Titre: Title:	Algorithme d'énumération par programmation dynamique pour la gestion d'une flotte de véhicules automatiques oeuvrant dans une mine souterraine
Auteur: Author:	Mathieu Beaulieu
Date:	2003
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Beaulieu, M. (2003). Algorithme d'énumération par programmation dynamique pour la gestion d'une flotte de véhicules automatiques oeuvrant dans une mine souterraine [Mémoire de maîtrise, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/7104/

In compliance with the
Canadian Privacy Legislation
some supporting forms
may have been removed from
this dissertation.

While these forms may be included
in the document page count,
their removal does not represent
any loss of content from the dissertation.

UNIVERSITÉ DE MONTRÉAL

ALGORITHME D'ÉNUMÉRATION PAR PROGRAMMATION DYNAMIQUE POUR LA GESTION D'UNE FLOTTE DE VÉHICULES AUTOMATIQUES OEUVRANT DANS UNE MINE SOUTERRAINE

MATHIEU BEAULIEU
DÉPARTEMENT DES GÉNIES CIVIL, GÉOLOGIQUE ET DES MINES
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE MINÉRAL)
JUIN 2003

National Library
of Canada

Acquisitions and
Bibliographic Services

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque nationale
du Canada

Acquisisitons et
services bibliographiques

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 0-612-86378-6
Our file Notre référence
ISBN: 0-612-86378-6

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé :

ALGORITHME D'ÉNUMÉRATION PAR PROGRAMMATION DYNAMIQUE POUR LA GESTION D'UNE FLOTTE DE VÉHICULES AUTOMATIQUES OEUVRANT DANS UNE MINE SOUTERRAINE

présenté par : BEAULIEU Mathieu
en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées
a été dûment accepté par le jury d'examen constitué de :

M. MARCOTTE Denis, Ph.D., président
M. GAMACHE Michel, Ph.D., membre et directeur de recherche
M. COHEN Paul, Ph.D., membre et codirecteur de recherche
M. SOUMIS Francois, Ph.D., membre

REMERCIEMENTS

Je remercie mon directeur de recherche, Michel Gamache, pour ses conseils, son support, ses encouragements et sa disponibilité qui m'ont permis de réaliser ce projet. Je remercie également mon codirecteur de recherche, Paul Cohen, pour les ressources matérielles et financières déployées pour la réalisation du projet. Je tiens à remercier Louis-Philippe Bigras et Renaud Grimard pour la qualité de leurs travaux effectués sur les systèmes de gestion de véhicules automatiques pour les opérations minières souterraines. De plus, je tiens à mentionner mon appréