démonstrations telle que proposée dans la section suivante. Pour illustrer notre propos, nous utilisons le problème du rectangle (figure 5.1), qui est un problème simple à résoudre, mais riche en ce qui concerne les défis informatiques qu'il présente.

5.3.1 Traitement des énoncés

Pour résoudre un problème, un élève doit entrer des énoncés qui correspondent à différents éléments de sa démonstration. Lorsqu'il inscrit des justifications (propriétés ou définitions), il ne fait habituellement pas référence à des éléments de la figure, donc pour chaque justification

Démontrer qu'un quadrilatère ayant trois angles droits est un rectangle.

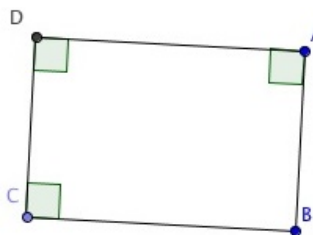


Figure 5.1 Problème du rectangle

correspond un énoncé unique. Par contre, dans le cas des résultats intermédiaires, il est possible de produire plusieurs énoncés équivalents pour décrire une observation unique. Par exemple, pour l'angle BCD de la figure 5.1, nous pourrions proposer les énoncés suivants :

Énoncé 1 : CD est perpendiculaire à BC

Énoncé 2 : BC est perpendiculaire à DC

Énoncé 3 : La mesure de l'angle DCB = 90

Énoncé 4 : L'angle BCD est droit

Ces quatre énoncés sont un sous-ensemble des énoncés possibles pour décrire l'angle droit. Nous observons premièrement que pour chaque élément de géométrie (angle, segment, quadrilatère, triangle, etc.), plusieurs noms équivalents peuvent lui être associés. Dans notre exemple, nous avons le segment CD (Énoncé 1) qui peut aussi être nommé DC (Énoncé 2). Pour les quadrilatères, huit noms équivalents sont possibles selon le point de départ et le sens de rotation. Il est donc impensable de demander à un usager d'entrer systématiquement tous les noms possibles des objets géométriques et espérer qu'il ne fasse aucune erreur ou omission. Pour résoudre le problème, nous avons choisi de transformer les noms sous une forme canonique. Par conséquent, nous avons conçu un algorithme qui permet de réorganiser les noms des objets géométriques tout en respectant les particularités de chacun. Par exemple, l'ordre des points permettant de représenter un carré est préservé.

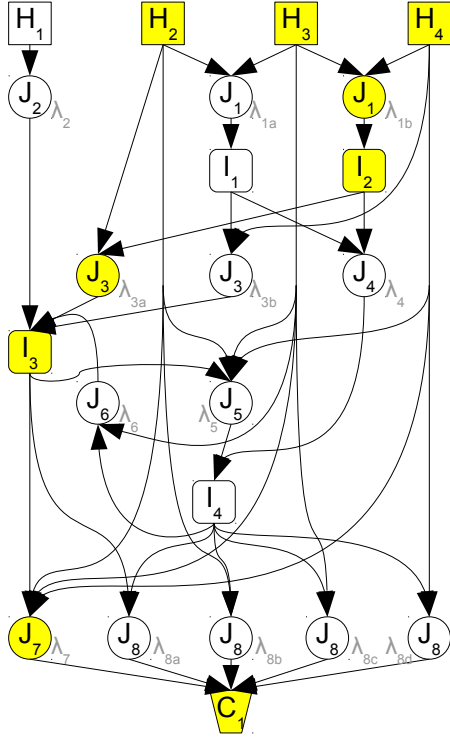
En rendant les noms des objets géométriques canoniques, il est possible de réduire le nombre d'énoncés nécessaires pour décrire le problème à résoudre. Cependant, en comparant l'énoncé 1 et l'énoncé 2, on remarque qu'ils sont presque identiques si nous utilisons les noms canoniques. En effet, il s'agit du même énoncé dont seule la position des objets à l'intérieur de celui-ci a été modifiée. Les éléments sont donc commutatifs alors que pour l'énoncé 3, ce n'est pas le cas, car BCD et 90 ne peuvent être déplacés. Pour éviter la multiplication inutile des énoncés, nous avons aussi opté pour une écriture canonique de chacun. La mise en oeuvre de notre algorithme permet donc de réorganiser les énoncés selon les contraintes fournies par

les didacticiens. Par conséquent, l'écriture canonique des énoncés est un premier pas pour contenir l'explosion combinatoire.

En appliquant notre stratégie sur les quatre énoncés de notre exemple, nous retrouvons trois énoncés canoniques équivalents, mais différents. Il n'est plus possible de réduire cette liste davantage sans perdre des informations essentielles à la compréhension des solutions potentielles de l'élève. Cependant, nous voulons éviter de créer des inférences pour toutes les combinaisons admissibles d'énoncés. Par exemple, si chacun des deux antécédents d'une inférence a trois énoncés équivalents et s'il en va de même pour le conséquent, il faudrait créer 27 inférences distinctes pour énumérer toutes les possibilités. Pour résoudre ce problème, nous avons associé à chaque élément de l'inférence un ensemble d'énoncés équivalents. Cette solution est acceptable, car tous les énoncés d'un ensemble correspondent à un concept géométrique commun, donc chaque élément de l'inférence fait référence à un concept et non à un énoncé précis. De cette façon, il est possible de reconnaître les différents énoncés que pourrait produire l'élève tout en minimisant le nombre d'inférences, donc de solutions, nécessaires pour représenter l'ensemble des solutions admissibles pour un problème donné.

Dans le cadre de QEDX, les inférences utiles à la résolution d'un problème sont encodées et regroupées sous forme d'un graphe HPDIC (voir chapitre 4). Dans celui-ci, chaque élément d'une inférence est représenté par un noeud qui contient un ensemble d'énoncés équivalents au concept associé. Les noeuds de justification (J) forment le pivot de chaque inférence, donc il y a autant de noeuds J qu'il y a d'inférences utiles pour résoudre le problème. Tous les liens entrants dans un noeud J sont connectés aux antécédents (H ou I) de l'inférence, alors que l'unique lien sortant pointe vers le conséquent (I ou C). Les noeuds I jouent donc un double rôle, car ils sont le conséquent de certaines inférences et ils sont recyclés comme antécédents dans d'autres, ce qui permet de construire une démonstration. Ainsi, le graphe HPDIC permet de représenter l'ensemble des solutions d'un problème donné de façon plutôt compacte. Pour extraire une solution particulière, nous choisissons premièrement une seule inférence dont le conséquent est la conclusion. Pour chacun de ses antécédents qui ne sont pas des hypothèses, nous sélectionnons aussi une seule inférence permettant de le générer. En appliquant de façon récursive ce procédé, nous obtenons une solution complète. L'ensemble des solutions provient donc de la combinatoire de tous ces choix. En résumé, les stratégies proposées jusqu'ici ont pour objectif de minimiser le nombre de noeuds dans le graphe HPDIC et nous croyons qu'il est très difficile de le réduire davantage sans risquer de perdre de l'information.

Le graphe HPDIC, présenté à la figure 5.2, contient les différentes solutions pour le problème du rectangle. Nous indiquons, dans le tableau à droite, un énoncé représentatif pour chacun des noeuds. Cependant, comme nous l'avons mentionné, chaque noeud contient un ensemble



(a) Graphe HPDIC

ID	Énoncé
H_1	ABCD est un quadrilatère
H_2	BC est perpendiculaire à CD
H_3	AD est perpendiculaire à CD
H_4	AB est perpendiculaire à AD
I_1	AD est parallèle à BC
I_2	AB est parallèle à CD
I_3	AB est perpendiculaire à BC
I_4	ABCD est un parallélogramme
C_1	ABCD est un rectangle
J_1	Deux droites perpendiculaires à une même troisième sont parallèles entre elles
J_2	La somme S des mesures des angles intérieurs d'un polygone à n côtés est donnée par la formule : $S = (n - 2) \times 180^\circ$
J_3	Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre
J_4	Un parallélogramme est un quadrilatère dont les côtés opposés sont parallèles
J_5	Un quadrilatère dont les angles opposés sont congrus est un parallélogramme
J_6	Les angles opposés d'un parallélogramme sont congrus
J_7	Un rectangle est un quadrilatère ayant quatre angles droits
J_8	Un parallélogramme ayant un angle droit est un rectangle

(b) Énoncés associés aux noeuds

- Comme **AB est perpendiculaire à AD**, et **AD est perpendiculaire à CD**, alors, selon la propriété qui dit «**Deux droites perpendiculaires à une même troisième sont parallèles entre elles**», **AB est parallèle à CD**. (λ_{1b})
- Comme **BC est perpendiculaire à CD**, et **AB est parallèle à CD**, alors, selon la propriété qui dit «**Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre**», **AB est perpendiculaire à BC**. (λ_{3a})
- Comme **AB est perpendiculaire à AD**, **AD est perpendiculaire à CD**, **BC est perpendiculaire à CD**, et **AB est perpendiculaire à BC**, alors, selon la propriété qui dit «**Un rectangle est un quadrilatère ayant quatre angles droits**», **ABCD est un rectangle**. (λ_7)

(c) Exemple d'une démonstration (colorée dans le graphe)

Figure 5.2 Solutions pour le problème du rectangle

d'énoncés équivalents. Ainsi, les énoncés utilisés dans l'exemple plus haut proviennent du noeud H_2 . Nous avons, de plus, attribué un identifiant unique à chaque inférence afin de les distinguer dans la suite du document. Chaque identifiant ($\lambda_{1a}, \lambda_{1b}, \dots, \lambda_{8d}$) est inscrit en gris et est adjacent au noeud de justification formant le pivot de l'inférence. Nous proposons enfin un exemple de démonstration pour lequel nous avons coloré les noeuds correspondants du graphe HPDIC en jaune et avons indiqué, pour chacune des inférences, l'identifiant rattaché entre parenthèses. Le graphe de la figure 5.2 contient 13 inférences qui se combinent pour former un total de 23 solutions possibles, telles qu'implantées dans QEDX. Cependant, nous avons bel et bien un graphe, car λ_5 et λ_6 forment un cycle passant par I_4 et I_3 . Le fait d'avoir un cycle nous oblige à vérifier chaque solution produite par notre algorithme pour la valider,

ce qui peut demander un temps de calcul non négligeable pour des problèmes plus complexes. Il est aussi impossible de stocker l'ensemble des solutions valides en mémoire pour certains problèmes. Notre but initial était donc de pouvoir évaluer le pourcentage d'achèvement de chaque solution sans avoir à les énumérer ou à les recalculer après chaque action de l'élève.

5.3.2 Décomposition d'un graphe en arbres

Le pourcentage d'achèvement est contenu dans le MIA (voir chapitre 4). Ce dernier inclut, de plus, la chronologie des actions de l'élève afin de modéliser son état cognitif. Cependant, même si le MIA indique une structure qui enregistre le pourcentage pour chaque solution, seule la solution la plus avancée est utilisée par les autres modules de QEDX. Notre but a donc été révisé et consiste à retrouver uniquement la solution la plus avancée. Les méthodes mises en oeuvre pour évaluer l'avancement des solutions sont décrites dans la prochaine section. Cependant, l'heuristique qui a été retenue dans le cadre de QEDX fonctionne seulement sur des arbres HPDIC. Pour être en mesure de l'utiliser, et ainsi limiter les problèmes reliés à l'explosion combinatoire du nombre de solutions, il fallait transformer notre graphe en un ou plusieurs arbres. Nous avons donc conçu et implanté l'algorithme 5.1 qui permet de créer la forêt équivalant au graphe HPDIC de départ, qui est ici défini par l'ensemble des solutions qu'il contient (*Sol*). Dans cet algorithme, nous utilisons trois fonctions qui ne sont pas décrites explicitement, soit FUSIONNE, qui permet de fusionner deux arbres, CONTIENTCYCLE, qui vérifie la présence de cycles dans un graphe et REMPLACE, qui remplace un arbre par un autre dans la forêt. Ces dernières fonctions ont fait l'objet d'optimisations afin de réduire le temps requis pour la création de la forêt.

Notre stratégie consiste donc à greffer une solution à un arbre (FUSIONNE) et à vérifier ensuite si cette opération crée un cycle (CONTIENTCYCLE). Nous testons une solution sur tous les arbres de la forêt, jusqu'à ce qu'elle puisse être ajoutée sans créer de cycle (TROUVERARBRECOMPATIBLE). Dans le cas contraire, la solution devient un nouvel arbre de la forêt, car chaque solution est, par définition, un arbre HPDIC. Il faut répéter l'opération pour toutes les solutions valides (*Sol*) pour un problème donné afin de créer la forêt équivalant au graphe HPDIC original. Notre algorithme ne permet donc pas d'éviter d'énumérer toutes les solutions. Cependant, la création de la forêt n'est exécutée qu'une seule fois pour chaque graphe HPDIC et les arbres résultants sont sauvegardés. Lors du choix d'un problème, la forêt rattachée est alors lue et utilisée par QEDX, donc elle n'a pas à être recrée.

Nous allons maintenant prouver que notre algorithme est valide en démontrant qu'aucune solution n'est perdue ou n'émerge lors de son exécution. Nous notons, premièrement, que c'est uniquement la fonction FUSIONNE qui a le potentiel d'altérer les solutions, car c'est la seule

```

fonction CRÉERFORÊT
  Forêt  $\leftarrow \{\}$ 
  pour tout  $s \in Sol$  faire
    ArbreCible = TROUVERARBRECOMPATIBLE(Forêt,  $s$ )
    si ArbreCible  $\neq \emptyset$  alors
      Forêt.REPLACE(ArbreCible, FUSIONNE(ArbreCible,  $s$ ))
    sinon
      Forêt  $\leftarrow$  Forêt  $\cup s$ 
    fin si
  fin pour
  retourne Forêt
fin fonction

fonction TROUVERARBRECOMPATIBLE(Forêt  $f$ , Solution  $s$ )
  pour tout  $Arbre \in f$  faire
    si  $\neg$ CONTIENTCYCLE( FUSIONNE( $Arbre$ ,  $s$ ) ) alors
      retourne  $Arbre$ 
    fin si
  fin pour
  retourne  $\emptyset$ 
fin fonction

```

Algorithme 5.1 Création d'une forêt d'arbres à partir d'un graphe HPDIC

qui manipule la structure des arbres. D'une part, il est impossible que cette fonction détruise une solution préalablement présente dans un arbre. En effet, selon la définition théorique proposée, la fonction FUSIONNE réalise, en fait, l'union de deux ensembles d'inférences, donc les deux ensembles de départ sont inclus dans l'ensemble résultant. Par conséquent, tous les sous-ensembles d'inférences, représentant les différentes solutions incluses dans l'arbre, sont toujours présents à la suite de l'union des ensembles. Nous pouvons donc conclure qu'aucune solution n'est perdue lors de l'exécution de la fonction FUSIONNE. D'autre part, les arbres produits ne contiennent pas de solutions qui ne sont pas aussi incluses dans le graphe HPDIC d'origine. Pour le démontrer, il faut se rappeler que chaque arbre de la forêt est constitué de l'union de plusieurs solutions et que ces dernières sont extraites du graphe d'origine. Il en résulte donc que seules les inférences faisant partie du graphe peuvent apparaître dans la définition des arbres. Par conséquent, chaque arbre est défini par un sous-ensemble d'inférences qui est inclus dans l'ensemble du graphe HPDIC d'origine. Cependant, seuls des sous-ensembles qui ne sont pas contenus dans ce dernier permettent de produire de nouvelles solutions. Nous pouvons alors conclure qu'aucune nouvelle solution ne peut émerger d'un arbre, car chaque sous-ensemble d'inférences inclus dans un arbre fait partie du graphe.

Étant donné que les traitements réalisés par la fonction FUSIONNE sont valides, il ne nous reste plus qu'à nous assurer que toutes les solutions contenues dans le graphe HPDIC sont ajoutées à la forêt. Dans notre algorithme, ce travail est pris en charge par la boucle qui

parcourt systématiquement l'ensemble des solutions ($s \in Sol$). Pour chacune, nous l'ajoutons dans la forêt, soit en l'intégrant à un arbre existant ou en l'utilisant en tant que nouvel arbre. Par conséquent, chaque solution se retrouve nécessairement dans au moins un des arbres. Dans le cas extrême où toutes les tentatives de fusion créent un cycle, nous aurons une forêt avec un nombre d'arbres égal au nombre de solutions du problème. À l'autre extrême, si notre graphe de départ est en fait un arbre, nous retrouverons le même arbre. Nous venons donc de confirmer que notre algorithme produit une forêt qui contient exactement les mêmes solutions valides que notre graphe HPDIC d'origine. Cependant, chaque arbre ne contient qu'un sous-ensemble de celles-ci qui dépend de l'ordre d'énumération des solutions et de la stratégie d'ajout dans les arbres.

Par exemple, en appliquant notre algorithme au graphe HPDIC du problème du rectangle (figure 5.2), nous obtenons une forêt, parmi toutes celles qui sont possibles, contenant deux arbres (figure 5.3). Nous remarquons que le cycle passant par λ_5 et λ_6 a été brisé, car chacune des inférences le formant se retrouve isolée dans son arbre.

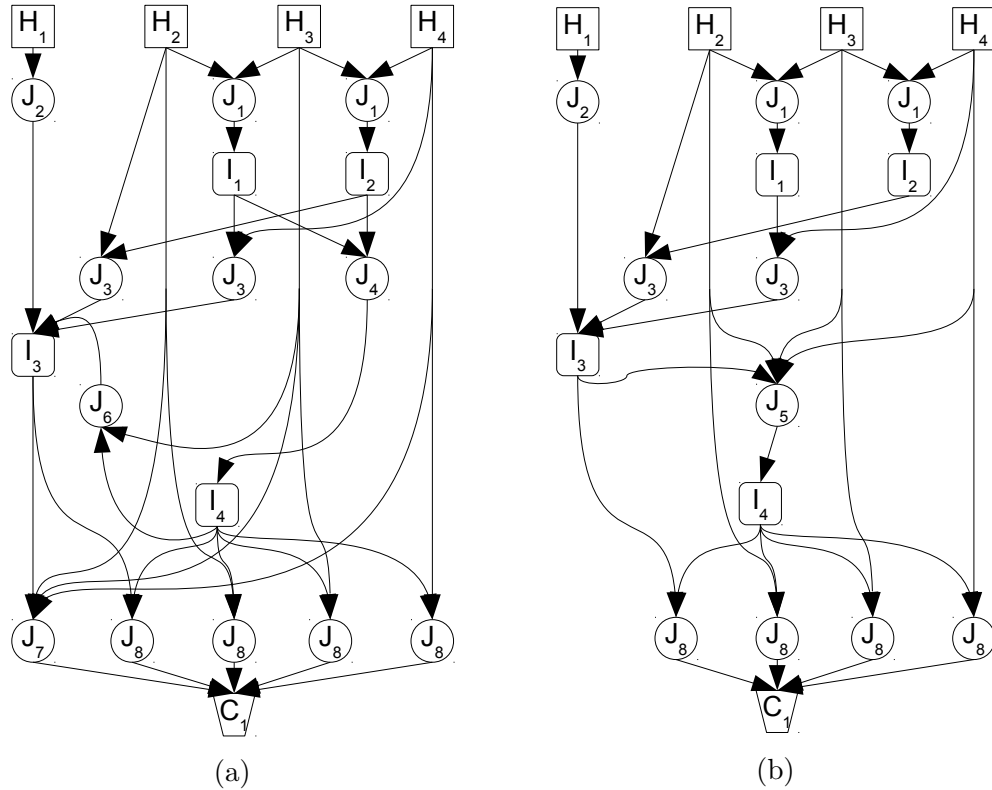


Figure 5.3 Arbres extraits du graphe pour le problème du rectangle

Dans cette section, nous avons exposé les différentes stratégies que nous avons utilisées pour limiter l'explosion combinatoire du nombre de solutions à traiter dans le cadre du logiciel

QEDX. Nous avons premièrement utilisé une écriture canonique pour le nom des objets géométriques et pour les énoncés que l'élève propose à l'interface du logiciel. Nous avons aussi permis de fournir plusieurs phrases équivalant à un concept commun, ou noeud du graphe HPDIC. À l'aide de ces stratégies, nous avons été en mesure de minimiser le nombre de noeuds nécessaires pour représenter l'ensemble des solutions pour un problème donné. À un niveau plus élevé d'abstraction, nous avons converti notre graphe HPDIC en une forêt d'arbres équivalente. Cette transformation nous permet d'utiliser l'heuristique présentée dans la section suivante pour retrouver la solution la plus avancée tout en évitant d'avoir à énumérer et stocker l'ensemble des solutions en permanence.

5.4 Évaluation de la progression

Dans la section précédente, une des stratégies utilisées pour contrôler l'explosion combinatoire consistait à créer une forêt d'arbres à partir d'un graphe HPDIC dans le but d'utiliser une heuristique qui permet d'estimer la progression de l'élève sans avoir à calculer l'ensemble des solutions. Avant toute chose, il est important de rappeler que QEDX utilise seulement la solution la plus avancée et le pourcentage de progression associé pour générer les messages du tuteur. Il n'est donc pas nécessaire de connaître la progression de chaque solution individuellement, même si celles-ci sont toutes représentées dans le MIA. Il est par contre primordial de détecter une solution complète dès son achèvement. Dans cette section, nous présentons trois méthodes pour mesurer la progression de l'élève. La première méthode, relative aux énoncés, correspond au calcul de progression que nous considérons comme exact. Les deux autres sont des heuristiques implantées au sein de QEDX dans le but d'estimer la progression, étant donné des contraintes temporelles ou des ressources limitées. Nous avons choisi d'utiliser trois solutions du problème du rectangle (S01, S04, S23) afin d'illustrer notre propos. Les noeuds faisant partie de chacune de ces solutions sont colorés en jaune dans le graphe correspondant de la figure 5.4. Dans cette dernière, les valeurs en gris sont relatives à la métrique de la méthode par propagation décrite plus loin.

5.4.1 Méthode par Énoncé (ME)

La méthode par énoncé a été définie formellement dans le chapitre 4. Elle consiste à faire le rapport entre le nombre d'énoncés distincts activés par l'élève dans une solution donnée et le nombre nécessaire pour produire la solution complète. Nous la qualifions d'exacte, car elle représente le travail réel nécessaire pour produire la démonstration. En effet, si l'on obtient une valeur de 50%, cela signifie que nous avons inscrit la moitié des énoncés nécessaires pour la compléter et qu'il nous en reste autant à produire.

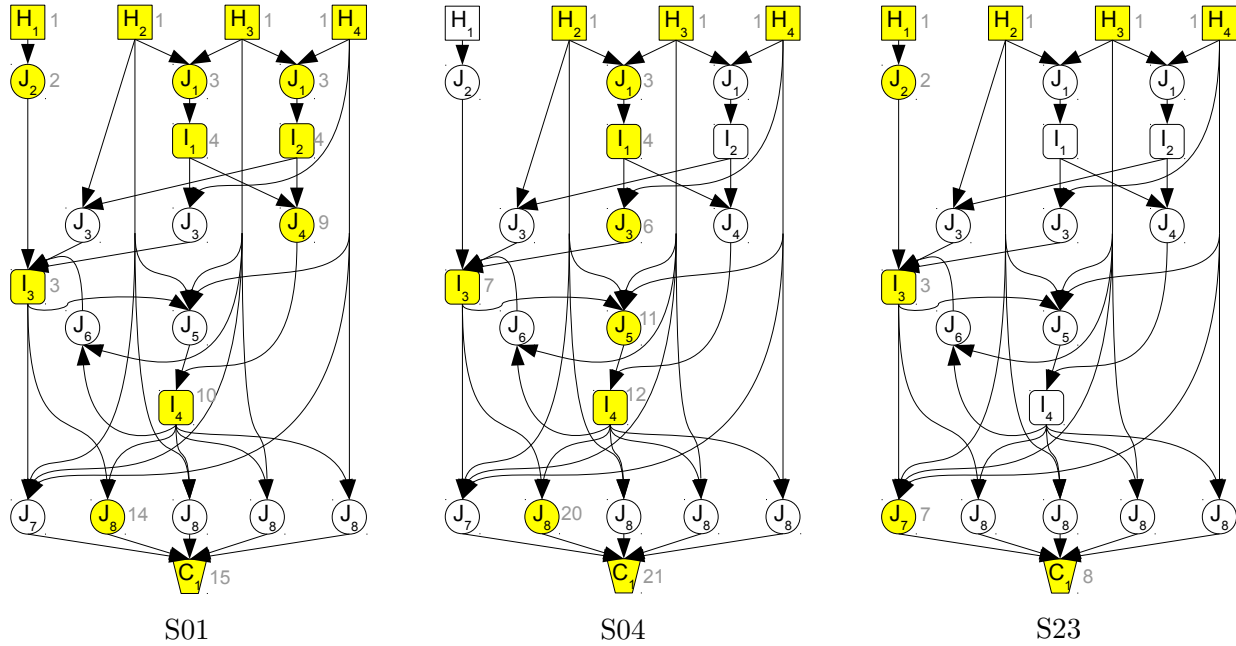


Figure 5.4 Trois solutions possibles au problème du rectangle

Concernant le graphe HPDIC du problème du rectangle (figure 5.2), il faut se rappeler que des noeuds de justification ayant le même indice numérique correspondent à un même énoncé de justification. Par exemple, un élève qui inscrit «Un parallélogramme ayant un angle droit est un rectangle» à l'interface de QEDX activera tous les noeuds J_8 , soit quatre noeuds. Identifier précisément un des noeuds est actuellement impossible, car nous n'avons pas les informations nécessaires par rapport au contexte de l'affirmation. En effet, le fait de permettre d'inscrire les énoncés dans un ordre quelconque simplifie la résolution du problème, mais cause une perte d'informations qui nous permettraient d'identifier avec certitude les intentions de l'élève. Par contre, tous les autres noeuds contiennent des énoncés distincts.

La liste suivante correspond aux énoncés nécessaires pour compléter chacune des trois solutions utilisées à titre d'exemple :

- S01 : $H_1, H_2, H_3, H_4, I_1, I_2, I_3, I_4, C_1, J_1, J_2, J_4, J_8$: 13 énoncés
- S04 : $H_2, H_3, H_4, I_1, I_3, I_4, C_1, J_1, J_3, J_5, J_8$: 11 énoncés
- S23 : $H_1, H_2, H_3, H_4, I_3, C_1, J_2, J_7$: 8 énoncés

5.4.2 Méthode par Noeud (MN)

Lors de la conception des graphes HPDIC, nous avons choisi de traiter tous les noeuds de justification comme étant uniques, même si certains contiennent des énoncés identiques,

car ils forment des inférences distinctes. Cette décision s'est avérée utile lors de la mise en oeuvre de l'algorithme d'énumération des solutions (voir chapitre 4), étant donné qu'elle simplifie le parcours dans le graphe. Cependant, l'ensemble des noeuds uniques représentant une solution peut alors contenir des noeuds faisant référence à des énoncés communs. Dans ce cas, pour être en mesure de calculer ME, il est nécessaire de détecter et d'ignorer les énoncés dupliqués. Cette détection, qui consiste à comparer des chaînes de caractères, demande un temps de traitement non négligeable pour des problèmes comportant quelques milliers de solutions. Dans le cadre de QEDX, un délai d'attente de plusieurs secondes avant d'obtenir une rétroaction de la part du tuteur n'est pas acceptable.

Toutes nos tentatives visant à optimiser la détection des énoncés dupliqués se sont avérées vaines. Nous avons donc éliminé cette détection, ce qui nous a amené à estimer la progression de l'élève en nous basant sur les noeuds uniques plutôt que sur les énoncés distincts. Par conséquent, ce nouvel algorithme a été nommé *méthode par noeud*. Celle-ci permet d'évaluer la progression directement à partir de chacune des solutions extraites du graphe sans que des traitements supplémentaires soient nécessaires. Elle consiste à calculer le rapport entre le nombre de noeuds activés par l'élève qui font partie de la solution et le nombre total de noeuds contenus dans cette dernière. Dans ce contexte, chaque résultat intermédiaire entré par l'élève est considéré par MN comme un pas de progression vers une solution complète, à l'instar de ME. En effet, par définition, chaque noeud de résultat intermédiaire contient un ensemble d'énoncés distincts et sera donc pris en compte une seule fois. Par contre, un même énoncé de justification peut apparaître dans plusieurs noeuds uniques de la solution. Lorsque l'élève produit une telle justification, plusieurs noeuds sont activés simultanément, ce qui est considéré, à tort par MN, comme une progression de plusieurs pas. Dans ce cas, MN surestime la progression de l'élève par rapport à ME. À l'inverse, l'écriture d'un énoncé qui n'est pas dupliqué produit une sous-estimation de la progression pour des solutions contenant plus de noeuds uniques que d'énoncés distincts. En effet, un nombre de noeuds plus élevé diminue l'apport relatif de chacun pour la production de la solution. Enfin, cette méthode de calcul permet, comme pour ME, de détecter les solutions complètes sans équivoque, même si l'estimation de la progression est moins précise.

Si nous reprenons l'exemple des trois solutions du problème du rectangle, la méthode MN nous donne la liste des noeuds suivants pour chacune :

- S01 : $H_1, H_2, H_3, H_4, I_1, I_2, I_3, I_4, C_1, J_1, J_1, J_2, J_4, J_8$: 14 noeuds
- S04 : $H_2, H_3, H_4, I_1, I_3, I_4, C_1, J_1, J_3, J_5, J_8$: 11 noeuds
- S23 : $H_1, H_2, H_3, H_4, I_3, C_1, J_2, J_7$: 8 noeuds

Nous observons que pour S04 et S23, le nombre de noeuds est égal au nombre d'énoncés, donc le calcul de ME ou de MN donne des résultats identiques. Cependant, dans le cas de S01, l'énoncé J_1 se retrouve dans deux inférences faisant partie de la solution, soit λ_{1a} et λ_{1b} . Il est alors inclus deux fois dans la liste et est, par conséquent, compté en double. C'est ce qui explique que le nombre de noeuds est supérieur au nombre d'énoncés pour S01. Dans ce cas, la progression évaluée par MN est de $1/14$ (7,1%) par énoncé, à l'exception de J_1 qui contribue à $2/14$ (14,2%) de la solution. Cette contribution est par contre constante pour ME, soit de $1/13$ (7,7%) par énoncé. Ces valeurs montrent que la surévaluation causée par la duplication d'énoncés est beaucoup plus importante que la sous-évaluation associée aux autres énoncés. Cependant, dans le cadre de QEDX, ce genre d'erreur est acceptable et la méthode MN a donc été utilisée avec succès.

5.4.3 Méthode par Propagation (MP)

Les méthodes de calcul précédentes nous permettent d'obtenir une métrique assez précise concernant la progression de l'élève. Cependant, pour être en mesure de trouver la solution la plus avancée, il est nécessaire de calculer la métrique choisie sur chaque solution individuellement. Cette stratégie est inefficace et impraticable pour des problèmes pouvant comporter plusieurs milliers, voire des millions, de solutions, alors que seule la solution la plus avancée nous intéresse. Nous avons donc décidé de proposer une nouvelle métrique qui est certes moins précise que les précédentes, mais pour laquelle il n'est pas nécessaire de parcourir la liste exhaustive de toutes les solutions pour retrouver la plus avancée.

L'heuristique que nous proposons consiste à propager une métrique qui permet d'estimer l'achèvement de solutions partielles à l'intérieur d'un arbre HPDIC. Nous la calculons de manière récursive pour tous les noeuds à partir des hypothèses jusqu'à ce que nous atteignons le noeud de conclusion. Par la suite, il s'agit de remonter dans l'arbre en utilisant la métrique calculée dans chaque noeud pour retrouver la solution la plus avancée. Concrètement, nous avons ajouté deux valeurs à chaque noeud de l'arbre HPDIC, soit CA qui représente une valeur cumulative du nombre de noeuds activés, ainsi que CP qui correspond au nombre cumulatif de noeuds parcourus. Ces deux valeurs sont associées à la solution partielle reliant les hypothèses au noeud courant.

$$N = \langle CA, CP \rangle \quad (5.1)$$

Ces valeurs se calculent différemment selon que l'on se trouve dans un noeud justification (N_J) ou dans un noeud résultat (N_R), contenant les hypothèses, les résultats intermédiaires et la conclusion. Premièrement, dans le cas N_J , on sait que pour avoir une inférence complète,

il faut écrire tous les antécédents. Le nombre de noeuds parcourus pour atteindre N_J sera donc la somme du nombre de noeuds parcourus pour chaque antécédent plus un.

$$N_J[CP] = \Sigma(Parents(N_J)[CP]) + 1 \quad (5.2)$$

En ce qui concerne le calcul de CA, on procède sensiblement de la même façon. En effet, on additionne les CA des antécédents, mais on incrémente le total d'une unité seulement si le noeud N_J est activé.

$$N_J[CA] = \Sigma(Parents(N_J)[CA]) + \begin{cases} 0 & \text{si } N_J \neg\text{activé} \\ 1 & \text{si } N_J \text{ activé} \end{cases} \quad (5.3)$$

Dans le cas de N_R , on sait que plusieurs inférences peuvent permettre de produire un même résultat. Or, pour produire une solution, une seule inférence, parmi toutes celles qui sont possibles, est requise. Nous devons donc choisir l'inférence qui correspond à la solution partielle la plus avancée. Pour ce faire, nous allons comparer les ratios CA/CP de tous les noeuds N_J ($Parents(N_R)$) qui permettent d'obtenir N_R et nous allons conserver celui ayant le ratio maximal. Dans le cas d'une égalité, nous privilégions la solution la plus courte (CP minimal).

$$BestPct(N_R) = \{n_j \mid n_j \in Parents(N_R) \wedge (n_j[CA]/n_j[CP]) = \max\{n[CA]/n[CP] \mid n \in Parents(N_R)\}\} \quad (5.4)$$

$$BestCh(N_R) = \{n_j \mid n_j \in BestPct(N_R) \wedge n_j[CP] = \min\{n[CP] \mid n \in BestPct(N_R)\}\} \quad (5.5)$$

L'ensemble $BestCh$ contient donc les N_J ayant des CA et des CP identiques pour un N_R donné. À ce stade, nous n'avons pas de critères pour identifier la solution qui correspond le mieux au travail de l'élève parmi celles-ci. Cette distinction n'est toutefois pas utile pour la propagation des valeurs de CA et CP, car ils sont de toute façon identiques pour toutes les solutions dans le groupe $BestCh$. La propagation des valeurs se produit de manière analogue aux noeuds N_J , sauf que l'on utilise seulement celle d'un parent au lieu de faire leur somme.

$$N_R[CA] = BestCh(N_R)[CA] + \begin{cases} 0 & \text{si } N_R \neg\text{activé} \\ 1 & \text{si } N_R \text{ activé} \end{cases} \quad (5.6)$$

$$N_R[CP] = BestCh(N_R)[CP] + 1 \quad (5.7)$$

En appliquant de façon récursive ces calculs sur chacun des noeuds, nous trouvons une estimation du pourcentage d'achèvement de la solution la plus avancée dans le noeud de conclusion. Cette méthode de calcul est moins précise que les précédentes, mais permet de détecter, sans équivoque, une solution complète dès qu'elle se présente, ce qui est essentiel dans le cadre de QEDX. En effet, un ratio CA/CP de 1, soit la valeur maximale, se propagera dans l'arbre HPDIC jusqu'à atteindre le noeud de conclusion.

Pour illustrer le fonctionnement du calcul par propagation, nous avons produit la figure 5.5. Celle-ci contient un des deux arbres HPDIC pour le problème du rectangle sur lequel nous avons indiqué, en jaune, des noeuds qui auraient pu être activés par un élève fictif. Nous avons juxtaposé à chaque noeud le ratio CA/CP, tel que calculé par notre méthode. La figure 5.5b montre la mise à jour des ratios à la suite de l'activation des noeuds J_3 indiqués en vert.

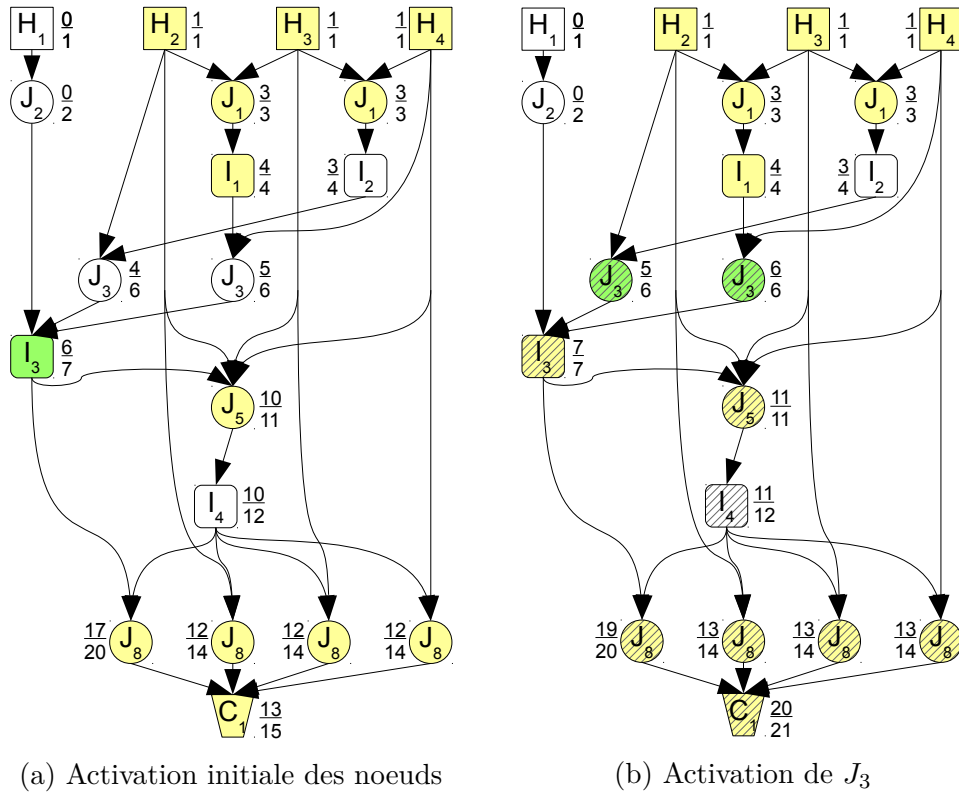


Figure 5.5 Exemple du calcul de l'avancement des solutions sur un arbre HPDIC

Pour retrouver la solution la plus avancée dans le cas de la figure 5.5a, nous partons de la conclusion avec un ratio de $13/15$. Étant donné que nous sommes dans un noeud N_R , il faut retrouver le N_J , parmi tous ceux qui sont possibles, qui a permis d'obtenir ce ratio. Nous cherchons donc un noeud ayant un CA de 12, soit un de moins que celui de la conclusion, étant donné qu'elle est activée, et un CP de 14. Ici, trois noeuds J_8 (λ_{8b} , λ_{8c} , λ_{8d}) contiennent

ce ratio, il n'est donc pas possible de les départager. Dans cet exemple, QEDX a choisi la solution utilisant λ_{8c} . Ensuite, nous ajoutons tous les noeuds parents de J_8 (cas d'un noeud N_J) à notre solution, soit I_4 et H_3 . En appliquant à rebours les équations, nous retrouvons la solution $\{\lambda_{8c}, \lambda_5, \lambda_{3b}, \lambda_{1a}\}$. Dans le cas de l'arbre de la figure 5.5b, nous trouvons la solution $\{\lambda_{8a}, \lambda_5, \lambda_{3b}, \lambda_{1a}\}$ (S04). Il est donc possible de retrouver la solution la plus avancée à l'intérieur d'un arbre HPDIC sans avoir à produire l'ensemble des solutions possibles pour un problème donné.

Nous avons affirmé précédemment que notre méthode de calcul par propagation est moins précise que ME et MN. Pour le démontrer, reprenons nos trois solutions, qui nous ont servi d'exemple, et voyons de quelle façon le nombre de noeuds estimé pour compléter chacune se compare aux méthodes précédentes. Dans la liste qui suit, nous avons indiqué le nombre de noeuds contenus dans chaque inférence utilisée pour produire la solution auquel nous avons ajouté la contribution des inférences permettant de produire les résultats intermédiaires utilisés comme antécédents. Ce calcul récursif nous permet de trouver le nombre de noeuds (CP), ainsi que la liste des noeuds pris en compte lors de l'estimation de la progression de l'élève. Les valeurs de CP pour chacun des noeuds contenus dans les trois solutions sont indiquées en gris dans la figure 5.4.

- S01 : I_3, I_4, C_1, J_8 : 4 noeuds (+ I_3+I_4) -> Total 15 noeuds
 - I_4 : I_1, I_2, J_4 : 3 noeuds (+ I_1+I_2) -> Total 9 noeuds
 - I_3 : H_1, J_2 : 2 noeuds
 - I_2 : H_3, H_4, J_1 : 3 noeuds
 - I_1 : H_2, H_3, J_1 : 3 noeuds
- S04 : I_3, I_4, C_1, J_8 : 4 noeuds (+ I_3+I_4) -> Total 21 noeuds
 - I_4 : H_2, H_3, H_4, I_3, J_5 : 5 noeuds (+ I_3) -> Total 11 noeuds
 - I_3 : H_4, I_1, J_3 : 3 noeuds (+ I_1) -> Total 6 noeuds
 - I_1 : H_2, H_3, J_1 : 3 noeuds
- S23 : $H_2, H_3, H_4, I_3, C_1, J_7$: 6 noeuds (+ I_3) -> Total 8 noeuds
 - I_3 : H_1, J_2 : 2 noeuds

On remarque premièrement que pour S23, le nombre de noeuds est identique pour les trois méthodes de calcul. La progression sera donc évaluée de façon exacte, peu importe la méthode choisie. Cependant, pour S01, on gagne 2 noeuds par rapport à la méthode par énoncé et l'on en gagne 10 dans le cas de S04. Ce phénomène apparaît lorsque l'on réutilise des hypothèses ou des résultats intermédiaires à plusieurs endroits dans une démonstration. Dans ce cas,

la contribution de ces noeuds sera ajoutée au total à chaque occurrence, ce qui augmente le poids de ces derniers dans le calcul de la progression. Par exemple, pour S04, l'ensemble des noeuds qui permettent de produire I_3 est comptabilisé à deux reprises pour produire la conclusion, car on l'utilise directement et il est utilisé pour produire I_4 . La progression de l'élève sera donc surestimée s'il inscrit des énoncés permettant de produire I_3 et sous-estimée pour les autres noeuds qui ne sont pas dupliqués. Pour éviter la duplication des noeuds lors de la propagation, il faudrait garder une trace des noeuds parcourus pour chacune des solutions, ce qui équivaut à produire toutes les solutions et ne règle pas notre problème de départ.

L'utilisation de la méthode par propagation est cependant limitée aux arbres HPDIC, car les cycles contenus dans les graphes peuvent produire une boucle infinie lors de l'application des calculs présentés. Pour bien comprendre ce qui se passe, imaginons un noeud N_R pour lequel deux inférences distinctes peuvent être utilisées pour le produire, une contenue dans un cycle et la seconde à l'extérieur du cycle. Pour calculer les valeurs de N_R , il nous faut la progression des deux inférences possible pour le produire dans le but de choisir la plus avancée. Il est cependant impossible de déterminer la valeur de la progression de l'inférence contenue dans le cycle, car pour y arriver, il faut connaître la valeur de N_R que nous tentons justement de déterminer. Une solution envisagée consistait à initialiser tous les noeuds avec des valeurs prédéfinies. Même en utilisant cette stratégie, il n'est pas possible d'éliminer le problème. En effet, si chaque inférence présente dans un cycle a un ratio de progression supérieur aux inférences qui entrent dans le cycle, nous ferons des mises à jour en boucle des inférences présentes dans le cycle, car *BestCh* provient toujours du cycle et modifie les valeurs du N_R courant. La stratégie consistant à conserver la liste des noeuds parcourus n'est d'aucun secours, car nous voulons être en mesure de traiter des solutions dans lesquelles des N_R sont utilisés à plusieurs endroits comme dans l'exemple de S04. Ce problème nous a donc contraints à transformer notre graphe en arbres, ce qui nous a menés à proposer l'algorithme présenté à la section précédente.

Dans le but d'estimer la progression de l'élève pour déterminer la solution qu'il désire vraisemblablement produire, nous avons proposé, dans cette section, trois méthodes de calcul. Les deux premières, soit la méthode par énoncé et la méthode par noeuds, produisent des résultats précis par rapport au travail à accomplir, mais doivent être appliquées sur chaque solution possible pour un problème donné. Pour des problèmes complexes, il est impossible de réaliser cette opération dans un laps de temps raisonnable pour préserver l'interactivité du logiciel. Nous avons donc proposé la méthode par propagation sur des arbres HPDIC qui donne des résultats moins précis, car des noeuds peuvent être comptabilisés à plusieurs reprises, mais qui évite l'énumération de toutes les solutions. Elle permet, comme pour les deux autres méthodes, de détecter une solution complète dès qu'elle est produite, car aucune

information n'est perdue concernant la solution la plus avancée. C'est par conséquent cette dernière qui a été implantée dans la version actuelle de QEDX et qui permet de travailler sur les cinq problèmes proposés. Nous utilisons, par la suite, la solution la plus avancée identifiée par cette méthode pour ordonner les inférences en appliquant la technique décrite dans la prochaine section, ce qui permet de déterminer la séquence des messages du tuteur.

5.5 Ordonnement des inférences

Le travail présenté dans les sections précédentes avait pour objectif de contenir l'explosion combinatoire associée à la recherche de la solution la plus avancée pour un problème donné. Cette dernière est utilisée par notre algorithme d'ordonnement des inférences, car elle correspond, selon nous, à la stratégie de résolution courante de l'élève. Par ailleurs, l'objectif de cette section consiste à présenter, de manière formelle, l'algorithme d'ordonnement. Afin de mieux comprendre ses principes, il est important de se rappeler que l'unité de travail de base utilisée par QEDX, ainsi que par les enseignants, est l'inférence. En effet, ces derniers ciblent, par exemple, l'inférence sur laquelle l'élève a travaillé le plus récemment avant de proposer des indices permettant de la compléter. Par la suite, l'enseignant choisit, au besoin, d'autres inférences dans le but de relancer le processus de résolution. Le défi consiste alors à classer les inférences en ordre de priorité, à la manière des enseignants, afin de proposer une aide qui respecte l'état cognitif de l'élève. Dans QEDX, c'est l'algorithme d'ordonnement qui réalise cette tâche en se basant sur la solution la plus avancée ainsi que sur la chronologie des actions de l'élève. Dans cette section, nous présentons, par conséquent, la description formelle de notre algorithme ainsi que des exemples qui permettront d'illustrer son fonctionnement. Nous exposons ses deux modes de fonctionnement et expliquons comment ceux-ci diffèrent et à quel moment chacun est utilisé. La liste ordonnée d'inférences produite par notre algorithme est par la suite utilisée dans le cadre du système de génération des messages du tuteur qui est présenté dans le chapitre 6.

5.5.1 Observations concernant la stratégie des enseignants

Avant de présenter formellement notre algorithme, nous allons justifier le choix de celui-ci. Premièrement, dans leur article, Matsuda et VanLehn (2003) constatent que les systèmes tutoriels en géométrie offrent rarement une aide qui ressemble à celle proposée par des enseignants. En effet, ces derniers ne travaillent pas seulement en chaînage avant ou arrière, comme le font les logiciels qu'ils ont analysés, mais ont une approche plus globale face au problème. Dans le cadre de la conception de QEDX, nous voulions proposer un système capable d'imiter les interventions des enseignants. Il nous fallait donc éviter d'utiliser les

stratégies des logiciels existants. Pour tenter de comprendre les stratégies d'interventions des enseignants, nous avons fait deux expérimentations dans des classes du secondaire (voir Tessier-Baillargeon (2016) pour les détails). Malheureusement, comme il fallait s'y attendre dans ce type de recherche, il ne nous a pas été possible d'identifier la stratégie précise utilisée pour chaque enseignant. Cependant, plusieurs principes généraux ont été découverts et nous avons tenté de les appliquer dans la conception du tuteur de QEDX.

Nous avons premièrement observé que les enseignants sont en mesure de déterminer, après une brève analyse du travail de l'élève, la stratégie probable de résolution de ce dernier. Il arrive parfois que la stratégie ne puisse pas être ciblée par l'enseignant et que, dans ce cas, ce dernier demande à l'élève de lui expliquer ce qu'il veut faire. Les enseignants semblent donc experts pour identifier la stratégie probable de l'élève, mais ils ne sont pas en mesure d'expliquer concrètement cette intuition. Pour aider l'élève, il était alors clair que QEDX se devait de déterminer sa stratégie de résolution. Faute de comprendre l'intuition des enseignants, nous avons utilisé un critère d'efficacité. En effet, nous considérons qu'un élève qui a une stratégie de résolution proposera des énoncés qui permettront de produire de manière efficace la solution correspondante. Nous avons donc choisi de prendre la solution la plus avancée comme étant la stratégie courante de l'élève. Celle-ci est extraite par les algorithmes de la section précédente et est utilisée dans notre algorithme pour ordonner les inférences.

La seconde observation qui a été implantée dans notre algorithme concerne la chronologie des événements. En effet, après avoir identifié la stratégie de l'élève, l'enseignant recherche généralement les pas de démonstration qui correspondent à celle-ci dans le travail récent de l'élève. Il l'incite alors à travailler sur des concepts qui sont probablement encore frais à sa mémoire. Si cette aide ne permet pas à l'élève de poursuivre sa résolution, l'enseignant pourra alors choisir des pas ou même des stratégies qui sont plus anciennes. De cette façon, il cherche à respecter l'état cognitif de l'élève tout en tentant de relancer le processus de résolution. Dans QEDX, nous conservons la chronologie d'activation des noeuds, ce qui peut nous permettre de reproduire en partie cette stratégie d'aide utilisée par des enseignants. L'heuristique que nous avons choisi de tester consiste à ordonner les inférences en ordre décroissant selon l'ensemble des temps d'activation des noeuds la composant. Nous allons donc privilégier des messages d'aides concernant le travail récent de l'élève avant de proposer d'autres stratégies. L'identification de la stratégie de l'élève ainsi que le respect de la chronologie des événements, tels qu'implantés dans la version actuelle de QEDX, constituent donc une approximation de la stratégie générale d'aide adoptée par les enseignants observés.

5.5.2 Retour sur quelques définitions formelles du HPDIC et du MIA

Pour être en mesure de bien comprendre l'algorithme d'ordonnancement proposé, il est nécessaire de revoir quelques définitions provenant du chapitre 4, où sont présentés le graphe HPDIC et le MIA. Premièrement, nous avons l'ensemble des inférences pour un problème donné ($\mathbf{I}$) qui est un sous-ensemble de toutes les inférences possibles. Une inférence est constituée d'un ensemble d'antécédents ($\mathcal{A}$), d'une justification (J) et d'un conséquent (C) qui proviennent de deux banques d'énoncés ($\mathcal{E}$ et $\mathcal{R}$). Pour construire le graphe HPDIC, il faut savoir que le conséquent d'une inférence peut être utilisé comme antécédent dans une autre, ce qui est décrit par l'ensemble $\mathbf{S}$ qui représente cette structure.

$$\mathbf{I} \subseteq \{ \langle \mathcal{A}, J, C \rangle \mid \mathcal{A} \subseteq \mathcal{R}, J \in \mathcal{E}, C \in \mathcal{R} \} \quad (5.8)$$

$$\mathbf{S} = \{ \langle \langle \mathcal{A}_1, J_1, C_1 \rangle, \langle \mathcal{A}_2, J_2, C_2 \rangle \rangle \mid C_1 \in \mathcal{A}_2 \wedge \langle \mathcal{A}_1, J_1, C_1 \rangle \in \mathbf{I} \wedge \langle \mathcal{A}_2, J_2, C_2 \rangle \in \mathbf{I} \} \quad (5.9)$$

Dans un graphe HPDIC, il y a un certain nombre de noeuds distincts ($\mathbf{ND}(HPDIC)$) et pour chacun un temps d'activation unique est associé ($\mathbf{TA}(HPDIC)$). Le temps d'activation est une valeur croissante qui est initialisée à 1 lors du démarrage du logiciel. On associe une valeur de -1 aux noeuds qui n'ont pas été activés par l'élève. Plus un noeud a été activé récemment, plus la valeur de son temps d'activation sera grande.

$$\mathbf{ND}(HPDIC) = \{ n \mid \langle \mathcal{A}, J, C \rangle \in HPDIC \wedge (n \in \mathcal{A} \vee n = J \vee n = C) \} \quad (5.10)$$

$$\begin{aligned} \mathbf{TA}(HPDIC) &= \{ \langle n, t \rangle \mid n \in \mathbf{ND}(HPDIC) \wedge t \in \{-1\} \cup [1.. \infty[\} \\ &\quad \neg \exists \langle n, t \rangle, \langle n, t' \rangle \mid \langle n, t \rangle \in \mathbf{TA}(HPDIC) \wedge \langle n, t' \rangle \in \mathbf{TA}(HPDIC) \wedge t < t' \} \end{aligned} \quad (5.11)$$

Par exemple, la séquence d'actions qui a mené à l'activation des noeuds en gris, gradués selon le moment d'activation, de la figure 5.6 est celle-ci :

$$— C_1, H_2, H_4, J_1, J_8, J_5, H_2, H_3, I_1$$

Dans cet exemple, pour chaque action, on lui associe un temps d'activation allant de 1, pour la première, jusqu'à 9, pour la plus récente. Ces temps sont indiqués en gris à côté des noeuds correspondant dans le graphe. Dans le cas des actions qui sont répétées, soit H_2 , on ne conserve que la valeur du temps d'activation la plus élevée, donc 7. Pour tous les autres noeuds du graphe qui ne sont pas mentionnés dans cette séquence, la valeur de -1 leur est associée. Par souci de clarté, les valeurs de -1 ont été omises dans la figure 5.6. Dans celle-ci, nous avons aussi surligné en jaune la solution dominante déterminée par le module d'évaluation de la progression à la suite de la séquence d'actions.

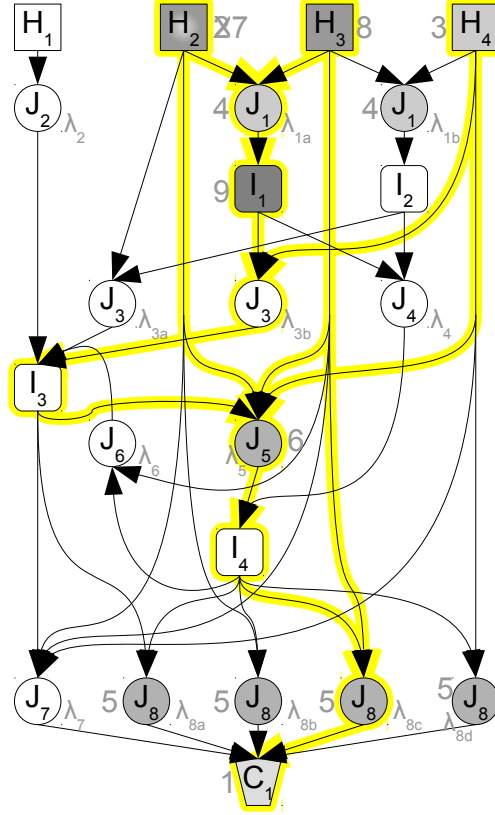


Figure 5.6 Temps d'activation associés à une séquence d'actions

5.5.3 Description de l'algorithme d'ordonnancement proposé

Le but de notre algorithme étant d'ordonner des inférences et non des noeuds individuels, il a été nécessaire de proposer une métrique qui représente le moment d'activation de celles-ci. Nous avons décidé de regrouper les temps d'activation des noeuds formant l'inférence sous forme d'une liste ordonnée de valeurs triées en ordre décroissant. La structure $\mathbf{LA}(\langle \mathcal{A}, J, C \rangle)$ représente donc cette liste pour chaque inférence.

$$\mathbf{LA}(\langle \mathcal{A}, J, C \rangle) = \text{TRIER}([t \mid \langle n, t \rangle \in \mathbf{TA}(\text{HPDIC}) \wedge (n \in \mathcal{A} \vee n = J \vee n = C)]) \quad (5.12)$$

Pour la séquence d'actions citée précédemment, le tableau 5.1 présente les $\mathbf{LA}(i)$ pour chaque inférence (i) acceptée pour la résolution du problème du rectangle. Dans ce tableau, les colonnes en gris correspondent aux inférences faisant partie de la solution la plus avancée, qui est surlignée en jaune dans la figure 5.6. Notre algorithme d'ordonnancement distingue trois types d'inférence qui sont définis selon le contenu du $\mathbf{LA}(i)$ associé. Le premier type, soit EI, est constitué des inférences que l'élève n'a pas encore travaillées, donc tous les temps

d'activations associés sont égaux à -1. Dans notre exemple, seul λ_2 fait partie de ce type. Nous avons, à l'autre extrême, les inférences complétées, soit celles qui n'ont aucun temps d'activation de -1, par exemple λ_{1a} . Celles-ci ne sont pas traitées par notre algorithme d'ordonnement, car aucune aide n'est nécessaire pour les compléter étant donné qu'elles le sont déjà. Toutes les autres inférences sont partiellement complétées et sont alors de type PI. Ces dernières sont importantes, car ce sont celles pour lesquelles l'élève a proposé quelques énoncés, mais qu'il n'a pas réussi à compléter, donc l'aide du tuteur est potentiellement utile et même nécessaire.

Tableau 5.1 Listes des temps d'activation à la suite d'une séquence d'actions

i	λ_{3b}	λ_4	λ_5	λ_7	λ_{8c}	λ_{1b}	λ_6	λ_{8b}	λ_{3a}	λ_{8d}	λ_{8a}	λ_2	λ_{1a}
LA (i)	9	9	8	8	8	8	8	7	7	5	5	-1	9
	3	-1	7	7	5	4	-1	5	-1	3	1	-1	8
	-1	-1	6	3	1	3	-1	1	-1	1	-1	-1	7
	-1	-1	3	1	-1	-1	-1	-1	-1	-1	-1		4
			-1	-1									
			-1	-1									
Type	PI	PI	PI	PI	PI	PI	PI	PI	PI	PI	PI	EI	-

Dans le but d'ordonner les listes d'activation des inférences obtenues, nous avons choisi d'utiliser une méthode similaire à celle utilisée pour ordonner les mots dans un dictionnaire. Nous proposons donc un comparateur de type lexicographique défini comme suit :

$a <_{lex} b$:

$$\begin{aligned}
 &\text{Si } |a| = |b| \text{ alors } (a_1, \dots, a_k) <_{lex} (b_1, \dots, b_k) \leftrightarrow \\
 &\quad a_1 < b_1 \vee [a_1 = b_1 \wedge (a_2, \dots, a_k) <_{lex} (b_2, \dots, b_k)] \\
 &\text{Si } |a| \neq |b| \text{ alors } (a_1, \dots, a_p) <_{lex} (b_1, \dots, b_q) \leftrightarrow \\
 &\quad (a_1, \dots, a_m) <_{lex} (b_1, \dots, b_m) \vee [(a_1, \dots, a_m) = (b_1, \dots, b_m) \wedge m < p] \\
 &\quad m = \min(p, q)
 \end{aligned} \tag{5.13}$$

Celui-ci compare donc, de façon séquentielle, les temps d'activation ayant des rangs identiques dans les **LA**(i) jusqu'à ce qu'une différence soit trouvée. À ce moment, on retourne le résultat de la simple comparaison de la valeur des deux temps. Si les deux listes sont de tailles différentes et que tous les éléments comparables sont identiques, nous avons décidé, à l'opposé de la règle du dictionnaire, de classer les inférences dans l'ordre inverse de la cardinalité des **LA**(i). Pour bien comprendre ce choix, il faut savoir que la liste ordonnée d'inférence sera parcourue à rebours pour privilégier celles pour lesquelles l'élève a récemment soumis des

énoncés. Cependant, lorsque deux inférences ont des temps d'activation identiques, mais que l'une contient moins de noeuds que la seconde, nous proposons de travailler sur la première en priorité, car elle requiert moins de travail de la part de l'élève pour la compléter. Par conséquent, il est logique de trier les $\mathbf{LA}(i)$ semblables en ordre inverse de leur cardinalité.

L'algorithme 5.2 présente la stratégie d'ordonnancement telle qu'elle a été implantée dans la version actuelle de QEDX. Dans celui-ci, nous utilisons SD(HPDIC), soit la stratégie dominante de résolution de l'élève qui nous semble la plus plausible. Actuellement, elle correspond à la solution la plus avancée qui est déterminée par la méthode par propagation décrite à la section précédente. Nous faisons aussi référence à HPDIC[I] qui correspond à l'ensemble des inférences du graphe HPDIC. De plus, pour signifier la concaténation d'éléments à la queue des listes ordonnées, soit ListeInfOrd et LIT, nous utilisons le symbole $\oplus$.

L'algorithme proposé peut s'adapter à deux principaux scénarios d'apprentissage, soit un mode d'exploration libre du problème et un mode dans lequel on privilégie le travail sur la solution dominante déterminée. Dans le premier cas, le système propose toujours de l'aide par rapport au travail récent de l'élève et s'adapte aux changements de stratégie instantanément. Il ne sera cependant pas très efficace pour inciter l'élève à compléter une preuve même s'il ne lui reste que quelques énoncés à soumettre pour y arriver. Ce mode est utile au début du problème, car l'élève n'a habituellement pas de stratégie claire et explore différentes pistes de solution pour voir la réaction du tuteur. Il est donc important de suivre son cheminement et de respecter son état cognitif. À un certain moment, l'élève croit avoir trouvé sa stratégie et veut compléter efficacement sa solution. Dans QEDX, nous avons choisi de quitter le mode d'exploration libre lorsque l'élève réussit à atteindre un seuil d'achèvement de 40% pour la solution la plus avancée. Ce seuil a été déterminé à la lumière de l'analyse des fichiers journaux recueillis au cours d'une expérimentation en classe (Tessier-Baillargeon et al., 2011). Dès que celui-ci est atteint, notre tuteur insiste alors sur les inférences qui permettent de compléter la solution déterminée, même si l'élève propose des énoncés correspondants à une autre stratégie. Le changement de stratégie sera détecté lorsque le pourcentage d'achèvement de la nouvelle solution dépassera celui de la stratégie dominante déterminée. Dans notre algorithme, il suffit de fournir une solution dominante vide pour utiliser le mode d'exploration libre ou de fournir la solution dominante déterminée pour privilégier le travail sur cette dernière. En effet, si une solution dominante est fournie, les inférences la composant seront ordonnées en priorité.

Pour créer la liste ordonnée des inférences qui sera utilisée par la couche GMD, nous y insérons d'abord la liste ordonnée des inférences de la solution dominante, si elle existe. Nous y ajoutons ensuite la liste ordonnée des inférences du graphe HPDIC qui ne font pas partie de la solution dominante. Pour chacune de ces deux dernières listes, une même procédure est

```

fonction ORDONNERINFÉRENCESGRAPHE
  ListeInfOrd  $\leftarrow \{\}$ 
  ListeInfOrd  $\leftarrow$  ORDONNERINFÉRENCES( SD(HPDIC) )
  ListeInfOrd  $\leftarrow$  ListeInfOrd  $\oplus$  ORDONNERINFÉRENCES( HPDIC[I]  $\setminus$  SD(HPDIC) )
  retourne ListeInfOrd
fin fonction

fonction ORDONNERINFÉRENCES(Inférence InfÀOrd)
  EI  $\leftarrow \{i \mid i \in \mathbf{InfÀOrd} \wedge \max(\mathbf{LA}(i)) < 0\}$ 
  PI  $\leftarrow \{i \mid i \in \mathbf{InfÀOrd} \wedge \max(\mathbf{LA}(i)) > 0 \wedge \min(\mathbf{LA}(i)) < 0\}$ 
  LIT  $\leftarrow$  TRIINVERSE( PI, OPÉRATEUR< )
  Cour  $\leftarrow$  LIT.PREMIER
  tant que Cour  $\neq \emptyset$  faire
    pour tout  $\{i \mid \langle i, \text{Cour} \rangle \in \mathbf{S} \wedge i \in \text{EI} \wedge i \notin \text{LIT}\}$  faire
      LIT  $\leftarrow$  LIT  $\oplus i$ 
    fin pour
    pour tout  $\{i \mid \langle \text{Cour}, i \rangle \in \mathbf{S} \wedge i \in \text{EI} \wedge i \notin \text{LIT}\}$  faire
      LIT  $\leftarrow$  LIT  $\oplus i$ 
    fin pour
    Cour  $\leftarrow$  LIT.SUIVANT
  fin tant que
  LIT  $\leftarrow$  LIT  $\oplus (\mathbf{InfÀOrd} \setminus \text{LIT})$ 
  retourne LIT
fin fonction

fonction OPÉRATEUR<(Inférence i1, Inférence i2)
  si  $\mathbf{LA}(\mathbf{i}_1) <_{lex} \mathbf{LA}(\mathbf{i}_2)$  alors
    retourne Vrai
  fin si
  si  $\mathbf{LA}(\mathbf{i}_1) = \mathbf{LA}(\mathbf{i}_2)$  alors
    si  $\langle \mathbf{i}_2, \mathbf{i}_1 \rangle \in \mathbf{S}$  alors
      retourne Vrai
    fin si
  fin si
  retourne Faux
fin fonction

```

Algorithme 5.2 Ordonnancement des inférences pour un graphe HPDIC

appliquée. Nous commençons par classer les inférences en trois catégories, soit les inférences complétées, les inférences pour lesquelles aucun travail n'a été fait (EI) et celles qui sont partiellement complétées (PI).

Nous trions ensuite l'ensemble PI en ordre décroissant pour accorder la priorité aux inférences activées récemment en utilisant le comparateur lexicographique proposé. Si celui-ci détermine que les deux inférences ont des temps d'activation identiques, nous avons ajouté un critère hiérarchique pour tenter de les départager. En effet, cette situation se produit habituellement lorsqu'un résultat intermédiaire est activé et que tous les autres noeuds des inférences adjacentes ne le sont pas. Dans ce cas, nous avons remarqué que les enseignants privilégient

l'inférence pour laquelle le résultat énoncé par l'élève est la conclusion de celle-ci. Le critère hiérarchique permet donc d'imiter cette stratégie d'aide. S'il n'y a pas de lien hiérarchique entre les deux inférences, nous considérons qu'elles ont une priorité équivalente et elles seront donc classées selon un ordre pseudo-aléatoire.

Après avoir ordonné les inférences du groupe PI, nous ajoutons celles d'EI en utilisant une stratégie précise. Étant donné que l'élève n'a pas encore travaillé sur celles-ci, il est normal qu'elles se retrouvent en fin de liste. Cependant, l'ordre de traitement de ces dernières n'est pas aléatoire et dépend directement du travail fait par l'élève. En effet, nous avons décidé d'utiliser un critère de proximité qui permet de donner une plus grande priorité aux inférences qui sont près de celles qui sont complétées ou partiellement complétées. Ce choix se justifie par le fait que les inférences adjacentes traitent habituellement de sujets semblables au travail réalisé par l'élève et nous semblent plus accessibles par rapport à l'état cognitif supposé de celui-ci. Pour y arriver, nous parcourons la liste des inférences déjà ordonnées et pour chaque inférence, nous ajoutons, en fin de liste, les inférences parentes et enfants faisant partie d'EI qui n'ont pas encore été traitées. La procédure est appliquée à l'ensemble de la liste, y compris les inférences d'EI qui viennent d'y être ajoutées. Cette stratégie permet donc de s'éloigner graduellement du travail récent de l'élève dans le but éventuel de lui proposer d'autres stratégies. À la suite de cette procédure, il peut arriver que des inférences d'EI n'aient pas été ajoutées à la liste ordonnée. Nous les ajoutons donc dans un ordre quelconque, car aucun critère ne nous permet de les départager.

À titre d'exemple, reprenons les données du tableau 5.1 qui correspondent au résultat de l'exécution des 9 actions présentées plus haut. Dans celui-ci, les inférences de type PI sont inscrites dans l'ordre inverse de celui prescrit par notre comparateur lexicographique. À partir de ces données, notre système est en mesure de déterminer une solution dominante qui contient quatre inférences, soit λ_{1a} , λ_{3b} , λ_5 et λ_{8c} . Selon le scénario d'apprentissage choisi, l'algorithme d'ordonnancement produit alors une des listes ordonnées d'inférences suivantes :

— Ordonnancement en mode d'exploration libre (OL) :

$$\lambda_{3b} \lambda_4 \lambda_5 \lambda_7 \lambda_{8c} \lambda_{1b} \lambda_6 \lambda_{8b} \lambda_{3a} \lambda_{8d} \lambda_{8a} \lambda_2$$

— Ordonnancement prioritaire pour la solution dominante (OD) :

$$\lambda_{3b} \lambda_5 \lambda_{8c} \lambda_4 \lambda_7 \lambda_{1b} \lambda_6 \lambda_{8b} \lambda_{3a} \lambda_{8d} \lambda_{8a} \lambda_2$$

Dans cet exemple, nous remarquons que l'inférence λ_{1a} n'a pas été triée, car elle est complète. En ce qui concerne les inférences partielles, le tri lexicographique a été suffisant pour les ordonner. On constate aussi que, dans le cas d'OD, ce sont les inférences de la solution dominante qui sont les plus prioritaires, même si d'autres correspondent mieux au travail récent de l'élève. On peut enfin voir que l'inférence qui n'a pas encore été traitée par l'élève,

soit λ_2 , se retrouve à la suite des inférences partielles, non pas à cause du tri lexicographique, mais bien à cause de l'ajout des EI par proximité. En effet, elle a été ajoutée, car elle est parente de λ_5 dans le cas d'OL. Pour OD, par contre, étant donné qu'elle ne fait pas partie de la solution dominante, l'analyse de λ_5 n'a pas permis de l'ajouter et il a fallu attendre l'analyse de λ_7 . Or, si λ_2 avait fait partie de la solution dominante, elle aurait été classée à la suite de λ_{8c} , donc avant les autres inférences du graphe ne faisant pas partie de la solution.

Le but de cette section était de présenter l'algorithme d'ordonnancement des inférences dont le résultat est utilisé par la couche GMD pour produire une suite de messages d'aide. Nous avons proposé une stratégie d'ordonnancement qui tente d'imiter les stratégies d'aide des enseignants. Plus particulièrement, nous avons proposé un algorithme qui tente d'aider l'élève à compléter la solution dominante de façon efficiente et qui respecte la chronologie de ses actions, comme semblent le faire les enseignants observés. Pour y arriver, notre algorithme peut permettre une exploration libre du problème et passer à un mode plus directif lorsqu'un certain seuil est atteint. Pour respecter la chronologie des actions, nous utilisons un tri lexicographique inverse sur le temps d'activation des inférences. Dans la section suivante, nous présentons les résultats obtenus par l'application des différentes stratégies visant à contenir l'explosion combinatoire pour identifier la solution dominante. Nous présentons aussi brièvement les résultats obtenus par notre algorithme d'ordonnancement. Cependant, celui-ci sera analysé plus en détail lors de la discussion.

5.6 Résultats obtenus et validations réalisées

L'application des algorithmes et des stratégies présentés dans ce chapitre, aux cinq problèmes contenus dans la version actuelle de QEDX, a permis de recueillir des données intéressantes et de contrôler, jusqu'à un certain point, l'explosion combinatoire. Cette section a pour objectif de présenter les résultats significatifs obtenus et de les analyser. Pour ce faire, nous commençons par indiquer les statistiques concernant les cinq problèmes ainsi que les résultats associés au découpage des graphes en arbres. Nous en profitons pour expliquer un résultat étonnant par rapport au nombre de solutions dans les arbres par rapport à celles contenues dans le graphe. Par la suite, nous proposons une démarche de résolution typique pour le problème du rectangle. Nous présentons, pour celle-ci, les différentes métriques permettant d'évaluer le pourcentage d'achèvement de chacune des solutions possibles. Nous illustrons, à l'aide de cet exemple représentatif des résultats obtenus lors de simulations, comment ces métriques diffèrent et quel est l'impact par rapport au choix de la solution dominante. Enfin, nous parlerons brièvement de l'algorithme d'ordonnancement et nous verrons qu'il respecte les spécifications élaborées en collaboration avec les didacticiens.

5.6.1 Réduction de l'explosion combinatoire

Premièrement, en ce qui concerne les stratégies permettant de minimiser l'explosion combinatoire par rapport au nombre de noeuds et d'inférences à l'intérieur du graphe HPDIC, nous n'avons que des résultats théoriques. En effet, ces stratégies ont été implantées dans la version initiale de notre système, donc nous n'avons aucune donnée concernant la taille des graphes et le nombre de solutions sans l'utilisation de celles-ci. Nous avons cependant calculé le nombre moyen de phrases canoniques distinctes par noeud de résultat intermédiaire et avons obtenu 1,49. Pour les noeuds de justification, on obtient toujours une moyenne de 1, car les différentes propriétés et définitions ne sont pas interchangeables, comme c'est, par contre, le cas pour certains énoncés de résultats. Sachant que la majorité des inférences contiennent un minimum de deux antécédents en plus de leur conséquent, nous obtenons 3,31 ($1,49^3$), soit environ 4 inférences équivalentes pour chaque inférence. Dans un cas extrême, tiré d'un problème inclus dans QEDX, nous avons deux antécédents ayant 13 énoncés équivalents, en plus d'un troisième en ayant 9, ce qui produirait un total de 1521 inférences équivalentes. Il est aussi important de noter que ces valeurs ne prennent pas en compte le nombre de permutations possibles de chacun des énoncés et de chacun des paramètres à l'intérieur de ces derniers. Il est donc évident que nos stratégies ont eu un impact significatif pour limiter l'explosion combinatoire. Cependant, il est très difficile d'évaluer l'économie en ce qui concerne le nombre de solutions, car nous n'avons pas trouvé de relation entre le nombre de noeuds ou d'inférences et le nombre de solutions présentes dans un graphe.

5.6.2 Données relatives aux arbres extraits des graphes implantés

Concernant la décomposition du graphe en une forêt d'arbres, le tableau 5.2 présente les résultats obtenus à la suite de l'exécution de l'algorithme 5.1 sur chacun des cinq graphes implantés dans QEDX (annexe A). Dans ce tableau, nous avons regroupé différentes statistiques concernant les graphes originaux (fond gris) ainsi que celles relatives aux arbres extraits (fond blanc) à l'aide de notre algorithme. La première colonne contient un identificateur (PID) pour chacun des graphes ou des arbres analysés. L'identificateur RT correspond au graphe du problème du rectangle, qui est utilisé à titre d'exemple dans ce chapitre, alors que RT_1 et RT_2 font référence aux arbres illustrés aux figures 5.3a et 5.3b respectivement. Nous donnons ensuite le nombre de noeuds d'hypothèses (Hyp), de résultats intermédiaires (RI) et de conclusions (Con) contenus dans chacune des structures. Pour sa part, la colonne **I** indique le nombre d'inférences alors que E correspond au nombre d'énoncés de justification distincts. En effet, un même énoncé de justification peut parfois être utilisé dans plus d'une inférence. Le nombre total de noeuds présents dans chaque structure est inscrit dans

la colonne Noeuds et est égal à la somme des colonnes Hyp, RI, Con et **I**. Enfin, le nombre de solutions valides et le pourcentage de solutions valides apparaissent dans la colonne Sol.

Tableau 5.2 Statistiques relatives aux arbres HPDIC extraits pour cinq problèmes

PID	Hyp	RI	Con	I	<i>E</i>	Noeuds	Sol (% Valide)
RT	4	4	1	13	8	22	23 (82,1)
<i>RT</i> ₁	4	4	1	12	7	21	11 (100)
<i>RT</i> ₂	4	4	1	10	5	19	12 (100)
TT	5	16	1	31	13	53	244 (52,1)
<i>TT</i> ₁	5	16	1	27	12	49	100 (100)
<i>TT</i> ₂	5	15	1	24	11	45	68 (100)
<i>TT</i> ₃	5	15	1	25	11	46	64 (100)
<i>TT</i> ₄	5	15	1	25	11	46	64 (100)
<i>TT</i> ₅	5	15	1	22	11	43	20 (100)
PT	3	34	1	76	23	114	1 056 (100)
PA	51	122	1	214	31	388	68 722 (100)
TS	14	47	1	116	44	178	5 504 194 (50,8)
<i>TS</i> ₁	14	47	1	113	44	175	3 741 633 (100)
<i>TS</i> ₂	14	47	1	108	44	170	759 122 (100)
<i>TS</i> ₃	14	45	1	99	41	159	227 237 (100)
<i>TS</i> ₄	14	45	1	97	41	157	69 927 (100)
<i>TS</i> ₅	14	45	1	96	40	156	260 842 (100)
<i>TS</i> ₆	11	32	1	53	25	97	9 416 (100)
<i>TS</i> ₇	9	29	1	48	20	87	7 758 (100)
<i>TS</i> ₈	8	25	1	35	18	69	264 (100)
<i>TS</i> ₉	13	36	1	66	30	116	377 675 (100)
<i>TS</i> ₁₀	13	37	1	67	32	118	410 060 (100)
<i>TS</i> ₁₁	11	29	1	51	21	92	56 365 (100)
<i>TS</i> ₁₂	10	26	1	40	21	77	1 760 (100)

Dans le tableau 5.2, il est possible de distinguer les arbres des graphes en se servant du pourcentage de solutions valides. En effet, toutes les solutions contenues dans un arbre sont valides (=100%), alors qu'en présence d'un graphe, notre algorithme d'énumération des solutions (voir chapitre 4) extrait des solutions qui contiennent des raisonnements en boucle et qui sont donc invalides (<100%). Ce résultat était cependant prévu, car une solution invalide est produite si et seulement si une boucle est présente, ce qui signifie qu'elle provient nécessairement d'un graphe. L'utilisation d'un algorithme de détection des cycles nous a permis de valider notre classification. Ainsi, nos résultats indiquent que, parmi les cinq problèmes implantés dans QEDX, nous retrouvons trois graphes (RT, TT, TS) et deux arbres (PT, PA).

De leur côté, les statistiques se rapportant aux arbres créés à partir des graphes à l'aide de l'algorithme 5.1 décrivent une forêt particulière parmi toutes celles qui sont possibles. En effet,

la composition de chacun des arbres générés dépend à la fois de la séquence de traitement des solutions ainsi que de l'ordre de parcours des arbres au moment de choisir celui dans lequel sera ajoutée la solution traitée. Dans le but d'illustrer ce phénomène, nous avons simulé deux exécutions de notre algorithme pour le problème du rectangle en choisissant un ordre différent de parcours des arbres pour chacune, tout en maintenant constante la séquence de traitement des solutions. Les résultats de ces simulations sont présentés dans le tableau 5.3. À l'intérieur de celui-ci, nous indiquons les inférences faisant partie de chacune des 23 solutions possibles, soit une par ligne. De plus, les solutions sont listées conformément à la séquence de traitement. Les inférences contenues dans chacun des deux arbres (A1 et A2) composant chacune des deux forêts (FF et FR) sont inscrites au bas du tableau. La forêt FF, implantée dans QEDX, correspond au parcours des arbres selon l'ordre de leur création, tandis que pour la forêt FR, les arbres sont parcourus dans l'ordre inverse. À droite de chaque solution, nous avons consigné, pour chaque forêt, le numéro de l'arbre dans lequel elle a été insérée. Les valeurs entre parenthèses de la colonne FR indiquent l'arbre dans lequel la solution apparaît sans y avoir été ajoutée explicitement. Enfin, en bas à droite, le nombre total de solutions contenues dans chaque arbre est inscrit.

Afin de comprendre les résultats de nos simulations, il est important de rappeler que, pour le problème du rectangle, ce sont les inférences λ_5 et λ_6 qui forment un cycle. Par conséquent, un couple de solutions dans lequel l'une contient λ_5 et l'autre, λ_6 , ne peut pas se retrouver dans un même arbre. Au début de notre simulation, les cinq premières solutions (S23, S14, S13, S12, S17) sont ajoutées à l'arbre A1 pour les deux forêts, car il n'y a aucun conflit. L'ajout de la solution S21 ne peut toutefois pas se faire dans A1, qui contient λ_6 (de S14), car elle utilise λ_5 . A2 est alors créé afin de recevoir S21. Maintenant que nous avons deux arbres, les comportements pour la création des deux forêts divergent. En effet, dans le cas de FF, nous tenterons d'ajouter chacune des solutions dans A1 et seules les solutions contenant λ_5 se retrouveront dans A2. Par contre, pour la forêt FR, les solutions seront testées dans A2 et uniquement celles contenant λ_6 devront être insérées dans A1. Les solutions contenant λ_5 ou λ_6 sont donc ajoutées dans A2 et A1 respectivement alors que les autres apparaîtront dans un arbre qui dépend de la politique associée à la forêt. Dans notre exemple, les solutions S18, S16, S01, S02 et S03 (fond gris) seront ajoutées dans A1 pour FF et dans A2 pour FR. Par conséquent, comme nous pouvons le constater, des arbres distincts sont produits. Cependant, les deux forêts sont équivalentes, car elles contiennent le même ensemble de solutions qui est néanmoins distribué différemment. De plus, les solutions S17, S01, S02 et S03 sont contenues dans les deux arbres de FR alors qu'aucune redondance n'existe dans FF. Il est donc possible que le nombre total de solutions contenues dans les arbres puisse dépasser celui du graphe original sans que de nouvelles solutions émergent.

Tableau 5.3 Production d'arbres à partir d'une liste ordonnée de solutions

Sol	λ_{1a}	λ_{1b}	λ_2	λ_{3a}	λ_{3b}	λ_4	λ_5	λ_6	λ_7	λ_{8a}	λ_{8b}	λ_{8c}	λ_{8d}	FF	FR
S23			X						X					1	1
S14	X	X				X		X	X					1	1
S13		X		X					X					1	1
S12	X				X				X					1	1
S17	X	X				X						X		1	1 (2)
S21			X				X					X		2	2
S10		X		X			X					X		2	2
S06	X				X		X					X		2	2
S18	X	X				X							X	1	2
S22			X				X						X	2	2
S11		X		X			X						X	2	2
S07	X				X		X						X	2	2
S16	X	X				X					X			1	2
S20			X				X				X			2	2
S09		X		X			X				X			2	2
S05	X				X		X				X			2	2
S01	X	X	X			X				X				1	2 (1)
S15	X	X				X		X		X				1	1
S02	X	X		X		X				X				1	2 (1)
S03	X	X			X	X				X				1	2 (1)
S19			X				X			X				2	2
S08		X		X			X			X				2	2
S04	X				X		X			X				2	2
FF A1	X	X	X	X	X	X		X	X	X	X	X	X	11	
FF A2	X	X	X	X	X		X			X	X	X	X	12	
FR A1	X	X	X	X	X	X		X	X	X		X			9
FR A2	X	X	X	X	X	X	X			X	X	X	X		18

Cette simulation permet, entre autres, d'illustrer le fait que la forêt produite à partir d'un graphe n'est pas unique. Nous pouvons cependant affirmer que, selon la preuve proposée à la fin de la section 5.3, toutes les forêts issues de notre algorithme sont équivalentes au graphe d'origine. Pourtant, le nombre total de solutions contenues dans certaines forêts peut dépasser celui du graphe. Dans le tableau 5.2, par exemple, nous obtenons un total de 316 et de 5 922 059 solutions pour la forêt des problèmes TT et TS respectivement. Ce résultat ne vient toutefois pas compromettre notre preuve, car les simulations réalisées nous indiquent que cette disparité est causée par la présence de solutions redondantes et non par l'émergence de nouvelles solutions. En ce qui concerne le choix de la forêt optimale, le nombre de solutions qu'elle contient n'a pas d'influence sur l'efficacité de l'algorithme MP, car ce dernier n'a pas à

énumérer toutes les solutions pour trouver la plus avancée. Par contre, le nombre d'arbres et le nombre total de noeuds ont un impact direct sur les performances de MP. Malheureusement, nous n'avons pas fait d'étude approfondie concernant le nombre minimal d'arbres ou de noeuds nécessaires afin de produire une forêt optimale. Des expérimentations ont cependant été réalisées et le nombre d'arbres obtenus pour chacun des problèmes s'est avéré constant, soit de 2, 5 et 12 arbres pour RT, TT et TS respectivement. Dans le cas du problème du rectangle (RT), il est facile de voir que 2 arbres sont nécessaires. En effet, seules les solutions contenant λ_5 et λ_6 doivent être isolées, donc il est nécessaire d'avoir un arbre pour chaque catégorie de solution. Les autres solutions ne contiennent aucune inférence qui participe à une boucle, donc elles peuvent être ajoutées dans n'importe quel arbre sans problèmes. Pour TT et TS, la situation n'est pas aussi claire à cause de la complexité des graphes. Malgré cela, il est encourageant de voir qu'il est possible de décomposer un graphe en un nombre relativement restreint d'arbres, même pour des problèmes complexes, ce qui nous permet de les traiter dans QEDX.

5.6.3 Comparaison des méthodes d'estimation de l'avancement

Nous avons ensuite produit des simulations de démarches de résolution d'un élève typique qui tenterait de résoudre le problème du rectangle dans le but de comparer nos trois méthodes pour identifier la solution dominante. Afin d'illustrer nos observations, nous utiliserons la chronologie des actions indiquée sur le graphe HPDIC de la figure 5.7. Dans celle-ci, nous avons inscrit le temps d'activation de chacun des noeuds et nous avons modulé la couleur de fond de ces derniers afin de mieux montrer la séquence des événements. De plus, pour être en mesure de comprendre les résultats présentés par la suite, nous avons produit la liste de toutes les solutions possibles pour le problème du rectangle (tableau 5.4). Dans ce tableau, chaque colonne correspond à une inférence alors que chaque ligne correspond à une solution unique. Les lignes horizontales servent à regrouper les solutions que QEDX n'est pas en mesure de discriminer, car elles utilisent un même ensemble d'énoncés. Par exemple, pour distinguer S02 et S03, il faudrait être en mesure de savoir si l'élève, en inscrivant l'énoncé de J_3 , avait l'intention d'utiliser λ_{3a} ou λ_{3b} , information que nous n'avons pas étant donné la structure de notre logiciel. La couleur de fond de chaque solution indique dans quel arbre elle se trouve, soit gris pour RT_1 et blanc pour RT_2 . Enfin, on indique le nombre de noeuds inclus dans l'évaluation de la progression de chaque solution, tel que calculé par les trois méthodes proposées (ME, MN et MP), dans les colonnes correspondantes.

Les tableaux 5.5, 5.6 et 5.7 présentent le pourcentage d'achèvement de chaque solution, tel que calculé respectivement par ME, MN et MP, pour une série d'actions typique qui permet

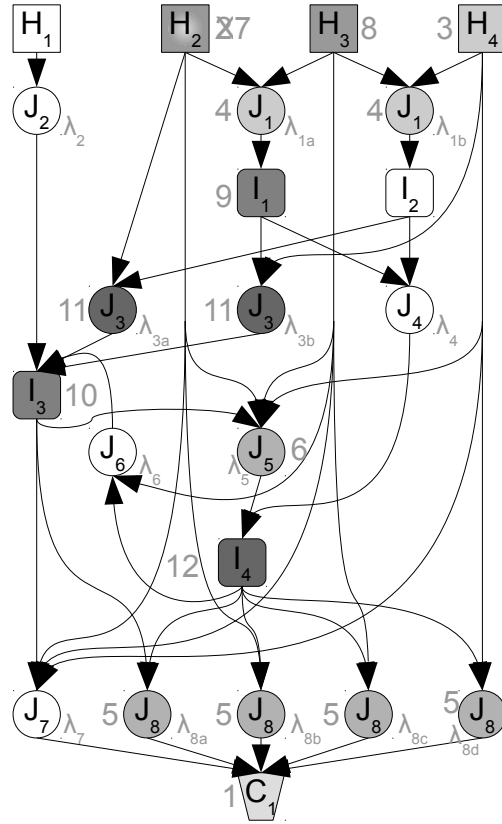


Figure 5.7 Chronologie d'actions typique produisant une démonstration

d'illustrer les principaux résultats issus de nos simulations. Nous avons indiqué, en dessous du temps d'activation associé, l'action réalisée par un élève fictif. Le pourcentage d'achèvement, arrondi à l'unité près, est indiqué pour chaque solution et pour chaque action. Les solutions ayant des progressions identiques ont été regroupées sur une même ligne par souci d'économie d'espace. Nous avons indiqué, par un fond gris, les valeurs maximales de chaque colonne, soit les solutions les plus avancées identifiées à la suite de chaque action de l'élève. Les valeurs sur fond gris foncé correspondent aux solutions dominantes choisies par QEDX. Enfin, les lignes pointillées verticales indiquent les endroits pour lesquels les jalons importants (25%, 40%, 50% et 75%) sont détectés par les différentes méthodes d'évaluation des solutions.

L'analyse sommaire des tableaux nous permet de constater que les solutions qui ne peuvent pas être discriminées par QEDX ont été regroupées dans le cas de ME et MN, alors que ce n'est pas le cas pour MP. En effet, dans les deux premiers cas, chaque noeud est comptabilisé au maximum une seule fois dans le calcul du pourcentage, même s'il contribue à produire plusieurs inférences de la solution. En ce qui concerne les justifications, on pourrait croire qu'elles sont comptées plus d'une fois dans le cas de MN. Il faut cependant se rappeler que

Tableau 5.4 Longueur des démonstrations pour le problème du rectangle telle que calculée par ME, MN et MP

Sol	λ_{1a}	λ_{1b}	λ_2	λ_{3a}	λ_{3b}	λ_4	λ_5	λ_6	λ_7	λ_{8a}	λ_{8b}	λ_{8c}	λ_{8d}	ME	MN	MP
S01	X	X	X			X				X				13	14	15
S02	X	X		X		X				X				12	13	19
S03	X	X			X	X				X				12	13	19
S04	X				X		X			X				11	11	21
S05	X				X		X				X			11	11	15
S06	X				X		X					X		11	11	15
S07	X				X		X						X	11	11	15
S08		X		X			X			X				11	11	21
S09		X		X			X				X			11	11	15
S10		X		X			X					X		11	11	15
S11		X		X			X						X	11	11	15
S12	X				X				X					9	9	12
S13		X		X					X					9	9	12
S14	X	X				X		X	X					12	13	18
S15	X	X				X		X		X				12	13	25
S16	X	X				X					X			10	11	13
S17	X	X				X						X		10	11	13
S18	X	X				X							X	10	11	13
S19			X				X			X				10	10	13
S20			X				X				X			10	10	11
S21			X				X					X		10	10	11
S22			X				X						X	10	10	11
S23			X						X					8	8	8

chaque énoncé, même s'il est utilisé dans plus d'un noeud, provient d'un noeud particulier et ce sont ces noeuds qui ne sont comptés qu'une seule fois. Par contre, dans le cas de MP, un noeud sera comptabilisé autant de fois qu'il est utilisé pour produire la solution. Plus un noeud sera près des hypothèses, plus il risque d'avoir une forte contribution, car plusieurs chaînes d'inférences faisant partie de la solution pourraient l'utiliser. Dans ce cas, pour que les solutions qui ne peuvent être discriminées par QEDX puissent être fusionnées, il faut que, en plus de contenir les mêmes noeuds, chaque noeud ait une contribution identique. Étant donné que chaque solution utilise des chaînes d'inférences uniques, la fusion n'est pas possible en utilisant MP. L'ordre d'entrée des énoncés a donc une plus grande influence sur le choix de la solution dominante dans le cas de MP, car il est possible de basculer entre des solutions équivalentes. Par contre, dans le cas de ME et MN, pour chaque groupe de solutions équivalentes, QEDX choisit toujours la même, car il parcourt toujours les solutions dans le même ordre. Par exemple, entre les temps 9 et 12, on aurait identifié la solution S06 pour

Tableau 5.5 Pourcentage d'avancement calculé par ME pour chaque solution à chaque étape d'une séquence typique de résolution

Temps Solution	1 C_1	2 H_2	3 H_4	4 J_1	5 J_8	6 J_5	7 H_2	8 H_3	9 I_1	10 I_3	11 J_3	12 I_4
S01	8	15	23	31	38	38	38	46	54	62	62	69
S02, S03	8	17	25	33	42	42	42	50	58	67	75	83
S04, S05, S06, S07	9	18	27	36	45	55	55	64	73	82	91	100
S08, S09, S10, S11	9	18	27	36	45	55	55	64	64	73	82	91
S12	11	22	33	44	44	44	44	56	67	78	89	89
S13	11	22	33	44	44	44	44	56	56	67	78	78
S14	8	17	25	33	33	33	33	42	50	58	58	67
S15	8	17	25	33	42	42	42	50	58	67	67	75
S16, S17, S18	10	20	30	40	50	50	50	60	70	70	70	80
S19, S20, S21, S22	10	20	30	30	40	50	50	60	60	70	70	80
S23	12	25	38	38	38	38	38	50	50	62	62	62

ME et MN alors que l'on oscille entre S06 et S04 pour MP. Ce phénomène influence donc les actions du tuteur et il pourrait avoir un impact sur les élèves, car la preuve affichée dans la fenêtre de rédaction de QEDX changera à des moments inattendus.

Si nous nous attardons maintenant à la progression des pourcentages pour chacune des solutions, nous remarquons qu'ils se comportent comme nous l'avions prévu. Premièrement, dans tous les cas ils forment une suite de valeurs croissante, ce qui est rassurant, car QEDX n'oublie aucun des énoncés soumis par l'élève. En ce qui concerne la comparaison des pourcentages, nous prenons ME comme référence, car il s'agit du calcul exact. Pour MN, nous observons une progression des pourcentages identique à ME dans le cas des solutions dont toutes les justifications utilisent des énoncés distincts. Pour les autres solutions, la progression est sous-évaluée pour tous les énoncés de résultat ainsi que pour les justifications qui ne sont pas répétées. Elle est par contre surévaluée pour les énoncés de justification répétés, comme nous l'avions prévu. Par exemple, pour S16, on observe un pas constant de progression de 10% pour ME, alors qu'il est de 9% ou 18% pour MN dépendant du type d'énoncé entré par l'élève. Dans le cas de MP, en général, nous nous attendons à une sous-évaluation de la progression pour les énoncés à proximité de la conclusion et à une surévaluation pour ceux près des hypothèses et c'est en gros ce que nous observons. En effet, les noeuds près des hypothèses sont souvent utilisés par plusieurs inférences contenues dans une démonstration et sont, par conséquent, comptabilisés à plusieurs reprises. Le poids relatif de l'un de ces noeuds dans le calcul de MP est ainsi plus important que celui d'un noeud à proximité de la conclusion, car le nombre de répétitions de ce dernier dans la démonstration est habituellement plus limité. S23 est par contre un cas particulier, car elle ne contient aucun noeud répété et elle se com-

Tableau 5.6 Pourcentage d'avancement estimé par MN pour chaque solution à chaque étape d'une séquence typique de résolution

Temps Solution	1 C_1	2 H_2	3 H_4	4 J_1	5 J_8	6 J_5	7 H_2	8 H_3	9 I_1	10 I_3	11 J_3	12 I_4
S01	7	14	21	36	43	43	43	50	57	64	64	71
S02, S03	8	15	23	38	46	46	46	54	62	69	77	85
S04, S05, S06, S07	9	18	27	36	45	55	55	64	73	82	91	100
S08, S09, S10, S11	9	18	27	36	45	55	55	64	64	73	82	91
S12	11	22	33	44	44	44	44	56	67	78	89	89
S13	11	22	33	44	44	44	44	56	56	67	78	78
S14	8	15	23	38	38	38	38	46	54	62	62	69
S15	8	15	23	38	46	46	46	54	62	69	69	77
S16, S17, S18	9	18	27	45	55	55	55	64	73	73	73	82
S19, S20, S21, S22	10	20	30	30	40	50	50	60	60	70	70	80
S23	12	25	38	38	38	38	38	50	50	62	62	62

porte donc de manière identique pour les trois méthodes, comme le montrent les résultats obtenus. En résumé, ce qui distingue la méthode MP des deux autres concerne la contribution répétée de certains noeuds dans le calcul de l'avancement d'une solution. En effet, plus des noeuds sont réutilisés dans une démonstration, plus la longueur de la solution estimée par MP diverge par rapport à ME ou MN, ce que l'on peut vérifier pour le problème du rectangle. La contribution répétée des noeuds dans une solution a donc une influence sur l'évaluation de la progression de l'élève, comme le montre notre séquence typique de résolution.

Ces disparités dans le calcul de la progression ont un impact à la fois au niveau des messages du tuteur qu'en ce qui concerne le choix de la solution dominante. En effet, QEDX propose des messages d'encouragement qui dépendent de la progression de l'élève. Nous avons indiqué le moment d'affichage des messages par des lignes pointillées verticales dans nos trois tableaux. Dans le cas de ME et MN, les messages se produisent aux mêmes moments, donc peu importe la méthode utilisée, l'élève de notre exemple n'observera pas de changement de comportement de la part de QEDX. Cependant, si MP est utilisé, tous les messages de progression, sauf le premier, seront affichés une action plus tôt par rapport au calcul exact. De plus, le passage du mode libre au mode plus directif est aussi avancé. Néanmoins, dans tous les cas on observe que le même ensemble de solutions complètes est identifié à la suite de la 12^e action, ce qui est un résultat important, car il confirme que toutes les méthodes d'estimations proposées sont en mesure d'identifier correctement les solutions complètes, malgré une précision variable pour le calcul de la progression. On pourrait tenter d'ajuster les seuils dans le cas de MP pour obtenir un comportement semblable à ME. Cependant, cette stratégie est vouée à l'échec, car les variations du moment d'affichage des messages dépendent grandement de l'ordre des

Tableau 5.7 Pourcentage d'avancement estimé par MP pour chaque solution à chaque étape d'une séquence typique de résolution

Temps Solution	1 C_1	2 H_2	3 H_4	4 J_1	5 J_8	6 J_5	7 H_2	8 H_3	9 I_1	10 I_3	11 J_3	12 I_4
S01	7	13	20	33	40	40	40	53	60	67	67	73
S02	5	16	26	42	47	47	47	63	68	74	79	84
S03	5	16	26	42	47	47	47	63	74	79	84	89
S04	5	19	33	43	48	52	52	67	76	86	95	100
S05	7	26	40	47	53	60	60	73	80	87	93	100
S06	7	20	33	40	47	53	53	73	80	87	93	100
S07	7	20	40	47	53	60	60	73	80	87	93	100
S08	5	19	33	43	48	52	52	67	67	76	86	90
S09	7	26	40	47	53	60	60	73	73	80	87	93
S10	7	20	33	40	47	53	53	73	73	80	87	93
S11	7	20	40	47	53	60	60	73	73	80	87	93
S12	8	25	42	50	50	50	50	67	75	83	92	92
S13	8	25	42	50	50	50	50	67	67	75	83	83
S14	6	17	28	39	39	39	39	61	67	72	72	78
S15	4	12	20	36	40	40	40	60	68	72	72	80
S16	8	23	31	46	54	54	54	69	77	77	77	85
S17	8	15	23	38	46	46	46	69	77	77	77	85
S18	8	15	31	46	54	54	54	69	77	77	77	85
S19	8	15	23	23	31	38	38	46	46	62	62	69
S20	9	27	36	36	45	55	55	64	64	73	73	82
S21	9	18	27	27	36	45	45	64	64	73	73	82
S22	9	18	36	36	45	55	55	64	64	73	73	82
S23	12	25	38	38	38	38	38	50	50	62	62	62

actions. Par exemple, la progression d'un élève qui aurait décidé de travailler en chaînage arrière, donc à partir de la conclusion, aurait été, au départ, sous-estimée par MP, donc les messages seraient probablement apparus plus tard par rapport à ME. Malgré cela, la précision de MP est acceptable, car, pour le type de rétroaction que nous proposons, une approximation grossière de la progression de l'élève est suffisante.

En ce qui concerne le choix de la solution dominante, les trois méthodes offrent des résultats relativement cohérents. En effet, à compter de la cinquième action, les trois méthodes proposent des groupes de solutions identiques. Il faut cependant se rappeler que dans le cas de MP, même si les solutions d'un même groupe ne progressent pas de manière identique, elles ne peuvent pas être discriminées par QEDX et seront complétées au même moment. Les variations observées ne sont donc pas significatives et sont en fait causées par des artefacts de la méthode d'évaluation. Pour les quatre premières actions, on observe plus de variabilité,

car il faut se rappeler qu’au début du problème peu de noeuds sont activés, donc le poids de chacun est important dans l’identification de la solution dominante. Par contre, nous considérons que l’élève est encore en phase d’exploration du problème, donc QEDX n’utilise pas la solution dominante pour aider celui-ci et opte pour une aide en exploration libre. Cependant, le seuil critique à partir duquel le choix de la solution reste relativement stable pour les trois méthodes d’évaluation de la progression, tel qu’observé dans notre exemple, semble être de 60%. Les résultats provenant de nos simulations, condensés dans l’exemple présenté, nous portent à croire que la méthode par propagation produit des résultats assez cohérents par rapport aux deux autres méthodes et qu’il est donc justifié de continuer à l’utiliser et à l’évaluer étant donné sa performance pour identifier la solution dominante. Il est aussi important de revoir le seuil de transition vers une aide plus directive, car il ne semble pas correspondre au moment où l’on observe une stabilisation de la solution dominante.

5.6.4 Validation de l’algorithme d’ordonnancement

Pour valider notre algorithme d’ordonnancement des inférences, nous avons réalisé des simulations, des tests unitaires et nous avons demandé à un expert de le vérifier par l’entremise des messages d’aide proposés par QEDX. Après avoir établi les spécifications avec les didacticiens, nous leur avons fourni des simulations sur papier pour vérifier si notre stratégie correspondait bien à leurs attentes, ce qui s’est avéré positif. Nous avons alors implanté notre algorithme ainsi que des tests unitaires qui permettent de vérifier sa validité face aux spécifications. Quelques simulations, semblables à celle présentée dans le tableau 5.1, ont aussi été réalisées, ce qui s’est avéré assez fastidieux même pour un problème modeste comme celui du rectangle. Enfin, nous avons demandé à un expert d’utiliser QEDX et de vérifier si les messages d’aide suivaient l’ordre prescrit, ce qui nous a permis de vérifier indirectement le bon fonctionnement de l’algorithme d’ordonnancement. Toutes ces vérifications se sont avérées concluantes, donc nous pouvons affirmer que notre algorithme respecte les spécifications.

Nous avons cependant observé, dans certains cas, au cours des tests et des expérimentations dans les écoles, un changement radical et inattendu de la stratégie proposée par le tuteur. En effet, il arrive que même en agissant conformément aux propositions de QEDX, ce dernier détecte un changement de solution dominante en cours de route, ce qui influence sa stratégie d’aide. Le tuteur propose alors un cheminement qui n’est plus conforme à la solution que nous tentions de produire et il est très difficile de revenir à cette dernière. Ce phénomène se produit lorsque, pour compléter notre solution, nous activons des noeuds qui font partie de solutions plus courtes ou de solutions qui ont déjà été partiellement travaillées. Dans ces cas, le calcul du pourcentage d’achèvement peut augmenter subitement pour les solutions plus

courtes et peut ainsi dépasser celui de la solution que nous tentons de produire. Il y a donc changement brusque de solution dominante et la nouvelle solution peut utiliser une stratégie sur laquelle nous n'avons pas travaillé. Le problème est donc relatif au choix de la solution dominante et non à notre algorithme d'ordonnancement, comme on avait cru au départ. En effet, ce dernier continue à trier correctement les inférences en utilisant la nouvelle solution dominante identifiée.

Dans cette section, nous avons premièrement présenté différents résultats obtenus en utilisant les stratégies proposées pour contrôler l'explosion combinatoire du nombre de solutions. En effet, le choix d'utiliser des formats de phrases canoniques et de permettre de regrouper des énoncés équivalents dans un même noeud dès la conception de notre système a permis de limiter le nombre de noeuds nécessaires pour construire l'arbre HPDIC des solutions. Malgré cela, le nombre de solutions pour certains problèmes était trop grand pour être traité efficacement dans QEDX. C'est pour cette raison que nous avons découpé le graphe en une forêt d'arbres. L'emploi de l'algorithme proposé a permis d'obtenir un nombre restreint d'arbres même pour les problèmes complexes testés. Nous avons prouvé qu'une solution peut se retrouver dans plus d'un arbre, ce qui explique le fait que le nombre total de solutions dans la forêt dépasse celui du graphe original. Par la suite, nous avons montré, à l'aide d'une séquence d'actions typique issue de nos simulations, que nos trois méthodes de calcul de la progression offrent des résultats cohérents. En effet, à partir d'une certaine action, les trois méthodes identifient des solutions dominantes similaires. Il y a certes une variabilité dans la valeur du pourcentage d'achèvement, ce qui influence le moment d'apparition des différents messages, mais les trois méthodes permettent de détecter correctement les solutions complètes. Enfin, nous avons indiqué que différents tests ont permis de vérifier que l'algorithme d'ordonnancement des inférences implanté correspondait bien à nos spécifications. La vérification par un expert et les évaluations en classe ont cependant permis de découvrir qu'il y a un problème pour l'identification de la solution dominante, ce qui peut rendre l'aide de QEDX incohérente. Nous discuterons, dans la prochaine section, des avantages, mais aussi des limites des stratégies proposées dans ce chapitre à la lumière des résultats obtenus.

5.7 Discussion

L'emploi des algorithmes proposés dans ce chapitre, sur les cinq problèmes implantés dans QEDX, a permis d'obtenir des résultats intéressants et encourageants. Nous allons maintenant montrer comment notre système se démarque des autres actuellement offerts. Nous discutons, dans la présente section, des avantages, mais aussi des limites de notre approche. Nous expliquons notamment comment notre contrôle de l'explosion combinatoire, dans le

but d'identifier la solution la plus avancée, permet à l'élève de travailler sur des solutions valides qui ne sont pas optimales. Elle permet en plus d'offrir un mode d'exploration libre du problème. Nous discutons aussi de l'impact positif de notre stratégie d'aide innovatrice qui consiste à proposer diverses pistes de solution. Concernant les limites de notre solution, nous discutons des problèmes causés par l'usage de phrases canoniques. De plus, nous présentons les problèmes et les défis soulevés pour l'identification de la stratégie dominante de l'élève. Enfin, une analyse de la stratégie d'ordonnement des inférences est proposée.

Notre stratégie pour contrôler l'explosion combinatoire du nombre de solutions, qui consiste en l'utilisation de phrases canoniques, n'est pas, en soi, une contribution importante de notre recherche. En effet, tous les systèmes doivent être en mesure de traiter des énoncés équivalents dans lesquels l'identification des éléments géométriques peut varier. Cependant, les systèmes recensés semblent ne permettre qu'un seul énoncé par noeud. Ceci oblige l'élève à travailler en utilisant une nomenclature précise qui a été choisie par l'auteur du problème. Par exemple, si l'on a décidé que l'élève doit nous parler d'un angle droit, l'affirmation que la mesure de l'angle est de 90° sera rejetée par le système. Par contre, dans QEDX nous avons choisi d'associer un concept par noeud dans lequel on peut mettre un ensemble d'énoncés équivalents. Il est donc possible, pour l'élève, de combiner diverses nomenclatures dans une même démonstration sans être pénalisé, ce qui est essentiel dans le contexte de l'enseignement au Québec. En effet, les enseignants peuvent présenter différentes nomenclatures selon le contexte et peuvent accepter leur utilisation, sans discrimination, lors de la résolution des problèmes. Notre décision rend donc QEDX plus flexible sans pour autant augmenter sa complexité.

Le fait d'accepter différentes nomenclatures simplifie le travail des élèves doués, mais les messages d'aide du tuteur peuvent parfois créer de la confusion. En effet, pour aider l'élève, nous faisons souvent référence à des noeuds qu'il a activés en lui indiquant l'énoncé correspondant à chacun. Dans le cas des noeuds qui ne contiennent qu'un seul énoncé, le choix est facile. Par contre, lorsqu'un ensemble d'énoncés est présent, il est difficile de choisir celui qui sera le plus apte à aider l'élève. Même si l'on sait que tous les énoncés représentent un même concept, le vocabulaire utilisé est très important, car il doit correspondre aux concepts que l'élève maîtrise. Par exemple, si l'élève nous indique que la mesure d'un angle vaut 90° , il est inapproprié de lui indiquer qu'il nous a affirmé que les deux droites sont perpendiculaires, même s'il s'agit du même concept. Dans le cas des noeuds activés, nous gardons l'énoncé qui a été soumis par l'élève en mémoire et nous l'utilisons pour produire l'aide, ce qui règle une partie du problème. Cependant, pour les noeuds qui ne sont pas encore activés, nous choisissons le premier énoncé rattaché au noeud, ce qui n'est pas optimal dans certains cas. Il reste donc du travail à faire pour choisir le bon énoncé selon le contexte et les compétences de l'élève. Nous nous sommes aussi aperçus que le seul fait de réorganiser les éléments

géométriques pour rendre les énoncés canoniques pouvait créer de la confusion chez certains élèves. Les stratégies pour contrôler l’explosion combinatoire sont importantes, mais ont eu un impact, insoupçonné au départ, concernant la pertinence de certains messages du tuteur. Pour identifier la stratégie de l’élève, nous avons choisi d’utiliser la solution la plus avancée. Pour ce faire, il est habituellement nécessaire d’énumérer toutes les solutions afin d’identifier celle qui maximise la métrique choisie. Lorsque le nombre de solutions à traiter est élevé, il est difficile de maintenir un temps de réponse raisonnable pour le tuteur. Pour résoudre le problème, dans le cas de Mentoniez (Py, 1996, 2001), ils font un prétraitement et ne conservent que les solutions ayant une taille plus petite que le maximum déterminé. Il n’est donc pas possible de travailler sur des solutions moins efficaces, mais tout de même valides. Proposer des problèmes relativement simples, comme dans Geometry Tutor (Anderson et al., 1985; Ritter et al., 2010), par exemple, permet aussi de limiter le nombre de solutions. Par contre, l’utilisation du chaînage avant et arrière permet d’éviter d’énumérer toutes les solutions et elle concentre les décisions lors de l’atteinte de moments clés. Toutes ces stratégies sont valides, mais imposent des restrictions sur le travail de l’élève, ce que nous voulions éviter dans le cadre de la mise en oeuvre de QEDX. La métrique proposée pour identifier la solution la plus avancée, bien qu’elle ne soit pas parfaite, est un apport important, car elle permet de l’identifier sans qu’il soit nécessaire d’énumérer toutes les solutions pourvu que l’on soit en présence d’un arbre HPDIC. Cette restriction nous a donc forcé à concevoir un nouvel algorithme qui transforme un graphe en une forêt d’arbres. Celui-ci demande tout de même d’énumérer l’ensemble des solutions, mais n’est exécuté qu’une seule fois pour générer la forêt qui est sauvegardée et utilisée dans QEDX. De plus, le contenu des arbres dépend de la stratégie d’ajout des solutions. Il serait donc intéressant de rechercher une méthode plus efficace pour créer la forêt. L’optimisation du nombre d’arbres et du contenu de chacun présente aussi des défis importants. L’algorithme proposé est tout de même une contribution notable, car il permet d’identifier la solution la plus avancée sans avoir à imposer de restrictions artificielles par rapport au travail de l’élève.

Un autre apport important de QEDX est le fait que l’on permette une exploration libre du problème, comme le propose Mentoniez (Py, 1996, 2001). En effet, des logiciels comme Geometry Tutor (Anderson et al., 1985; Ritter et al., 2010) ou Géométrie (Gressier, 2015) forcent les élèves à travailler dans un ordre déterminé qui correspond habituellement à la phase de rédaction de la preuve. Ils ne permettent donc pas d’explorer le problème, ce qui est par ailleurs une phase importante de la résolution que même les experts utilisent pour organiser leur stratégie de rédaction. Dans le cas de Mentoniez, la phase d’exploration est présente, mais l’aide du tuteur est limitée à l’identification des erreurs sur les inférences proposées. Dans le cadre de QEDX, l’élève est libre en tout temps et notre tuteur respecte

son travail en le guidant et non en lui imposant des actions. Cette aide respectueuse a été rendue possible grâce à l'utilisation des algorithmes d'identification de la solution dominante et par l'ordonnancement des inférences en utilisant les données du MIA. En effet, QEDX interagit, en tout temps, en indiquant les énoncés erronés soumis par l'élève, mais aussi en proposant des pistes pour lui permettre de résoudre le problème posé. Même si le tuteur propose à l'élève, à partir d'un certain moment, de travailler sur la solution dominante, ce dernier est toujours libre de changer de stratégie. De plus, notre tuteur tente de respecter l'état cognitif de l'élève en lui offrant de l'aide par rapport à son travail récent au lieu de l'inciter à travailler, par exemple, sur la solution optimale. QEDX intègre donc un tuteur intelligent et respectueux en permettant une exploration libre du problème en tout temps, ce qui est innovateur par rapport aux systèmes recensés.

Même si notre tuteur ne force pas l'élève à réaliser les actions proposées, il n'en demeure pas moins que ce dernier est grandement influencé par les suggestions de QEDX. De plus, la solution dominante est affichée sous forme de texte à trous dans l'onglet *Rédaction* et est utilisée comme outil d'aide à la rédaction par les élèves. Il est donc important que l'aide ainsi que la rédaction proposée soient cohérentes par rapport à la stratégie de l'élève. Il a cependant été observé que l'identification de la solution dominante est parfois déficiente et provoque des changements radicaux dans la stratégie d'aide du tuteur. Lorsque cela se produit, l'aide du tuteur devient inutile, car ce dernier insiste sur une solution qui n'est parfois pas accessible aux élèves. Le problème est causé par notre heuristique d'identification de la solution dominante, qui n'utilise que le pourcentage d'achèvement ou une valeur estimée de celui-ci. Cette méthode est simple et elle privilégie la solution optimale à l'instar des experts, mais elle s'est avérée plus instable que nous ne l'avions imaginé. Nous avons donc testé une version du logiciel qui n'utilisait jamais la solution dominante pour ordonner les inférences dans le but d'obtenir de l'aide plus pertinente et elle a reçu de meilleures critiques. Par contre, l'onglet *Rédaction* est un outil qui est très utile et apprécié des élèves. Il est donc primordial de rendre l'identification de la solution dominante plus robuste, car elle a un impact important sur l'acceptation et le succès de QEDX. Une piste de solution possible serait d'utiliser les données du MIA pour identifier la ou les solutions les plus récentes pour lesquelles nous avons observé une progression, même si elles ne sont pas les plus avancées. L'utilisation du concept de stratégie, défini au chapitre 4 comme étant l'ensemble des justifications qui permettent de caractériser une démonstration, pourrait aussi être envisagée, car il contient les éléments clés d'une démonstration, et ce même s'il ne permet pas de détecter les solutions complètes. Nous pourrions également penser offrir un choix de rédactions à l'élève par l'entremise de l'onglet *Rédaction*. Il est cependant clair que peu importe notre choix, seuls des tests en classe pourront nous permettre de valider la stratégie choisie.

Dans QEDX, nous offrons, comme dans plusieurs autres systèmes, une gradation des messages d'aide allant des plus génériques aux plus précis. Par contre, nous avons innové en proposant différentes pistes de solution au lieu de ne proposer de l'aide que pour la prochaine étape idéale, comme c'est habituellement le cas. Cette initiative vise à permettre une véritable exploration du problème et offre l'occasion de produire des preuves complètes même si elles ne sont pas optimales. De plus, elle tente d'imiter les actions d'un enseignant qui, voyant que l'élève n'est plus en mesure de progresser malgré ses suggestions, lui propose une nouvelle stratégie de résolution. Les pistes qui sont proposées à l'élève sont en fait liées directement à la liste ordonnée des inférences produite par l'algorithme décrit dans ce chapitre. En effet, notre tuteur propose, au départ, des indices qui correspondent aux dernières actions de l'élève pour s'en éloigner progressivement et lui proposer de nouvelles stratégies. Dans un cas extrême, après une longue attente, le tuteur nous aura donné des indices pour compléter chacune des inférences présentes dans le graphe HPDIC. Notre stratégie d'aide permet donc, en théorie, à QEDX d'agir de manière semblable à un enseignant ce qui nous démarque véritablement des autres systèmes.

Dans les faits, cependant, la politique d'aide de notre tuteur, que nous avons utilisée lors des expérimentations en classe, ne nous a pas permis de véritablement tester l'utilité des différentes stratégies offertes. En effet, QEDX proposait automatiquement un message d'aide chaque minute et l'élève n'avait pas la possibilité de modifier cette attente. De plus, dans le but de permettre à l'élève de résoudre le problème à l'intérieur du temps alloué, notre système fournissait une partie de la réponse lorsqu'un délai de dix minutes s'était écoulé depuis la dernière action valide. Il en résulte donc que QEDX ne pouvait traiter que deux ou trois inférences avant d'abandonner et d'attendre la prochaine action valide. Il faut aussi souligner que les inférences qui concernent des stratégies similaires et qui partagent donc un ensemble d'énoncés auront tendance à être regroupées dans la liste ordonnée. Enfin, les élèves n'avaient pas tendance à attendre les messages du tuteur ou à regarder l'historique à la suite d'une discussion de quelques minutes. Par conséquent, nos expérimentations ne nous permettent pas de tirer de conclusion par rapport à l'utilité ou à la pertinence de proposer de multiples stratégies. Il est cependant important de revoir les politiques de notre tuteur pour qu'il soit en mesure de proposer une aide adéquate et qui est plus représentative des agissements d'un enseignant. Il serait probablement pertinent de permettre à l'élève d'obtenir des messages d'aide à la demande, mais de ne permettre d'obtenir des réponses explicites qu'après un certain laps de temps. Il serait aussi impératif de revoir notre algorithme d'ordonnancement pour qu'il soit en mesure de proposer d'autres pistes de solution de façon plus intelligente. Nous pourrions, par exemple, penser à choisir une inférence représentative d'un ensemble d'inférences similaires et trier ces dernières. Bref, notre idée qui consiste à proposer diverses

pistes de solution, à la manière d'un enseignant, est innovatrice et a beaucoup de potentiel, mais il faudra retravailler l'implantation de celle-ci afin qu'elle soit plus représentative des stratégies d'aide des enseignants.

Le but de cette section était de proposer une analyse critique de notre travail en faisant ressortir les avantages, mais aussi les limites de la solution proposée. Nous avons premièrement montré qu'en associant plusieurs énoncés à chaque noeud du graphe HPDIC, il est possible de limiter l'explosion combinatoire du nombre de noeuds tout en permettant à l'élève d'utiliser différentes nomenclatures. De plus, nous avons expliqué comment notre algorithme d'identification de la solution dominante permet d'éviter, en partie, d'énumérer toutes les solutions. Nous l'avons utilisé dans le but d'offrir une exploration libre du problème sans imposer de restrictions par rapport aux solutions acceptables ou à l'ordre de résolution. Nous avons aussi innové en offrant un tuteur qui propose de multiples pistes de solution. Toutes ces caractéristiques permettent à QEDX de véritablement se démarquer des autres logiciels recensés. Cependant, comme nous l'avons expliqué, nos algorithmes comportent encore plusieurs lacunes, même s'ils sont performants et que QEDX a été, en général, bien accepté auprès des élèves. Le principal problème concerne l'identification de la solution dominante. En effet, nous avons observé que la métrique utilisée était instable lorsque des solutions de taille différentes partagent des inférences, ce qui amène le tuteur à proposer une aide moins pertinente. Il est aussi évident qu'il faudra réviser la politique d'aide ainsi que l'algorithme d'ordonnancement des inférences pour offrir véritablement des pistes distinctes de solution. Enfin, nous avons mentionné l'importance d'utiliser une nomenclature conforme au travail de l'élève pour éviter de créer de la confusion. Nous avons donc proposé des stratégies et des algorithmes, qui, même s'ils sont loin d'être parfaits, nous ont permis d'implanter QEDX, un système tutoriel intelligent innovateur en géométrie.

5.8 Conclusion

L'objectif de notre recherche est de proposer un système tutoriel intelligent en géométrie qui soutient l'élève dans une démarche d'exploration lors de l'élaboration d'une démonstration. Pour y arriver, nous avons choisi de ne pas limiter l'aide à la prochaine étape optimale, comme le font habituellement les autres systèmes, mais de proposer différentes pistes de solution que l'élève pourrait utiliser. Le but de ce chapitre était donc de présenter les algorithmes qui permettent d'ordonner les inférences pour produire des messages d'aide adaptés au cheminement de l'élève. Pour être en mesure de déterminer l'ordre des inférences, nous avons cependant besoin de la solution dominante, ce qui nous a amenés à affronter un problème d'explosion combinatoire du nombre de solutions. Les algorithmes et stratégies que nous avons présentés

font partie de la couche EDOI de QEDX. Ils utilisent les données du graphe HPDIC et du MIA pour produire la liste ordonnée des inférences.

Dans ce chapitre, nous avons, dans un premier temps, recensé les principaux systèmes existants et nous avons observé que la majorité, à l'exception de Mentoniez (Py, 1996, 2001), n'offre pas de phase d'exploration du problème. Ils forcent habituellement l'élève à travailler dans un ordre déterminé qui correspond à la phase de rédaction et offrent un nombre restreint de solutions possibles, ce qui limite artificiellement les problèmes d'explosion combinatoire. Pour être en mesure d'offrir une phase exploratoire avec un minimum de restrictions, nous nous sommes donc attaqué à la gestion des solutions pour déterminer la solution dominante. Nous avons proposé une écriture canonique des énoncés et la possibilité d'inscrire un ensemble d'énoncés équivalents dans chaque noeud, ce qui a permis de minimiser le nombre de noeuds nécessaires pour représenter l'ensemble des inférences possibles pour résoudre un problème donné. Cette stratégie a eu un impact certain, mais difficile à mesurer. Par contre, l'utilisation des phrases canoniques peut parfois causer de la confusion chez certains élèves.

Nous avons, par la suite, proposé trois méthodes pour évaluer la progression de l'élève et identifier la solution dominante. La première, soit ME, était celle considérée exacte, car elle représente précisément le pourcentage d'énoncés qu'il reste à produire pour obtenir une solution complète. Dans le cas de MN, chaque noeud d'une solution était compté une seule fois, mais les noeuds contenant des énoncés identiques n'étaient pas éliminés. L'écriture de certains énoncés provoquait donc une progression artificiellement plus rapide. Pour obtenir la solution dominante, il était nécessaire d'appliquer la méthode choisie, parmi ces dernières, sur toutes les solutions et garder celle ayant la valeur maximale. Pour des problèmes complexes, et ayant des millions de solutions, il n'était pas possible de faire les calculs dans un temps raisonnable et c'est pour cette raison que nous avons développé la méthode MP, qui n'est applicable que sur des arbres HPDIC. Il a donc été nécessaire de produire un algorithme qui transforme notre graphe en une forêt d'arbres équivalents. Cette dernière est produite lors d'un traitement préalable, mais nécessite tout de même l'énumération de l'ensemble des solutions. MP est une grossière approximation de ME, car elle peut comptabiliser un même noeud à plusieurs reprises selon la topologie de la solution, ce qui associe une valeur de progression variable à chacun. Les résultats obtenus montrent que pour les cinq problèmes implantés dans QEDX, la décomposition des graphes en arbres a permis d'obtenir un nombre relativement restreint d'arbres mêmes pour les problèmes les plus complexes. Les simulations effectuées montrent aussi que les trois méthodes offrent des résultats similaires pour l'identification de la solution dominante, ce qui est encourageant. Cependant, nous avons observé un changement radical de la solution dominante identifiée dans certains contextes d'utilisation. Même si notre méthode n'est pas parfaite, elle est tout de même une contribution importante

de notre projet de recherche, car elle contribue à permettre d'identifier la solution dominante sans nécessiter l'énumération continue de l'ensemble des solutions.

L'algorithme d'ordonnement des inférences a finalement été présenté. Celui-ci a pour but d'offrir différentes pistes de solution à un élève qui n'est plus en mesure de cheminer dans sa résolution d'un problème, comme le fait un enseignant. Il s'agissait donc de classer les inférences dans un ordre qui respecte l'état cognitif de l'élève tout en l'aidant à compléter la solution dominante. L'algorithme proposé a deux modes de fonctionnement, soit un mode d'exploration libre et un autre plus directif pour compléter la solution identifiée. Pour trier les inférences, on utilise les données des temps d'activation associés à chaque noeud d'une inférence, inscrit dans le MIA, que l'on regroupe dans une liste triée en ordre décroissant. On peut ensuite utiliser le comparateur lexicographique proposé pour obtenir une liste d'inférence ordonnée. Dans cette dernière, les inférences sur lesquelles l'élève a travaillé récemment se trouvent en tête de liste, car leur temps d'activation est le plus élevé. Ce sont donc celles-ci qui seront privilégiées pour proposer les premiers messages d'aide. Les inférences qui n'ont aucun noeud activé sont ajoutées en fin de liste, avec une priorité qui dépend de leur proximité par rapport aux inférences ayant été travaillées, afin d'offrir des pistes de solution portant sur le travail récent de l'élève. Ce comportement a été proposé par des didacticiens et la conformité de notre algorithme a été validée par un expert et par des simulations. QEDX, donc notre algorithme d'ordonnement, a été utilisé avec succès en classe. Notre algorithme est innovateur et permet ainsi, en théorie, d'imiter le comportement d'un enseignant. Cependant, la politique actuelle d'aide du tuteur ainsi que le problème d'identification de la solution dominante ne nous permettent pas de nous prononcer sur l'utilité réelle de notre algorithme.

Même si les idées et algorithmes proposés dans ce chapitre sont innovateurs, il faut se rappeler que ce n'est qu'une première itération dans le but d'obtenir un tuteur imitant les interventions d'un enseignant humain. Le principal problème qu'il faut résoudre concerne l'identification de la solution dominante, car c'est celle-ci qui influence les premiers messages du tuteur et qui détermine la solution affichée dans l'onglet *Rédaction*. Elle affecte donc la crédibilité de QEDX en tant que tuteur intelligent. Pour résoudre le problème, il serait intéressant d'examiner une identification par thèmes. Par exemple, une solution ne peut pas être identifiée comme dominante si les principaux thèmes en faisant partie n'ont pas été mentionnés par l'élève. On pourrait aussi penser implanter un critère d'inertie qui conserve la même solution dominante pour un certain nombre d'actions dans le cas où son pourcentage d'achèvement ne soit plus le plus élevé. Il y a aussi du travail à réaliser en ce qui concerne l'ordonnement des inférences. En effet, actuellement le temps nécessaire pour proposer des pistes de solution véritablement différentes est trop élevé. La modification des politiques d'aide du tuteur ne règlera qu'une partie du problème. Le véritable défi est d'éviter de proposer des séries de

messages d'aide concernant des inférences similaires. Il serait peut-être utile de tenter de proposer de l'aide pour une inférence typique qui représente un ensemble d'inférences similaires. Le critère de similitude pourrait concerner autant les thèmes contenus dans chaque inférence que les éléments géométriques utilisés. Son choix n'est pas sans importance, car il influencera directement la performance du tuteur, et pourrait donc faire l'objet d'une nouvelle recherche.

Enfin, rappelons-nous que les éléments présentés dans ce chapitre font partie de la couche EDOI de QEDX et ont pour objectif de proposer une liste d'inférences ordonnée qui sera utilisée par la couche GMD. En effet, tout le travail réalisé ici ne produit aucun message d'aide discursif directement. C'est la couche GMD qui organise et produit les messages du tuteur, donc c'est elle qui rend QEDX interactif. Elle utilise une machine à états pour tenter d'imiter les interactions des enseignants observés. La description formelle et détaillée de la couche GMD est présentée dans le chapitre 6 .

CHAPITRE 6 GÉNÉRATION DES MESSAGES DISCURSIFS (GMD)

«La simplicité est la sophistication suprême.»

— Léonard de Vinci

6.1 Introduction

Lors de la conception des systèmes tutoriels, il est important de bien choisir les messages qui guident l'élève au cours de la résolution d'un problème. En effet, il faut proposer des messages clairs, pour éviter de créer de la confusion, tout en s'abstenant, si possible, de donner les réponses en même temps que les questions (Richard et al., 2014). Dans le cadre de l'élaboration de QED-Tutrix (QEDX), un logiciel d'aide à la démonstration en géométrie, nous avons opté pour une approche multidisciplinaire afin de relever ce défi. Celui-ci est donc issu d'une collaboration entre des experts en didactique des mathématiques et en informatique. Pour bien comprendre le processus qui permet à QEDX de proposer des messages pertinents à l'élève, nous allons, tout d'abord, décrire brièvement l'architecture de notre système, qui comprend quatre couches principales (HPDIC, MIA, EDOI et GMD).

La couche de base, soit HPDIC, correspond à la modélisation du domaine. Elle contient l'ensemble des preuves acceptables, dans le cadre scolaire, qui permet de résoudre le problème posé. Une démonstration est constituée d'un ensemble de pas, appelés inférences. Chacune propose un conséquent qui est le résultat de l'emploi d'une justification sur un ensemble d'antécédents. Dans QEDX, on associe une structure en arbre à chaque inférence. Il faut aussi savoir que les antécédents peuvent être des conséquents d'autres inférences afin de former une preuve. La couche HPDIC contient donc l'ensemble des inférences utiles à la résolution d'un problème, telles qu'établies par l'enseignant ou le didacticien, ainsi que les liens qui les unissent sous forme d'un graphe orienté (graphe HPDIC). Des algorithmes, présentés dans le chapitre 4, permettent d'énumérer toutes les solutions contenues dans celui-ci.

À l'opposé de la couche HPDIC, qui ne contient qu'une structure statique associée à un problème donné, la couche MIA maintient une structure dynamique qui sert à modéliser l'apprenant. En effet, pour être en mesure de respecter l'état cognitif de celui-ci lors de nos interventions, nous enregistrons la chronologie de ses actions. Pour y arriver, nous associons un temps d'activation à chaque noeud du graphe HPDIC qui équivaut au temps le plus récent d'écriture, à l'interface de QEDX, de l'énoncé correspondant. Il est donc possible d'identifier à quel moment l'élève a probablement travaillé sur chacune des inférences contenues dans

le graphe. Ces informations sont utilisées dans les couches supérieures afin de proposer des messages mieux adaptés à la stratégie de résolution adoptée par l'élève.

Pour sa part, la couche EDOI sert à planifier la stratégie d'intervention visant à aider un élève bloqué dans son processus de résolution (voir chapitre 5). Lors de cette planification, c'est l'inférence qui est utilisée comme unité de base. En effet, l'algorithme implanté dans QEDX trie les inférences en ordre chronologique, selon les données contenues dans le MIA, dans le but d'identifier celle qui est incomplète et qui a été travaillée le plus récemment par l'élève. C'est donc celle-ci qui sera utilisée en priorité pour la génération des messages discursifs dans la couche GMD. Nous avons aussi mis en oeuvre une reconnaissance de plan qui est basée sur l'identification de la solution la plus avancée, c'est-à-dire celle pour laquelle le pourcentage d'énoncés nécessaires pour la compléter est minimal. Lorsque nous croyons avoir reconnu la solution dominante de l'élève, déterminé par un seuil sur le pourcentage d'achèvement, nous privilégions alors les inférences incomplètes faisant partie de celle-ci. Cette stratégie permet donc de respecter l'état cognitif de l'élève en lui proposant des indices concernant son travail récent tout en le guidant vers l'achèvement de la solution la plus avancée.

Les couches HPDIC, MIA et EDOI travaillent exclusivement sur des données internes et n'ont, par conséquent, aucune interaction directe avec l'élève. C'est la couche de génération des messages discursifs (GMD) qui analyse les actions de l'élève et qui produit les rétroactions pertinentes à l'interface de QEDX. Pour y parvenir, elle utilise, entre autres, la stratégie d'intervention issue de la couche EDOI afin de générer une série d'indices visant à relancer le processus de résolution de l'élève. L'objectif du présent chapitre consiste donc à décrire, de façon technique et détaillée, la couche GMD ainsi que ses fondements didactiques. Dans le but de l'atteindre, ce chapitre comprend deux sections principales. La première propose les résultats du travail réalisé en collaboration avec les didacticiens pour identifier la structure des messages utilisés par un enseignant typique lors de ses interventions. Il est intéressant de noter que ce dernier peut proposer différentes pistes de solution avant de donner une réponse à l'élève, ce qui le distingue par rapport aux approches adoptées par les systèmes tutoriels recensés (voir section 6.2). Dans la seconde section, nous présentons les structures, implantées dans QEDX, qui permettent de produire des rétroactions semblables à celles qui ont été observées. Notre défi consistait donc à proposer une architecture assez riche pour être en mesure de modéliser les interactions tout en étant facile à manipuler afin que les didacticiens puissent modifier les messages au besoin.

Plus précisément, ce chapitre débute par une brève revue de littérature dans laquelle nous nous concentrons sur les interactions offertes par les différents systèmes tutoriels intelligents. Nous y analysons essentiellement ceux en géométrie. De plus, nous proposons d'autres textes

pertinents concernant la structure des rétroactions. Les résultats des analyses didactiques sont décrits dans la section suivante. Nous expliquons, entre autres, le processus qui a permis de transformer des interactions brutes en une structure plus ordonnée qui est tout de même valide du point de vue de la didactique. Dans la troisième section, nous proposons une architecture basée sur une machine à états finis, implantée dans QEDX, qui permet d'encoder la structure des messages issue de notre étude. De plus, nous traitons d'autres types de rétroactions offertes par notre système et qui proviennent aussi d'analyses didactiques. Dans la section concernant les résultats, nous décrivons le travail requis pour implanter les messages et nous donnons des exemples de séquences d'indices produits par QEDX. Nous traitons, de plus, des tests réalisés par un expert ainsi que de l'expérimentation en classe qui a servi à valider notre système. Par la suite, nous présentons une discussion qui fait ressortir les avantages, mais aussi les limites de notre approche. Nous proposons également une adaptation de la structure classique, et linéaire, des messages d'aide qui reflète les résultats de notre travail. Nous terminons par une brève conclusion qui contient quelques pistes à explorer.

6.2 Interactivité des systèmes existants

Dans cette section, nous présentons une brève revue de la littérature dans le but de faire ressortir les stratégies habituellement utilisées pour produire des rétroactions permettant d'aider un élève en cours de résolution d'un problème. Cependant, dans le cadre de ce chapitre, nous nous concentrons uniquement sur les rétroactions offertes par les différents systèmes, par exemple l'ordre et le nombre de messages, sans nous soucier de la stratégie d'aide, c'est-à-dire l'identification de la prochaine étape à réaliser. En effet, cette dernière, ainsi que la revue de la littérature associée, a été traitée en détail dans le chapitre 5 concernant la couche EDOI. Au cours des prochains paragraphes, nous montrons donc que les systèmes recensés, même s'ils mettent en oeuvre des stratégies d'aide différentes, proposent un ensemble de rétroactions relativement uniforme. Celles-ci sont, toutefois, généralement limitées par rapport aux interactions observées entre un élève et son enseignant. La section débute par une description des trois types de rétroactions habituellement disponibles. Nous présentons ensuite les systèmes les plus intéressants, dans le cadre de notre recherche, et nous décrivons les types de rétroactions que chacun met en oeuvre. Nous étendons aussi notre analyse à quelques systèmes tutoriels en langue naturelle. Nous nous attardons plus particulièrement à la structure des messages utilisés pour relancer le processus de résolution de l'élève, car c'est le principal défi que nous devons relever pour l'implantation de la couche GMD. Finalement, nous décrivons une structure de messages d'aide à deux niveaux qui permet de s'adapter à la performance de l'élève et qui résume bien la stratégie d'aide traditionnellement employée.

Dans son article, VanLehn (2006) propose une terminologie pour décrire les systèmes tutoriels intelligents. Il les décompose en deux boucles, soit celle qui est externe et celle qui est interne. La boucle externe concerne le choix des problèmes, selon un itinéraire qui peut être adaptatif, ainsi que l'évaluation et la modélisation des compétences de l'élève. Elle n'est pas pertinente dans le cadre de notre projet, car les activités qui y sont associées sont gérées par l'enseignant dans la version actuelle de QEDX. Par contre, la boucle interne s'intéresse aux rétroactions offertes à l'élève au cours de la résolution d'un problème particulier, ce qui correspond précisément au sujet du présent chapitre. Dans cette boucle, il définit trois types de rétroactions possibles. Le premier consiste en une rétroaction minimale, qui permet, par exemple, d'indiquer des erreurs d'orthographe en modifiant la couleur du texte. Ces rétroactions, peu intrusives, se contentent habituellement d'indiquer les erreurs sans pour autant donner d'explications concernant leur source. Pour obtenir des informations plus détaillées au sujet des erreurs détectées, VanLehn parle alors de rétroaction propre associée à l'erreur. Dans ce cas, il s'agit de proposer un court texte, ou une série de textes, qui présente une explication par rapport à l'erreur commise ainsi que, dans certains cas, des indices permettant à l'élève de la corriger. Cette forme de rétroaction peut être offerte à la demande de l'élève ou automatiquement lorsque certaines catégories d'erreurs sont détectées. Enfin, le dernier type de rétroaction défini consiste à fournir des indices, automatiquement ou à la demande, concernant la prochaine étape à réaliser pour permettre de relancer le processus de résolution de l'élève bloqué. Il s'agit généralement, dans ce cas, de messages de plus en plus précis qui ont pour objectif d'inciter l'élève à exécuter une action pertinente identifiée par le système. Ces définitions des trois types de rétroactions résument bien les comportements des systèmes tutoriels et nous les avons utilisées pour analyser et comparer les systèmes recensés.

En ce qui concerne les rétroactions, Andes (Conati et al., 2002; VanLehn et al., 2005, 2010), un système tutoriel pour la physique classique, peut être considéré comme une référence, car il met en oeuvre les trois types de rétroaction définis dans la boucle interne. En effet, il offre premièrement une rétroaction minimale par l'entremise d'une coloration des équations inscrites par l'élève. Lorsque ce dernier sélectionne une équation erronée (rouge), il peut demander une indication concernant la source de l'erreur et a accès à trois niveaux hiérarchiques de messages, allant du plus vague au plus précis. Andes peut aussi afficher automatiquement un message d'aide associé aux erreurs courantes. Enfin, un élève bloqué en cours de résolution a accès à une hiérarchie de trois messages qui lui indiquent la prochaine action à exécuter. Andes propose donc des rétroactions variées qui ont un impact important sur la motivation ainsi que sur les notes des élèves. Or, son application se limite principalement aux problèmes de physique qui se résolvent par une série d'équations. De plus, il ne propose pas d'autres pistes de solution, car il donne des indices uniquement par rapport à la démarche optimale.

Du côté des systèmes tutoriels en géométrie, aucun, parmi ceux qui ont été recensés, n'offre des rétroactions aussi complètes qu'Andes. Ceux qui s'en rapprochent le plus sont le Geometry Tutor (Anderson et al., 1985; Ritter et al., 2010) et le Geometry Explanation Tutor (Alevan et al., 2001b,a). En effet, ceux-ci offrent une rétroaction minimale à l'aide de couleurs ainsi qu'une indication de la prochaine étape en utilisant une hiérarchie de messages qui propose entre cinq et sept niveaux. Par contre, les rétroactions propres signalant la source de l'erreur ne sont accessibles que pour les erreurs modélisées par l'expert du domaine et se limitent à un seul message. Cette limitation est relativement mineure, étant donné le nombre important d'erreurs modélisées, et elle n'a pas empêché ces systèmes d'avoir un succès important auprès des élèves. Ces systèmes forcent cependant l'élève à travailler en respectant un ordre strict et n'offrent pas d'aide pour d'autres stratégies de résolution.

ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991), pour sa part, offre aussi une gamme assez complète de rétroactions. En effet, il propose des explications concernant la source de l'erreur ainsi qu'une aide à la prochaine étape en utilisant une hiérarchie de messages. Cependant, l'identification des erreurs, soit la rétroaction minimale, se fait par l'intermédiaire de messages qui doivent être acquiescés, ce qui peut déconcentrer l'élève. L'aspect intéressant de ce système est qu'il est en mesure d'offrir de l'aide pertinente tout en permettant une exploration libre du problème. Par contre, son élaboration a été abandonnée, car, malgré le fait qu'il proposait un paradigme de résolution différent, basé sur le travail des experts, aucune différence de performance significative n'a été observée par rapport à Geometry Tutor.

Des systèmes comme DEFI (Gras, 1988; Almouloud, 1992), Géométrie (Gressier, 2015) et Mentoniez (Py, 1994, 1996, 2001) n'offrent pas d'explications concernant la source d'une erreur ou d'aide à la prochaine étape. Ceux-ci proposent tout de même une rétroaction, sous forme de message, indiquant la validité de l'inférence proposée par l'élève ainsi que la liste des erreurs identifiées. Dans le cas de DEFI et de Géométrie, on parle véritablement de validation locale, car seule la structure de l'inférence courante est vérifiée. Par contre, Mentoniez utilise une reconnaissance de plan afin d'identifier plus précisément les erreurs de l'élève dans les cas où plusieurs explications sont plausibles. Étant donné que ces systèmes n'offrent pas d'aide à la prochaine étape et qu'ils valident seulement des inférences complètes, ils ne sont pas pertinents dans le cadre de ce chapitre.

Nous nous sommes enfin intéressés aux systèmes tutoriels en langue naturelle, par exemple AutoTutor (Graesser et al., 2004; Olney et al., 2010; Nye et al., 2014), afin de comprendre la façon dont ceux-ci gèrent une discussion. Les défis à relever dans ce domaine concernent principalement la compréhension du texte entré par l'utilisateur et la production d'une rétroaction sous forme de phrases cohérentes et valides. La stratégie d'aide est cependant

relativement semblable à celle utilisée par les autres systèmes tutoriels et est gérée par un ensemble de scénarios ou par une machine à états finis. Ces systèmes sont, en général, en mesure de reconnaître les erreurs de l'élève, lui donner des explications supplémentaires et lui proposer le prochain sujet à travailler. Dans ce dernier cas, ils produisent, comme pour les systèmes traditionnels, une hiérarchie de messages concernant un élément précis de la réponse complète. Habituellement, le dernier message de la hiérarchie indique explicitement l'élément recherché. Par conséquent, même si ces systèmes proposent des discussions qui nous semblent naturelles, ils utilisent une approche d'aide classique. De plus, ce type de système fonctionne bien pour des questions qualitatives qui requièrent des réponses complexes, mais s'adapte difficilement aux domaines plus formels comme les mathématiques ou la géométrie.

Dans son article, Anohina-Naumeca (2007) propose une organisation des messages à deux niveaux (figure 6.1) qui résume bien la structure d'aide utilisée par la majorité des systèmes tutoriels. En effet, dans le bas de la figure on retrouve la hiérarchie typique des messages, allant du plus général au plus spécifique. La couche supérieure permet de regrouper les messages selon différents échelons afin de rendre possible une aide adaptative comme celle offerte dans l'Advanced Geometry Tutor (Matsuda et VanLehn, 2005; Matsuda, 2004). Dans ce cas, on choisit l'échelon de départ approprié, selon la compétence de l'élève, et l'on affiche le premier message correspondant pour ensuite descendre dans la hiérarchie. Il s'agit donc, encore une fois, d'une vision très linéaire de la structure des messages qui n'offre pas d'autres pistes de solution, comme le fait un enseignant.

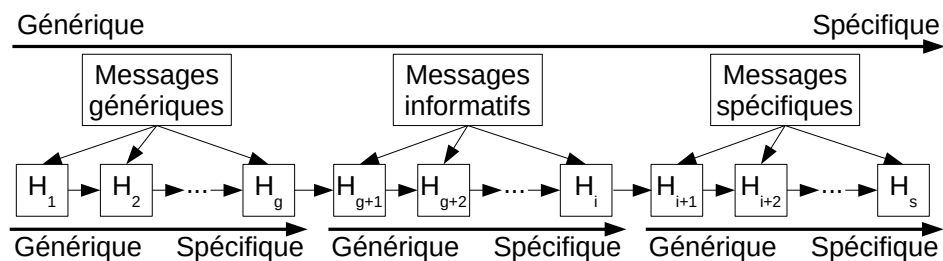


Figure 6.1 Organisation des messages sur deux niveaux (adapté de Anohina-Naumeca, 2007)

En résumé, on peut classer l'aide proposée dans les différents systèmes tutoriels en trois catégories, soit la rétroaction minimale, l'explication de l'erreur et l'aide à la prochaine étape. Parmi les systèmes recensés, Andes (Conati et al., 2002; VanLehn et al., 2005, 2010) est une référence, car il offre les trois types de rétroaction. En ce qui concerne la géométrie, c'est le Geometry Tutor (Anderson et al., 1985; Ritter et al., 2010) qui propose l'aide la plus complète, mais il ne permet pas à l'élève d'explorer librement le problème et il utilise le paradigme de la géométrie formelle, ce qui est incompatible avec nos objectifs de recherche. Nous nous sommes

cependant inspiré principalement des rétroactions proposées par ces deux systèmes, car ils sont des références dans leur domaine respectif. Par contre, aucun système recensé ne propose de multiples pistes de solution à un élève qui est bloqué dans sa résolution. Ils utilisent tous, incluant les tuteurs en langue naturelle, une structure de message hiérarchique, illustrée à la figure 6.1, qui ne concerne que l'étape optimale à exécuter. Nous devons donc proposer une structure de messages différente dans le but d'imiter la stratégie d'aide de l'enseignant. Dans la section suivante, nous présentons les résultats d'analyses des interactions entre les élèves et leurs enseignants pour ensuite décrire la mise en oeuvre de ceux-ci dans QEDX.

6.3 Analyse des interactions enseignants-élèves

À la lumière de notre analyse des systèmes tutoriels intelligents existants, il en ressort qu'aucun d'entre eux n'offre une aide aussi riche et diversifiée que celle proposée par un enseignant. Ce constat était basé, au départ, sur une compréhension intuitive des stratégies de ce dernier ainsi que sur des expériences personnelles. Dans le but d'appuyer notre critique, mais aussi de recueillir des informations utiles à la mise en oeuvre de l'agent-tuteur de notre système, nous avons réalisé une expérimentation dans des classes de niveau secondaire. Elle visait, entre autres, à recueillir des traces d'interactions entre les enseignants et leurs élèves dans le cadre de la résolution de problèmes de preuve en géométrie. L'objectif de cette section est donc de présenter cette expérimentation, qui a, en fait, servi à clore le premier cycle de conception de QEDX (figure 3.2), ainsi que les résultats obtenus. L'analyse des données recueillies lors de celle-ci, accomplie essentiellement par l'équipe de didactique, nous a permis d'identifier certaines stratégies d'aide à partir desquelles nous avons dégagé une stratégie dominante qui a été modélisée et implantée dans QEDX. La section débute par la formulation du problème du rectangle, qui est utilisé pour illustrer nos propos, ainsi que d'une solution acceptable. Par la suite, nous décrivons le contexte de l'expérimentation au cours de laquelle GeoGebraTUTOR (GGBT), une version rudimentaire de QEDX, a été employée. Un résumé de la stratégie d'aide d'un enseignant typique, basé sur nos observations, est ensuite proposé. Il est à noter que les différentes rétroactions implantées dans notre système sont, en fait, inspirées de cette stratégie. Enfin, nous exposons la méthode qui a permis d'identifier la structure générale des messages d'aide adoptée par l'enseignant dans un contexte précis. Un exemple de cette structure, adaptée au problème du rectangle, est aussi proposé.

6.3.1 Problème du rectangle

Dans le cadre du projet, cinq problèmes distincts ont été encodés dans QEDX par l'équipe de didactique. L'énoncé du plus simple d'entre eux, soit le problème du rectangle, est donné

à la figure 6.2. Nous l'utilisons, d'une part, pour permettre aux élèves de se familiariser avec le système, mais aussi pour démontrer le fonctionnement interne de QEDX.

Démontrer qu'un quadrilatère ayant trois angles droits est un rectangle.

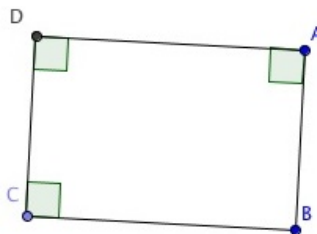


Figure 6.2 Problème du rectangle

Pour résoudre le problème du rectangle, 23 preuves différentes sont acceptées dans son implantation actuelle. Nous proposons d'utiliser la preuve suivante afin d'illustrer nos propos :

- Comme **AB est perpendiculaire à AD**, et **AD est perpendiculaire à CD**, alors, selon la propriété qui dit «**Deux droites perpendiculaires à une même troisième sont parallèles entre elles**», **AB est parallèle à CD**.
- Comme **BC est perpendiculaire à CD**, et **AB est parallèle à CD**, alors, selon la propriété qui dit «**Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre**», **AB est perpendiculaire à BC**.
- Comme **AB est perpendiculaire à AD**, **AD est perpendiculaire à CD**, **BC est perpendiculaire à CD**, et **AB est perpendiculaire à BC**, alors, selon la propriété qui dit «**Un rectangle est un quadrilatère ayant quatre angles droits**», **ABCD est un rectangle**.

Cette démonstration comporte trois phrases, ou *inférences*, qui correspondent chacune à un pas de la preuve. Chaque phrase débute par un certain nombre de prémisses, soit les antécédents. Ceux-ci sont suivis par une justification qui permet de déduire un conséquent. On remarque que le conséquent d'une inférence peut être recyclé comme antécédent d'une autre pour produire une démonstration complète. Dans le cadre de ce chapitre, nous nous servons de l'inférence comme élément de base de nos analyses, car elle permet d'obtenir une décomposition naturelle de la preuve. De plus, les résultats et analyses présentés dans cette section s'appliquent aisément à toute inférence étant donné qu'elles partagent une structure et une fonction similaires.

Afin de pouvoir manipuler les inférences à l'intérieur de QEDX, nous les avons encodées sous forme d'arbres. Cette structure permet de décrire l'organisation de chaque inférence, ainsi que celle de la preuve, de façon graphique, ce qui en facilite la compréhension. La figure

6.3 présente l'arbre associé à la deuxième inférence de la démonstration précédente et sert de référence pour la majorité des analyses et des exemples proposés par la suite. Celle-ci montre clairement la relation hiérarchique qui existe entre les antécédents, la justification et le conséquent. Dans le but d'alléger le texte, nous avons, de plus, associé à chaque énoncé un identifiant qui est inscrit dans le noeud correspondant.

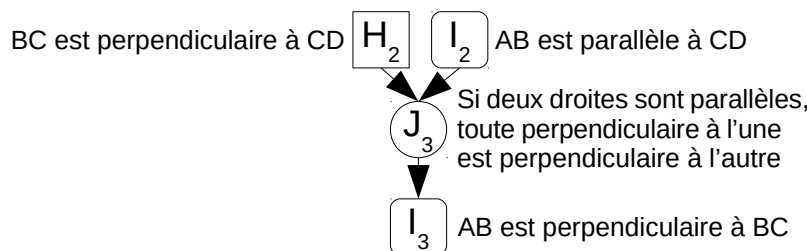


Figure 6.3 Encodage d'une inférence acceptée pour le problème du rectangle

6.3.2 Description de l'expérimentation initiale

La conception de QEDX consiste en une série d'aller-retour entre des analyses didactiques et la mise en oeuvre informatique des conclusions obtenues. Le logiciel produit à chaque cycle est soumis à une expérimentation dans le but de le valider, mais aussi de recueillir des données pour le cycle suivant de son élaboration. La première expérimentation réalisée dans le cadre du projet visait donc deux objectifs principaux. Le premier consistait à valider les solutions de quatre problèmes implantés dans notre système, incluant le problème du rectangle. L'autre objectif, qui est relatif aux travaux présentés dans ce chapitre, était de recueillir des informations concernant les interactions entre les élèves, le logiciel et l'enseignant. Les données obtenues ont servi de bases lors de l'élaboration d'une première version du tuteur discursif dans QEDX.

L'expérimentation s'est déroulée à l'École d'Éducation Internationale de Laval où nous avons pu compter sur la collaboration de deux enseignants. Nous avons eu accès à quatre classes d'élèves doués de 15 et 16 ans, pour un total de 54 binômes. Les classes ont été choisies afin de maximiser la probabilité d'atteindre nos objectifs, étant donné les rétroactions rudimentaires proposées dans GGBT. Pour chaque classe, quatre périodes de 75 minutes nous ont été allouées. Lors de la première période, nous avons fait un rappel des notions concernant la preuve et les élèves ont résolu des problèmes sur papier. Ces exercices avaient pour but de les informer quant au style de démonstration accepté par GGBT. Au cours des deux séances suivantes, les élèves ont résolu les quatre problèmes implantés dans notre système. C'est lors de celles-ci que nous avons récolté l'essentiel des informations relatives aux stra-

tégies d'aide employées par les enseignants. La dernière période a servi à faire un retour sur l'expérimentation et à recueillir les commentaires des élèves et de leurs enseignants.

L'interface de GGBT était composée de trois parties principales (figure 6.4), soit l'énoncé du problème, la figure dynamique gérée par GeoGebra (Hohenwarter et Fuchs, 2004; Hohenwarter, 2013; International GeoGebra Institute, 2015) et le dialogue avec GGBT. C'est dans cette dernière que l'on affichait les actions de l'élève qui étaient immédiatement suivies par la liste des pourcentages d'achèvement de chacune des solutions possibles pour le problème. Afin de proposer un énoncé au système, l'élève devait le choisir à l'intérieur d'un répertoire classé par thème et le compléter au besoin. Si des éléments devaient être inscrits dans l'énoncé, la couleur de fond (vert ou rouge) associée à chacun indiquait s'il respectait certaines règles simples, par exemple, la casse et le nombre de caractères attendus. Il est à noter qu'aucune vérification par rapport à la figure dynamique n'est effectuée. Les rétroactions étant rudimentaires, l'élève doit, par conséquent, faire preuve d'autonomie et demander l'assistance de l'enseignant s'il n'est plus en mesure de faire progresser les pourcentages d'achèvement. Ce dernier a accès à la chronologie des actions de l'élève, et des rétractions de GGBT, et a donc la possibilité de donner des indices qui respectent son cheminement. Pour la conception de notre tuteur, ce sont principalement ces interventions qui nous intéressent.

Lors de l'expérimentation, plusieurs outils de cueillette de données ont été utilisés. D'une part, GGBT génère des fichiers journaux qui contiennent l'ensemble des énoncés soumis par l'élève ainsi que les rétroactions du système. Ceux-ci nous permettent notamment d'analyser les démarches de résolution et de valider les solutions implantées. De plus, le travail de quatre binômes par classe a été enregistré à l'aide du logiciel ScreenFlow qui permet de capturer, en simultané, l'écran, la caméra intégrée et les conversations. Ces données, plus riches, constituent la principale source d'informations qui a servi à construire le modèle du tuteur de QEDX. Chaque enseignant portait aussi un dictaphone afin d'enregistrer ses interventions verbales. Dans le but d'en faciliter l'analyse, des verbatims de tous les enregistrements ont été produits et annotés par l'équipe de didactique. Enfin, les commentaires des enseignants et des élèves ont été recueillis. Les résultats obtenus nous ont, entre autres, permis de constater que GGBT a généralement été bien accueilli, malgré quelques erreurs dans les solutions et une interface causant certaines frustrations (Tessier-Baillargeon, 2016). Ces problèmes, qui ont été corrigés depuis, n'ont toutefois pas affecté significativement les interventions de l'enseignant concernant la résolution du problème posé et n'ont donc pas porté préjudice à l'analyse des interactions. L'objectif relatif à la cueillette d'informations a ainsi été atteint. Les résultats et analyses menant à l'identification d'une stratégie d'aide, basés sur les données récoltées, sont décrits dans la suite de cette section.

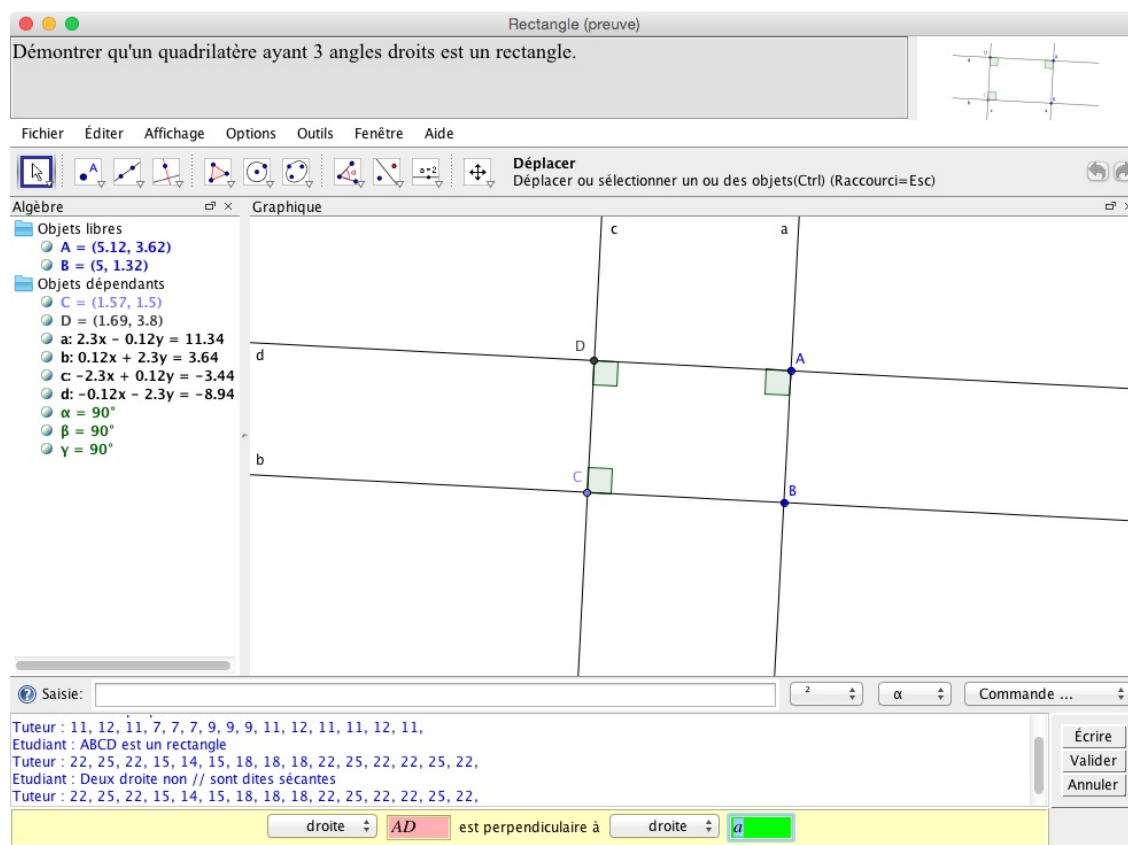


Figure 6.4 Interface de GGBT lors de la première expérimentation (2010)

6.3.3 Stratégie de l'enseignant typique observé

Les observations réalisées ainsi que les outils mis en place lors de l'expérimentation ciblaient particulièrement l'identification des stratégies d'aide adoptées par les enseignants. Celles-ci ont été mises en lumière lors des analyses effectuées essentiellement par les didacticiens de l'équipe (Tessier-Baillargeon, 2016). Nous présentons maintenant une description sommaire d'une stratégie typique que nous avons identifiée. Dans le cadre de la résolution d'un problème de géométrie, un élève bloqué dans son processus de résolution fait habituellement appel à l'enseignant. À ce moment, ce dernier regarde l'ensemble des énoncés inscrits par l'élève et tente de se faire une idée par rapport à sa stratégie de résolution en repérant des éléments clés. Il ne lui est pas nécessaire de connaître en détail chaque solution, comme c'est le cas dans QEDX qui les modélise par des graphes HPDIC. Afin de préciser la stratégie de l'élève, l'enseignant peut lui poser des questions choisies judicieusement. Dans notre système, la stratégie est ciblée différemment par la couche EDOl qui recherche la solution la plus avancée. À partir de la stratégie déduite et des éléments inscrits par l'élève, l'enseignant tente alors de l'aider efficacement à compléter une solution. Dans le but de respecter l'état cognitif de l'élève,

il choisit habituellement de fournir des indices par rapport à son travail récent, soit la dernière inférence incomplète sur laquelle l'élève a travaillé. S'il ne parvient pas à relancer le processus de résolution, il lui propose alors de travailler sur d'autres inférences, soit différentes pistes de solution, choisies en se référant à l'historique des actions de l'élève avant de lui donner une partie de la réponse. Dans QEDX, la chronologie des actions est conservée dans la couche MIA tandis que le plan d'intervention est établi par la couche EDOI.

Les interventions de l'enseignant, issues de son plan d'action, sont composées autant de messages verbaux que de rétroactions non verbales. Par exemple, lorsque l'élève répond à une question posée par l'enseignant, nous avons remarqué que le regard de ce dernier peut souvent être un bon indicateur en ce qui concerne la qualité de la réponse proposée. Il arrive aussi régulièrement que l'enseignant pointe des éléments du travail de l'élève ou lui demande de pointer un élément afin de répondre à une question. En ce qui concerne le discours, des mots d'encouragements sont utilisés pour inciter l'élève à persévérer, mais servent aussi à le rassurer par rapport à la démarche choisie. De plus, l'enseignant peut réagir lorsqu'il détecte une erreur courante et peut proposer des astuces pour l'éviter. Finalement, il pose des questions plus ou moins précises provenant de son plan d'intervention qui ont pour objectif de relancer le processus de résolution de l'élève. Une analyse plus détaillée de ces derniers messages est présentée dans la dernière partie de cette section. Ce qu'il faut retenir ici est que la stratégie d'intervention d'un enseignant est composée d'un ensemble d'actions et de rétroactions complexes qui reposent sur un plan d'intervention adapté. Les différentes rétroactions, inspirées de nos observations et mises en oeuvre par la couche GMD de QEDX, sont décrites dans la section suivante.

6.3.4 Structure des messages

Dans le but de concevoir l'agent-tuteur de QEDX, nous avons recueilli et analysé les interventions de deux enseignants dans un contexte d'enseignement réel. Nous nous sommes rapidement aperçus que les interactions étaient très riches et complexes. Pour être en mesure d'en retirer des informations utiles à l'élaboration de notre système, il a été nécessaire d'isoler certains types d'interventions. Dans le cadre de ce projet, nous avons choisi de nous restreindre aux messages associés à chaque inférence, prise individuellement, faisant partie du plan d'intervention de l'enseignant. En effet, tout nous porte à croire que ce sont ces messages qui sont les plus susceptibles de relancer le processus de résolution de l'élève, car ils contiennent généralement des indices concernant le travail à réaliser.

Notre analyse nous a alors permis de déterminer que les enseignants utilisent habituellement trois niveaux de messages pour aider les élèves, soit, dans l'ordre, les messages de type neutre,

indicatif et actif/proactif. Lors de l'expérimentation, nous n'avions pas spécifié d'ordre d'intervention par rapport aux différents niveaux et, malgré cela, les enseignants avaient tendance à proposer des messages neutres au départ. De plus, un enseignant offrait plus régulièrement des messages actifs alors que l'autre privilégiait des messages indicatifs. Par contre, en ce qui concerne le contenu des questions, tous deux avaient tendance à demander les justifications en priorité, comme si elles jouaient un rôle de premier plan dans la démonstration à construire. Une fois celle-ci identifiée, l'enseignant passait ensuite au conséquent pour terminer par les antécédents de l'inférence choisie. Il semble donc y avoir une stratégie assez systématique de la part des enseignants autant en ce qui concerne l'ordre des éléments ciblés que pour l'ordre de différents niveaux d'aide proposés. Nous avons alors tenté de mettre à profit cette régularité pour la conception de l'intelligence de notre système.

Pour être en mesure d'implanter les messages discursifs relatifs à chaque inférence dans QEDX, il nous fallait néanmoins déterminer leur organisation et repérer, si possible, des structures récurrentes. Dans un premier temps, nous avons répertorié, pour une inférence générique, les différents cas de figure possibles associés à la présence, ou à l'absence, de ses divers éléments dans le travail de l'élève. Par exemple, celui-ci peut avoir inscrit une partie des antécédents ainsi que la justification. Il lui reste donc à trouver les antécédents manquants et le conséquent. Notre analyse nous a permis de recenser un total de quatorze situations possibles, qui sont communes à toutes les inférences. Pour chacune, nous avons ensuite produit, en étroite collaboration avec l'équipe de didactique, une liste ordonnée de messages. Ceux-ci sont, en fait, extraits ou extrapolés de l'étude des interventions observées et correspondent aux messages que l'enseignant typique pourrait vraisemblablement proposer à un élève bloqué dans le but de l'aider à compléter l'inférence ciblée. Il est à noter que nous ne nous sommes imposé aucune contrainte quant à la structure ou à la nature des messages générés lors de cette tâche afin que ces derniers reflètent le plus fidèlement possible les stratégies d'aide analysées.

Nonobstant cette absence de contraintes, les messages que nous avons composés respectaient, dans la majorité des situations, une certaine structure récurrente qui a été mise en évidence lors de notre analyse. Nous avons, par la suite, modifié et ajusté quelques messages afin d'uniformiser leur structure et leur contenu. Par conséquent, des corrections mineures, qui n'ont aucun impact par rapport à la validité didactique de la structure des messages proposée, ont été effectuées. Le tableau 6.1 contient le résultat de ce travail, soit la séquence finale des messages associée à dix des quatorze situations recensées. Les quatre autres situations sont traitées par la suite. Dans le tableau, nous avons inscrit, à titre d'exemple, les messages relatifs à l'inférence présentée à la figure 6.3, issue du problème du rectangle.

Tableau 6.1 Séquence des messages associés à l'inférence de la figure 6.3 (* : au besoin)

		H	0	0	0	+	+	+	+	1	1	1
		J	0	1	1	0	0	1	1	0	0	1
		C	1	0	1	0	1	0	1	0	1	0
C_j	Quelle propriété ou définition te permet de conclure : I_3 ?	1										
C_j	Pour quelle justification le fait de connaître les hypothèses : H_2 ; I_2 est utile ?				1					1		
C_j	Quelle propriété ou définition te permet de conclure : I_3 à partir des hypothèses suivantes : H_2 ; I_2 ?					1					1	
N_j	Recherche des justifications concernant la perpendicularité et le parallélisme de droites ou de segments entre eux.	2			2	2				2	2	
N_j	La justification recherchée fait aussi allusion à deux droites ou segments parallèles tous deux perpendiculaires à une troisième droite (ou segment).	3			3	3				3	3	
C_c	Quelle conclusion peut découler de cette justification : J_3 ?		1									
C_c	Quelle conclusion peut découler de ces hypothèses : H_2 ; I_2 ?				4					4		
C_c	Quelle conclusion peut découler de l'utilisation de cette propriété : J_3 en prenant en compte ces hypothèses : H_2 ; I_2 ?						1					1
N_c	Que peux-tu déduire sur la position relative des segments AB et BC ?		2		5		2			5		2
N_c	As-tu soumis le résultat concernant la perpendicularité des segments AB et BC ?		3		6		3			6		3
C_h	Quelles hypothèses sont nécessaires pour employer la justification suivante : J_3 ?		4				4					
C_h	Quelles hypothèses sont nécessaires pour déduire cette conclusion : I_3 ?	4				4						
C_h	Quelles hypothèses sont nécessaires pour déduire cette conclusion : I_3 à l'aide du théorème : J_3 ?			1				1				
N_{h1}	Que peux-tu dire sur la position relative des segments BC et CD ?	5	5	2	7*	5*	5*	2*				
N_{h1}	As-tu soumis une affirmation concernant la perpendicularité des segments BC et CD ?	6	6	3	8*	6*	6*	3*				
N_{h2}	Que peux-tu dire concernant la position relative des segments AB et CD ?	7	7	4	9*	7*	7*	4*				
N_{h2}	As-tu soumis une affirmation portant sur le parallélisme des segments AB et CD ?	8	8	5	10*	8*	8*	5*				

Dans le haut du tableau 6.1, on retrouve des groupes de trois valeurs qui sont, en fait, un encodage pour chacune des situations. La valeur en face de chaque élément de l'inférence (H, J ou C) indique si l'ensemble des énoncés correspondant à l'élément en question a été inscrit par l'élève (1) ou s'il est totalement absent (0). Dans le cas des antécédents (H), il arrive aussi qu'un sous-ensemble partiel des énoncés nécessaires ait été inscrit par l'élève (+). On retrouve, dans la seconde colonne, l'ensemble des messages uniformisés résultant de notre analyse. Nous avons attribué une catégorie à chacun afin d'indiquer s'il s'agit d'une mise en contexte (C) ou d'un indice par rapport à l'énoncé, contenu dans un noeud (N) (figure 6.3), ciblé par la mise en contexte. En indice, on indique le noeud de référence à l'intérieur de l'inférence. Ces catégories sont utilisées dans la description de l'implantation du système tutoriel présenté à la section suivante. Enfin, pour chaque situation, on retrouve une valeur associée aux différentes phrases qui correspond à l'ordre chronologique de l'utilisation de celles-ci dans le discours de l'enseignant typique. Certaines phrases, fournissant des indices par rapport aux antécédents, seront utilisées seulement si les antécédents associés n'ont pas encore été proposés par l'élève (*). Le tableau 6.1 propose donc une synthèse de la structure

des messages d'un enseignant typique dans dix situations déterminées. La mise en oeuvre du module de génération des messages discursifs a été basée, en grande partie, sur celui-ci.

En ce qui concerne les quatre situations qui ne sont pas documentées dans le tableau, il s'agit, en fait, de cas particuliers. En effet, elles correspondent au traitement des inférences qui n'ont pas encore été travaillées par l'élève (000), soit avant qu'un premier énoncé valide n'ait été soumis (début) ou en cours de résolution, et à celles qui sont déjà complétées (111), soit en cours de résolution ou lorsqu'une solution est achevée. Pour chacune de ces situations, nous n'avons pas été en mesure d'uniformiser les messages sans affecter leur validité sur le plan didactique. Par exemple, dans le cas 000 en début de problème, l'enseignant a tendance à proposer des indices concernant les stratégies générales de résolution à utiliser et s'assure que l'élève a bien compris non seulement l'énoncé, mais aussi les enjeux du problème. Il ne fait donc pas référence, comme dans les autres situations, à une inférence particulière. À l'opposé, dans le cas 111, l'enseignant n'a rien à faire, mais il peut parfois en profiter pour encourager ou féliciter l'élève. Par conséquent, ces cas sont traités de façon distincte dans QEDX.

En plus des quatre situations décrites précédemment, le message qui consiste à donner une partie de la réponse à l'élève, que nous appelons le message final, n'a pas été inscrit dans le tableau 6.1. En effet, celui-ci ne semble pas être produit selon une stratégie fixe, mais dépend plutôt du déroulement de l'interaction entre l'élève et l'enseignant ainsi que de la personnalité de ce dernier. De plus, il est rare que le message final soit proposé à la suite d'une seule série hiérarchique d'indices, comme c'est habituellement le cas dans les systèmes tutoriels que nous avons recensé. Nous avons, par conséquent, décidé de ne pas les indiquer dans le tableau, car ils ne sont généralement pas associés aux indices concernant une inférence particulière, mais sont plutôt produits après avoir exploré différentes pistes de solution. Il a cependant été essentiel de les intégrer dans QEDX, car ils forment l'aide de dernier recours pour relancer le processus de résolution de l'élève.

La dernière observation intéressante concerne le degré de précision des messages. En effet, la stratégie générale consiste à proposer des indices très génériques pour se diriger graduellement vers des messages fournissant une partie de la réponse en utilisant un nombre variable de niveaux intermédiaires. Cette hiérarchie a été observée pour les quatorze situations analysées, si nous isolons la structure des messages pour un noeud particulier de l'inférence. Par exemple, les indices permettant de déterminer la justification de l'inférence de la figure 6.3 débutent par une mise en contexte (C_j) qui dépend du travail réalisé par l'élève. Celle-ci ne fournit, cependant, aucune information particulière concernant la justification à produire. Par contre, les messages relatifs au noeud (N_j) fournissent graduellement des informations qui sont de plus en plus précises. Enfin, un message final, du genre «As-tu pensé soumettre la justification

J_3 », pourrait être proposé par l’enseignant. Par ailleurs, nous avons observé que l’enseignant a tendance à utiliser, en alternance, des messages de niveau C et N avant de donner un message final afin d’inciter l’élève à explorer différentes pistes de solution, comme on peut l’observer dans le tableau 6.1. Nous désirons donc implémenter cette stratégie dans QEDX afin d’imiter le plus fidèlement possible le travail de l’enseignant, ce qui est innovateur par rapport à l’approche préconisée par les systèmes tutoriels actuellement offerts.

L’objectif principal de cette section consistait à exposer les résultats pertinents pour l’élaboration du GMD obtenus lors de l’expérimentation qui a servi à recueillir les informations concernant les interactions entre les enseignants et leurs élèves. Nous avons d’abord présenté le problème du rectangle, ainsi qu’une de ses solutions, car c’est un problème simple, implanté dans QEDX, qui permet d’illustrer nos propos. Nous avons ensuite décrit le contexte de l’expérimentation qui s’est déroulée dans une école secondaire. Quatre classes d’élèves ont utilisé GGBT, une version rudimentaire de QEDX, ce qui nous a permis de recueillir des données sur les interactions entre ceux-ci et leurs enseignants. L’analyse des données, décrite en détail dans Tessier-Baillargeon (2016), a, d’une part, permis de mieux comprendre la stratégie d’intervention de l’enseignant. Nous avons montré le lien entre celle-ci et les quatre couches de QEDX qui, en fait, en sont inspirées. Toutefois, le principal but de l’analyse était de comprendre la structure des messages de l’enseignant. Il est important de rappeler que celui-ci utilise à la fois des rétroactions non verbales et verbales pour aider et encourager l’élève. Or, ce sont les messages verbaux qui véhiculent le plus d’informations et qui sont les plus ardues à analyser. Nous avons donc choisi de nous concentrer sur les messages relatifs à une inférence particulière afin de simplifier leur étude. Nous avons alors recensé quatorze situations possibles pour lesquelles nous avons produit des séquences de messages. Par la suite, un travail d’uniformisation, qui n’affecte pas la validité didactique des messages, a conduit à la production du tableau 6.1. Des cas particuliers ont, par contre, été identifiés pour quatre situations ainsi que pour le traitement des messages finaux. Néanmoins, en isolant les messages associés à un noeud, nous retrouvons une hiérarchie qui ressemble à celle qui est implantée dans les systèmes tutoriels usuels. Par contre, dans le cadre de QEDX, nous voulons proposer, à la manière de l’enseignant, diverses pistes de solution avant de fournir un élément de la réponse. La majeure partie des informations présentées dans cette section ont été exploitées lors de la conception du module de génération des messages qui est décrit à la section suivante.

6.4 Mise en oeuvre des rétroactions de QEDX

L’analyse des données issues de la première expérimentation nous a, entre autres, permis de déterminer les rétroactions ainsi que les messages d’aide typiques utilisés par les enseignants.

Il fallait maintenant, pour atteindre un des objectifs de notre projet, les implanter à l'intérieur de QEDX. Dans cette section, nous décrivons donc les techniques et les algorithmes utilisés afin d'offrir un système tutoriel inspiré du travail des enseignants. Nous avons divisé la présente section en deux parties principales. Dans la première, nous traitons des messages instantanés, soit ceux qui sont immédiatement produits à la suite de la soumission d'un énoncé par l'élève et qui en indique la validité. Nous présentons à la fois les rétroactions minimales et les messages propres à l'erreur, tels que décrits par VanLehn (2006). Concrètement, nous faisons un retour sur les rétroactions sémaphoriques, utilisées dans GGBT, avant d'expliquer notre utilisation des émoticônes. Les rétroactions plus complexes, sous forme de phrases courtes, sont ensuite abordées. La deuxième partie traite des messages d'aide concernant la prochaine étape du processus de résolution, soit le troisième type de message répertorié par VanLehn. Nous décrivons, plus précisément, le travail réalisé afin de transformer les stratégies contenues dans le tableau 6.1 en une machine à états finis. De plus, nous expliquons les caractéristiques de cette dernière qui ont rendu possible la prise en charge des situations particulières qui ont été exclues du tableau. Tout au long de cette section, nous allons, enfin, justifier nos principaux choix en utilisant les résultats présentés dans la section précédente.

Avant de passer à la description de notre implantation pour les différents types de messages, nous jugeons opportun de présenter brièvement l'interface actuelle de QEDX (figure 6.5). Celle-ci a été construite en réponse aux commentaires recueillis et aux observations réalisées lors de l'expérimentation décrite précédemment. Une première amélioration importante concerne l'ajout des onglets. En effet, ceux-ci permettent à l'élève de basculer entre le répertoire des phrases et la figure sans perdre son travail. Nous avons aussi ajouté la preuve courante, à compléter sous forme de texte à trous, afin d'indiquer clairement aux élèves les éléments manquant pour obtenir une démonstration complète. L'ajout du tuteur intelligent correspond à la seconde amélioration majeure. Afin de rendre la conversation entre l'élève et le tuteur plus facile à lire, nous avons déplacé la fenêtre de clavardage simulé à droite de la figure. À l'intérieur de cette dernière, nous pouvons voir différents messages proposés par notre système (vert) en réaction au travail de l'élève (bleu). La majorité des types de messages décrits par la suite sont présents dans cette figure et nous y faisons donc référence afin d'illustrer nos propos.

6.4.1 Rétroactions instantanées

Dans cette première partie, nous décrivons la façon dont nous avons implanté les messages qui apparaissent immédiatement à la suite de la proposition d'un énoncé par l'élève, que nous qualifions de rétroactions instantanées. Elles ne contiennent habituellement aucun in-

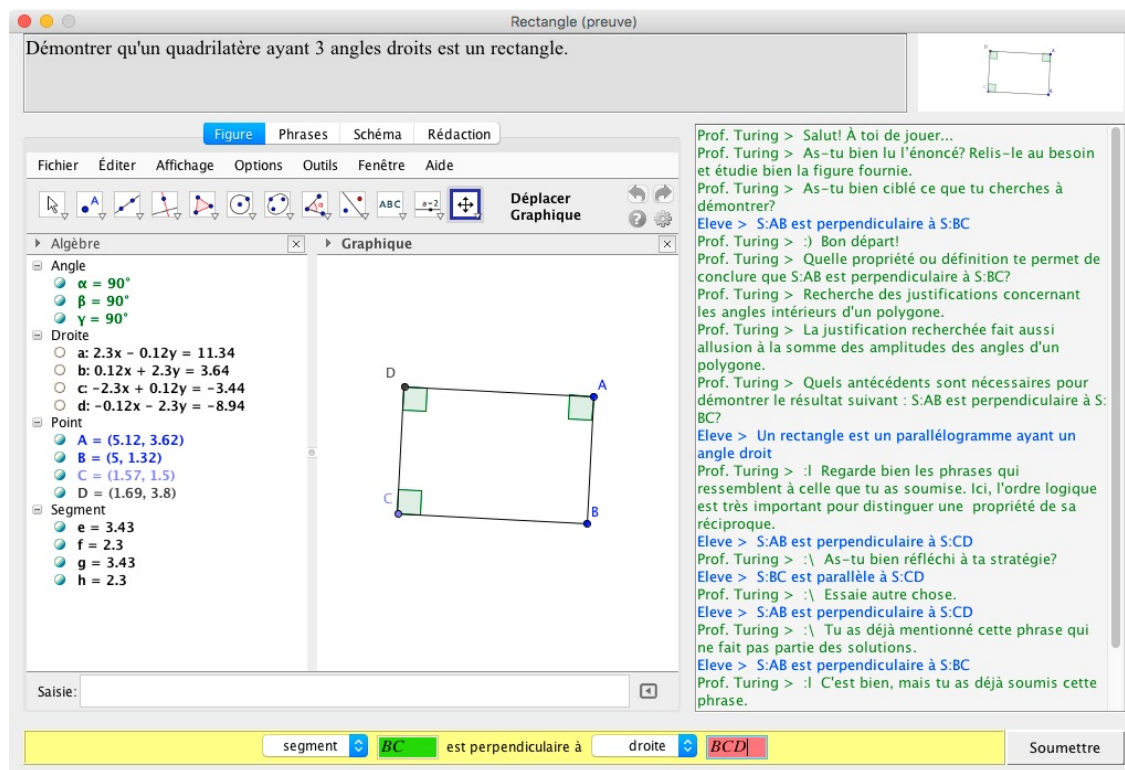


Figure 6.5 Interface actuelle de QEDX

dice concernant la stratégie de résolution, contrairement aux messages d'aide à la prochaine étape dont l'implantation est décrite dans la seconde partie. En fait, les messages instantanés correspondent, en général, à la réaction spontanée, verbale ou non verbale, de l'enseignant en réponse à une affirmation de l'élève. L'enseignant peut, par exemple, donner son approbation en esquissant un sourire ou en offrant quelques mots d'encouragement. Par contre, lorsque l'élève fait une erreur, il peut lui signifier et lui proposer un correctif approprié s'il le juge utile. Au cours des paragraphes qui suivent, nous décrivons donc la mise en oeuvre de ces différentes rétroactions à l'interface de QEDX. Nous y présentons les trois types de rétroactions instantanées implantées dans notre système, soit les rétroactions sémaphoriques, les émoticônes et les messages discursifs.

Le premier type de rétroaction auquel l'élève a accès, que nous nommons la *rétroaction sémaphorique*, permet d'indiquer, en temps réel, la validité des éléments géométriques inscrits dans un énoncé avant de le soumettre. On peut voir, au bas de la figure 6.5, un exemple de cette rétroaction. Celle-ci existait dans GGBT et a été conservée, car les élèves nous ont indiqué, dans leurs commentaires, qu'elle leur évite des pertes de temps. En effet, on indique, par un changement de la couleur de fond de la zone de texte, si l'élément géométrique respecte des règles d'orthographe simples associées à son type. Nous vérifions, par exemple, si la

casse et le nombre de caractères attendus sont respectés. Cette fonctionnalité permet donc d'éviter de soumettre des énoncés qui sont jugés invalides par le système, à cause d'une faute de frappe. Par contre, la rétroaction ne donne aucune indication en ce qui concerne la validité de l'élément par rapport à la figure, gérée par GeoGebra (Hohenwarter et Fuchs, 2004; Hohenwarter, 2013; International GeoGebra Institute, 2015), qui est affichée dans l'onglet *Figure*. Nous avons fait ce choix, car l'interface logicielle offerte par GeoGebra, à l'époque, ne nous permettait pas d'effectuer facilement une telle vérification. Par conséquent, la rétroaction sémaphorique indique que l'élément est valide si, par exemple, l'élève entre deux lettres majuscules pour nommer un segment, car il respecte les règles d'orthographe, même si celui-ci n'est pas présent sur la figure.

La rétroaction sémaphorique est le seul type de rétroaction, implanté dans QEDX, qui n'a pas un lien direct avec les actions d'un enseignant en classe. En effet, ce dernier ne corrige pas systématiquement tous les éléments géométriques écrits par l'élève, car il est souvent en mesure de déduire l'intention de celui-ci, même si des erreurs sont présentes. Cependant, certains proposent parfois, dans des contextes d'enseignement précis, des phrases à trous qui contiennent une case pour chaque caractère à inscrire. Ces indices s'apparentent, en fait, aux rétroactions sémaphoriques implantées, mais leur utilité est limitée. À l'opposé de l'enseignant, qui est en mesure d'interpréter les énoncés de l'élève, QEDX, pour sa part, les analyse de manière formelle et est donc intolérant aux fautes de frappe. La rétroaction sémaphorique est, par conséquent, un outil pour aider l'élève à respecter la structure plus rigide imposée par notre système.

Les émoticônes, affichées en réaction à la soumission d'un énoncé par l'élève, forment le second type de rétroaction instantanée offert dans QEDX. Elles tirent leur inspiration à la fois des observations réalisées lors de la première expérimentation, ainsi que des réactions non verbales de l'enseignant. D'une part, lors de l'expérimentation, les élèves avaient accès au pourcentage d'achèvement de chacune des solutions pour évaluer leur progression. Même si cette information est très rudimentaire, il n'en demeure pas moins qu'elle leur offrait une rétroaction immédiate concernant la validité de l'énoncé proposé. Comme celle-ci semblait avoir un effet positif sur la motivation des élèves à compléter la preuve associée au problème posé, nous avons voulu intégrer une information équivalente dans la version actuelle de QEDX. D'autre part, c'est l'observation des réactions non verbales des enseignants qui nous a offert une piste de solution. En effet, ces derniers adoptent différentes expressions faciales qui permettent à l'élève de juger de la validité de l'énoncé proposé. Par exemple, un sourire indique à celui-ci qu'il est dans la bonne voie, alors qu'un regard déçu est souvent associé à une erreur. L'utilisation des émoticônes nous a donc paru appropriée, car elles permettent de reproduire les principales expressions faciales de l'enseignant en réaction au travail de l'élève.

Dans notre système, une émoticône est toujours le premier élément affiché en réaction à chaque énoncé introduit par l'élève, comme on peut le voir, dans la fenêtre de droite, dans la figure 6.5. En effet, à la suite de chaque entrée de l'élève (bleue), Prof Turing (vert) répond par une émoticône avant de produire au besoin un message. Le choix de celle à afficher dépend du verdict, à la suite de l'analyse de l'énoncé soumis, mais aussi d'une métrique associée à ce verdict. Par exemple, si l'énoncé n'est pas reconnu dans le contexte de résolution d'un problème, celui-ci sera considéré comme invalide et la métrique associée correspond au nombre d'énoncés invalides produits depuis le dernier énoncé valide. Par contre, si l'énoncé fait partie du graphe des solutions, il est donc valide et, dans ce cas, on y associe le pourcentage d'achèvement de la solution la plus avancée. Par conséquent, il est possible d'ajuster l'expression de Prof Turing lorsque l'élève atteint, par exemple, un certain seuil d'achèvement. Dans la version actuelle de QEDX, cinq verdicts sont possibles et, pour chacun, on y associe une métrique caractéristique, comme indiqué dans le tableau 6.2. Les émoticônes associées à chaque combinaison de verdict et de métrique sont décrites dans un fichier texte afin de permettre à l'équipe de didactique de les modifier aisément.

Tableau 6.2 Métrique rattachée à chacun des verdicts possibles

ID	Verdict	Métrique
VA	Énoncé valide	% d'achèvement de la solution la plus avancée
VR	Répétition d'un VA	Nombre de VR depuis le dernier VA
RE	Réciproque	Nombre de RE depuis le début du problème
IN	Énoncé invalide	Nombre de IN depuis le dernier VA
IR	Répétition d'un IN	Nombre de IR depuis le dernier VA

Le dernier type de rétroaction instantanée, facultative, implantée dans la version actuelle de QEDX, consiste en un message discursif, juxtaposé à l'émoticône, qui concerne directement l'énoncé proposé par l'élève. Ce message s'affiche au besoin et sert soit à encourager celui-ci ou à lui proposer un correctif par rapport à une erreur identifiée. En ce qui concerne les rétroactions visibles dans la figure 6.5, elles proposent toutes, néanmoins, un message discursif. Les messages sont composés en collaboration avec l'équipe de didactique et sont inscrits dans un fichier en format texte afin de permettre à ces derniers de les ajuster au besoin. De plus, on associe à chacun un contexte d'utilisation qui fait référence au verdict, incluant la métrique correspondante, ou à un énoncé particulier. D'une part, il est possible d'utiliser le verdict concernant l'énoncé ainsi que la métrique associée, tels qu'établis dans le cas des émoticônes, pour produire des messages d'encouragement ou des messages génériques. Nous permettons, d'autre part, de proposer des messages particuliers concernant des énoncés précis identifiés. Par exemple, si nous savons que les élèves confondent le concept de parallélisme et

de perpendicularité, il est possible d'indiquer un message unique lorsque l'énoncé erroné est détecté dans le cadre de la résolution d'un problème donné. Il est aussi intéressant de noter qu'il est possible de proposer un ensemble de messages pour chaque contexte et QEDX en choisira un au hasard afin de permettre un peu de variabilité dans les rétroactions.

Ces messages discursifs instantanés sont, dans leur ensemble, inspirés de nos observations concernant le discours spontané de l'enseignant typique en réaction aux réponses proposées par les élèves. En effet, celui-ci peut, par exemple, offrir des mots d'encouragement pour faire part de la progression de l'élève. Par contre, dans le cas d'une mauvaise réponse, un message est généralement de mise pour éviter à ce dernier du travail inutile. Dans QEDX, ces messages discursifs sont donc un complément aux émoticônes, comme le sont les messages spontanés des enseignants par rapport à leur expression faciale. Ils correspondent aussi, en partie, aux rétroactions propres à l'erreur telles que décrites par VanLehn (2006). En effet, les indices propres, concernant une erreur particulière, qui sont proposés par les enseignants et offerts dans notre système, font partie de cette catégorie.

Dans cette partie, nous avons présenté les trois types de rétroaction instantanée offerts à l'interface de QEDX. Premièrement, la rétroaction sémaphorique sert à indiquer si les éléments géométriques, inscrits à l'intérieur des énoncés, respectent certaines règles afin d'éviter à l'élève des pertes de temps causées par des fautes de frappe. Elle permet, généralement, de compenser la rigueur de QEDX, lors de l'analyse des énoncés, par rapport à la souplesse adoptée par les enseignants. Par contre, les émoticônes et les messages discursifs instantanés sont grandement inspirés du comportement des enseignants observés. Leur mise en oeuvre dans QEDX permet d'offrir la flexibilité nécessaire pour produire des rétroactions adaptées aux différents contextes identifiés tout en étant faciles à modifier par l'équipe de didactique. Ces rétroactions correspondent aux deux premières catégories définies par VanLehn (2006), alors que l'aide à la prochaine étape est présentée dans la partie suivante.

6.4.2 Machine à états finis produisant les messages d'aide discursifs

Les rétroactions instantanées sont très commodes pour motiver l'élève et lui éviter du travail inutile le menant vers une solution invalide. Cependant, lorsque le processus de résolution de l'élève est bloqué, elles n'offrent qu'une aide très limitée pour le relancer. Pour remédier à la situation, l'enseignant engage alors une conversation avec l'élève afin de lui fournir les indices utiles à la poursuite de son travail. Dans cette partie, nous décrivons donc la machine à états finis, développée dans le cadre du projet, qui permet de reproduire certains éléments de ces interventions à l'interface de QEDX. L'architecture de cette machine découle principalement des analyses présentées à la section 6.3.4, concernant la structure des messages

d'aide. Dans les paragraphes qui suivent, nous proposons, tout d'abord, un encodage des informations contenues dans le tableau 6.1 sous forme d'états et de transitions. Par la suite, nous expliquons la pertinence des autres états inclus dans notre machine par rapport aux différents cas particuliers répertoriés. Nous utilisons aussi la conversation de la figure 6.5 dans le but d'illustrer nos propos.

On se rappelle que le tableau 6.1 a été produit à la suite de l'analyse des interventions des enseignants observés afin d'inventorier les séquences de messages typiques qui pourraient être proposées dans quatorze situations recensées. Cependant, celui-ci ne présente que dix situations ayant une structure de messages relativement uniforme alors que pour chacun des quatre autres cas, qui ont été omis, une structure spécifique leur est associée. Dans ce tableau, on retrouve une liste de messages ainsi que l'ordre séquentiel de ceux-ci pour chaque situation. Nous avons aussi regroupé les messages en différentes catégories que nous avons indiquées dans la colonne de gauche. Ceux dans la catégorie C correspondent à la mise en contexte qui permet d'indiquer le noeud de l'inférence qui sera traité par les messages suivants, soit ceux dans la catégorie N . De plus, l'indice associé à chaque catégorie fait référence au noeud qui lui est rattaché : hypothèse ($h1, h2, \dots, h_n$), justification (j) ou conclusion (c).

En analysant les séquences de messages, nous remarquons qu'un message de type C précède toujours ceux de type N relatifs au noeud ciblé. En fait, dans notre tableau, un seul message de type C est produit pour chaque noeud de justification (N_j) ou de conclusion (N_c) ou pour l'ensemble des noeuds d'hypothèses (N_{hn}) et son contenu dépend de l'état de l'inférence. Effectivement, celui-ci reprend les énoncés déjà produits par l'élève, sans plus, pour éviter de lui donner des réponses lors de la mise en contexte. En ce qui concerne les messages de type N correspondants à un noeud particulier, ils sont, en général, tous produits dans un ordre précis, afin de fournir graduellement des éléments relatifs à l'énoncé recherché. Il est aussi important de noter que les messages (C et N), associés à un noeud particulier, ne sont pas proposés si l'énoncé correspondant a été préalablement inscrit par l'élève. Cependant, dans le cas des hypothèses, le message de type C est produit dès que l'ensemble est incomplet, mais seuls les messages de type N utiles pour le compléter sont ensuite proposés. Enfin, on remarque que l'ordre de visite des noeuds est toujours le même. En effet, on s'intéresse en priorité à la justification pour ensuite rechercher la conclusion et terminer par les hypothèses.

L'organisation des messages contenue dans le tableau 6.1 a été modélisée à l'aide d'une machine à états finis illustrée à la figure 6.6. Concrètement, chaque catégorie de messages issue du tableau correspond à l'état du même nom dans la machine. Par exemple, l'ensemble des messages de type C_j est géré par l'état C_j , alors que la liste de tous les messages de type N_j se retrouve dans l'état N_j . Afin de montrer les liens qui existent entre notre machine et la struc-

ture des messages, l'état C_j sera, pour l'instant, notre point de départ. En effet, il est relatif au noeud de justification qui est habituellement privilégié par les enseignants observés. Dans cet état, deux transitions sont possibles. D'une part, la transition Act (énoncé(s) Activé(s)) sert à éviter l'affichage des messages concernant la justification dans le cas où elle aurait déjà été soumise par l'élève. Le système passe alors directement au noeud de conclusion (C_c), ce qui correspond, par exemple, à la situation «010» du tableau. Si, par contre, la justification est absente, nous affichons alors tous les messages contenus dans l'état C_j , relatifs à l'état de l'inférence, en respectant les délais déterminés en collaboration avec l'équipe de didactique. Lorsque tous les messages ont été affichés (transition $M=\{\}$), on passe aux messages particuliers au noeud de justification (état N_j).

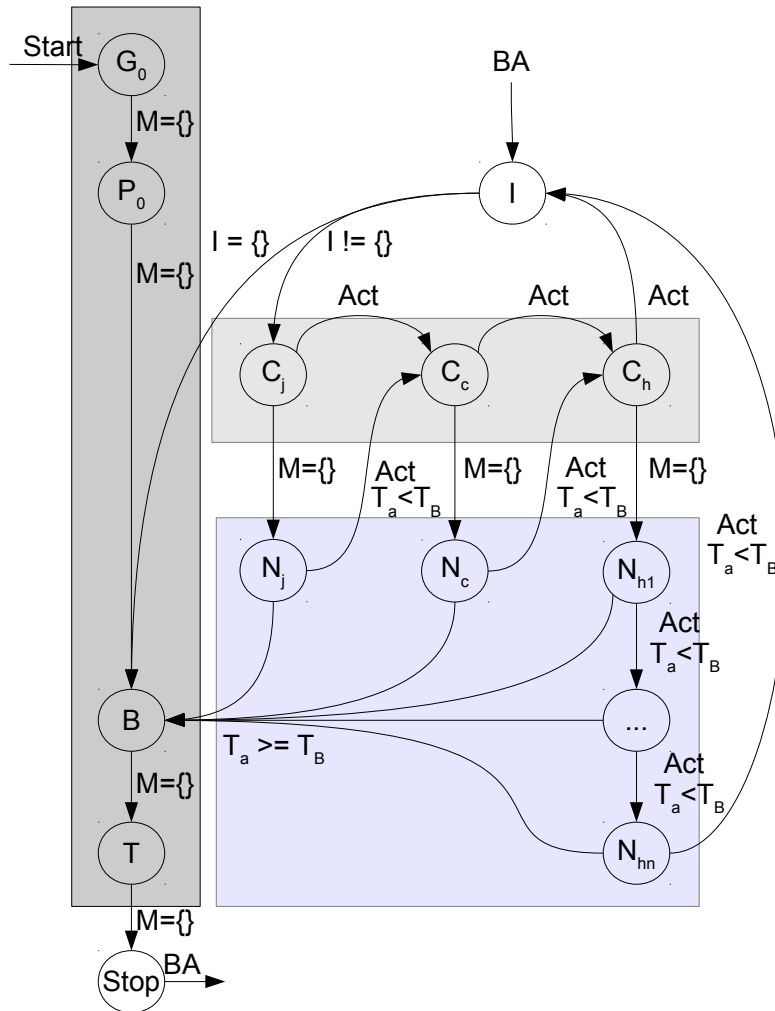


Figure 6.6 Schéma général de la machine à états finis générant les messages

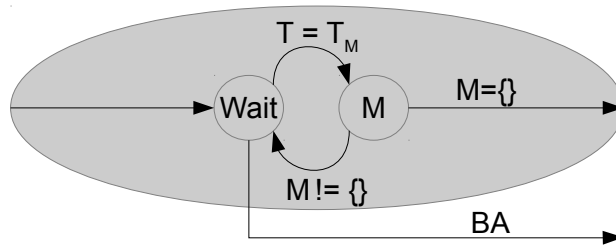
Dans l'état N_j , à la suite de l'affichage de tous les messages qui y sont contenus, deux transitions sont possibles. Celle qui est empruntée dépend de la comparaison entre le temps écoulé

depuis l'écriture, par l'élève, du dernier énoncé valide (T_a) et le temps d'attente minimum avant de fournir, explicitement, une partie de la solution (T_B). En ce qui concerne la séquence des messages indiquée dans le tableau 6.1, nous considérons que $T_a < T_B$, c'est-à-dire que l'enseignant propose, patiemment, différents indices à l'élève dans l'espoir de relancer son processus de résolution, tout en évitant de lui donner une partie de la preuve. Nous supposons donc que, après l'affichage de tous les messages de l'état N_j , nous passons à l'état C_c . Au sujet de la transition *Act*, elle est indiquée par souci d'uniformité, mais elle n'est réellement utile que pour les états N_{hn} . En effet, elle permet d'éviter d'afficher les messages associés à l'état N_j si l'énoncé a déjà été proposé par l'élève. Cependant, cette vérification est ici redondante, car elle a préalablement été réalisée par l'état C_j .

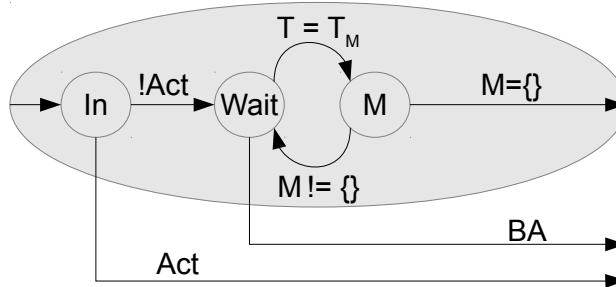
Le fonctionnement des états C_c et N_c est similaire à celui des états C_j et N_j respectivement, sauf que ceux-ci traitent les messages concernant la conclusion de l'inférence. On se retrouve, par la suite, dans l'état C_h , peu importe le chemin parcouru. Pour ce dernier, la transition *Act* a une signification particulière. En effet, elle sera empruntée uniquement si toutes les hypothèses relatives à l'inférence ont été soumises par l'élève. Par conséquent, dès qu'une hypothèse est manquante, nous affichons tous les messages de l'état C_h avant de passer à l'état N_{h1} qui contient l'ensemble des messages d'aide pour la première hypothèse à traiter. Il y a, en fait, autant d'états N_{hn} qu'il y a d'hypothèses dans l'inférence considérée. Chacun de ces états a un comportement analogue à l'état N_j ou N_c . Cependant, la transition *Act* est, dans ce cas, nécessaire, car elle permet d'éviter d'afficher les messages concernant les hypothèses déjà produites par l'élève étant donné que la chaîne des états N_{hn} est parcourue en entier dès qu'une hypothèse est manquante. Tous les états et toutes les transitions décrits jusqu'à présent nous permettent donc de reproduire la structure des messages associée aux différentes situations répertoriées dans le tableau 6.1 pour une inférence déterminée.

Avant de passer à la description des autres éléments de notre machine, nous présentons l'architecture interne des différents états, telle qu'illustrée à la figure 6.7. Nous allons nous concentrer sur l'état N (figure 6.7c), car il est le plus complexe et les autres états ne sont, en fait, que des simplifications de celui-ci. Son point d'entrée est l'état In qui ne sert qu'à éviter les traitements et les messages superflus lorsque l'énoncé rattaché au noeud a déjà été soumis par l'élève (*Act*). Dans le cas contraire (!*Act*), on entre dans la boucle d'affichage des messages. Dans le cadre de QEDX, pour éviter les abus, il n'était pas possible d'obtenir des messages d'aide à la demande, par exemple, en appuyant sur un bouton. Ils étaient, par conséquent, affichés en respectant des délais (T_M) que nous avons imposés. C'est pour cette raison que nous avons intégré un état *Wait* qui permet de mettre le système tutoriel en attente jusqu'à ce que l'échéance pour l'affichage du prochain message soit atteinte ($T = T_M$). L'autre transition qui nous permet de sortir de cet état, soit *BA* (Bonne Action), est activée de

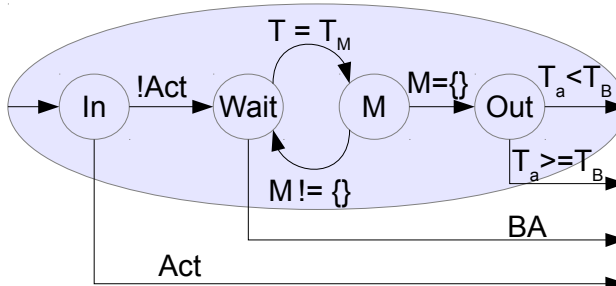
façon asynchrone seulement lorsque l'élève produit un énoncé valide. Dans ce cas, la stratégie du tuteur est mise à jour afin d'adapter immédiatement l'aide offerte. Pour tous les autres verdicts concernant l'énoncé proposé, le système tutoriel poursuit sa stratégie d'aide comme si l'élève n'avait réalisé aucune action. Pour sa part, l'état M permet d'afficher le prochain message de sa liste interne. Par la suite, s'il reste d'autres messages qui n'ont pas encore été affichés ($M \neq \{\}$), on retourne alors en attente. Dans le cas contraire ($M = \{\}$), on vérifie, dans l'état Out , s'il est temps de donner une partie de la réponse à l'élève ($T_a \geq T_B$) ou si l'on doit lui proposer une autre piste de solution ($T_a < T_B$). Cette dernière décision ne se produit qu'après avoir affiché tous les messages concernant un noeud particulier de l'inférence dans le but de respecter la hiérarchie usuelle des messages.



(a) Structure des états G, P, B, T



(b) Structure des états C



(c) Structure des états N

Figure 6.7 Structure interne de la machine illustrée à la figure 6.6

Dans la fenêtre de clavardage simulé de la figure 6.5, on retrouve un exemple d'une séquence de messages générée par les éléments de notre machine à états finis décrits précédemment. Dans cette fenêtre, le premier énoncé soumis par l'élève fait partie de la démonstration, donc le système l'accepte et produit un mot d'encouragement. Notre machine s'active alors et se prépare à offrir un indice à l'élève dans l'éventualité où il ne parviendrait pas à poursuivre son travail. Comme ce dernier n'a réalisé aucune action valide dans le délai prescrit, le système écrit un premier message. Celui-ci est de type C_j , comme on pouvait s'y attendre. Par la suite, notre système tutoriel enchaîne par deux messages de type N_j concernant le noeud de justification ciblé par C_j . En l'absence d'une action valide, notre machine passe alors à l'état C_c . Cependant, la conclusion de l'inférence considérée a déjà été inscrite par l'élève. Par conséquent, la machine emprunte la transition *Act* pour éviter d'afficher des messages inutiles qui ne causeraient que de la confusion. La machine se retrouve alors dans l'état C_h et propose, comme prévu, après un délai déterminé, un message de ce type. On voit que l'élève se hâte ensuite et toutes ses actions subséquentes se produisent avant l'affichage du prochain message issu de notre machine.

Jusqu'à maintenant, nous avons décrit l'organisation des états C et N qui permet de reproduire la structure des messages pour les dix situations, relatives à une inférence ciblée, contenues dans le tableau 6.1. Il existe ainsi un tableau distinct et, par analogie, une machine à états finis unique pour chaque inférence faisant partie du graphe des solutions. Il faut cependant être conscient que ce n'est pas la structure de la machine qui la rend unique, mais plutôt les informations contenues dans chacun de ses états, comme la liste des messages, l'état de l'inférence, etc. Afin de proposer des indices appropriés, il est nécessaire de déterminer l'inférence courante sur laquelle l'élève travaille afin de mettre en branle la machine concernée. L'algorithme d'ordonnancement de la couche EDOI de QEDX sert justement à placer les inférences en ordre de priorité, soit, notamment, selon la chronologie du travail de l'élève. On obtient alors une liste ordonnée d'inférences que nous traitons dans l'état I de notre machine. Le rôle de ce dernier est de choisir une inférence prioritaire n'ayant pas encore été traitée et de peupler les états C et N avec les informations pertinentes avant de passer à l'état C_j . Par conséquent, tant que la liste d'inférences n'a pas été épuisée ($I! = \{\}$), la machine produit des messages concernant l'inférence courante avant de passer à la suivante. Enfin, à la suite de l'écriture d'un énoncé valide par l'élève, la liste est mise à jour par la couche EDOI et l'ensemble des inférences traitées, remis à zéro, afin de respecter le nouvel état cognitif de l'élève.

En l'absence d'une action valide de l'élève, notre machine génère un enchaînement de messages, mais, éventuellement, deux situations peuvent se présenter. La première survient lorsque toutes les inférences contenues dans l'état I ont été utilisées pour produire des mes-

sages ($I = \{\}$). Dans ce cas, aucun autre message d'aide, de type C ou N , associé à une inférence, ne peut être engendré. D'autre part, il est possible de fixer un délai (T_B) au-delà duquel on considère que l'élève pourrait bénéficier d'un message lui indiquant une partie de la preuve. Ce critère a été adopté, car nos analyses ne nous ont pas permis d'identifier, de façon précise, la stratégie de l'enseignant concernant la suggestion explicite d'un élément de la réponse. Comme on peut le voir sur la figure 6.7, l'expiration du délai ($T_a \geq T_B$) n'est vérifiée qu'à la sortie des différents états N afin de préserver la structure hiérarchique des messages. Les transitions qui correspondent à ces deux situations nous conduisent à l'état B , soit celui qui offre une partie de la réponse à l'élève. Cet état produit habituellement un message unique dans lequel on indique textuellement un énoncé à inscrire à l'interface de QEDX. Si cette ultime tentative est infructueuse, il est probable que l'élève ne soit pas en mesure d'utiliser notre système adéquatement ou qu'il ait abandonné. Par conséquent, dans l'état T , nous lui proposons, en dernier ressort, de consulter son enseignant, car QEDX a épuisé toutes les ressources mises à sa disposition pour relancer son processus de résolution. La machine se retrouve enfin dans l'état *Stop* en attente d'une action valide de l'élève (BA).

Pour terminer cette description du fonctionnement de notre machine à états finis, il ne nous reste qu'à présenter les états G_0 et P_0 . Ceux-ci permettent de proposer soit des messages généraux (G_0), qui font habituellement référence à des stratégies de résolution de haut niveau, applicables à tous les problèmes, ou des messages particuliers au problème posé (P_0), qui incitent l'élève à réfléchir sur des caractéristiques propres au problème à résoudre. Dans le premier cas, nous pouvons, par exemple, lui suggérer de relire l'énoncé alors que dans le second, nous pouvons lui poser une question qui fait référence à un élément précis de la figure. Les deux états (G_0 et P_0) sont visités, une seule fois, en début de problème (*Start*). En guise d'illustration, les deuxième et troisième messages de la figure 6.5 ont été produits dans l'état G_0 . L'ajout de ces deux états permet de reproduire le comportement d'un enseignant qui s'assure de l'engagement de l'élève avant de l'aider, de façon plus ciblée, à trouver des énoncés utiles pour compléter sa démonstration. Leur organisation, à l'intérieur de la machine, permet aussi de respecter la hiérarchie usuelle des messages en commençant par les plus génériques.

Afin de permettre la gestion des quatre situations particulières qui ont été recensées lors de l'analyse des messages (voir section 6.3.4), mais qui sont exclues du tableau 6.1, des traitements spéciaux ont, dans certains cas, été nécessaires. Premièrement, les messages proposés lorsque l'élève n'a pas encore commencé son travail sont produits par les états G_0 et P_0 , et sont, par conséquent, pris en charge, de façon transparente, par notre machine. Par contre, si, en cours de résolution, nous devons fournir des indices concernant une inférence dont aucun énoncé n'a été inscrit par l'élève (cas 000), cela signifie généralement que ce dernier n'a pas encore de stratégie de résolution claire. Dans ce cas, l'enseignant typique l'aide à construire

le plan de sa solution en lui suggérant de faire une liste des justifications qui lui semblent utiles. Ce comportement peut être simulé par la production des messages associés exclusivement à l'état N_j pour l'inférence ciblée. Par conséquent, lorsqu'une inférence n'ayant pas été travaillée par l'élève (cas 000) est choisie dans l'état I , nous devons court-circuiter tous les états C et N , sauf N_j . Il est possible de gérer ce cas à l'aide de notre machine à états finis simplement en produisant des listes de messages vides pour les états que nous désirons court-circuiter. Enfin, les inférences déjà complétées (cas 111) ne nécessitent aucune intervention de la part du système tutoriel. En fait, celles-ci sont absentes de la liste enregistrée dans l'état I , car elles ont été retirées de cette dernière par l'algorithme d'ordonnancement des inférences de la couche EDOI de QEDX. La seule exception concerne l'achèvement de la dernière inférence nécessaire pour terminer une solution. Dans ce cas, on détecte la solution complète et l'on affiche un message instantané confirmant la réussite du problème. Par conséquent, aucun message associé à une inférence complète n'est véritablement nécessaire. Ceci conclut donc la description de l'architecture de la machine à états finis qui a permis de mettre en oeuvre, de façon relativement transparente, les quatorze situations répertoriées dans le cadre de notre travail.

Notre but, dans cette section, était de décrire l'architecture informatique du module de génération des messages discursifs (GMD) conçu à la suite de l'analyse des interactions entre les enseignants et les élèves en situation de résolution de problème. Nous avons divisé la section en deux parties principales, soit les messages instantanés et les messages d'aide discursifs. Les premiers apparaissent immédiatement, en réponse à chaque action de l'élève, afin d'indiquer la validité de celle-ci. Nous avons implanté, d'une part, des rétroactions sémaphoriques qui permettent de signaler la conformité, relativement à certaines règles, des éléments inscrits à l'intérieur des énoncés, ainsi que, d'autre part, des émoticônes qui reproduisent, en quelque sorte, l'expression faciale de l'enseignant en réaction à une réponse de l'élève. Ces dernières permettent d'attester le traitement, par le système, de l'énoncé soumis et d'informer l'élève quant à sa pertinence par rapport aux solutions acceptables. Ces deux rétroactions sont, selon la nomenclature de VanLehn (2006), des rétroactions dites minimales. Nous avons, de plus, implanté des rétroactions propres à l'erreur, sous forme de messages instantanés facultatifs qui accompagnent les émoticônes. Ceux-ci sont associés à des énoncés précis, représentant des erreurs typiques, identifiés par l'équipe de didactique. Nous avons, dans un deuxième temps, décrit notre machine à états finis qui permet de fournir des indices suggérant la prochaine étape du processus de résolution, soit le troisième type de rétroaction identifié par VanLehn. Celle-ci est dérivée des stratégies répertoriées dans le tableau 6.1, qui forment un sous-ensemble important des principales interventions de l'enseignant typique observé. Dans l'architecture proposée, chaque état permet d'afficher, de manière séquentielle, les messages

du type associé et ces derniers sont régénérés dynamiquement à la suite d'une action valide de l'élève. De plus, nous avons expliqué la stratégie adoptée pour gérer, de façon relativement transparente, les quatre situations particulières qui ne respectent pas la même structure que celles contenues dans le tableau 6.1. Dans la section suivante, nous exposons les tests réalisés ainsi que les résultats obtenus afin de confirmer la pertinence et la validité du GMD.

6.5 Évaluation du GMD

Les rétroactions instantanées et la machine à états finis décrites dans la section précédente permettent, théoriquement, de reproduire un sous-ensemble important des interventions des enseignants observés. Elles ont donc été implantées dans QEDX afin de proposer une aide pertinente à l'élève en processus de résolution de problème. Nous devons maintenant vérifier si notre mise en oeuvre permet véritablement d'aider ces derniers. Toutefois, l'implantation seule des structures n'est pas suffisante pour produire un tuteur intelligent, car des messages doivent aussi être composés pour les différentes situations. Ainsi, cette section est divisée en trois volets, soit une présentation des principales statistiques concernant les messages, une vérification préliminaire du système et une expérimentation permettant de valider le GMD. Plus précisément, nous proposons, tout d'abord, un tableau indiquant le nombre de messages qui ont été produits pour les différentes situations. Nous en profitons pour expliquer les mécanismes implantés dans QEDX qui permettent de minimiser le nombre de messages nécessaires tout en offrant une grande flexibilité pour traiter des cas particuliers. Dans un deuxième temps, les procédures de vérification du GMD qui ont été employées dans le cadre du projet sont brièvement décrites. En effet, la mise en oeuvre de celui-ci a été validée à l'aide de simulations et de vérifications par un expert en didactique pour s'assurer que les messages générés respectent la structure du tableau 6.1. Enfin, une expérimentation, réalisée en classe afin d'attester de la pertinence du GMD, est décrite. Les principaux résultats, essentiellement qualitatifs, sont résumés et leur signification brièvement expliquée.

6.5.1 Rédaction des messages d'aide

Dans le premier volet de cette section, nous nous intéressons au travail requis pour produire l'ensemble des messages de QEDX ainsi qu'aux mécanismes mis en oeuvre pour rendre cette tâche moins laborieuse. En effet, pour chaque état de notre machine à états finis (figure 6.6), nous devons fournir une liste de messages qui pourront ensuite être proposés à l'élève afin de l'aider à résoudre un problème donné. Le tableau 6.3 présente donc le nombre de messages composés pour chacun des états et pour chacun des cinq problèmes implantés dans la version actuelle de QEDX (annexe A).

Tableau 6.3 Données reliées aux messages créés pour les cinq problèmes implantés

PID	$\bar{A}$	J	It	G_0	P_0	C	N	B	T
RT	2,23	8 (13)	–	–	3	–	54 (27)	–	–
TT	1,87	13 (31)	–	–	3	–	108 (54)	–	–
PT	2,16	23 (76)	–	–	3	–	228 (114)	–	–
PA	2,15	31 (214)	2	–	3	–	246 (123)	–	–
TS	1,99	44 (116)	–	–	3	–	480 (240)	–	–
Général	–	–	40	4	–	29	–	1	1
Total	–	119 (450)	42	4	15	29	1116 (558)	1	1
Unique	–	86	42	4	15	29	852 (426)	1	1

Dans la colonne de gauche du tableau 6.3, nous retrouvons l'identifiant de cinq problèmes, ainsi qu'une catégorie pour les messages généraux. En effet, dans QEDX, il est possible de composer des messages génériques pour chacun des états, qui sont utilisés lorsqu'aucun message plus précis n'est fourni. Ceci permet de réduire le nombre de messages nécessaires tout en donnant la possibilité de les personnaliser pour des contextes particuliers. Par exemple, le message faisant appel à l'enseignant (T) n'a été codé qu'une seule fois pour l'ensemble des problèmes, alors que l'aide spécifique au problème (P_0) est distincte pour chacun. Ensuite, nous indiquons le nombre total de messages ainsi que le nombre de messages uniques pour chaque catégorie. Nous faisons cette distinction, car il arrive qu'un même message soit réutilisé à l'intérieur de différents problèmes. Cette situation est possible, car les messages ne sont pas associés à des énoncés précis, mais plutôt aux gabarits qui permettent de les engendrer. Par exemple, l'énoncé H_2 de l'inférence de la figure 6.3 a été produit à l'aide du gabarit : « ? est perpendiculaire à ? ». Par conséquent, des énoncés construits à l'aide de gabarits identiques font référence à un même ensemble de messages. Il est toutefois possible d'inclure, dans ces derniers, des liens vers les différents éléments du gabarit afin de les personnaliser.

Les deux colonnes suivantes fournissent des statistiques par rapport aux inférences acceptées pour la résolution de chacun des cinq problèmes. Nous indiquons, d'une part, le nombre d'antécédents moyen par inférence ($\bar{A}$). Étant donné qu'il n'y a, par définition, qu'une seule justification et un seul conséquent par inférence, nous n'avons pas inclus leur nombre moyen dans le tableau. D'autre part, la colonne J indique le nombre d'énoncés de justification uniques présents dans le graphe des solutions, mais aussi, entre parenthèses, le nombre total d'inférences admissibles pour chaque problème. Pour comprendre cette nuance, il faut savoir qu'une infinité d'inférences distinctes, utilisant une même justification, peuvent être générées à partir d'un ensemble particulier de gabarits d'énoncés. Comme certains énoncés de justification sont utilisés dans différents problèmes, nous avons aussi indiqué, sur la dernière ligne,

le nombre d'énoncés uniques, tous problèmes confondus. Ces statistiques nous permettent à la fois de juger de la complexité relative de chaque problème, en comparant le nombre d'inférences, mais aussi d'estimer le nombre de messages, associés aux états N , à produire. En effet, ce dernier est proportionnel au nombre d'énoncés de justification distincts multiplié par le nombre de noeuds moyen par inférence pour un problème donné.

Pour sa part, la colonne *It* contient le nombre de messages instantanés, ou les rétroactions immédiates, proposés dans la version actuelle de QEDX. Ceux-ci sont formés, en fait, des émoticônes qui peuvent être accompagnées d'un message, lorsque jugé opportun. Nous avons produit des rétroactions différentes pour chacun des huit verdicts (valide, invalide, répétition, etc.) potentiels concernant l'énoncé soumis par l'élève. Par conséquent, des messages adaptés sont offerts lorsque l'élève propose, par exemple, une réciproque ou qu'il répète un énoncé ne faisant pas partie des démonstrations. Ces messages peuvent généralement être utilisés pour tous les problèmes. Cependant, il est aussi permis d'en créer pour réagir à des erreurs particulières d'un élève dans un contexte donné et cette possibilité a été testée dans le cadre du problème *PA*. De plus, il est intéressant de noter que 15 des 40 messages sont associés à la production d'un énoncé valide. Ce nombre peut paraître élevé, mais, en fait, nous avons profité du potentiel qu'offre QEDX de choisir aléatoirement un message lorsqu'un groupe est associé à un contexte particulier afin de varier les réactions du tuteur.

En ce qui concerne la machine à états finis, on remarque d'abord que, comme on pouvait s'y attendre, les messages généraux (G_0) sont universels alors que ceux particuliers au problème (P_0) ont été produits indépendamment pour chaque problème. Dans le cas de l'état T , un seul message universel, invitant l'élève à consulter son professeur, est suffisant. Par contre, pour les états B et C , il est légitime de penser que des messages doivent être produits pour l'ensemble des situations qui peuvent survenir. Par exemple, il faudrait, en théorie, composer un message de type B distinct pour chaque état N possible. Afin d'éviter cette explosion du nombre de messages nécessaires, nous avons implanté une syntaxe qui accepte des paramètres prédéfinis qui font référence aux différents éléments de l'inférence traitée. C'est grâce à ce mécanisme qu'il a été possible de produire un seul message de type B général qui indique l'énoncé exact qui doit être inscrit par l'élève. Néanmoins, dans le cas des états C , 29 messages ont été requis pour s'assurer qu'ils sont écrits dans un français convenable.

La dernière colonne du tableau 6.3, qui n'a pas encore été décrite, concerne les états N . Dans celle-ci, nous avons inscrit, pour chaque problème, le nombre total de messages produits et, entre parenthèses, le nombre d'états N associés. Nous savons que le nombre total, théorique, d'états N à générer, pour un problème donné, est calculé en dénombrant les noeuds composant chaque inférence du graphe des solutions, prise individuellement, et en faisant ensuite la

somme de ces valeurs. Par exemple, dans la preuve proposée pour le problème du rectangle (section 6.3.1), nous obtenons un total théorique de 14 états N ($4 + 4 + 6$), si nous supposons que c'est l'unique solution acceptée. Cependant, comme nous l'avons mentionné plus haut, les messages sont rattachés à des gabarits d'énoncés et non à des énoncés précis, ce qui permet de réduire sensiblement le nombre d'états N à traiter. Dans notre exemple, un même gabarit est utilisé pour tous les antécédents de la dernière inférence, donc un seul état N permet de tous les traiter, ce qui réduit de 6 à 3 le nombre d'états relatifs à celle-ci. Par conséquent, le nombre d'états N inscrit dans le tableau 6.3 correspond, en fait, au total en excluant les répétitions à l'intérieur de chaque problème. Le nombre de messages de type N a été indiqué séparément pour chaque problème afin d'apprécier le travail nécessaire pour implanter des problèmes de complexité diverse de façon indépendante. En vérité, cependant, tous ces messages peuvent être considérés comme étant génériques, car ils font habituellement référence aux gabarits des énoncés et non à des situations particulières concernant le problème posé. Il est donc possible de recycler les messages dans le cas d'inférences construites en utilisant un même ensemble de gabarits, sans égard au problème. La ligne *Total* correspond donc au nombre total de messages qu'il aurait été nécessaire de composer si tous les problèmes avaient été implantés en vase clos, alors que la ligne *Unique* indique le nombre d'états et de messages distincts réellement traités afin de fournir une aide pour les cinq problèmes proposés.

Près de vingt heures ont été nécessaires, à l'équipe de didactique, pour implanter environ neuf cents messages dans QEDX. La vaste majorité de ceux-ci sont relatifs aux états de type N et, pour ces derniers, deux messages ont systématiquement été composés. La quantité de travail peut sembler considérable pour mettre en oeuvre cinq problèmes, mais il faut se rappeler que notre banque de messages était initialement vierge. Toutefois, lors de l'ajout de nouveaux problèmes, seuls les messages concernant les inférences qui n'ont pas déjà été traitées dans les problèmes précédents devront être produits, car les messages de type N peuvent être réutilisés. Par contre, si nous ne sommes pas en mesure de recycler les messages déjà composés, par exemple dans le cadre d'une traduction, le tableau 6.3 nous indique qu'il y a un facteur variant de 3,5 à 5,5 entre le nombre de justifications uniques et le nombre d'états N à traiter. Le facteur précis dépend du nombre de gabarits distincts contenu à l'intérieur de chaque inférence. On peut cependant affirmer que le facteur minimal théorique est de trois, car chaque inférence contient, au minimum, un antécédent, une justification et un conséquent, ce qui produit au moins trois états N uniques. Ces données nous permettent donc de dresser un portrait assez précis des ressources qui ont été nécessaires pour rendre fonctionnel notre système tutoriel.

6.5.2 Résultats des simulations du GMD

Après avoir implanté la machine à états finis ainsi que les messages associés, nous avons réalisé des simulations afin de confirmer que la séquence des messages générés par QEDX correspond bien à celle prévue à la section 6.3.4. La figure 6.8 contient les messages produits pour un échantillon de trois situations simulées dans le cadre de nos tests dans le but d'illustrer la stratégie d'aide de notre système. Dans cette figure, nous avons indiqué, entre crochets, le type de chacun des messages proposé par *ProfTuring*. La première situation présentée (figure 6.8a) se produit lorsque l'élève choisit un problème, mais n'inscrit, par la suite, aucun énoncé. Dans ce cas, on remarque l'absence de message de type *B*, bien qu'il soit présent dans la machine, car nous avons décidé de ne pas en proposer dans ce contexte particulier. Pour les deux autres situations (figure 6.8b et 6.8c), l'élève inscrit respectivement un ou deux énoncés avant d'attendre les suggestions du tuteur. QEDX génère alors un message instantané (*It*) en réaction à chaque énoncé soumis. Les messages subséquents sont produits à raison d'un à la minute. Ceux au-dessus de la ligne de démarcation correspondent à deux cas (+00 et +10) traités dans le tableau 6.1. Par la suite, le système choisit une nouvelle inférence et génère des messages relatifs à celle-ci jusqu'à ce que le délai maximal de dix minutes soit atteint. Il termine alors la séquence de messages prévue pour le noeud courant et propose enfin les messages de dernier recours (*B* et *T*). La figure 6.8 donne donc un aperçu de la séquence des messages offerts par QEDX ainsi que des différentes pistes de résolution abordées. Un travail d'annotation similaire à celui proposé dans cette figure a été réalisé afin d'étudier les messages générés pour les quatorze situations répertoriées ainsi que pour les messages instantanés et a permis de confirmer que le GMD met en oeuvre les comportements attendus.

Étant donné le bilan positif obtenu à la suite des tests unitaires réalisés, nous avons décidé de faire vérifier notre système de manière rigoureuse par un expert indépendant. Son mandat consistait à produire des preuves, à la manière d'un élève typique, pour les quatre problèmes opérationnels à l'époque (*RT*, *TT*, *PT*, *PA*), et à noter tout message du tuteur qui ne respectait pas la structure issue de nos analyses. Le tableau 6.4 présente des statistiques concernant les différents tests réalisés. Pour chaque problème, l'expert a fait plusieurs tentatives de résolution (*Tentatives*) et, dans 93% des cas, il a été en mesure de compléter une démonstration (*Réussites*). Ce sont des erreurs dans les solutions implantées qui l'ont forcé à abandonner la résolution à trois occasions. Afin de produire les démonstrations, plus de 900 énoncés ont été écrits (*Élève*), ce qui a généré autant de messages instantanés (*MI*) ainsi qu'en un peu plus de 1400 indices relatifs à la prochaine étape (*PE*). L'analyse des 2344 messages générés par le système a permis de confirmer qu'ils respectent tous scrupuleusement la structure prescrite à l'intérieur d'une inférence, malgré quelques coquilles syntaxiques. Par contre, l'expert a

[G₀]Prof. Turing > Salut ! À toi de jouer...
 [G₀]Prof. Turing > As-tu bien lu l'énoncé ? Relis-le au besoin et étudie bien la figure fournie.
 [G₀]Prof. Turing > As-tu bien ciblé ce que tu cherches à démontrer ?
 [G₀]Prof. Turing > Quels éléments de l'énoncé peuvent t'être utiles pour amorcer ta démonstration ?
 [P₀]Prof. Turing > On te demande de démontrer qu'un quadrilatère avec 3 angles droits est nécessairement un rectangle. Sais-tu comment le prouver ?
 [P₀]Prof. Turing > Quelles justifications (propriétés ou définitions) pourraient t'être utiles pour démontrer qu'il s'agit sans doute d'un rectangle ? Recherche des justifications par rapport aux angles du rectangle.
 [P₀]Prof. Turing > Tu cherches à démontrer que le quadrilatère ABCD est nécessairement un rectangle. Essaie d'écrire le résultat (conclusion), une hypothèse ou une justification.
 [T]Prof. Turing > Consulte ton professeur.

(a) Situation 000 (début)

Élève > S :BC est perpendiculaire à S :CD
 [It]Prof. Turing > :) Bon départ !
 [C_j]Prof. Turing > Pour quelle justification le fait : S :BC est perpendiculaire à S :CD, peut-il s'avérer utile ?
 [N_j]Prof. Turing > Recherche des justifications concernant la perpendicularité et le parallélisme de droites ou de segments entre eux.
 [N_j]Prof. Turing > La justification recherchée fait aussi allusion à deux droites ou segments parallèles tous deux perpendiculaires à une troisième droite (ou segment).
 [C_c]Prof. Turing > L'antécédent : S :BC est perpendiculaire à S :CD, peut t'aider à démontrer quel résultat ?
 [N_c]Prof. Turing > Que peux-tu déduire sur la position relative des segments S :AB et S :BC ?
 [N_c]Prof. Turing > As-tu soumis le résultat concernant la perpendicularité des segments S :AB et S :BC ?
 [C_h]Prof. Turing > Quels faits, accompagnant S :BC est perpendiculaire à S :CD, pourraient t'aider à faire avancer ta démonstration.
 [N_{h2}]Prof. Turing > Que peux-tu dire concernant la position relative des segments S :AB et S :CD ?
 [N_{h2}]Prof. Turing > As-tu soumis une affirmation portant sur le parallélisme des segments S :AB et S :CD ?
 [C_j]Prof. Turing > Pour quelle justification le fait : S :BC est perpendiculaire à S :CD, peut-il s'avérer utile ?
 [N_j]Prof. Turing > Recherche des justifications concernant le rectangle.
 [N_j]Prof. Turing > La justification recherchée fait aussi allusion aux quatre angles droits du rectangle.
 [B]Prof. Turing > Essaie d'écrire : Un rectangle est un quadrilatère ayant quatre angles droits
 [T]Prof. Turing > Consulte ton professeur.

(b) Situation +00

Élève > S :BC est perpendiculaire à S :CD
 [It]Prof. Turing > :) Bon départ !
Élève > Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre
 [It]Prof. Turing > :) Continue !
 [C_c]Prof. Turing > Quel résultat peut découler de l'utilisation de la justification suivante : Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre, en prenant en compte l'antécédent : S :BC est perpendiculaire à S :CD ?
 [N_c]Prof. Turing > Que peux-tu déduire sur la position relative des segments S :AB et S :BC ?
 [N_c]Prof. Turing > As-tu soumis le résultat concernant la perpendicularité des segments S :AB et S :BC ?
 [C_h]Prof. Turing > Quels faits en plus de S :BC est perpendiculaire à S :CD, sont nécessaires afin d'employer la justification Si deux droites sont parallèles, toute perpendiculaire à l'une est perpendiculaire à l'autre ?
 [N_{h2}]Prof. Turing > Que peux-tu dire concernant la position relative des segments S :AB et S :CD ?
 [N_{h2}]Prof. Turing > As-tu soumis une affirmation portant sur le parallélisme des segments S :AB et S :CD ?
 [C_j]Prof. Turing > Pour quelle justification le fait : S :BC est perpendiculaire à S :CD, peut-il s'avérer utile ?
 [N_j]Prof. Turing > Recherche des justifications concernant le rectangle.
 [N_j]Prof. Turing > La justification recherchée fait aussi allusion aux quatre angles droits du rectangle.
 [C_c]Prof. Turing > L'antécédent : S :BC est perpendiculaire à S :CD, peut t'aider à démontrer quel résultat ?
 [N_c]Prof. Turing > Quel résultat concernant la nature du quadrilatère Q :ABCD peut être déduit ?
 [N_c]Prof. Turing > Tu peux déduire que le quadrilatère Q :ABCD est un rectangle.
 [B]Prof. Turing > Essaie d'écrire : Q :ABCD est un rectangle
 [T]Prof. Turing > Consulte ton professeur.

(c) Situation +10

Figure 6.8 Exemples de messages générés pour le problème du rectangle

déecté des lacunes évidentes autant en ce qui concerne l'ordre de traitement des inférences que pour la détermination de la solution la plus avancée. Celles-ci concernent toutefois la couche EDOI de QEDX et ont été analysées en détail dans le chapitre 5. Par conséquent, les conclusions de l'expert nous permettent d'affirmer que la couche GMD est en mesure de reproduire une structure de messages inspirée du travail de l'enseignant.

6.5.3 Expérimentation en classe

À la suite de la vérification par l'expert, les solutions ainsi que les coquilles syntaxiques ont été corrigées. Cependant, pour améliorer la performance de la couche EDOI, des informations supplémentaires, dont nous ne disposons pas actuellement, sont nécessaires. Malgré les lacunes, nous considérons que QEDX est assez robuste pour être utilisé par des élèves dans un contexte d'enseignement réel. Nous avons donc réalisé une expérimentation dont l'objectif principal était de valider les messages discursifs proposés par notre système tutoriel. Il s'agis-

Tableau 6.4 Statistiques relatives aux tests réalisés par l'expert

PID	Tentatives	Réussites	Élève	MI	PE	Total tuteur
RT	13	13	154	154	239	393
TT	10	9	278	278	569	847
PT	11	11	179	179	277	456
PA	8	6	327	327	321	648
Total	42	39	938	938	1406	2344

sait, en fait, de vérifier si la structure des messages construite à partir d'un sous-ensemble des interventions des enseignants observés lors de l'expérimentation précédente (section 6.3) permettait réellement d'aider l'élève en situation de résolution de problème. Par conséquent, cette expérimentation ne visait pas à tester l'effet de l'utilisation de notre système sur les résultats scolaires, mais bien à valider une étape de sa mise en oeuvre. Nous appuyons ainsi nos conclusions sur des analyses qualitatives plutôt que sur une évaluation à l'aide des prétests et post-tests traditionnels. Ce type d'analyse est avantageux dans le cadre de l'élaboration de QEDX, car il permet, entre autres, d'obtenir des informations qui seront réinvesties lors des étapes subséquentes de sa conception.

Pour réaliser cette expérimentation, deux enseignants de l'Académie Ste-Thérèse nous ont donné accès à un total de trois classes ordinaires de quatrième secondaire (15-16 ans). Environ 30 équipes de deux élèves ont utilisé QEDX, dont l'interface est illustrée à la figure 6.5, durant deux périodes de 60 minutes. Ils devaient tenter de résoudre les quatre problèmes implantés, à l'époque, dans notre système tout en se faisant aider par l'agent-tuteur que nous avons conçu. Si les messages proposés par QEDX ne permettaient pas de relancer le processus de résolution de l'élève, celui-ci était invité à faire appel à son enseignant. Afin de recueillir des données, nous avons d'abord utilisé le logiciel ScreenFlow dans le but d'enregistrer, simultanément, la caméra, les actions à l'écran et le son du microphone interne de l'ordinateur. Nous avons capturé, de cette façon, le travail de quatre équipes par classe. Des fichiers journaux, contenant toutes les phrases inscrites par les élèves ainsi que les rétroactions de QEDX, ont aussi été produits pour toutes les équipes. Ces fichiers ne contiennent toutefois qu'une partie des interactions, car ils n'incluent pas, par exemple, les changements d'onglets ou les actions sur la figure dynamique. Les enseignants portaient, de plus, un dictaphone qui permettait d'enregistrer toutes leurs interventions verbales. Nous avons enfin recueilli les commentaires des enseignants et des élèves en plus de noter nos propres observations.

Selon les données et les commentaires recueillis, nous pouvons affirmer que QEDX a généralement reçu un accueil favorable, autant par les élèves que par leurs enseignants, et que les améliorations apportées à la suite de la première expérimentation ont été bénéfiques. Par

exemple, la navigation par l'intermédiaire des onglets et l'ajout de l'onglet *Rédaction* ont aidé plusieurs élèves à compléter leurs preuves. En ce qui concerne la couche GMD, nous notons premièrement que les émoticônes, ainsi que les messages instantanés associés, ont une influence positive sur les élèves, car elles les encouragent tout en leur évitant du travail inutile. Ceux-ci découvrent rapidement qu'une émoticône souriante signifie que leur démonstration progresse. Ils se permettent alors d'entrer différents énoncés afin de vérifier si leurs idées sont utiles à la résolution du problème posé. Étant donné que QEDX ne pénalise pas l'écriture d'énoncés erronés, nous avons remarqué que les élèves sont plus enclins à explorer diverses pistes potentielles de solution. De plus, il aurait été intéressant de vérifier l'impact des messages qui sont relatifs à des erreurs précises, mais nous n'avons pas eu le temps d'en composer avant l'expérimentation. Par contre, les messages plus génériques concernant les propriétés réciproques ou la répétition des énoncés ont été utiles pour plusieurs élèves. Notre modélisation des expressions faciales et des réactions spontanées de l'enseignant encourage donc l'élève à explorer librement les différentes solutions au problème posé.

Les opinions concernant les messages d'aide à la prochaine étape sont toutefois partagées. D'une part, le délai imposé entre chaque message a causé des désagréments à plusieurs élèves. En effet, ceux qui sont doués le trouvent trop court, car ils ont l'impression de se faire bombarder de messages alors qu'ils réfléchissent à leur stratégie et explorent la figure. Par contre, les élèves plus faibles, qui sont en attente d'un message qu'ils seront en mesure d'exploiter pour poursuivre leur résolution, le trouvent trop long. En fait, aucun élève ne nous a indiqué que le temps d'attente était adéquat. De plus, nous avons remarqué que certains élèves isolent un concept contenu dans un des premiers messages du tuteur afin de trouver une phrase valide à inscrire et ignorent ainsi la suite de la conversation. Ils oublient même de consulter le message leur fournissant un élément de la réponse. Ces observations nous indiquent donc que l'imposition d'un temps d'attente entre chaque message limite, dans plusieurs cas, l'efficacité de notre tuteur. D'autre part, le contenu des messages a aussi été critiqué. Cependant, l'appréciation de ceux-ci dépend habituellement du niveau de compétence de l'élève. En effet, ce sont en majorité des élèves faibles qui trouvent les messages difficiles à comprendre, alors que les autres considèrent que ces mêmes messages sont généralement utiles. Ces résultats n'invalident pas le travail réalisé, mais nous indiquent plutôt que le système tutoriel devra être en mesure de s'adapter à la compétence de l'élève.

Nous avons toutefois été témoins de nombreuses situations dans lesquelles QEDX permettait de relancer le processus de résolution d'un élève bloqué. Celles-ci se produisaient généralement lorsque deux conditions étaient respectées. Il faut premièrement que le système détecte correctement la stratégie de résolution de l'élève. En effet, lorsque la stratégie, déterminée par la couche EDOI, est impertinente, les messages proposés sont incohérents par rapport à

l'état cognitif de l'élève et ils ont tendance à créer de la confusion. En second lieu, il faut que l'élève prenne le temps de lire et de comprendre les indices du tuteur. Dans les cas où ces conditions sont réunies, les actions des élèves observés nous indiquent que la séquence des messages proposés par QEDX les guide adéquatement. En effet, les élèves raffinent graduellement leurs critères de recherche des énoncés, en se référant aux messages du tuteur, pour éventuellement trouver une phrase qui fait progresser leur preuve. La structure des messages implantée permet, comme prévu, d'offrir des informations de plus en plus précises sans pour autant donner les réponses de manière prématurée. Les résultats obtenus lors de l'expérimentation tendent donc à confirmer que les rétroactions générées par la couche GMD ont un impact positif sur le travail de l'élève et permettent d'aider ce dernier à compléter les preuves demandées. Néanmoins, des correctifs devront, entre autres, être apportés afin d'ajuster les échéances entre les messages et d'adapter l'aide selon le niveau de compétence de l'élève.

L'objectif de cette section était de présenter les résultats des tests réalisés afin de valider la pertinence des messages générés par le GMD de QEDX. Toutefois, avant d'être en mesure d'effectuer les différents tests, il était essentiel de composer les messages du tuteur. Il a fallu près de vingt heures, à l'équipe de didactique, pour produire un peu plus de neuf cents messages appropriés pour résoudre les cinq problèmes implantés. Des mécanismes, mis en oeuvre dans QEDX, permettent néanmoins d'alléger cette tâche en offrant, par exemple, la possibilité de définir des messages génériques ou de réutiliser ceux qui sont associés à des états similaires. En fait, lors de l'ajout d'un nouveau problème, seuls les messages relatifs aux inférences uniques à celui-ci doivent être ajoutés, car tous les autres sont recyclés et peuvent être ajustés. Une fois ce travail accompli, nous avons produit des simulations afin de vérifier si les messages proposés par notre machine à états finis correspondaient à ceux qui sont attendus. Nous avons donc simulé les quatorze situations répertoriées, ce qui nous a permis de confirmer que notre système est conforme aux spécifications. Dans un deuxième temps, nous avons fait vérifier notre système par un expert indépendant. Celui-ci a analysé plus de deux mille messages du tuteur et a conclu que les messages instantanés ainsi que les messages associés à une inférence sont conformes à la structure attendue. Des problèmes concernant l'EDOI ont, par contre, été trouvés. Enfin, une expérimentation a été réalisée en classe et QEDX a reçu, généralement, un accueil positif. Nos observations nous indiquent que, dans de nombreuses situations, les messages du tuteur permettent de relancer le processus de résolution d'un élève bloqué. Néanmoins, l'appréciation du contenu et de la vitesse d'affichage des messages semble dépendre de la compétence de l'élève. Ces résultats démontrent donc que les interventions des enseignants modélisés par le GMD permettent déjà d'aider l'élève, malgré les problèmes qui se sont manifestés lors de l'expérimentation. Une analyse détaillée des forces, mais aussi des lacunes de notre système est proposée dans la section suivante.

6.6 Discussion

Les résultats, majoritairement positifs, décrits dans la section précédente, tendent à démontrer que notre système est en mesure de proposer une aide pertinente à partir d'un sous-ensemble important des interventions typiques des enseignants observés. Cependant, il est important de se rappeler que QEDX est en constante évolution et qu'il comporte encore plusieurs limites. Par conséquent, l'objectif de cette section est de proposer une analyse critique des principales forces, mais aussi des faiblesses du système actuel. D'une part, concernant les aspects positifs, nous montrons, entre autres, comment la mise en oeuvre de la machine à états et de la gestion des messages permet d'offrir un système robuste, tout en étant flexible. De plus, nous expliquons de quelle façon notre système se distingue des autres tout en proposant les différentes catégories de message décrites par VanLehn (2006). D'autre part, nous discutons de l'absence d'adaptation dans le discours du tuteur, qui semble rendre les interventions de ce dernier moins efficaces dans le cas d'élèves plus faibles. Nous mettons aussi en question notre politique de production des messages à intervalles fixes. Nous terminons la section en reprenant la figure 6.1, concernant l'organisation des messages, et nous en proposons une adaptation qui reflète la structure qui est implantée dans QEDX.

Premièrement, l'approche multidisciplinaire adoptée a permis de concevoir une couche GMD qui propose des messages qui aident les élèves en situation de résolution de problème. Ce constat démontre qu'il est possible de construire un système tutoriel efficace en appliquant des théories didactiques reconnues. Toutefois, il est illusoire de penser qu'un tel résultat a été obtenu sans effort et sans une véritable collaboration entre les experts en didactique et en informatique. Par exemple, nous savons que les interventions des enseignants sont extrêmement complexes. Nous avons donc choisi de nous concentrer sur un sous-ensemble de messages déterminés qui sont tout de même pertinents. Dans ce contexte, l'équipe de didactique avait pour mandat de s'assurer que la structure des messages extraite de nos analyses respectait les principes didactiques. Néanmoins, les experts en informatique pouvaient suggérer des modifications afin de rendre l'implantation de la structure possible. Ce dialogue constant nous assure donc que l'organisation des messages modélisée est conforme aux théories didactiques. Par conséquent, cette approche se démarque de celle plus traditionnelle qui consiste à se baser sur une simple analyse à priori de la démarche de résolution d'un expert afin d'en extraire les principales stratégies. C'est cette dernière approche qui a permis de concevoir, entre autres, le Geometry Tutor (Anderson et al., 1985; Ritter et al., 2010), ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991) ou Andes (Conati et al., 2002; VanLehn et al., 2005, 2010). Malgré leurs succès probants, évalués par la méthodologie des prétests et post-tests, il arrive que l'on s'aperçoive, à la suite d'une analyse à posteriori, que les stratégies du système ne soient pas en

accord avec celles d'un enseignant typique, comme il a été observé dans le cas de la première version d'Andes (VanLehn et al., 2005). Par contre, l'approche que nous avons utilisée nous permet, très probablement, d'éviter de telles surprises. Nous nous exposons, cependant, au risque que la structure des rétroactions déterminée ne puisse pas être entièrement implantée, étant donné la complexité apparente des interventions. Heureusement, nous avons été en mesure d'extraire une certaine régularité, ce qui nous a permis de construire, par exemple, une machine à états finis qui reproduit l'ensemble des situations répertoriées. Par conséquent, l'approche collaborative préconisée nous démarque des systèmes plus traditionnels et nous permet de certifier que la couche GMD est didactiquement valide.

La validité didactique est un pas dans la bonne direction, mais elle ne nous assure pas que notre système propose, à un élève en difficulté, des rétroactions variées et pertinentes. En effet, à cause de la complexité des interventions de l'enseignant, seule une partie des messages a, pour l'instant, été modélisée. Afin de vérifier si les rétroactions offertes par QEDX sont adéquates, nous les avons comparées à celles d'autres systèmes en utilisant la nomenclature conseillée par VanLehn (2006). Dans son article, celui-ci traite, en premier lieu, des rétroactions immédiates associées aux différentes actions de l'utilisateur. Dans les systèmes traditionnels, comme Andes (Conati et al., 2002; VanLehn et al., 2005, 2010) ou Geometry Tutor (Anderson et al., 1985; Ritter et al., 2010), on utilise un sémaphore chromatique (rouge ou vert) pour indiquer si l'élément de réponse est valide. Cette forme de rétroaction est très efficace, car elle évite à l'élève de se lancer dans un chemin sans issues. Dans notre cas, nous avons aussi adopté le sémaphore pour aider l'élève à compléter les énoncés. De plus, nous avons ajouté les émoticônes afin d'imiter les réactions faciales de l'enseignant face aux actions des élèves. Nous sommes, selon nos recherches, les pionniers dans l'utilisation des émoticônes à l'intérieur d'un système tutoriel en géométrie. Le second type de rétroaction, cité par VanLehn, concerne celles propres aux erreurs répertoriées. Pour celles-ci, des systèmes comme Andes et ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991) offrent la possibilité, en appuyant sur un bouton, d'obtenir plusieurs messages hiérarchiques d'explication qui permettent à l'élève de comprendre et de corriger ses erreurs. Dans notre cas, par contre, un seul message, relatif aux erreurs modélisées, est affiché automatiquement, ce qui est semblable, par exemple, au comportement de Geometry Tutor. Cependant, même si QEDX permet ce type de rétroaction, aucune erreur n'a actuellement été cataloguée. VanLehn classe enfin les indices concernant la prochaine étape de résolution comme le troisième type de rétroaction. L'approche classique utilisée, par exemple, dans Geometry Tutor et même dans AutoTutor (Graesser et al., 2004; Olney et al., 2010; Nye et al., 2014), consiste à proposer une série de messages, allant du plus générique au plus précis, selon un nombre de niveaux déterminé. Cette approche force, en quelque sorte, l'élève à travailler en utilisant une stratégie optimale

préétablie. Dans le cas de notre système, le nombre de niveaux est variable, car nous proposons de multiples pistes de solution avant de fournir une partie de la réponse, ce qui permet à l'élève d'explorer le problème et de travailler sur une solution qui n'est pas forcément optimale. Cette approche nous démarque donc d'autres systèmes tutoriels recensés. Les trois catégories de rétroactions décrites par VanLehn sont, par conséquent, présentes dans QEDX, même si la modélisation des erreurs reste à faire. Elles sont aussi analogues à celles offertes par des systèmes tutoriels populaires. Nous avons, de plus, innové en intégrant des émoticônes et des indices proposant différentes stratégies de résolution. Cette comparaison nous permet d'affirmer que l'aide offerte par QEDX est diversifiée et pertinente.

Même si la conception de QEDX est basée sur des théories didactiques reconnues et que les rétroactions proposées se comparent avantageusement à celles offertes par d'autres systèmes tutoriels, il n'en demeure pas moins que certains élèves ont eu une réaction mitigée face au système. Par exemple, les élèves plus faibles ont du mal à utiliser efficacement le répertoire d'énoncés et trouvent généralement que les messages du tuteur sont difficiles à comprendre. Cette dernière remarque était cependant prévisible, sachant que la structure des messages de QEDX est basée sur des interactions entre un enseignant typique et des élèves moyens ou doués. Il en résulte donc des messages qui sont, en général, efficaces pour aider un élève moyen, mais qui ne contiennent pas assez d'informations pour un élève faible. Afin de relancer le processus de résolution de ces derniers, des indices plus détaillés, fournissant parfois une partie de la réponse, sont souvent nécessaires. Parmi les systèmes recensés, on note l'Advanced Geometry Tutor (Matsuda et VanLehn, 2005; Matsuda, 2004) qui s'est inspiré de ce principe afin d'offrir une aide adaptative. En effet, il produit des messages d'aide hiérarchiques, dont le niveau initial est ajusté en fonction de la compétence de l'élève, ce qui permet d'éviter de proposer des messages d'une efficacité moindre. Malheureusement, la version actuelle de notre système n'intègre pas un tel mécanisme, ce qui limite sa performance auprès des élèves ayant certaines difficultés. De plus, lors de nos expérimentations, nous avons observé que les enseignants, selon leur personnalité, utilisaient des stratégies d'aide distinctes. Par exemple, certains adoptent une approche proactive, c'est-à-dire qu'ils posent des questions, parfois surprenantes, tout en évitant de donner les réponses. À l'opposé, d'autres enseignants fournissent graduellement des éléments de réponse jusqu'à ce que l'élève soit en mesure de compléter sa preuve. Il est légitime de penser qu'une approche proactive est mieux adaptée aux élèves doués, car elle leur propose des défis, alors que l'autre approche est rassurante pour les élèves inquiets. Cependant, pour la version actuelle de QEDX, il a été nécessaire de faire un compromis, donc nous avons adopté une approche à mi-chemin entre ces deux extrêmes. Notre tuteur propose, par conséquent, une personnalité générique, ce qui contribue à limiter son efficacité. Pour tenter de remédier à ces problèmes, il serait intéressant de proposer à

la fois un tuteur dont la personnalité peut être adaptée ainsi qu'un réglage dynamique du niveau des messages générés en fonction de la compétence, ou du profil, de l'élève.

Du côté technologique, l'utilisation d'une machine à états finis afin de modéliser l'organisation uniforme des messages, s'est avérée une décision judicieuse. D'une part, cette structure de données permet de proposer une représentation simple du fonctionnement du tuteur qui peut être interprétée par l'ensemble des membres de notre équipe multidisciplinaire. Il a donc été possible, pour l'équipe de didactique, de proposer des correctifs afin que le système réagisse en conformité avec les interventions des enseignants observés. De plus, comme le comportement du tuteur est facilement prévisible, la composition des messages relatifs aux différents états fut relativement aisée même si plusieurs heures de travail ont été nécessaires. D'autre part, notre mise en oeuvre de la machine offre une grande flexibilité pour les créateurs de contenus, car nous avons adopté une approche basée sur les données. En effet, tous les messages se retrouvent à l'intérieur de fichiers textes, ce qui rend la modification, et même la traduction, de ceux-ci possible sans avoir à modifier le code source de QEDX. Il est aussi permis d'ajuster l'échéance pour l'affichage des messages successifs et même d'associer, à ces derniers, des contextes d'utilisation précis. Par exemple, il a été possible de bloquer la production de messages de type *B* lorsque l'élève n'a pas encore commencé sa preuve, et ce, sans l'intervention des programmeurs. Une autre décision technique qui fut favorable concerne la gestion des messages. Premièrement, nous donnons la possibilité de composer des messages dont certains éléments sont générés dynamiquement, ce qui a permis de réduire considérablement la quantité de messages à produire. De plus, nous déterminons dynamiquement la liste des messages rattachée à chaque état de la machine selon un ordre de priorité, de la plus générique à la plus précise. La priorité est déterminée en fonction de la précision du contexte associé à la liste. Lors de la création de nouveaux contenus, les messages génériques sont utilisés, mais peuvent être facilement remplacés par des messages particuliers, au besoin, ce qui réduit sensiblement le temps nécessaire pour la rédaction des messages du tuteur. Par conséquent, la simplicité de la machine à états finis ainsi que la flexibilité offerte dans la gestion des messages ont contribué à mettre en oeuvre un GMD qui peut être aisément manipulé par une équipe multidisciplinaire afin de reproduire diverses interventions des enseignants.

Même si la gestion des messages a été implantée de façon parcimonieuse et qu'elle est conforme à la structure définie, il semble que la décision d'afficher les indices en appliquant des délais prédéterminés est discutable. Nous avons fait ce choix dans le but d'éviter que les élèves n'obtiennent des réponses simplement en appuyant à répétition sur un bouton permettant de produire les messages du tuteur. Seulement, cette stratégie est efficace uniquement si les messages apparaissent lorsque l'élève en a vraiment besoin, donc le choix des échéances est crucial. Dans le cadre du projet, nous n'avons pas fait d'analyse concernant les temps d'attentes op-

timaux, alors nous les avons fixés à une minute entre chaque message, ce qui nous semblait raisonnable. Cependant, lors de l'expérimentation en classe, tous les commentaires des élèves concernant l'affichage des messages mentionnaient que la vitesse était soit trop rapide ou trop lente. Par conséquent, si nous désirons maintenir cette politique, des études approfondies seront nécessaires. Par contre, d'autres stratégies sont possibles. Par exemple, Andes (Conati et al., 2002; VanLehn et al., 2005, 2010) pénalise l'élève en fonction du nombre de messages demandés alors qu'Advanced Geometry Tutor (Matsuda et VanLehn, 2005; Matsuda, 2004) attend une action erronée de la part de l'élève pour déclencher l'affichage du prochain message. Toutefois, toutes les stratégies mentionnées ont pour but d'empêcher l'élève d'abuser du système lorsque le nombre de messages pour obtenir une partie de la réponse est petit, ce qui n'est assurément pas le cas dans QEDX. En effet, plusieurs centaines de messages sont accessibles afin de traiter toutes les inférences du graphe des solutions. C'est donc pour éviter une attente interminable que nous court-circuitons, en quelque sorte, la machine après une échéance de dix minutes et que nous affichons alors une partie de la réponse. Il serait cependant intéressant de proposer un bouton de demande d'aide sans contraintes afin d'analyser les modèles d'utilisation étant donné le nombre imposant de messages qui peuvent être générés par notre système. Néanmoins, le grand nombre de messages produits limite souvent la pertinence de l'aide offerte. Par exemple, on propose environ dix messages concernant les éléments d'une inférence particulière avant de passer à la suivante. Par contre, les élèves en lisent généralement moins de trois. La proposition d'une autre inférence arrive donc à un moment où l'élève a déjà perdu le fil de la conversation. Ainsi, malgré notre volonté, innovatrice, de proposer diverses pistes de solution à un élève bloqué, il semble que la structure actuelle des messages ne permet pas de démontrer la pertinence et l'efficacité d'une telle approche.

Avant de conclure cette section, nous proposons une adaptation de la figure 6.1, concernant l'organisation des messages, qui s'inspire de l'architecture de notre système. Dans la figure originale, on se rappelle que le niveau supérieur identifie des catégories de messages auxquelles sont rattachés un ou plusieurs messages concrets du niveau inférieur. Cette organisation, à deux niveaux, avait pour but de permettre d'ajuster dynamiquement le contenu des messages, comme le fait l'Advanced Geometry Tutor (Matsuda et VanLehn, 2005; Matsuda, 2004), en utilisant une approche plus grossière. Cependant, la caractéristique principale qui nous intéresse ici concerne le parcours linéaire des messages, soit des génériques vers les spécifiques. Cette organisation ne permet habituellement pas de proposer de multiples pistes de solution, comme c'est le cas dans QEDX. Dans le but de corriger cette lacune, nous avons intégré, à la figure 6.1, une boucle autour des messages informatifs (figure 6.9). Celle-ci permet de représenter l'utilisation, par un enseignant, d'une série de messages d'échelons relativement uniformes afin de proposer à l'élève diverses pistes de solution. Nous avons, de plus, pris la

liberté de renommer les messages du niveau inférieur afin qu'ils correspondent aux catégories de messages définies dans ce chapitre. Il est ainsi facile d'y reconnaître l'architecture de notre machine à états finis. En effet, on retrouve bien l'ordre déterminé des états et la boucle autour des états C et P qui permet, en théorie, de proposer différentes solutions. Il faut cependant noter que, dans la figure 6.9, l'ensemble des messages associé à chaque état est énuméré explicitement. Malgré les modifications réalisées, l'organisation des messages proposée permet néanmoins de modéliser toutes les situations prises en compte par la figure originale. En effet, nous retrouvons la séquence originale lorsque nous choisissons de ne pas emprunter la boucle. L'organisation des messages révisée que nous proposons permet donc d'intégrer la possibilité d'offrir divers chemins de solution au modèle linéaire original, ce qui aboutit à un modèle amélioré, car il est plus général.

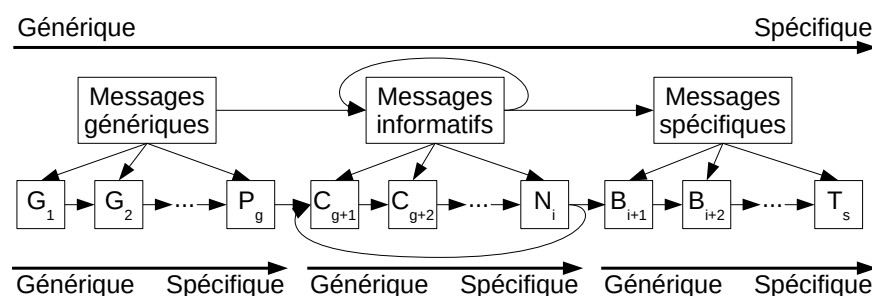


Figure 6.9 Organisation des messages selon deux niveaux intégrant un cycle

À la lumière des résultats, généralement positifs, obtenus lors des évaluations de QEDX, notre objectif, dans cette section, consistait à proposer une analyse critique concernant l'implantation de la couche GMD. Afin d'atteindre celui-ci, nous avons, tout d'abord, expliqué comment notre approche multidisciplinaire se démarque de la conception, plus traditionnelle, basée sur des analyses à priori. Grâce aux interactions constantes entre les didacticiens et les informaticiens, nous avons été en mesure de bâtir un tuteur qui met en oeuvre des théories didactiques reconnues. De plus, nous avons montré de quelle façon notre système intègre les trois types de rétroactions répertoriées par VanLehn (2006). En fait, nous offrons des rétroactions comparables aux systèmes populaires en plus de proposer deux innovations, inspirées des interventions des enseignants, soit des émoticônes ainsi que des indices associés à différentes pistes de solution. Les décisions techniques, concernant l'implantation des messages, se sont aussi avérées avantageuses. D'une part, l'utilisation d'une machine à états finis permet de représenter, de façon simple, le fonctionnement du tuteur, ce qui est intéressant dans le cadre d'un projet multidisciplinaire. L'association d'un contexte, plus ou moins générique, à chaque message, autorise, d'autre part, une grande flexibilité en ce qui concerne les messages proposés par le tuteur tout en minimisant le travail nécessaire pour l'ajout de nouveaux

problèmes. Malgré tous ces avantages, notre système comporte plusieurs lacunes. La principale concerne la production d'indices après un délai prescrit, sans possibilité, pour l'élève, d'en faire la demande au besoin. Il s'est avéré que cette politique est difficile à implanter de manière adéquate et elle limite aussi l'efficacité des autres pistes de solution. Nous avons, de plus, mentionné que notre tuteur ne s'adapte pas au style et à la compétence de l'élève. En effet, nous avons implanté un tuteur adoptant une stratégie basée sur celle d'un enseignant typique qui est adaptée pour les élèves moyens. Idéalement, par contre, le niveau des messages devrait s'ajuster dynamiquement selon la compétence de l'élève. Afin d'améliorer les performances du tuteur, il serait alors utile de mettre en oeuvre des profils d'enseignants et d'élèves en plus de revoir la politique d'affichage des messages. Nous avons enfin proposé une organisation des messages à deux niveaux révisée qui s'inspire de l'architecture de notre système en intégrant une boucle permettant d'offrir diverses pistes de solution.

6.7 Conclusion

Le but de ce chapitre consistait à exposer notre mise en oeuvre de la couche GMD de QEDX, soit celle qui permet de générer les rétroactions ainsi que les messages discursifs du tuteur. Pour atteindre ce but, nous avons d'abord fait une brève revue de la littérature. Notre principale conclusion est qu'aucun des systèmes recensés ne propose diverses pistes de résolution à un élève bloqué, ce qui limite la possibilité, pour ce dernier, d'explorer librement l'espace des solutions. Toutefois, les analyses des interactions entre les élèves et leurs enseignants, réalisées essentiellement par l'équipe de didactique (Tessier-Baillargeon, 2016), nous indiquent clairement que ces derniers s'adaptent à la démarche entreprise par l'élève. Ces interactions étant extrêmement riches et complexes, nous avons choisi de nous intéresser principalement aux interventions relatives aux inférences traitées isolément. Des messages, concernant quatorze situations répertoriées, ont alors été composés et une certaine régularité en est ressortie. À la suite de quelques corrections, qui n'ont pas mis en péril la validité didactique de la démarche, nous avons été en mesure de modéliser la structure des messages à l'aide d'une machine à états finis. Celle-ci, en plus de proposer différents indices concernant une inférence ciblée, comporte un cycle qui permet de proposer d'autres pistes de solution. De plus, des rétroactions sémaphoriques, des émoticônes ainsi que des messages propres, liés aux erreurs encodées, ont aussi été implantés dans le but de proposer les trois types d'interactions décrits par VanLehn (2006). Afin de rendre notre système fonctionnel, plus de neuf cents messages, rattachés à cinq problèmes, ont été composés. À la suite de tests unitaires satisfaisants, nous avons demandé à un expert de réaliser une vérification plus formelle et rigoureuse de QEDX. Ses tests l'ont amené à conclure que les rétroactions, produites par le GMD, permettent de

reproduire adéquatement toutes les situations répertoriées, mais que l'ordonnancement des inférences, géré par l'EDOI, est lacunaire. Les réactions des élèves, lors d'une expérimentation en classe, ont aussi été majoritairement positives. De plus, nous avons observé, à plusieurs reprises, que notre système est en mesure de relancer le processus de résolution d'un élève bloqué, ce qui nous permet d'affirmer que notre approche est pertinente et prometteuse. Cependant, la politique d'affichage des messages selon des délais fixes ainsi que l'absence de réglage du niveau des messages en fonction de la compétence de l'élève limitent l'efficacité de QEDX, surtout dans le cas d'élèves plus faibles. La conception du système par une équipe multidisciplinaire a néanmoins permis de proposer une structure d'aide innovatrice qui offre diverses pistes de solution afin de permettre à l'élève d'explorer librement le problème.

Malgré les résultats encourageants obtenus, plusieurs changements sont envisagés afin d'améliorer la performance de QEDX. Dans un premier temps, une nouvelle étude des interventions des enseignants serait utile afin de mieux comprendre la dynamique concernant la suggestion des différentes pistes de résolution. En effet, nous produisons actuellement un nombre relativement élevé de messages pour chaque inférence, donc la proposition suivante apparaît, en général, alors que l'élève ne suit plus la conversation. Ce problème est amplifié par la politique actuelle d'affichage des messages basée sur des délais prédéfinis qui impose une attente de près de dix minutes avant de passer à l'inférence suivante. Afin de recueillir des données pour nous aider à résoudre ce problème, nous pourrions proposer un système sans contraintes dans lequel l'élève est en mesure de demander de l'aide au besoin et de solliciter des indices concernant une autre solution. Ces informations, combinées à l'analyse des interventions des enseignants, pourront permettre d'ajuster la machine à états finis ou de proposer une nouvelle structure de données mieux adaptée. Une autre modification envisagée concerne l'implantation de différentes personnalités du tuteur en plus de l'adaptation du niveau des messages selon la compétence de l'élève. En effet, nous savons que chaque enseignant a une approche qui lui est propre et qu'il sait adapter son discours pour chaque élève, alors que la version actuelle de QEDX propose un tuteur générique. Une étude devra donc être menée afin de classer les enseignants ainsi que les élèves selon divers profils et les données recueillies pourront être mises à contribution pour proposer un tuteur personnalisé. L'adaptation du niveau des messages devra être analysée en parallèle, car nous croyons qu'il dépend à la fois de la compétence de l'élève et du profil de l'enseignant. Enfin, nous envisageons de remplacer certains messages discursifs par des problèmes connexes. Cette stratégie consiste, en fait, à proposer un problème, ciblant une difficulté, que l'élève doit résoudre avant de pouvoir reprendre la résolution du problème original. Il s'agit d'une stratégie que certains enseignants utilisent pour éviter de donner directement des réponses aux élèves. Ce dernier projet n'en est qu'à ses débuts, mais il permettra certainement d'enrichir les interventions de QEDX.

CHAPITRE 7 RÉSULTATS CONNEXES ET DISCUSSION GÉNÉRALE

*«Toute personne qui n'a jamais commis
d'erreurs n'a jamais tenté d'innover»*

— Albert Einstein

7.1 Introduction

Dans les chapitres précédents, nous avons notamment analysé les avantages et les limites technologiques de chacune des couches de QED-Tutrix (QEDX) prises isolément. Par contre, les constats concernant, entre autres, l'interface du système ou sa conformité face aux théories didactiques n'ont pas été traités. Ainsi, ce chapitre présente les principaux résultats qui sont relatifs au système dans sa globalité, et qui ne sont donc rattachés à aucune couche particulière, en plus d'une discussion générale. Dans un premier temps, nous nous intéressons aux commentaires recueillis au cours des tests et des expérimentations à propos de l'ergonomie de l'interface de QEDX et à l'évolution de cette dernière. Nous montrons ensuite comment l'interface du système actuel permet de mettre en oeuvre différentes démarches confirmant ainsi son statut d'Espace de Travail Géométrique (ETG) (Kuzniak, 2006). En second lieu, nous proposons une discussion en trois volets. D'une part, nous traitons des apports de la mise en oeuvre de la conception dans l'usage (Rabardel, 1995), soit qu'elle nous ait permis d'ajuster les schèmes d'utilisation de QEDX et de mettre en lumière une utilisation inattendue de certains onglets. Par la suite, nous discutons des choix concernant l'architecture informatique du système et nous montrons comment ils ont simplifié son élaboration. Nous terminons par une brève réflexion relative à la collaboration multidisciplinaire et au recours aux méthodes qualitatives dans le cadre de notre recherche.

7.2 Résultats se rapportant à l'ergonomie de QEDX

Au fil de notre démarche de recherche, nous avons eu recours à deux expérimentations en classe (voir section 3.4) dans le but de valider et de faire évoluer notre système. De plus, nous avons fait appel à un expert en didactique afin qu'il réalise une vérification rigoureuse du système avant la seconde expérimentation. Lors de ces tests et de ces expérimentations, nous avons, entre autres, recueilli des commentaires et des suggestions à propos de l'interface de QEDX. Dans cette section, nous présentons d'abord ceux-ci, ainsi qu'une description sommaire des correctifs effectués, dans un ordre chronologique. Nous donnons notamment

les raisons qui nous ont amené à ajouter l'onglet *Rédaction* et à anticiper l'implantation de l'onglet *Schéma*. Dans un deuxième temps, nous montrons comment les démarches de découverte, de validation et de modélisation sont essentiellement rattachées respectivement aux onglets *Figure*, *Phrases* et *Rédaction*. Nous proposons, en plus, quelques exemples de la mise en oeuvre des démarches par les élèves lors de l'utilisation de la version actuelle de notre système. Ces observations nous permettent alors de valider le statut d'ETG de QEDX.

7.2.1 Commentaires relatifs à l'interface

Les données recueillies lors de la première expérimentation nous ont, entre autres, permis de valider les graphes HPDIC (voir chapitre 4) et de développer les messages discursifs de QEDX (voir chapitre 6). Nous avons aussi fait des observations et recueilli des commentaires par rapport à l'interface de GGBT. Nous avons remarqué que, malgré leur aspect rébarbatif, les pourcentages d'achèvement véhiculaient des informations importantes quant à la proximité d'une démonstration complète en plus de valider que Prof Turing a bien pris en compte l'énoncé soumis. Nous les avons donc remplacés par des émoticônes dans la version subséquente de GGBT, soit QEDX. On nous a aussi suggéré des modifications à l'interface, pour le rendre plus ergonomique. Par exemple, le positionnement de la fenêtre de dialogue a été modifié. De plus, plusieurs élèves, face au peu de structure qu'impose GGBT, ont eu recours à une planification de leur démonstration sur papier et on nous a suggéré d'intégrer une fenêtre équivalente dans notre logiciel. Nous avons donc réservé un espace pour réaliser cette tâche au sein de QEDX, soit l'onglet *Schéma* qui est en développement, ainsi que l'onglet *Rédaction* qui se veut un moyen plutôt qu'une fin pour l'élaboration des démonstrations.

À la suite de cette première expérimentation, l'aide discursive ainsi que la majorité des commentaires recueillis ont été intégrés dans QEDX, à l'exception de l'onglet *Schéma*. Avant de soumettre notre système à une nouvelle expérimentation en classe, nous avons demandé à un expert en didactique de le vérifier. En plus de valider la structure et le contenu des messages (voir chapitre 6), il nous a fait des commentaires au sujet de la convivialité de QEDX. Il nous a, entre autres, indiqué que les thèmes proposés pour certains énoncés pourraient être mieux choisis. En effet, par souci d'économie, nous avons imposé une limite de quatre thèmes par énoncé ce qui ne s'est pas avéré suffisant pour bien les identifier dans plus de 75% des cas. Par exemple, initialement nous avons associé à l'énoncé «Deux triangles ayant des bases et des hauteurs homologues congrues ont la même aire» les thèmes *Aire*, *Base*, *Hauteur* et *Triangle*. Toutefois, pour bien le caractériser, il nous fallait aussi inclure les thèmes *Mesure*, *Égalité*, *Congru* et *Homologue*. Nous avons alors retiré cette limite en modifiant la structure de données qui gère les thèmes dans QEDX, ce qui a corrigé le problème.

Pour la seconde expérimentation, nous avons intégré dans QEDX les corrections proposées par l'expert. L'analyse des commentaires des élèves à la suite de l'expérimentation nous indique que l'onglet *Rédaction* semble être un ajout apprécié et important, car il permet de voir la progression de la preuve. En ce qui concerne l'onglet *Phrases*, une majorité d'élèves le trouvent utile, mais proposent d'ajouter des liens pour avoir plus d'informations au sujet des énoncés. Ceux-ci nous ont indiqué qu'ils avaient appris de nouveaux termes mathématiques grâce au classement par catégories. Quelques élèves ont fait de vagues commentaires à propos de l'interface, comme la rendre plus attrayante ou permettre une recherche en langue naturelle. Des élèves ont aussi mentionné que les preuves étaient longues à produire. Ceci est causé par la granularité fine des preuves incluses dans QEDX. Il faudrait donc permettre d'ajuster la granularité afin de l'adapter aux demandes des enseignants dans différentes situations. Enfin, les données recueillies concernant les profils d'intervention des enseignants serviront dans un projet ultérieur et n'ont, par conséquent, pas été analysées dans le cadre de nos travaux.

7.2.2 Précision du statut d'ETG

À la suite de la validation de l'interface à l'aide des commentaires des élèves, nous avons précisé à posteriori le statut d'ETG (Kuzniak, 2006) de QEDX. Pour ce faire, les didacticiens ont tenté de trouver des moments clés dans les traces d'interactions qui démontrent comment les démarches de découverte, de validation et de modélisation sont rendues possibles par l'utilisation de QEDX. Premièrement, la démarche de découverte, qui s'inscrit dans la phase heuristique de la résolution, se déroule pendant la recherche d'une stratégie de preuve par l'élève. Selon les données recueillies, il semble que cette démarche se déroule principalement dans l'onglet *Figure* de QEDX. En effet, l'élève peut simultanément étudier l'énoncé et la figure statique pour ensuite tester des conjectures sur la figure dynamique, pendant que Prof Turing peut lui fournir des pistes de solution. Par exemple, dans le cas du problème de comparaison des aires (A.4), nous avons observé que les élèves avaient tendance à expérimenter avec un cas particulier, en déplaçant le point E vers le sommet C, afin de mieux distinguer les deux triangles et de positionner les hauteurs associées à l'intérieur de ceux-ci. Cette manipulation leur permet donc de découvrir les éléments communs et permet de préciser une stratégie de preuve. Par contre, pour le problème du triangle inscrit dans le parallélogramme (A.3), la figure est statique pour préserver l'illusion d'optique. Dans ce cas, l'utilisation des outils de mesure aurait pu être envisagée, mais, selon nos données, ce ne fut pas le cas, car les élèves n'ont probablement pas eu assez de temps pour apprivoiser ceux-ci.

Le passage de l'intuition à l'axiomatisation s'inscrit dans la démarche de validation dans la nomenclature des ETG. Dans QEDX, c'est l'onglet *Phrases* qui permet aux élèves de

soumettre leurs idées, sous forme d'énoncés de géométrie formalisés, et de vérifier leur validité. Nous avons observé que les élèves qui croient avoir trouvé une stratégie de démonstration prometteuse à la suite de la manipulation de la figure soumettent leurs idées à l'aide de l'onglet *Phrases* et observent la réaction de Prof Turing. Ce dernier joue donc un rôle primordial, car il devient un instrument au service de l'élève dans sa démarche de validation. Par exemple, pour le problème A.3, les élèves ont tenté de proposer que le triangle ABC soit équilatéral, ce qui a été refusé par Prof Turing. Partant des mêmes thèmes, ils ont indiqué que le triangle est isocèle, ce qui a été accepté, et ils ont ensuite construit leur démonstration autour de ce fait. Nous avons aussi observé que le classement par thèmes permet à certains élèves de découvrir de nouvelles justifications qui peuvent être utiles dans le cadre du problème à résoudre. L'interface actuelle de l'onglet *Phrases*, combinée aux réactions du système tutoriel, permet à l'élève de réaliser une démarche de validation et pourrait aussi devenir un environnement d'apprentissage qui permet de découvrir les énoncés de géométrie euclidienne au besoin.

Lorsque l'élève est prêt à communiquer sa solution, il lui est nécessaire de l'organiser et cette étape s'articule autour d'une démarche de modélisation. Malgré la liberté que QEDX offre par rapport à l'ordre de soumission des énoncés, nous avons observé que les élèves avaient tendance à soumettre les éléments d'une inférence en groupe pour bien communiquer leur raisonnement au tuteur. On devine que les élèves ont en tête une structure qui les aide à organiser les idées qu'ils désirent communiquer. Nous avons observé que la plupart des élèves utilisent une démarche traditionnelle allant de la figure vers la communication de leurs observations à l'aide des énoncés formels. Cependant, certains élèves ont utilisé la démonstration trouée, affichée dans l'onglet *Rédaction*, comme moyen pour cibler les éléments importants de la figure. Par exemple, une équipe a utilisé l'onglet *Rédaction*, et son code chromatique, ainsi que le message de Prof Turing pour déterminer l'hypothèse manquante et ainsi compléter leur démonstration. L'onglet *Rédaction* n'est donc pas une fin, mais bien un outil qui aide l'élève à produire une démonstration au sein de QEDX. Ces observations concernant la mise en oeuvre des démarches de découverte, de validation et de modélisation nous permettent donc de confirmer le statut d'ETG de QEDX.

Dans cette section, nous avons d'abord présenté une synthèse des commentaires au sujet de l'ergonomie de l'interface de QEDX consignés durant les deux expérimentations en classe ainsi que lors des tests réalisés par l'expert. D'une part, quelques modifications de nature esthétique ont été réalisées afin d'améliorer la convivialité du système. Cependant, il s'est avéré que l'ajout des onglets, et plus particulièrement de l'onglet *Rédaction*, a eu un impact important et positif sur le travail des élèves et sur leur appréciation de QEDX. De plus, l'ajout de thèmes supplémentaires, pour mieux caractériser les énoncés, a permis de simplifier leur recherche. Les prochaines étapes qui permettront d'enrichir l'interface de notre système concernent

principalement l'ajout d'informations pédagogiques dans l'onglet *Phrases* et l'implantation de l'onglet *Schéma*. Dans la seconde partie de cette section, nous avons illustré, à l'aide d'exemples, la mise en oeuvre, par les élèves, des démarches définies dans le cadre des ETG à l'interface de QEDX. Il en est ressorti que la démarche de découverte s'opère essentiellement dans l'onglet *Figure*, celle de validation, dans l'onglet *Phrases* et celle de modélisation, dans l'onglet *Rédaction*. Il nous est donc possible de conclure, en s'appuyant sur ces observations, que QEDX est un ETG. Dans la section suivante, nous proposons une discussion générale à propos des principales décisions qui ont eu un impact sur l'ensemble de notre démarche.

7.3 Discussion

Les résultats présentés dans la section précédente relatent l'évolution de l'interface de notre système au cours des deux premiers cycles de sa conception (figure 3.2). Ils nous ont aussi permis de confirmer le statut d'ETG de QEDX. Nous proposons maintenant une discussion générale relative à la signification de ces résultats ainsi qu'une critique concernant certaines décisions techniques et méthodologiques adoptées dans le cadre du projet. Nous traitons d'abord des découvertes que nous avons faites à la suite de l'analyse des interactions entre les élèves et notre système. Nous expliquons notamment de quelle façon l'interface de QEDX rend possible une véritable exploration du problème et comment la rédaction est utilisée en tant qu'outil qui soutient le processus d'élaboration de la preuve. Du côté informatique, nous discutons des avantages, mais aussi des limites, de l'approche préconisée, centrée sur les données. Nous traitons, de plus, de l'architecture modulaire de notre système qui est basée sur le concept d'inférence ainsi que du niveau de granularité des solutions acceptées par QEDX qui limite, pour l'instant, sa pertinence. La section se termine par une réflexion sommaire par rapport à la collaboration multidisciplinaire avec des experts en didactique et à la mise en oeuvre de méthodes d'analyses qualitatives dans le cadre de notre projet.

7.3.1 Aspects ergonomiques de l'interface

Premièrement, notre logiciel est innovateur, car en plus d'avoir été développé selon le paradigme de la conception dans l'usage (Rabardel, 1995), il s'intègre parfaitement dans le cadre théorique des ETG (Kuzniak, 2006), car il est en fait un ETG. Il se distingue donc des logiciels comme ANGLE (Koedinger et Anderson, 1993; Koedinger, 1991) ou Mentoniez (Py, 1994, 1996, 2001), qui malgré leurs aspects innovateurs par rapport aux systèmes tutoriels traditionnels, ont été conçus de façon classique, soit par la succession d'une analyse, de l'implantation et de tests de validation. Même si ce travail a été réalisé par des experts du domaine, il est difficile de prévoir comment les élèves vont réagir au système ou d'anticiper

des schèmes d'utilisations différents de ceux qui ont été prévus. Par exemple, dans ANGLE, la dualité entre une figure et un énoncé semblait utile, mais a causé des problèmes lors de l'élaboration des preuves par les élèves à l'interface du logiciel. Dans notre cas, les différentes expérimentations ne visent pas à juger de la valeur de notre travail, mais sont en fait un tremplin pour la poursuite du développement. De plus, il était difficile de préciser le statut d'ETG de GGBT, car c'est l'ajout du tuteur discursif et de l'onglet *Rédaction* qui a véritablement permis aux élèves d'appliquer des démarches de validation et de modélisation au sein de notre système.

L'onglet *Rédaction* constitue un bon exemple de l'utilité de la conception dans l'usage. En effet, initialement, notre but était d'utiliser une approche assez traditionnelle pour l'élaboration des démonstrations à l'interface de QEDX, soit de permettre à l'élève d'explorer le problème en proposant des énoncés au tuteur et de passer, par la suite, à la rédaction proprement dite. Lors de la première expérimentation, aucun outil d'organisation de la démonstration n'était disponible dans GGBT et les élèves ont eu recours au support papier/crayon pour être en mesure de produire une démonstration complète. Ils nous ont donc suggéré d'ajouter un outil permettant d'avoir un environnement complet de résolution à l'interface de GGBT et c'est ainsi que nous avons eu l'idée d'ajouter l'onglet *Rédaction*. Comme celui-ci est accessible en tout temps et s'ajuste en temps réel, la rédaction n'est plus l'étape finale de la résolution du problème, mais bien un outil qui permet de produire une démonstration complète. L'utilisation de la rédaction en tant qu'outil est devenue évidente lors de la seconde expérimentation et c'est aussi à partir de nos observations que nous avons conclu que cet onglet permet véritablement à certains élèves de réaliser la démarche de modélisation. Cette nouvelle vision face à la rédaction est un autre aspect qui distingue QEDX des autres systèmes tutoriels. L'ajout de l'onglet *Schéma*, dans une version future, aura certainement un impact sur l'utilisation de l'onglet *Rédaction*, mais il nous est impossible, actuellement, de le prévoir.

L'utilisation de la rédaction en tant qu'outil a été possible grâce au design ouvert de l'interface de QEDX, qui permet l'accès, tout au long de l'élaboration d'une démonstration, à tous les onglets. En effet, dans les tuteurs classiques, par exemple Geometry Tutor (Anderson et al., 1985), le logiciel s'attend à ce que l'élève produise une rédaction sans avoir exploré le problème au préalable. Dans les systèmes dans lesquels on propose une étape d'exploration (Mentoniezh (Py, 1994, 1996, 2001)) ou de construction (Géométrie (Gressier, 2015)), l'élève qui décide de passer à la prochaine étape ne peut pas revenir en arrière dans le cas où il voudrait changer sa stratégie de résolution. Par contre, dans QEDX, l'élève peut naviguer librement et il lui est possible de tester plusieurs stratégies en parallèle avant de produire sa preuve finale. Il peut donc suivre en temps réel sa progression et le système ajuste la démonstration lorsqu'il croit avoir détecté un changement de stratégie. Il peut aussi tester de nouvelles hypothèses sur la

figure dynamique au besoin. L'utilisation des différents onglets a demandé un apprentissage, mais les élèves se sont rapidement ajustés pour tirer profit de tous les outils offerts. Cependant, ce qui a déstabilisé plusieurs élèves est le fait que QEDX accepte les énoncés dans un ordre quelconque et qu'il ne demande pas de respecter explicitement la structure des inférences. L'accès en tout temps aux différents onglets, et en particulier l'accès à l'onglet *Rédaction*, a permis de résoudre le problème, car la structure de l'inférence est bien visible à l'intérieur de la démonstration. L'interface de QEDX permet donc à l'élève de résoudre le problème posé sans le contraindre à une stratégie unique en lui offrant la possibilité de naviguer dans les onglets sans restriction et de retourner à une phase exploratoire au besoin.

Comme pour l'onglet *Rédaction*, l'utilisation de l'onglet *Phrases* par les élèves a mené à des observations intéressantes. Au départ, nous avons décidé d'implanter un répertoire d'énoncés avec un outil de recherche par thèmes. Lors des premières minutes d'utilisation, les élèves avaient du mal à repérer la phrase voulue, car ils n'arrivaient pas à cibler les thèmes pertinents. Ils ont rapidement compris que certains thèmes permettaient de cibler un nombre plus restreint de phrases et ont utilisé cette observation pour faire des recherches plus efficaces. Par la suite, certains élèves ont utilisé l'onglet *Phrases* pour découvrir de nouvelles propriétés et définitions, ce qui n'était pas le but initial. En effet, à partir des observations faites sur la figure dynamique, ils étaient en mesure de cibler les thèmes pertinents et choisissaient ensuite la phrase qui semblait s'appliquer à la situation. Ils étaient donc en mesure de retrouver une propriété connue, mais aussi d'en découvrir de nouvelles qui n'avaient pas été traitées dans le cadre du programme d'enseignement. Ils pouvaient donc explorer des solutions qui n'auraient pas été accessibles dans un environnement traditionnel. De plus, le fait que les élèves nous aient demandé d'ajouter des explications pour les phrases offertes montre que l'onglet *Phrases* n'est pas seulement un répertoire d'énoncés, mais peut offrir un environnement d'apprentissage qu'il faudra développer.

7.3.2 Examen de l'architecture informatique

En ce qui concerne l'architecture du logiciel, notons premièrement que celui-ci a été conçu selon une approche centrée sur les données, c'est-à-dire que la majorité du contenu peut être modifié par l'édition de fichiers en format texte. Dans les faits, ce choix avait pour objectif de permettre un contrôle accru, de la part des didacticiens, sur les rétroactions de Prof Turing. Un utilisateur expérimenté est en mesure de créer les fichiers contenant la description du graphe des démonstrations (HPDIC) ainsi que les messages du tuteur associés aux différents états de la machine (GMD). Concrètement, la didacticienne qui a participé au projet a créé les fichiers nécessaires à la description des cinq problèmes implantés dans QEDX et les a

corrigés au besoin. Cependant, cette approche, telle qu'implantée dans QEDX, est sensible aux erreurs, car aucun mécanisme de validation de la cohérence des inférences n'a été mis en oeuvre afin d'accepter des démonstrations non formelles. De plus, lors de la conception des formats de fichiers, nous avons tenté d'anticiper les besoins futurs et nous avons alors créé une structure plus flexible, mais aussi plus complexe que ce qui est actuellement requis. Il est donc possible, par exemple, de faire apparaître des messages à des moments qui ne correspondent pas à la séquence prévue par le GMD. La structure implantée a néanmoins grandement facilité la mise en oeuvre des rétroactions de QEDX. L'approche centrée sur les données a ainsi été bénéfique pour notre projet, car elle a permis de fournir une certaine indépendance aux producteurs de contenu en leur permettant de modifier le système de messages sans avoir recours à la programmation. Si nous voulons éventuellement rendre disponible QEDX dans les écoles, il sera essentiel de créer une interface pour faire la gestion des fichiers de données et probablement les transformer en XML pour les rendre plus facilement manipulables.

D'autre part, l'architecture de QEDX est aussi centrée sur le concept d'inférence. Ce n'est donc pas par hasard que le graphe HPDIC forme la base de notre structure en couches. Initialement, il ne servait qu'à représenter l'ensemble des démonstrations d'un problème, mais il est rapidement devenu une structure unificatrice, car il contient toutes les inférences utiles à la résolution d'un problème. La manipulation des inférences nous permet de simplifier l'architecture de notre logiciel, car chacune encapsule ses éléments constitutifs. De plus, c'est un concept central, du point de vue de la didactique, lors de l'élaboration des démonstrations. Ce choix s'est donc fait naturellement et nous avons même envisagé de l'étendre à la description des constructions sur la figure pour que celles-ci puissent s'intégrer aisément dans QEDX. Ancrée sur cette représentation, l'architecture de QEDX est modulaire. La couche GMD, par exemple, nécessite une liste d'inférences pour fonctionner correctement. La couche EDOII a pour fonction de lui fournir cette liste et c'est le seul lien qui existe entre ces deux couches. Cette modularité a simplifié le développement lors des différentes phases du projet et a rendu possible l'utilisation de tests unitaires pour vérifier le fonctionnement et la validité de nos algorithmes. QEDX, en plus d'être valide du point de vue de la didactique, a été conçu de façon à être facilement maintenu et étendu en ce qui concerne son architecture informatique.

Le niveau de granularité des solutions actuellement incluses dans QEDX limite néanmoins son usage dans des classes de niveau secondaire. Pourtant, notre but initial était de proposer un système adapté à l'enseignement de la géométrie au Québec, c'est-à-dire qu'il devait permettre de produire des démonstrations qui peuvent contenir des raccourcis inférentiels. Nous avons donc évité d'utiliser un démonstrateur automatique qui produit des preuves formelles et c'est pour cela que les graphes HPDIC ont été développés. Ceux-ci n'imposent aucune restriction au niveau du contenu d'une inférence, c'est-à-dire que la cohérence entre

les antécédents, la justification et le conséquent n'est pas vérifiée par QEDX. Ceci permet d'introduire des inférences acceptables dans une démonstration, mais qui ne sont pas nécessairement valides du point de vue formel. Les graphes HPDIC permettent donc d'encoder les différentes démonstrations acceptables et offrent même la possibilité d'accepter, à la limite, une argumentation en tant que démonstration valide. Cependant, dans les faits, les didacticiens ont produit, pour les cinq problèmes implantés, des inférences qui sont très près des raisonnements formels afin que toute solution valable soit reconnue par le système. Il en résulte donc des démonstrations très longues et élaborées, ce qui va à l'encontre de notre objectif de départ. Ce n'est pas mauvais en soi, mais la version actuelle ne correspond pas à la réalité des classes de quatrième secondaire. Il reste donc du travail à faire, tant de la part des didacticiens que des informaticiens pour proposer des démonstrations adaptées aux objectifs d'apprentissage déterminés par les enseignants.

7.3.3 Décisions méthodologiques

Nous désirons finalement aborder brièvement deux sujets concernant la méthodologie qui ont eu une influence majeure sur notre démarche, soit le travail au sein d'une équipe multidisciplinaire et le recours à des analyses qualitatives. D'une part, il est important de rappeler que l'élaboration de QEDX ne s'est pas faite à la suite d'une simple analyse à priori du travail des experts, mais bien par une collaboration étroite et continue entre des experts en didactique et en informatique. Cette collaboration fut enrichissante, car elle a servi à élargir notre champ de compétence en nous initiant à différents principes didactiques qui ont ensuite été intégrés dans la conception de notre système. De plus, elle nous a donné l'occasion d'exercer nos habiletés de vulgarisation afin que les didacticiens soient aptes à entrevoir les principaux défis informatiques. Outre l'enrichissement personnel, une approche multidisciplinaire permet généralement d'obtenir une solution plus complète. Ceci s'explique par le fait que la collaboration entre des personnes ayant des expertises diverses aide à mieux cerner les différents aspects d'un problème. Aussi, les néophytes dans un domaine posent parfois des questions étonnantes, mais pertinentes, qui peuvent amener l'expert à réévaluer la solution. Par exemple, le traitement de problèmes contenant des contradictions n'avait pas été prévu au départ par les informaticiens, mais ils ont été intégrés à la suite d'une demande des didacticiens. Seulement, il demeure difficile de faire reconnaître la valeur d'une telle démarche en milieu académique, car un travail parfaitement multidisciplinaire est habituellement incomplet dans chacun des domaines traités, même si l'ensemble forme un projet de recherche complet et cohérent. Néanmoins, un projet multidisciplinaire a généralement des répercussions positives autant sur la solution proposée que sur les membres de l'équipe, comme nous avons eu l'occasion d'en faire l'expérience.

Notre approche multidisciplinaire nous a amené, d'autre part, à employer des analyses qualitatives pour valider notre système, ce qui n'est pas commun en génie. Or, nous ne prétendons pas que QEDX, dans son état actuel, a une influence sur les résultats scolaires, car il est, en fait, en cours d'élaboration. Lors des expérimentations, nous souhaitons plutôt recenser les schèmes d'utilisation (Rabardel, 1995) prévus ou inopinés afin de nous assurer que notre système met en oeuvre le contrat didactique. Pour ce faire, les analyses qualitatives sont toutes indiquées pourvu qu'elles soient réalisées de façon rigoureuse en utilisant des cadres comme ceux suggérés par Paillé et Mucchielli (2012). De plus, afin d'éviter de proposer des scénarios qui restreignent le travail de l'élève, nous avons mis en oeuvre la théorisation ancrée (Glaser et Strauss, 1967), dans laquelle le chercheur aborde l'expérimentation avec un minimum d'*aprioris* pour ainsi construire sa théorie lors des analyses. Dans le cadre de notre recherche, les didacticiens ont fait les analyses (Tessier-Baillargeon, 2016) en ayant recours aux catégories conceptualisantes, qui consistent à résumer une séquence d'actions à l'aide de concepts déterminés par le chercheur, afin d'extraire, par exemple, les différentes démarches de l'élève. Qui plus est, elles ont aussi permis de découvrir, entre autres, des utilisations inattendues de l'onglet *Phrases* et de l'onglet *Rédaction* qui n'auraient potentiellement pas pu être détectées par l'utilisation de méthodes quantitatives. Or, il est vrai que les résultats issus de nos analyses sont difficilement généralisables, notamment à cause de la taille d'échantillon restreint. Toutefois, comme nous n'avons pas l'ambition d'extrapoler nos observations, mais bien de répertorier des schèmes d'utilisation, les méthodes qualitatives se sont avérées adéquates. Cependant, lorsque QEDX aura atteint une certaine maturité, nous envisageons de réaliser une étude quantitative afin de comparer ses performances à celles d'autres systèmes.

Dans cette discussion, nous avons notamment présenté quelques aspects innovateurs de QEDX autant par rapport au volet didactique qu'au volet informatique. Ainsi, notre système se distingue, car il a été conçu en appliquant la méthodologie de la conception dans l'usage, ce qui nous a permis de le façonner en tant qu'ETG authentique, tel que confirmé par des observations faites en classe. De plus, alors que les systèmes tutoriels classiques considèrent la rédaction comme une fin, nous l'avons proposée comme un outil d'aide à la démonstration en autorisant l'accès à celle-ci en tout temps. Nous avons même observé que l'onglet *Phrases*, initialement élaboré comme un répertoire d'énoncés, a été utilisé en qualité de véritable instrument d'apprentissage et de découverte de la géométrie. En ce qui concerne le volet informatique, l'architecture modulaire par couches et l'approche centrée sur les données ont facilité le développement du système, à la fois pour les informaticiens et les didacticiens. Par contre, le niveau de granularité des preuves actuellement implantées dans QEDX n'est pas adapté au degré d'exigence des enseignants de niveau secondaire. Enfin, nous avons indiqué que l'approche multidisciplinaire a eu des impacts positifs sur notre formation et sur

la solution proposée. Elle nous a aussi amené à mettre en oeuvre des analyses qualitatives rigoureuses, peu orthodoxes en informatique, mais qui sont cohérentes avec notre démarche.

7.4 Conclusion

L'objectif de ce chapitre consistait à présenter des résultats et une discussion concernant les aspects de QEDX qui ne sont rattachés à aucune couche logicielle particulière, notamment l'ergonomie de son interface. Nous tenons d'abord à rappeler que notre système est en constante évolution, car nous avons appliqué le paradigme de la conception dans l'usage. Les deux expérimentations réalisées nous ont, entre autres, permis de recueillir des informations concernant l'ergonomie de son interface, mais aussi de confirmer que les schèmes d'utilisation prévus ou originaux permettent de mettre en oeuvre le contrat didactique. De plus, ce processus nous autorise à faire évoluer l'interface à la suite de chacune des expérimentations, ce qui nous a donné l'occasion de créer un ETG. Nous avons même découvert que les élèves utilisent la rédaction en tant qu'outil et que l'onglet *Phrases* contribue à l'apprentissage de la géométrie. QEDX se distingue donc des systèmes tutoriels traditionnels par son processus de conception itératif qui intègre des considérations didactiques et par son interface qui permet une véritable exploration du problème tout en soutenant de nouveaux schèmes d'utilisation. Cette vision innovatrice du système tutoriel n'aurait pas été possible sans une collaboration multidisciplinaire étroite et l'emploi de méthodes d'analyses qualitatives. L'architecture informatique par couches ainsi que le choix d'une approche centrée sur les données ont aussi été influencés par cette collaboration. Néanmoins, le traitement actuel des solutions à l'intérieur de QEDX force le didacticien à produire des démonstrations dont le niveau de granularité n'est pas compatible avec les exigences des enseignants du secondaire, ce qui limite l'utilité de notre système. Il nous apparaît alors que du travail sera nécessaire pour offrir, par exemple, une interface qui permettra aux enseignants de choisir des canevas de démonstrations qu'ils jugent acceptables selon les objectifs qu'ils auront sélectionnés. D'autre part, en plus de l'onglet *Rédaction*, les élèves nous ont suggéré d'ajouter un espace dans lequel ils pourraient organiser graphiquement leur démonstration, soit l'onglet *Schéma*. Nous envisageons, entre autres, d'offrir une interface de type babillard électronique dans laquelle l'élève pourrait construire lui-même l'arbre de sa démonstration à partir des énoncés qu'il a soumis. L'onglet *Schéma* n'a cependant pas été implanté dans le cadre de notre recherche, car nous n'avons pas les informations requises afin de proposer une interface qui autorise des schèmes d'utilisation qu'il nous faut déterminer. Bref, du travail sera nécessaire, autant par rapport au volet didactique qu'au volet informatique, pour ajouter de telles fonctionnalités dans QEDX et ainsi rendre celui-ci plus efficace pour soutenir l'élève dans son processus de résolution.

CHAPITRE 8 CONCLUSION GÉNÉRALE

«Ça ne veut pas dire que ce n'est pas fini.»

— Pierre Gauthier

Pour conclure cette thèse, notons d'abord qu'elle consiste essentiellement à présenter le second cycle de conception de QED-Tutrix (QEDX), soit un système tutoriel intelligent qui soutient l'élève lors de l'élaboration de preuves en géométrie. Ce système a été développé dans le cadre d'une recherche multidisciplinaire au cours de laquelle des experts en didactique des mathématiques et en informatique ont collaboré. Il met en oeuvre des théories didactiques reconnues, en plus de proposer des rétroactions qui sont inspirées des interventions d'enseignants observés. L'approche multidisciplinaire préconisée ainsi que la stratégie d'aide proposée par QEDX en font un système tutoriel intelligent innovateur par rapport à ceux actuellement offerts (voir chapitre 2). Afin d'appuyer cette affirmation, nous présentons, dans une première section, une synthèse de nos travaux. Dans celle-ci, nous reprenons indépendamment nos cinq objectifs secondaires afin de démontrer qu'ils ont été atteints. Nous indiquons aussi les aspects innovateurs de QEDX qui sont associés à chacun. Nous faisons ensuite de même pour notre objectif principal, ce qui nous permet de clore la synthèse. Dans la section suivante, nous résumons les limites fondamentales de notre système, que nous divisons en trois catégories. De plus, pour chacune, nous proposons des pistes de solution à explorer dans le cadre de travaux futurs. Enfin, nous proposons de nouveaux axes de recherche relatifs aux cycles de conception de QEDX et d'autres, connexes, dans une ultime section.

8.1 Synthèse des travaux

Dans cette première section, nous présentons une synthèse de nos travaux qui est structurée autour des cinq objectifs secondaires et de l'objectif principal de notre recherche. Nous avons pris cette initiative, car chacun correspond essentiellement à un chapitre de la présente thèse. Seul le chapitre 2, relatant la revue de littérature, n'est rattaché à aucun objectif. Toutefois, les principales conclusions issues de ce dernier sont intégrées dans l'analyse des objectifs. De plus, pour chacun, nous soulignons les innovations notables résultant de nos travaux. Il est à noter que l'ordre de traitement des objectifs secondaires correspond à celui des chapitres de la thèse alors que l'examen de l'objectif principal fait office de conclusion à la section.

L'objectif secondaire que nous allons d'abord examiner est notamment relatif aux aspects didactiques de nos travaux et consiste à «Construire un système tutoriel qui met en oeuvre des

théories didactiques reconnues, en faisant appel à une équipe multidisciplinaire». Nous pouvons affirmer que cet objectif a été atteint, car, d'une part, la collaboration multidisciplinaire a été un véritable succès. En effet, à l'opposé de l'approche traditionnelle de conception qui ne se base que sur des études à priori, nous avons mis à profit un dialogue continu entre les didacticiens et les informaticiens afin d'obtenir un système plus raffiné. De plus, cette approche a élargi le cadre de nos connaissances et a créé une symbiose entre les chercheurs. D'autre part, la mise en oeuvre de théories didactiques pour l'élaboration de QEDX a aussi été une réussite. En effet, nos cycles de conception sont inspirés de la conception dans l'usage afin de créer un système dont les schèmes d'utilisation peuvent évoluer au cours de son élaboration. Un exemple en est l'onglet *Phrases* qui a étonnamment servi d'outil d'apprentissage. Ensuite, nos analyses des interactions entre QEDX et l'élève s'ancrent dans la TSD. Finalement, nous avons démontré, à l'aide des résultats d'une expérimentation en classe, que l'interface de notre système est un ETG authentique. Bien que quelques systèmes aient mis en application des théories didactiques, nous sommes les premiers à en proposer un qui soit conforme à la théorie des ETG. Nous avons également eu recours à des analyses qualitatives, effectuées essentiellement par l'équipe de didactique (Tessier-Baillargeon, 2016), afin de valider notre système, ce qui est cohérent avec notre démarche, mais peu usuel pour une recherche en informatique. QEDX propose donc des interactions conformes aux théories didactiques grâce à une collaboration multidisciplinaire, ce qui confirme l'atteinte de l'objectif.

Les quatre autres objectifs secondaires correspondent respectivement aux quatre couches logicielles de QEDX. Premièrement, l'objectif qui consiste à «Identifier une manière adéquate d'encoder l'ensemble des démonstrations acceptables, qui ne sont pas nécessairement formelles, pour un problème donné» a été atteint par l'entremise de la couche HPDIC. Le défi de cet objectif concerne le traitement des preuves non formelles issues du paradigme de géométrie cognitive (Richard et Fortuny, 2007) en usage au Québec. Cette contrainte rendait presque impossible l'utilisation d'un moteur de déduction automatique comme ceux mis en oeuvre dans d'autres systèmes tutoriels intelligents recensés. Nous avons alors proposé les graphes HPDIC qui sont construits à partir de l'ensemble des inférences admissibles pour résoudre un problème. Ces dernières sont déterminées par l'enseignant ou le didacticien et elles ne sont soumises à aucune contrainte formelle. Par conséquent, il est possible de traiter, entre autres, la géométrie cognitive, les raccourcis inférentiels, les contradictions, etc. Le graphe HPDIC résultant contient, en fait, l'ensemble des solutions acceptables au problème posé dans un minimum d'espace mémoire. Il n'a donc pas été nécessaire de restreindre le nombre de solutions valides, comme c'est souvent le cas. Chaque solution peut être extraite à l'aide des algorithmes que nous avons proposés et elle contient toutes les informations requises pour produire les principales représentations de la preuve inventoriées. Il est aussi à noter que les

graphes HPDIC sont analogues aux réseaux déductifs enseignés à l'école secondaire. Ainsi, l'objectif a été réalisé, car le graphe HPDIC contient l'ensemble des preuves, engendrées par les inférences répertoriées, qui peuvent être admises dans divers contextes d'apprentissage.

Pour sa part, le MIA permet de «Modéliser l'aspect dynamique de la démarche de résolution de l'élève», ce qui correspond à l'objectif secondaire suivant. Dans celui-ci, c'est essentiellement le caractère dynamique des actions qui constitue l'obstacle majeur à surmonter. En effet, dans les systèmes imposant un ordre de résolution, seul l'ensemble des actions réalisées, sans égards à leur séquence, est requis afin de déterminer l'aide à offrir. Par contre, si nous acceptons que les énoncés soient inscrits dans un ordre quelconque, par exemple, pour autoriser une exploration libre, la modélisation traditionnelle ne permet pas de générer des indices qui respectent l'état cognitif de l'élève. Nous avons alors innové en conservant la chronologie des énoncés soumis par ce dernier. Ces traces correspondent donc à l'historique de l'un des indicateurs de son état cognitif qui peut nous permettre de comprendre son parcours de résolution, soit les différentes approches qu'il a explorées. Ainsi, notre modélisation accepte les changements de stratégie de résolution contrairement à d'autres systèmes. À partir de ces données, il est possible, grâce aux couches supérieures de QEDX, d'analyser le cheminement de l'élève et de lui proposer des messages qui correspondent à son état cognitif ou même d'autres pistes de solution. Par conséquent, le fait de consigner la chronologie des actions permet d'atteindre l'objectif, car nous conservons l'aspect dynamique de la démarche tout en permettant une exploration libre du problème.

En ce qui concerne l'objectif qui consiste à «Interpréter la chronologie des actions de l'élève afin de reconnaître sa stratégie de résolution», c'est la couche EDOl qui permet de l'atteindre. En réalité, la reconnaissance de la stratégie de résolution fait référence à la fois à l'identification de la solution que l'élève tente de compléter ainsi qu'à l'estimation de son état cognitif, soit la détermination des inférences sur lesquelles il a récemment travaillé. D'une part, nous considérons que la preuve la plus avancée est celle qu'il désire achever. Le défi consistait alors à la retrouver sans avoir à énumérer toutes les preuves qui sont possibles. Nous avons donc employé différentes stratégies afin de contenir l'explosion combinatoire du nombre de solutions, mais ces efforts n'ont pas été suffisants pour traiter certains problèmes implantés. Par conséquent, nous avons proposé deux algorithmes innovateurs, soit un qui permet de décomposer le graphe HPDIC en une forêt d'arbres et un autre qui estime efficacement la solution la plus avancée dans chacun. Nous n'avons ainsi pas eu recours à des moyens artificiels pour réduire le nombre de solutions, comme c'est le cas dans d'autres systèmes. Notre algorithme d'ordonnancement des inférences, basé sur la chronologie des actions et sur la preuve la plus avancée, a, d'autre part, permis d'estimer l'état cognitif de l'élève et est inspiré des interventions d'enseignants observés. Celui-ci est aussi un apport important, car il

nous donne la possibilité de proposer une aide qui respecte l'état cognitif, ainsi que d'autres pistes de résolution, lors d'une exploration libre du problème au lieu d'imposer une séquence déterminée d'actions. Ainsi, l'EDOI permet de reconnaître la stratégie de résolution de l'élève par la mise en oeuvre de différents algorithmes innovateurs qui utilisent les données du MIA. Finalement, la couche GMD a permis de réaliser le dernier objectif secondaire qui visait à «Apporter le soutien nécessaire à l'élève par l'entremise de messages discursifs inspirés des interventions d'enseignants réels». Cet objectif impliquait donc l'observation d'enseignants dans le but de modéliser leurs interventions sous forme de messages d'aide pertinents. Pour l'atteindre, nous avons fait une première expérimentation en classe afin de recueillir des traces d'interactions entre les enseignants, les élèves et GeoGebraTUTOR (GGBT). Étant donné la complexité de celles-ci, nous avons choisi de modéliser essentiellement les messages associés à une inférence particulière. Nous les avons implantés sous forme d'une machine à états finis compte tenu de la régularité associée à ce type d'intervention. Notre machine intègre aussi la proposition de différentes pistes de solution qui est basée sur la liste ordonnée d'inférences issue de la couche EDOI. Cette séquence d'affichage des messages, qui inclut de multiples propositions, n'est offerte dans aucun autre système recensé. De plus, nous proposons des émoticônes ainsi que des messages d'encouragement qui sont inspirés des réactions spontanées des enseignants et qui sont aussi uniques à QEDX. Enfin, des rétroactions propres aux erreurs répertoriées sont également possibles. En fait, nous offrons les trois catégories de rétroactions déterminées par VanLehn (2006) et le soutien proposé à l'élève se compare avantageusement à celui des principaux systèmes offerts. Une seconde expérimentation nous a permis de confirmer que QEDX apporte une aide pertinente et efficace lorsqu'il est en mesure d'estimer adéquatement l'état cognitif de l'élève. Les messages discursifs, inspirés des interventions d'enseignants et produits par la couche GMD, permettent donc d'offrir un soutien adéquat aux élèves ce qui confirme l'atteinte de l'objectif.

À la suite de l'analyse de chacun des cinq objectifs secondaires qui nous a permis de conclure qu'ils ont tous été réalisés avec succès, il ne nous reste qu'à vérifier que le système résultant permet d'atteindre l'objectif principal. Ce dernier visait à «Proposer un système tutoriel intelligent qui soutient l'élève dans une démarche d'exploration, plutôt que de rédaction, dans le cadre de l'élaboration d'une preuve en géométrie». Il a été énoncé en réponse aux contraintes des systèmes tutoriels existants qui forcent habituellement l'élève à travailler sur une démonstration formelle optimale selon une séquence déterminée qui correspond, en fait, à une démarche de rédaction. Ceux-ci n'intègrent donc pas la phase exploratoire du processus de résolution ou, lorsqu'ils le font, n'offrent aucun soutien à l'élève. Par contre, le système que nous avons élaboré, soit QEDX, permet de combler ce manque et ainsi d'atteindre notre objectif. Premièrement, il s'adresse véritablement aux élèves du secondaire en offrant la

possibilité de produire des preuves en géométrie cognitive par l'entremise des graphes HPDIC. En fait, l'enseignant peut adapter le niveau d'exigence, ainsi que les messages du tuteur, selon l'apprentissage ciblé, sans être contraint à demander des preuves formelles. De plus, notre système accepte l'écriture des énoncés dans un ordre quelconque, ce qui autorise l'élève à explorer le problème et ce qui l'encourage même à le faire. En effet, celui-ci peut suggérer des conjectures qui ne font pas partie de sa preuve courante pour éventuellement changer de stratégie de résolution sans restrictions. QEDX est en mesure, à l'aide de l'EDOI, de suivre son cheminement et de lui proposer une aide inspirée des interventions d'enseignants observés, qui respecte son état cognitif tout au long de sa démarche. Il encourage également l'exploration en proposant d'autres pistes de solution à un élève bloqué, ce qui est innovateur dans le domaine des systèmes tutoriels intelligents. Il est aussi important de rappeler que dans le cadre de ce projet multidisciplinaire, nous avons mis en oeuvre des théories didactiques reconnues et nous avons démontré que l'interface de QEDX est un ETG, c'est-à-dire qu'il permet de réaliser les trois démarches spécifiées dans ce cadre conceptuel. Enfin, une expérimentation en classe a permis de valider la pertinence de l'aide apportée par notre système. En général, les élèves ont su profiter de l'exploration libre offerte et plusieurs ont trouvé les messages d'aide utiles et adéquats. Nous pouvons alors conclure que l'objectif a été atteint, car la phase exploratoire est possible et même encouragée par l'entremise des messages générés par QEDX, ce qui est d'ailleurs innovateur par rapport aux systèmes recensés.

8.2 Limites de la solution proposée

Dans la section précédente, afin de démontrer la réalisation des différents objectifs, nous avons insisté sur les aspects positifs ainsi que sur les éléments innovateurs concernant QEDX. Toutefois, celui-ci est loin d'être parfait, comme nous l'ont révélé les résultats des tests et des expérimentations. Nous allons donc, dans les paragraphes qui suivent, faire un rappel des principales limites de notre système en plus de proposer des solutions qui pourraient être envisagées. Nous les avons regroupées en trois principales catégories qui seront traitées séquentiellement. La première concerne la gestion du contenu, comme l'ajout de nouveaux problèmes ou la création des messages du tuteur. Les limites relatives à l'architecture logicielle ainsi qu'à la performance des algorithmes proposés constituent la seconde catégorie. Enfin, la dernière regroupe les lacunes reliées aux messages d'aide proposés par QEDX.

Tout d'abord, le nombre restreint de problèmes qui sont actuellement offerts à l'interface de QEDX constitue une des principales limites relatives à son utilisation. Cette situation découle du fait que l'ajout de chaque problème demande plusieurs heures de travail de la part des experts. En effet, comme nous avons choisi de travailler selon le paradigme de la

géométrie cognitive et de permettre les raccourcis inférentiels, il est alors nécessaire d'inscrire manuellement chacune des inférences acceptées dans des fichiers textes. Cette procédure est fastidieuse en plus d'être propice aux erreurs. Aussi, si un enseignant désire, par exemple, modifier le niveau de granularité de la preuve demandée, il doit adapter le fichier correspondant. Comme QEDX ne vérifie pas la cohérence des preuves et n'accepte que les énoncés issus des inférences inscrites par le didacticien, les erreurs présentes peuvent alors empêcher certains élèves de compléter le problème. Pour chaque nouveau problème, il faut également produire les messages du tuteur qui sont, eux aussi, contenus dans des fichiers textes. Ainsi, la procédure pour ajouter ou modifier du contenu demande actuellement un travail rigoureux de la part d'experts, autant en didactique qu'en manipulation des données de QEDX. Afin de simplifier cette tâche, il serait d'abord utile de construire une interface pour les enseignants dans laquelle ils auraient une représentation visuelle des inférences ainsi que des messages qui y sont associés. De plus, l'intégration d'un moteur formel pourrait, entre autres, permettre de vérifier la cohérence des fichiers de données, automatiser la construction des graphes HP-DIC, etc. Cependant, des études seront nécessaires afin de produire des raccourcis inférentiels valides pour les solutions générées automatiquement. Par conséquent, l'ajout de nouveaux problèmes est actuellement difficile à cause de l'approche centrée sur les données qui a été retenue, mais pourrait être simplifié par la production et l'intégration de différents outils.

En ce qui concerne les limites de l'architecture et des algorithmes proposés, elles émergent essentiellement de la couche EDOI. D'une part, l'identification de la stratégie de résolution de l'élève est à l'occasion déficiente, ce qui amène QEDX à proposer une aide incohérente. En effet, notre algorithme d'évaluation de la solution dominante détecte parfois des changements inopinés de stratégie. Ceci est causé à la fois par un manque d'informations relatives aux actions de l'élève, mais aussi par notre heuristique d'estimation de l'achèvement des solutions que nous avons implanté afin d'éviter de toutes les énumérer. Néanmoins, l'énumération est tout de même nécessaire lors de la création des arbres HPDIC, ce qui limite la complexité des problèmes que notre système peut traiter. Afin de déterminer précisément la stratégie de résolution, il serait utile de recueillir des données plus exhaustives relatives aux actions de l'élève à l'intérieur du MIA. On pourra alors les analyser afin de produire une heuristique robuste qui nous éviterait potentiellement d'énumérer l'ensemble des preuves. D'autre part, l'ordonnancement des inférences, qui permet de proposer différentes pistes de résolution, gagnerait aussi à être amélioré. En effet, nous nous sommes aperçu que des priorités homologues sont associées à des inférences similaires, car elles partagent un même ensemble d'énoncés. Ces inférences sont alors traitées successivement par le GMD, mais elles ne permettent pas véritablement de proposer d'autres démarches, même s'il s'agit d'inférences distinctes, étant donné qu'elles ont plusieurs énoncés en commun. Pour augmenter la diversité des suggestions

de QEDX, on pourrait envisager de regrouper les inférences similaires et ne proposer des indices que pour une inférence typique de chaque groupe. L'algorithme d'ordonnancement pourrait aussi prendre en compte une métrique de similitude. Bref, bien que notre système ait été en mesure d'aider certains élèves, des améliorations concernant la stratégie d'assistance sont envisageables.

Les dernières lacunes que nous jugeons opportun d'aborder portent sur la politique d'affichage des messages ainsi que sur leur contenu. Premièrement, le fait d'utiliser une écriture canonique pour les énoncés a perturbé certains élèves, car les messages du tuteur ne semblaient pas faire directement référence aux énoncés soumis. De plus, comme nous n'avons modélisé qu'un sous-ensemble des interventions de l'enseignant typique, le contenu des messages est relativement uniforme et est destiné à un élève moyen ou doué. Il est aussi à noter que le niveau des messages proposés n'était pas ajusté en fonction de la performance de l'élève. Par conséquent, les élèves faibles avaient généralement du mal à comprendre les indices, ce qui rendait les interventions de QEDX inefficaces. Toutefois, le principal irritant concernait notre politique d'affichage des messages selon des échéances fixes. En effet, plusieurs élèves nous ont indiqué que le temps d'attente était soit trop court ou trop long. Cette politique a été mise en oeuvre afin d'éviter que les élèves n'abusent du système, mais il semble qu'elle soit inadéquate. Elle provoque aussi un affichage tardif des messages concernant les autres pistes de résolution qui sont alors ignorés par les élèves. Ces constats démontrent donc qu'il est essentiel de revoir la politique d'affichage des messages afin d'en produire lorsque l'élève en a réellement besoin. Ceci implique l'instauration de profils d'élèves qui pourraient aussi permettre de moduler le niveau des messages selon la compétence de ceux-ci. Des profils d'enseignants, associés à des indices directifs ou proactifs, donneraient la possibilité de générer des messages mieux adaptés tout en augmentant leur variété. Des rétroactions associées aux erreurs modélisées pourraient enfin être ajoutées, car l'infrastructure nécessaire est en place, mais elle n'a pas été mise à profit dans le cadre de nos expérimentations. En définitive, malgré toutes les lacunes, dont certaines importantes, recensées dans cette section, il n'en demeure pas moins que l'aide offerte par QEDX a soutenu efficacement plusieurs élèves, mais elle pourrait être améliorée en explorant les différentes suggestions mentionnées.

8.3 Améliorations futures

Dans le cadre de notre recherche, nous avons aussi dressé une liste d'améliorations qu'il serait intéressant d'implanter dans notre système et que nous résumons ici. Premièrement, une communication bidirectionnelle, entre QEDX et un logiciel de géométrie dynamique, devrait être développée afin de permettre la mise en oeuvre d'un tuteur de construction. Il faudra aussi

travailler à arrimer les protocoles de construction au graphe HPDIC de manière à ce que notre tuteur puisse soutenir l'élève dans cette démarche, car les manipulations ou les constructions de celui-ci ne sont actuellement pas prises en compte par QEDX. De plus, comme nous avons observé que le répertoire d'énoncés pouvait faire office d'outil d'apprentissage, il serait utile d'enrichir son contenu en y ajoutant, par exemple, des bulles d'informations, des animations, des vidéos, etc. Dans ce domaine, Desmarais (2008) nous donne quelques exemples d'interfaces pour l'apprentissage juste à temps qui pourraient être envisagées. L'onglet *Schéma* constitue un autre outil qu'il serait intéressant d'implémenter, car il permettrait de proposer une représentation différente de la preuve, ce qui pourrait aider certains élèves (Wong et al., 2011). En effet, il donnerait l'occasion à ceux-ci de construire l'arbre HPDIC, ou le réseau déductif, qui correspond à leurs preuves, tout en étant assistés par le tuteur de QEDX. Ce dernier pourrait aussi mettre à profit le schéma de l'élève afin d'améliorer la reconnaissance de son plan de résolution, car il permet de cibler une solution précise grâce à sa structure. Enfin, étant donné que l'arbre HPDIC d'une démonstration contient toutes les informations nécessaires afin de produire ses différentes représentations, celles-ci pourraient être proposées à l'interface de QEDX. On pourrait aussi envisager de permettre de sélectionner les éléments de la preuve manquants pour lesquels on désire obtenir des indices. Par conséquent, chacun des onglets de QEDX peut être enrichi par la mise en oeuvre de nos suggestions.

Bien que toutes les améliorations proposées jusqu'à maintenant dans ce chapitre puissent avoir un effet bénéfique sur l'efficacité de QEDX, seuls les profils d'enseignants et d'élèves ont un lien direct avec les cycles de conception prévus (figure 3.2). En fait, ceux-ci font partie du cycle qui suit immédiatement les travaux présentés dans cette thèse. Ainsi, des informations permettant de construire les divers profils ont déjà été recueillies et doivent être analysées afin d'extraire une structure des profils d'interventions. Concrètement, il s'agit de déterminer de quelle façon les différents types d'enseignants adaptent leur discours selon la compétence de l'élève. Par la suite, les profils devront être implantés dans QEDX et validés lors d'une expérimentation. De plus, il faudra profiter de cette occasion pour colliger des données concernant l'usage des problèmes connexes servant à relancer le processus de résolution d'un élève bloqué. Il sera donc nécessaire de choisir des enseignants qui utilisent cette stratégie d'enseignement qui peut paraître étonnante, car il s'agit, dans les faits, de répondre à une question par une autre. Le défi consistera alors à déterminer des critères de connexité qui permettront de construire une structure des problèmes semblable à celle mise en oeuvre par l'expert. Une expérimentation permettra de clore ce quatrième cycle et de recueillir des informations concernant l'évaluation des compétences qui sera implantée dans QEDX. Ces dernières données pourraient aussi amener QEDX à offrir des itinéraires de problèmes, c'est-à-dire qu'il pourrait proposer, de façon autonome, le prochain problème

que l'élève devra résoudre selon ses compétences, à la manière d'ASSISTments (Feng et al., 2011), par exemple. À ce stade de conception, il sera alors possible de mettre en oeuvre des évaluations quantitatives afin de comparer les performances de QEDX à celles d'autres systèmes. Les prochains cycles d'élaboration de QEDX présentent donc des défis intéressants et permettront d'offrir un système qui propose un environnement d'apprentissage unique.

RÉFÉRENCES

S. Ainsworth, N. Major, S. Grimshaw, M. Hayes, J. Underwood, B. Williams, et D. Wood, “REDEEM : Simple intelligent tutoring systems from usable tools”, dans *Authoring Tools for Advanced Technology Learning Environments*, T. Murray, S. B. Blessing, et S. Ainsworth, édés. Netherlands : Springer Netherlands, 2003, ch. 8, pp. 205–232.

V. Aleven et K. R. Koedinger, “Knowledge component (KC) approaches to learner modeling”, dans *Design Recommendations for Intelligent Tutoring Systems*, série Adaptive Tutoring, R. A. Sottolare, A. Graesser, X. Hu, et H. Holden, édés. Orlando, FL, USA : U.S. Army Research Laboratory, Août 2013, vol. 1, ch. 15, pp. 165—182.

——, “Limitations of student control : Do students know when they need help ?” dans *Intelligent Tutoring Systems*, série Lecture Notes in Computer Science, G. Gauthier, C. Frasson, et K. VanLehn, édés. Springer Berlin Heidelberg, Juin 2000, vol. 1839, pp. 292–303.

V. Aleven, K. R. Koedinger, H. C. Sinclair, et J. Snyder, “Combating shallow learning in a tutor for geometry problem solving”, dans *Intelligent Tutoring Systems*, série Lecture Notes in Computer Science, B. P. Goettl, H. M. Halff, C. L. Redfield, et V. J. Shute, édés. San Antonio, Texas, USA : Springer Berlin Heidelberg, Août 1998, vol. 1452, pp. 364–373.

V. Aleven, O. Popescu, et K. R. Koedinger, “Pedagogical content knowledge in a tutorial dialogue system to support self-explanation”, dans *Papers of the AIED-2001 Workshop on Tutorial Dialogue Systems*, série Artificial Intelligence in Education (AIED), Mai 2001, pp. 59–70.

——, “Towards tutorial dialog to support self-explanation : Adding natural language understanding to a cognitive tutor”, dans *AI-ED in the Wired and Wireless Future, Proceedings of AI-ED 2001*, série Artificial Intelligence in Education (AIED), J. D. Moore, C. L. Redfield, et W. L. Johnson, édés. Amsterdam, Netherlands : IOS Press, 2001, pp. 246–255.

V. Aleven, B. McLaren, I. Roll, et K. Koedinger, “Toward meta-cognitive tutoring : A model of help seeking with a cognitive tutor”, *International Journal of Artificial Intelligence in Education*, vol. 16, no. 2, pp. 101–128, Avr. 2006.

V. Aleven, B. M. McLaren, J. Sewall, et K. R. Koedinger, “A new paradigm for intelligent tutoring systems : Example-tracing tutors”, *International Journal of Artificial Intelligence in Education*, vol. 19, no. 2, pp. 105–154, 2009.

V. A. Aleven et K. R. Koedinger, “An effective metacognitive strategy : Learning by doing and explaining with a computer-based cognitive tutor”, *Cognitive Science*, vol. 26, no. 2, pp. 147–179, Mars 2002.

M. Allard, “Les classes débordent !” *La Presse*, Nov. 2008.

R. Allen, C. Desmoulins, et L. Trilling, “Tuteurs intelligents et intelligence artificielle : Problèmes posés en construction de figures géométriques”, dans *Intelligent Tutoring Systems*, série Lecture Notes in Computer Science, C. Frasson, G. Gauthier, et G. I. McCalla, eds. Springer Berlin Heidelberg, Juin 1992, vol. 608, pp. 325–334.

S. A. Almouloud, “DEFI : Outil informatique de révélation du rôle de la figure et d’apprentissage de la démonstration au collège.” *Repères-IREM*, vol. 9, pp. 35–60, Oct. 1992.

J. R. Anderson, “ACT : A simple theory of complex cognition.” *American Psychologist*, vol. 51, no. 4, pp. 355–365, Avr. 1996.

J. R. Anderson et B. J. Reiser, “The LISP Tutor : It approaches the effectiveness of a human tutor”, *Byte*, vol. 10, no. 4, pp. 159–175, Avr. 1985.

J. R. Anderson et C. D. Schunn, “Implications of the ACT-R learning theory : No magic bullets”, dans *Educational Design and Cognitive Science*, série Advances in Instructional Psychology, R. Glaser, éd. Mahwah, NJ : Lawrence Erlbaum Associates, 2000, vol. 5, ch. 1, pp. 1–34.

J. R. Anderson, C. F. Boyle, et G. Yost, “The geometry tutor”, dans *Proceedings of the Ninth International Joint Conference on Artificial Intelligence*, vol. 1. Los Angeles, CA, USA : Morgan Kaufmann Publishers Inc., Août 1985, pp. 1–7.

J. R. Anderson, C. F. Boyle, A. T. Corbett, et M. W. Lewis, “Cognitive modeling and intelligent tutoring”, *Artificial Intelligence*, vol. 42, no. 1, pp. 7–49, Fév. 1990.

J. R. Anderson, D. Bothell, M. D. Byrne, S. Douglass, C. Lebiere, et Y. Qin, “An integrated theory of the mind”, *Psychological Review*, vol. 111, no. 4, pp. 1036–1060, 2004.

A. Anohina-Naumeca, “Advances in intelligent tutoring systems : Problem-solving modes and model of hints”, *International Journal of Computers, Communication and Control*, vol. 2, no. 1, pp. 48–55, 2007.

I. Arroyo, C. Beal, T. Murray, R. Waller, et B. P. Woolf, “Web-based intelligent multimedia tutoring for high stakes achievement tests”, dans *Intelligent Tutoring Systems*, série Lecture

Notes in Computer Science, J. C. Lester, R. M. Vicari, et F. Paraguaçu, édés. Springer Berlin Heidelberg, Sep. 2004, vol. 3220, pp. 468–477.

M. Artigue, *Didactique des mathématiques*. Lausanne : Delachaux et Niestlé, 1996, ch. Ingénierie didactique.

S. Autexier, D. Dietrich, et M. R. G. Schiller, “Towards an intelligent tutor for mathematical proofs”, dans *Proceedings First Workshop on CTP Components for Educational Software, THedu’11*, série EPTCS, P. Quaresma et R. Back, édés., vol. 79, Wroclaw, Poland, 2012, pp. 1–28.

N. Balacheff, “Processus de preuve et situations de validation”, *Educational Studies in Mathematics*, vol. 18, no. 2, pp. 147–176, Mai 1987.

N. Balacheff et C. Margolinas, “cKc modèle de connaissances pour le calcul de situations didactiques”, dans *Balises pour la Didactique des Mathématiques*, A. Mercier et C. Margolinas, édés. Grenoble, France : La Pensée Sauvage, 2005, pp. 75–106.

N. Balacheff, R. Caferra, M. Cerulli, N. Gaudin, M. Maracci, M. A. Mariotti, J.-P. Muller, J.-F. Nicaud, M. Ocello, F. Olivero, N. Peltier, S. Pesty, S. Soury-Lavergne, R. Sutherland, J. Trgalova, et C. Webber, *Baghera Assessment Project, Designing an Hybrid and Emergent Educational Society*, série Les Cahiers du laboratoire Leibniz, S. Soury-Lavergne, éd. Grenoble, France : Laboratoire Leibniz-IMAG, Avr. 2003, vol. 81.

T. Barnes et J. Stamper, “Automatic hint generation for logic proof tutoring using historical data”, *Journal of Educational Technology & Society*, vol. 13, no. 1, pp. 3–12, Jan. 2010.

Y. Baulac, “Un micromonde de géométrie, Cabri-Géomètre”, Thèse de doctorat, Université Joseph Fourier - Grenoble I, Fév. 1990.

J. Beck, M. Stern, et E. Haugsjaa, “Applications of ai in education”, *Crossroads*, vol. 3, no. 1, pp. 11–15, Sep. 1996.

L. Becker, M. Palmer, S. van Vuuren, et W. Ward, “Question ranking and selection in tutorial dialogues”, dans *Proceedings of the Seventh Workshop on Building Educational Applications Using NLP*. Stroudsburg, PA, USA : Association for Computational Linguistics, 2012, pp. 1–11.

C. Benz Müller, A. Fiedler, M. Gabsdil, H. Horacek, I. Kruijff-Korbayová, M. Pinkal, J. Siekmann, D. Tsovaltzi, B. Q. Vo, et M. Wolska., “Tutorial dialogs on mathematical proofs”,

dans *Proceedings of IJCAI-03 Workshop on Knowledge Representation and Automated Reasoning for E-Learning Systems*, P. Baumgartner, P. A. Cairns, M. Kohlhase, et E. Melis, édés., Acapulco, Mexico, 2003, pp. 12–22.

P. Bernat, “CHYPRE : Un logiciel d’aide au raisonnement”, *Repères-IREM*, vol. 10, pp. 25–46, 1993.

Y. Bertot, F. Guillot, et L. Pottier, “Visualizing geometrical statements with GeoView”, *Electronic Notes in Theoretical Computer Science*, vol. 103, pp. 49–65, Nov. 2004.

S. B. Blessing, S. B. Gilbert, S. Ourada, et S. Ritter, “Authoring model-tracing cognitive tutors”, *International Journal of Artificial Intelligence in Education*, vol. 19, no. 2, pp. 189–210, 2009.

F. Botana, M. Hohenwarter, P. Janičić, Z. Kovács, I. Petrović, T. Recio, et S. Weitzhofer, “Automated theorem proving in GeoGebra : Current achievements”, *Journal of Automated Reasoning*, vol. 55, no. 1, pp. 39–59, Juin 2015.

G. Brousseau, *Théorie des situations didactiques : Didactique des mathématiques 1970-1990*. Grenoble : La Pensée sauvage, 1998.

P. Brusilovsky, E. Schwarz, et G. Weber, “ELM-ART : An intelligent tutoring system on world wide web”, dans *Intelligent Tutoring Systems*, série Lecture Notes in Computer Science, C. Frasson, G. Gauthier, et A. Lesgold, édés. Montreal, Canada : Springer Berlin Heidelberg, Juin 1996, vol. 1086, pp. 261–269.

M. Buckley et C. Benz Müller, “A dialogue manager supporting natural language tutorial dialogue on proofs”, dans *Electronic Notes in Theoretical Computer Science*, Edinburgh, Scotland, Avr. 2005.

Cabri-Java Project, *Site web du Projet Cabri-Java*, 2004. En ligne : <http://www.cabri.net/cabrijava/>

CABRILOG SAS, *Site web de Cabri-Géomètre*, 2009. En ligne : <http://www.cabri.com/fr/>

S. Carberry, “Techniques for plan recognition”, *User Modeling and User-Adapted Interaction*, vol. 11, no. 1-2, pp. 31–48, Mars 2001.

Carnegie Mellon University, *Site web de Cognitive Tutor Authoring Tools (CTAT)*, 2015. En ligne : <http://ctat.pact.cs.cmu.edu>

F. Caron et S. René de Cotret, “Un regard didactique sur l’évaluation en mathématiques : genèse d’une perspective”, dans *Actes du colloque 2007 du Groupe des Didacticiens des Mathématiques du Québec*, Rimouski, QC, Juin 2007, pp. 123–134.

M. Chi et K. VanLehn, “Porting an intelligent tutoring system across domains”, dans *Artificial Intelligence in Education : Building Technology Rich Learning Contexts That Work*, série Frontiers in Artificial Intelligence and Applications, R. Luckin, K. R. Koedinger, et J. Greer, édés., vol. 158. Amsterdam, Netherlands : IOS Press, 2007, pp. 551–553.

S.-C. Chou, X.-S. Gao, et J.-Z. Zhang, “A deductive database approach to automated geometry theorem proving and discovering”, *Journal of Automated Reasoning*, vol. 25, no. 3, pp. 219–246, Oct. 2000.

K. Chrysafiadi et M. Virvou, “Student modeling approaches : A literature review for the last decade”, *Expert Systems with Applications*, vol. 40, no. 11, pp. 4715–4729, Sep. 2013.

J. claude Falmagne, E. Cosyn, J. paul Doignon, et N. Thiéry, “The assessment of knowledge, in theory and in practice”, dans *Formal Concept Analysis*, série Lecture Notes in Computer Science, R. Missaoui et J. Schmidt, édés., vol. 3874. Springer Berlin Heidelberg, 2006, pp. 61–79.

P. Cobo et J. M. Fortuny, “Social interactions and cognitive effects in contexts of area-comparison problem solving”, *Educational Studies in Mathematics*, vol. 42, no. 2, pp. 115–140, Mars 2000.

P. Cobo, J. M. Fortuny, E. Puertas, et P. R. Richard, “AgentGeom : a multiagent system for pedagogical support in geometric proof problems”, *International Journal of Computers for Mathematical Learning*, vol. 12, no. 1, pp. 57–79, 2007.

C. Conati et K. VanLehn, “Teaching meta-cognitive skills : Implementation and evaluation of a tutoring system to guide self-explanation while learning from examples”, dans *Proceedings of AIED ‘99, 9th World Conference of Artificial Intelligence and Education*, S. P. Lajoie et M. Vivet, édés. Amsterdam, Netherlands : IOS Press, 1999, pp. 297–304.

C. Conati, A. Gertner, et K. VanLehn, “Using bayesian networks to manage uncertainty in student modeling”, *User Modeling and User-Adapted Interaction*, vol. 12, no. 4, pp. 371–417, Nov. 2002.

Coq Development Team, *Site web de Coq Proof Assistant*, 2015. En ligne : <https://coq.inria.fr>

A. Corbett, M. McLaughlin, et K. C. Scarpinato, “Modeling student knowledge : Cognitive tutors in high school and college”, *User Modeling and User-Adapted Interaction*, vol. 10, no. 2-3, pp. 81–108, Juin 2000.

Coursera Inc., *Site web de Coursera*, 2015. En ligne : <https://www.coursera.org>

S. Coutat et P. R. Richard, “Les figures dynamiques dans un espace de travail mathématique pour l’apprentissage des propriétés géométriques”, *Annales de didactique et de sciences cognitives*, vol. 16, pp. 97–126, 2011.

M. C. Desmarais, “Just-in-time knowledge for effective hybrid learning”, dans *Hybrid Learning and Education*, série Lecture Notes in Computer Science, J. Fong, R. Kwan, et F. L. Wang, édés. Springer Berlin Heidelberg, Août 2008, vol. 5169, pp. 31–39.

M. C. Desmarais et R. S. J. d. Baker, “A review of recent advances in learner and skill modeling in intelligent learning environments”, *User Modeling and User-Adapted Interaction*, vol. 22, no. 1-2, pp. 9–38, Avr. 2012.

M. C. Desmarais, L. Giroux, et S. Larochelle, “An advice-giving interface based on plan-recognition and user-knowledge assessment”, *International Journal of Man-Machine Studies*, vol. 39, no. 6, pp. 901–924, Déc. 1993.

C. Desmoulins, “Étude et réalisation d’un système tuteur pour la construction de figures géométriques”, Thèse de doctorat, Université Joseph Fourier - Grenoble I, Fév. 1994.

DFKI & Saarland University, *Site web de ActiveMath*, 2007. En ligne : <http://www.activemath.org>

R. Duval, *Sémiosis et pensée humaine : Régistres sémiotiques et apprentissages intellectuels*, série Exploration. Berne : Peter Lang, 1995.

M. Dzikovska, N. Steinhauser, E. Farrow, J. Moore, et G. Campbell, “BEETLE II : Deep natural language understanding and automatic feedback generation for intelligent tutoring in basic electricity and electronics”, *International Journal of Artificial Intelligence in Education*, vol. 24, no. 3, pp. 284–332, Sep. 2014.

S. D’Mello et A. Graesser, “Design of dialog-based intelligent tutoring systems to simulate human-to-human tutoring”, dans *Where Humans Meet Machines*, A. Neustein et J. A. Markowitz, édés. Springer New York, 2013, ch. 11, pp. 233–269.

edX Inc., *Site web de edX*, 2015. En ligne : <https://www.edx.org>

- S. El-Khoury, P. R. Richard, E. Aïmeur, et J. M. Fortuny, “Development of an intelligent tutorial system to enhance students’ mathematical competence in problem solving”, dans *Proceedings of E-Learn : World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education 2005*, G. Richards, éd. Vancouver, Canada : Association for the Advancement of Computing in Education (AACE), Oct. 2005, pp. 2042–2049.
- C. Elliott, J. Rickel, et J. Lester, “Lifelike pedagogical agents and affective computing : An exploratory synthesis”, dans *Artificial Intelligence Today*, série Lecture Notes in Computer Science, M. J. Wooldridge et M. Veloso, édés. Springer Berlin Heidelberg, 1999, vol. 1600, pp. 195–211.
- M. Feng, N. Heffernan, et K. Koedinger, “Student modeling in an intelligent tutoring system”, dans *Intelligent Tutoring Systems in E-Learning Environments : Design, Implementation and Evaluation*, S. Stankov, V. Glavinic, et M. Rosic, édés. Hershey, PA, USA : IGI Global, 2011, pp. 208–236.
- D. Fossati, B. D. Eugenio, S. Ohlsson, C. Brown, et L. Chen, “Data driven automatic feedback generation in the iList intelligent tutoring system”, *Technology, Instruction, Cognition and Learning*, vol. 10, no. 1, pp. 5—26, 2015.
- Y. Gal, S. Reddy, S. M. Shieber, A. Rubin, et B. J. Grosz, “Plan recognition in exploratory domains”, *Artificial Intelligence*, vol. 176, no. 1, pp. 2270–2290, Jan. 2012.
- X.-S. Gao et Q. Lin, “MMP/Geometer - a software package for automated geometric reasoning”, dans *Automated Deduction in Geometry*, série Lecture Notes in Computer Science, F. Winkler, éd. Springer Berlin Heidelberg, 2004, vol. 2930, pp. 44–66.
- L.-M. Gervais, “Mathématiques : l’hypothèse du pragmatisme”, *Le Devoir*, Juin 2011.
- S. B. Gilbert, S. B. Blessing, et E. Guo, “Authoring effective embedded tutors : An overview of the extensible problem specific tutor (xPST) system”, *International Journal of Artificial Intelligence in Education*, vol. 25, no. 3, pp. 428–454, Sep. 2015.
- B. G. Glaser et A. L. Strauss, *The Discovery of Grounded Theory. Strategies for Qualitative Research*. Chicago : Aldine, 1967.
- A. C. Graesser, K. VanLehn, C. P. Rosé, P. W. Jordan, et D. Harter, “Intelligent tutoring systems with conversational dialogue”, *AI Magazine*, vol. 22, no. 4, pp. 39–51, Oct. 2001.
- A. C. Graesser, S. Lu, G. T. Jackson, H. H. Mitchell, M. Ventura, A. Olney, et M. M.

Louwerse, “AutoTutor : A tutor with dialogue in natural language”, *Behavior Research Methods, Instruments, & Computers*, vol. 36, no. 2, pp. 180–192, Mai 2004.

R. Gras, “Aide logicielle aux problèmes de démonstration géométriques dans l’enseignement secondaire”, *Petit x*, vol. 17, pp. 65–83, 1988.

J. Gressier, *Site web de Géométrie*, 2015. En ligne : <http://geometrix.free.fr/site/>

D. Guin, M.-A. Egret, B. Koch, G. Kuntz, G. Métivier, et N. Vogel, “Modélisation de la démonstration géométrique dans Geometry Tutor”, *Annales de Didactique et de Sciences Cognitives*, vol. 4, pp. 5–40, 1991.

M. Hohenwarter, “GeoGebra 4.4 – from desktops to tablets”, *Indagatio Didactica*, vol. 5, no. 1, pp. 8–18, Juil. 2013.

M. Hohenwarter et K. Fuchs, “Combination of dynamic geometry, algebra and calculus in the software system GeoGebra”, dans *Computer Algebra Systems and Dynamic Geometry Systems in Mathematics Teaching Conference 2004*, Pecs, Hungary, Juil. 2004, pp. 128–133.

E. Horvitz, J. Breese, D. Heckerman, D. Hovel, et K. Rommelse, “The Lumière project : Bayesian user modeling for inferring the goals and needs of software users”, dans *Proceedings of the Fourteenth Conference on Uncertainty in Artificial Intelligence*, série UAI’98. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1998, pp. 256–265.

M. Hvidsten, *Site web de Geometry Explorer*, 2005. En ligne : <http://homepages.gac.edu/~hvidsten/explorer/>

International GeoGebra Institute, *Site web de GeoGebra*, 2015. En ligne : <http://www.geogebra.org>

S. Isotani et L. de Oliveira Brandão, “An algorithm for automatic checking of exercises in a dynamic geometry system : iGeom”, *Computers & Education*, vol. 51, no. 3, pp. 1283–1303, Nov. 2008.

P. Janičić et P. Quaresma, “System description : GCLCprover + GeoThms”, dans *Automated Reasoning*, série Lecture Notes in Computer Science, U. Furbach et N. Shankar, éd. Springer Berlin Heidelberg, 2006, vol. 4130, pp. 145–150.

S. Jean-Daubias, “Un système d’assistance au diagnostic de compétences en algèbre élémentaire”, *Sciences et Techniques Educatives*, vol. 9, no. 1.2, pp. 171–200, 2002.

——, “PÉPITE : un système d’assistance au diagnostic de compétences”, Thèse de doctorat, Université du Maine, Le Mans, FRANCE, Jan. 2000.

P. W. Jordan, B. Hall, M. Ringenberg, Y. Cue, et C. Rosé, “Tools for authoring a dialogue agent that participates in learning studies”, dans *Proceedings of the 2007 Conference on Artificial Intelligence in Education : Building Technology Rich Learning Contexts That Work*, série Frontiers in Artificial Intelligence and Applications, R. Luckin, K. R. Koedinger, et J. Greer, édés., vol. 158. Amsterdam, Netherlands : IOS Press, 2007, pp. 43–50.

H. A. Kautz, “A formal theory of plan recognition”, Thèse de doctorat, University of Rochester, Rochester, NY, USA, Mai 1987.

——, “A formal theory of plan recognition and its implementation”, dans *Reasoning About Plans*, R. J. Brachman, J. F. Allen, H. A. Kautz, R. N. Pelavin, et J. D. Tenenbergs, édés. San Francisco, CA, USA : Morgan Kaufmann Publishers Inc., 1991, ch. 2, pp. 69–124.

Khan Academy, *Site web de Khan Academy*, 2015. En ligne : <https://www.khanacademy.org>

K. R. Koedinger, “Tutoring concepts, percepts, and rules in geometry problem-solving”, Thèse de doctorat, Carnegie Mellon University, Pittsburgh, PA, USA, 1991.

K. R. Koedinger et J. R. Anderson, “Abstract planning and perceptual chunks : Elements of expertise in geometry”, *Cognitive Science*, vol. 14, no. 4, pp. 511–550, 1990.

——, “Reifying implicit planning in geometry : Guidelines for model-based intelligent tutoring system design”, dans *Computers as cognitive tools*, S. P. Lajoie et S. J. Derry, édés. Hillsdale, NJ, USA : Lawrence Erlbaum Associates, 1993, pp. 15–46.

K. R. Koedinger, J. R. Anderson, W. H. Hadley, et M. A. Mark, “Intelligent tutoring goes to school in the big city”, *International Journal of Artificial Intelligence in Education (IJAIED)*, vol. 8, pp. 30–43, 1997.

K. R. Koedinger, V. Aleven, N. Heffernan, B. McLaren, et M. Hockenberry, “Opening the door to non-programmers : Authoring intelligent tutor behavior by demonstration”, dans *Intelligent Tutoring Systems*, série Lecture Notes in Computer Science, J. C. Lester, R. M. Vicari, et F. Paraguaçu, édés. Springer Berlin Heidelberg, 2004, vol. 3220, pp. 162–174.

M. Kordaki et A. Mastrogiannis, “The potential of multiple-solution tasks in e-learning environments : Exploiting the tools of Cabri Geometry II”, dans *Proceedings of E-Learn :*

World Conference on E-Learning in Corporate, Government, Healthcare, and Higher Education 2006, T. Reeves et S. Yamashita, éd. Honolulu, Hawaii, USA : Association for the Advancement of Computing in Education (AACE), Oct. 2006, pp. 97–104.

U. Kortenkamp, “Foundations of dynamic geometry”, Thèse de doctorat, Swiss Federal Institute of Technology, Zurich, Switzerland, 1999.

——, *Site web de Cinderella*, 2013. En ligne : <http://www.cinderella.de>

U. Kortenkamp et J. Richter-Gebert, “Using automatic theorem proving to improve the usability of geometry software”, dans *Proceedings of Mathematical User Interfaces 2004*, P. Libbrecht, éd., 2004, p. 12.

A. Kuzniak, “Paradigmes et espaces de travail géométriques. Éléments d’un cadre théorique pour l’enseignement et la formation des enseignants en géométrie”, *Canadian Journal of Science, Mathematics and Technology Education*, vol. 6, no. 2, pp. 167–187, 2006.

——, “L’espace de travail mathématique et ses genèses”, *Annales de Didactique et de Sciences Cognitives*, vol. 16, pp. 9–24, 2011.

I. Lakatos, *Preuves et Réfutations. Essai sur la Logique de la Découverte Mathématique*. Paris, France : Hermann, 1984.

L. Lesta et K. Yacef, “An intelligent teaching assistant system for logic”, dans *Intelligent Tutoring Systems*, série Lecture Notes in Computer Science, S. A. Cerri, G. Gouardères, et F. Paragauçu, éd. Springer Berlin Heidelberg, Juin 2002, vol. 2363, pp. 421–431.

B. Li et L. Li, “Elementary algebra proof exercises using a theorem proving system”, dans *Practical Applications of Intelligent Systems*, série Advances in Intelligent and Soft Computing, Y. Wang et T. Li, éd. Springer Berlin Heidelberg, 2012, vol. 124, no. Part III, pp. 275–280.

N. Li, N. Matsuda, W. W. Cohen, et K. R. Koedinger, “Integrating representation learning and skill learning in a human-like intelligent agent”, *Artificial Intelligence*, vol. 219, pp. 67–91, Fév. 2015.

S.-H. Liao, P.-H. Chu, et P.-Y. Hsiao, “Data mining techniques and applications – a decade review from 2000 to 2011”, *Expert Systems with Applications*, vol. 39, no. 12, pp. 11 303–11 311, Sep. 2012.

V. Luengo, “Cabri-Euclide : un micromonde de preuve intégrant la réfutation. principes didactiques et informatiques. réalisation”, Thèse de doctorat, Université Joseph Fourier - Grenoble I, Grenoble, France, Sep. 1997.

——, “Some didactical and epistemological considerations in the design of educational software : The Cabri-Euclide example”, *International Journal of Computers for Mathematical Learning*, vol. 10, no. 1, pp. 1–29, 2005.

V. Luengo et N. Balacheff, “Contraintes informatiques et environnements d’apprentissage de la démonstration en géométrie”, *Sciences et Techniques Educatives*, vol. 5, no. 1, pp. 15–45, 1998.

C. J. MacLellan, E. S. Wiese, N. Matsuda, et K. R. Koedinger, “SimStudent : Authoring expert models by tutoring”, dans *Proceedings of the 2nd Annual GIFT Users Symposium*, série Adaptive Tutoring, R. A. Sottolare, éd. Orlando, FL, USA : U.S. Army Research Laboratory, Juin 2015, pp. 25–32.

M. Macrelle et C. Desmoulins, “Macro-definitions, a basic component for interoperability between iles at the knowledge level : Application to geometry”, dans *Intelligent Tutoring Systems*, série Lecture Notes in Computer Science, B. P. Goettl, H. M. Halff, C. L. Redfield, et V. J. Shute, éd. Springer Berlin Heidelberg, Août 1998, vol. 1452, pp. 46–55.

Mathway, *Site web de Mathway*, 2015. En ligne : <http://www.mathway.com>

N. Matsuda, “The impact of different proof strategies on learning geometry theorem proving”, Thèse de doctorat, University of Pittsburgh, Pittsburgh, PA, USA, 2004.

N. Matsuda et K. VanLehn, “GRAMY : A geometry theorem prover capable of construction”, *Journal of Automated Reasoning*, vol. 32, no. 1, pp. 3–33, 2004.

——, “Modeling hinting strategies for geometry theorem proving”, dans *User Modeling 2003*, série Lecture Notes in Computer Science, P. Brusilovsky, A. Corbett, et F. de Rosis, éd. Johnstown, PA, USA : Springer Berlin Heidelberg, Juin 2003, vol. 2702, pp. 373–377.

——, “Advanced Geometry Tutor : An intelligent tutor that teaches proof-writing with construction”, dans *Artificial Intelligence in Education : Supporting Learning through Intelligent and Socially Informed Technology*, série Frontiers in Artificial Intelligence and Applications, C.-K. Looi, G. McCalla, B. Bredeweg, et J. Breuker, éd., vol. 125. Amsterdam, Netherlands : IOS Press, 2005, pp. 443–450.

N. Matsuda, W. W. Cohen, et K. R. Koedinger, “Teaching the teacher : Tutoring simstudent leads to more effective cognitive tutor authoring”, *International Journal of Artificial Intelligence in Education*, vol. 25, no. 1, pp. 1–34, Mars 2015.

McGraw-Hill Education, *Site web de Geometer’s Sketchpad*, 2014. En ligne : <http://www.dynamicgeometry.com>

McGraw-Hill Global Education Holdings, LLC., *Site web de ALEKS*, 2015. En ligne : <http://www.aleks.com/>

E. Melis, G. Gogvadze, P. Libbrecht, et C. Ullrich, “ActiveMath – a learning platform with semantic web features”, dans *Semantic Web Technologies for e-Learning*, série The Future of Learning, D. Dicheva, R. Mizoguchi, et J. Greer, éd. Amsterdam, Netherlands : IOS Press, 2009, vol. 4, ch. 9, pp. 159–177.

Ministère de l’Éducation, du Loisir et du Sport, *Programme de formation de l’école québécoise : Enseignement secondaire, deuxième cycle*. Montréal : Gouvernement du Québec, 2009, ch. 6 : Mathématique, p. 152.

———, *Cadre d’évaluation des apprentissages : Mathématique*, Ministère de l’Éducation, du Loisir et du Sport, éd. Montréal : Gouvernement du Québec, 2011.

A. Mitrovic, “Fifteen years of constraint-based tutors : What we have achieved and where we are going”, *User Modeling and User-Adapted Interaction*, vol. 22, no. 1-2, pp. 39–72, Avr. 2012.

A. Mitrovic, B. Martin, et M. Mayo, “Using evaluation to shape its design : Results and experiences with SQL-Tutor”, *User Modeling and User-Adapted Interaction*, vol. 12, no. 2-3, pp. 243–279, Juin 2002.

A. Mitrovic, B. Martin, P. Suraweera, K. Zakharov, N. Milik, J. Holland, et N. McGuigan, “ASPIRE : An authoring system and deployment environment for constraint-based tutors”, *International Journal of Artificial Intelligence in Education*, vol. 19, no. 2, pp. 155–188, 2009.

Moodle Pty Ltd, *Site web de Moodle*, 2015. En ligne : <https://moodle.org>

B. Mostafavi, T. Barnes, et M. Croy, “Automatic generation of proof problems in deductive logic”, dans *Proceedings of the 4th International Conference on Educational Data Mining (EDM 2011)*, Eindhoven, Netherlands, Juil. 2011, pp. 289—294.

T. Murray, “Eon : Authoring tools for content, instructional strategy, student model and interface design”, dans *Authoring Tools for Advanced Technology Learning Environments*, T. Murray, S. B. Blessing, et S. Ainsworth, édés. Netherlands : Springer Netherlands, 2003, vol. Section III, ch. 11, pp. 309–339.

J. Narboux, *Site web de GeoProof*, 2006. En ligne : <http://home.gna.org/geoproof/>

——, “A graphical user interface for formal proofs in geometry”, *Journal of Automated Reasoning*, vol. 39, no. 2, pp. 161–180, Août 2007.

B. D. Nye, A. C. Graesser, et X. Hu, “AutoTutor and family : A review of 17 years of natural language tutoring”, *International Journal of Artificial Intelligence in Education*, vol. 24, no. 4, pp. 427–469, Déc. 2014.

A. M. Olney, “GnuTutor : An open source intelligent tutoring system based on AutoTutor”, dans *AAAI Fall Symposium Series*, 2009, pp. 70–75.

A. M. Olney, A. C. Graesser, et N. K. Person, “Tutorial dialog in natural language”, dans *Advances in Intelligent Tutoring Systems*, série Studies in Computational Intelligence, R. Nkambou, J. Bourdeau, et R. Mizoguchi, édés. Springer Berlin Heidelberg, 2010, vol. 308, ch. 9, pp. 181–206.

P. Paillé et A. Mucchielli, *L’Analyse Qualitative en Sciences Humaines et Sociales*, 3e éd., série Collection U : Sciences Humaines & Sociales. Paris, France : Armand Colin, 2012.

J. J. Paneque, P. Cobo, et J. M. Fortuny, “Intelligent tutoring and the development of argumentative competence”, *Technology, Knowledge and Learning*, pp. 1–22, Mai 2016.

S. Papert, *Mindstorms : Children, Computers, and Powerful Ideas*. New York, NY, USA : Basic Books, Inc., 1980.

L. Paquette, “Un modèle pour la génération d’indices par une plateforme de tuteurs par traçage de modèle”, Thèse de doctorat, Université de Sherbrooke, Sherbrooke, QC, CA, Juil. 2013.

L. Paquette, J.-F. Lebeau, et A. Mayers, “Authoring problem-solving tutors : A comparison between ASTUS and CTAT”, dans *Advances in Intelligent Tutoring Systems*, série Studies in Computational Intelligence, R. Nkambou, J. Bourdeau, et R. Mizoguchi, édés. Springer Berlin Heidelberg, 2010, vol. 308, ch. 19, pp. 377–405.

L. Paquette, J.-F. Lebeau, G. Beaulieu, et A. Mayers, “Automating next-step hints generation using ASTUS”, dans *Intelligent Tutoring Systems*, série Lecture Notes in Computer Science, S. Cerri, W. Clancey, G. Papadourakis, et K. Panourgia, eds. Springer Berlin Heidelberg, 2012, vol. 7315, pp. 201–211.

——, “Designing a knowledge representation approach for the generation of pedagogical interventions by mths”, *International Journal of Artificial Intelligence in Education*, vol. 25, no. 1, pp. 118–156, Mars 2015.

A. Peña-Ayala, “Educational data mining : A survey and a data mining-based analysis of recent works”, *Expert Systems with Applications*, vol. 41, no. 4, Part 1, pp. 1432–1462, Mars 2014.

N. Pinkwart, V. Aleven, K. Ashley, et C. Lynch, “Toward legal argument instruction with graph grammars and collaborative filtering techniques”, dans *Intelligent Tutoring Systems*, série Lecture Notes in Computer Science, M. Ikeda, K. D. Ashley, et T.-W. Chan, eds. Springer Berlin Heidelberg, 2006, vol. 4053, pp. 227–236.

N. Pinkwart, K. Ashley, C. Lynch, et V. Aleven, “Evaluating an intelligent tutoring system for making legal arguments with hypotheticals”, *International Journal of Artificial Intelligence in Education*, vol. 19, no. 4, pp. 401–424, Déc. 2009.

D. Py, “Reconnaissance de plan pour la modélisation de l’élève : le projet Mentoniez”, *Recherches en Didactique des Mathématiques*, vol. 14, no. 1.2, pp. 113–138, 1994.

——, “Aide à la démonstration en géométrie : le projet Mentoniez”, *Sciences et Techniques Educatives*, vol. 3, no. 2, pp. 227–256, 1996.

——, “Environnements interactifs d’apprentissage et démonstration en géométrie”, Université de Rennes I, Habilitation à Diriger des Recherches, Juil. 2001.

P. Rabardel, *Les Hommes et les Technologies ; Une Approche Cognitive des Instruments Contemporains*. Armand Colin, 1995.

M. A. Rau, “Conceptual learning with multiple graphical representations : Intelligent tutoring systems support for sense-making and fluency-building processes”, Thèse de doctorat, Carnegie Mellon University, Pittsburgh, PA USA, Août 2013.

M. A. Rau, V. Aleven, N. Rummel, et S. Rohrbach, “Sense making alone doesn’t do it : Fluency matters too! its support for robust learning with multiple representations”, dans

Intelligent Tutoring Systems, série Lecture Notes in Computer Science, S. A. Cerri, W. J. Clancey, G. Papadourakis, et K. Panourgia, édés. Springer Berlin Heidelberg, 2012, vol. 7315, pp. 174–184.

P. R. Richard, “L’inférence figurale : Un pas de raisonnement discursivo-graphique”, *Educational Studies in Mathematics*, vol. 57, no. 2, pp. 229–263, Sep. 2004.

——, *Raisonnement et stratégies de preuve dans l’enseignement des mathématiques*, série Exploration. Berne : Peter Lang, 2004, vol. 125.

P. R. Richard et J. M. Fortuny, “Amélioration des compétences argumentatives à l’aide d’un système tutoriel en classe de mathématique au secondaire”, *Annales de Didactique et de Sciences Cognitives*, vol. 12, pp. 83–116, 2007.

P. R. Richard, J. M. Fortuny, M. Gagnon, N. Leduc, E. Puertas, et M. Tessier-Baillargeon, “Didactic and theoretical-based perspectives in the experimental development of an intelligent tutorial system for the learning of geometry”, *ZDM Mathematics Education*, vol. 43, no. 3, pp. 425–439, Juil. 2011.

P. R. Richard, M. Gagnon, et J. M. Fortuny, “Gestion interactive de problèmes en géométrie pour le développement des compétences des Élèves et l’acquisition du savoir mathématique”, dans *Actes Quatrième Symposium Espaces de Travail Mathématique (ETM4)*, I. M. Gomez-Chacon, J. Escribano, A. Kuzniak, et P. R. Richard, édés., San Lorenzo de El Escorial, Madrid, Espagne, Juin 2014, pp. 497–517.

S. Ritter, B. Towle, R. C. Murray, R. G. M. Hausmann, et J. Connelly, “A cognitive tutor for geometric proof”, dans *Intelligent Tutoring Systems*, série Lecture Notes in Computer Science, V. Alevén, J. Kay, et J. Mostow, édés. Springer Berlin Heidelberg, 2010, vol. 6095, pp. 453–453.

I. Roll, R. S. J. d. Baker, V. Alevén, et K. R. Koedinger, “On the benefits of seeking (and avoiding) help in online problem-solving environments”, *Journal of the Learning Sciences*, vol. 23, no. 4, pp. 537–560, Nov. 2014.

C. Romero et S. Ventura, “Educational data mining : A survey from 1995 to 2005”, *Expert Systems with Applications*, vol. 33, no. 1, pp. 135–146, Juil. 2007.

C. P. Rosé, P. Jordan, M. Ringenber, S. Siler, K. VanLehn, et A. Weinstein, “Interactive conceptual tutoring in Atlas-Andes”, dans *AI-ED in the Wired and Wireless Future, Procees-*

dings of AI-ED 2001, série Artificial Intelligence in Education (AIED), J. D. Moore, C. L. Redfield, et W. L. Johnson, édés. Amsterdam, Netherlands : IOS Press, 2001, pp. 256–266.

K. E. Sabo, R. K. Atkinson, A. L. Barrus, S. S. Joseph, et R. S. Perez, “Searching for the two sigma advantage : Evaluating algebra intelligent tutors”, *Computers in Human Behavior*, vol. 29, no. 4, pp. 1833–1840, Juil. 2013.

R. Scheines et W. Sieg, “Computer environments for proof construction”, *Interactive Learning Environments*, vol. 4, no. 2, pp. 159–169, 1994.

O. Scheuer, S. Niebuhr, T. Dragon, B. M. McLaren, et N. Pinkwart, “Adaptive support for graphical argumentation - the LASAD approach”, *IEEE Learning Technology Newsletter*, vol. 14, no. 1, pp. 8–11, 2012.

W. Sieg, “The AProS project : Strategic thinking & computational logic”, *Logic Journal of IGPL*, vol. 15, no. 4, pp. 359–368, 2007.

K. Sijtsma et B. W. Junker, “Item response theory : Past performance, present developments, and future expectations”, *Behaviormetrika*, vol. 33, no. 1, pp. 75–102, 2006.

J. C. Stamper, “Automatic generation of intelligent tutoring capabilities via educational data mining.” Thèse de doctorat, University of North Carolina at Charlotte, Charlotte, NC, USA, 2010.

Succès Scolaire, *Site web de Succès Scolaire*, 2014. En ligne : <http://www.successscolaire.ca>

G. Sukthankar, R. P. Goldman, C. Geib, D. V. Pynadath, et H. H. Bui, “An introduction to plan, activity, and intent recognition”, dans *Plan, Activity, and Intent Recognition*, G. Sukthankar, R. Goldman, C. Geib, D. Pynadath, et H. Bui, édés. Philadelphia, PA, USA : Elsevier, 2014, p. 22.

J. Tadinada, “Interactive tutoring system for high school geometry”, Mémoire de maîtrise, Indian Institute of Technology, Bombay, India, Juin 2011.

D. Tall, “The psychology of advanced mathematical thinking”, dans *Advanced Mathematical Thinking*, série Mathematics Education Library, D. Tall, éd. Springer Netherlands, 1991, vol. 11, pp. 3–21.

D. Tanguay, “Learning proof : From truth towards validity”, dans *Proceedings of the Xth Conference on Research in Undergraduate Mathematics Education (RUME)*. San Diego,

CA, USA : San Diego State University, 2007, p. 15.

—, “Apprentissage de la démonstration et graphes orientés”, *Annales de Didactique et de Sciences Cognitives*, vol. 10, pp. 55–93, 2005.

M. Tessier-Baillargeon, “GeoGebraTUTOR : Modélisation d’un système tutoriel autonome pour l’accompagnement d’élèves en situation de résolution de problèmes de démonstration en géométrie plane et genèse d’un espace de travail géométrique idoine”, Thèse de doctorat, Université de Montréal, Montréal, Mars 2016.

M. Tessier-Baillargeon, N. Leduc, et P. R. Richard, “Niveaux d’intervention enseignante pour le développement d’un système tutoriel : Une expérience didactique à l’École secondaire avec le système GéogebraTUTOR”, dans *Actes des Journées Mathématiques de l’Institut Français de l’Éducation (IFÉ)*, L. Trouche, H. Chaachoua, M. Hersant, Y. Matheron, et G. Psycharis, édés. Lyon, France : École Normale Supérieure (ENS) de Lyon, Juin 2011, pp. 201–208.

M. Tessier-Baillargeon, P. R. Richard, N. Leduc, et M. Gagnon, “Conception et analyse de GeoGebraTUTOR, un système tutoriel intelligent : Genèse d’un espace de travail géométrique idoine”, *Revista Latinoamericana de Investigacion en Matematica Educativa (Relime)*, vol. 17, no. 4-II, pp. 303–326, Déc. 2014.

The Teaching Company, LLC. , *Site web de The Great Courses*, 2015. En ligne : <http://www.thegreatcourses.com>

D. Tsovaltzi, “MENON : Automating a socratic teaching model for mathematical proofs”, Thèse de doctorat, Universität des Saarlandes, Saarbrücken, Mars 2010.

Tutor.com, Inc., *Site web de Tutor.com*, 2015. En ligne : <http://www.tutor.com>

University of Cambridge et Technische Universität München, *Site web de Isabelle*, 2015. En ligne : <https://www.cl.cam.ac.uk/research/hvg/Isabelle/>

F. Vailles, “Plus d’élèves par classe, vraiment ?” *La Presse*, Jan. 2015.

K. VanLehn, “The behavior of tutoring systems”, *International Journal of Artificial Intelligence in Education*, vol. 16, no. 3, pp. 227–265, Août 2006.

K. VanLehn, D. Bhembé, M. Chi, C. Lynch, K. Schulze, R. Shelby, L. Taylor, D. Treacy, A. Weinstein, et M. Wintersgill, “Implicit versus explicit learning of strategies in a non-procedural cognitive skill”, dans *Intelligent Tutoring Systems*, série Lecture Notes in Com-

puter Science, J. C. Lester, R. Vicari, et F. Paraguaçu, édés. Springer Berlin Heidelberg, 2004, vol. 3220, pp. 521–530.

K. VanLehn, C. Lynch, K. Schulze, J. A. Shapiro, R. Shelby, L. Taylor, D. Treacy, A. Weinstein, et M. Wintersgill, “The Andes physics tutoring system : Lessons learned”, *International Journal of Artificial Intelligence in Education*, vol. 15, no. 3, pp. 147–204, Août 2005.

K. VanLehn, B. van de Sande, R. Shelby, et S. Gershman, “The Andes physics tutoring system : An experiment in freedom”, dans *Advances in Intelligent Tutoring Systems*, série Studies in Computational Intelligence, R. Nkambou, J. Bourdeau, et R. Mizoguchi, édés. Springer Berlin Heidelberg, 2010, vol. 308, pp. 421–443.

L. S. Vygotsky, *Mind in Society : The Development of Higher Psychological Processes*, M. Cole, V. John-Steiner, S. Scribner, et E. Souberman, édés. Cambridge, MA, USA : Harvard University Press, 1978.

W. Ward, R. Cole, D. Bolaños, C. Buchenroth-Martin, E. Svirsky, S. V. Vuuren, T. Weston, J. Zheng, et L. Becker, “My Science Tutor : A conversational multimedia virtual tutor for elementary school science”, *ACM Transactions on Speech and Language Processing*, vol. 7, no. 4, pp. 18 :1–18 :29, Août 2011.

C. Webber, L. Bergia, S. Pesty, et N. Balacheff, “Baghera project : a multi-agent architecture for human learning”, dans *Workshop - Multi-Agent Architectures for Distributed Learning Environments.*, série Proceedings of the International Conference on AI and Education (AIED), San Antonio, TX, USA, 2001, pp. 12–17.

G. Weber, *Site web de ELM-ART*, 2014. En ligne : <http://art2.ph-freiburg.de/Lisp-Course>

G. Weber et P. Brusilovsky, “ELM-ART : An adaptive versatile system for web-based instruction”, *International Journal of Artificial Intelligence in Education (IJAIED)*, vol. 12, pp. 351–384, 2001.

A. Weerasinghe, “A general model of adaptive tutorial dialogues for intelligent tutoring systems”, Thèse de doctorat, University of Canterbury, Christchurch, New Zeland, 2013.

M. Wenzel, *The Isabelle/Isar Reference Manual*, Avr. 2015.

S. Wilson et J. D. Fleuriot, “Combining dynamic geometry, automated geometry theorem proving and diagrammatic proofs”, dans *Proceedings of the European Joint Conferences*

on Theory and Practice of Software (ETAPS) Satellite Workshop on User Interfaces for Theorem Provers (UITP). Edinburgh : Springer, 2005.

D. Winterstein, A. Bundy, et C. Gurr, “Dr.Doodle : A diagrammatic theorem prover”, dans *Automated Reasoning*, série Lecture Notes in Computer Science, D. Basin et M. Rusinowitch, édts. Springer Berlin Heidelberg, 2004, vol. 3097, pp. 331–335.

C. R. Wolfe, C. L. Widmer, V. F. Reyna, X. Hu, E. M. Cedillos, C. R. Fisher, P. G. Brust-Renck, T. C. Williams, I. Damas Vannucchi, et A. M. Weil, “The development and analysis of tutorial dialogues in AutoTutor Lite”, *Behavior Research Methods*, vol. 45, no. 3, pp. 623–636, Sep. 2013.

W.-K. Wong, S.-K. Yin, H.-H. Yang, et Y.-H. Cheng, “Using computer-assisted multiple representations in learning geometry proofs”, *Journal of Educational Technology & Society*, vol. 14, no. 3, pp. 43–54, Juil. 2011.

Yahoo! Inc., *Site web de Yahoo! Answers*, 2015. En ligne : <http://answers.yahoo.com>

Z. Ye, S.-C. Chou, et X.-S. Gao, “An introduction to Java Geometry Expert”, dans *Automated Deduction in Geometry*, série Lecture Notes in Computer Science, T. Sturm et C. Zengler, édts. Springer Berlin Heidelberg, 2011, vol. 6301, pp. 189–195.

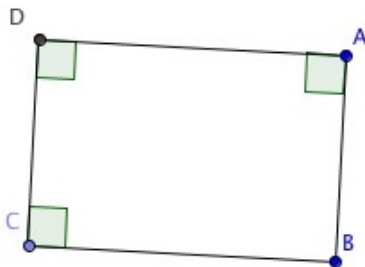
YouTube, LLC, *Site web de YouTube*, 2015. En ligne : <https://www.youtube.com>

ANNEXE A Problèmes implantés dans QEDX

Cette annexe contient le libellé des cinq problèmes implantés dans QED-Tutrix (QEDX). L'encodage inscrit entre parenthèses, qui est rattaché à chacun, est utilisé à titre de référence dans les différents tableaux présentés dans cette thèse.

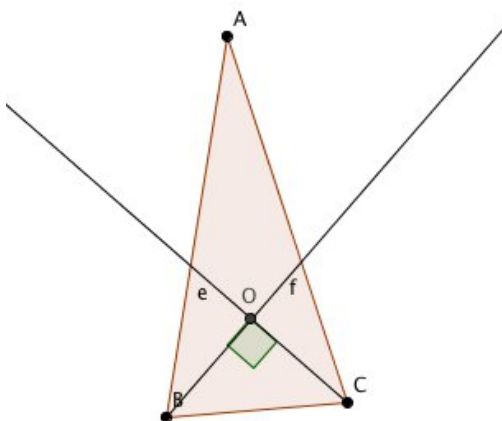
A.1 Preuve du rectangle (Richard et Fortuny, 2007) (RT)

Démontrer qu'un quadrilatère ayant trois angles droits est un rectangle.



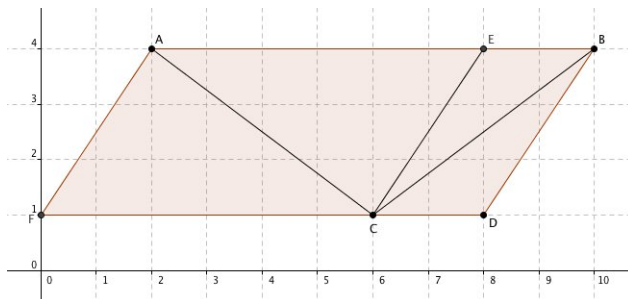
A.2 Construction d'un triangle impossible (Richard, 2004b) (TT)

Dans un triangle ABC isocèle en A, on considère les bissectrices des angles à la base qui se coupent à angle droit en O. Quelle est la nature du triangle BOC ?



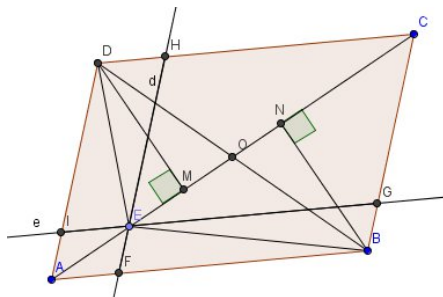
A.3 Triangle inscrit dans un parallélogramme (Richard, 2004b) (PT)

Sachant que $AECF$ et $EBDC$ sont des parallélogrammes, que les axes Ox et Oy sont perpendiculaires et que chaque division vaut 1 ; déterminez la nature du triangle ABC .



A.4 Comparaison des aires de triangles (Cobo et Fortuny, 2000) (PA)

Si E est un point quelconque de la diagonale AC du parallélogramme ci-dessous, quelle relation y a-t-il entre les aires des triangles AEB et AED ?



A.5 Triangle inscrit dans un cercle (Richard, 2004b) (TS)

Soit un cercle de diamètre BC , d la médiatrice du segment BC et A un point d'intersection du cercle et de d . Quelle est la nature du triangle ABC ?

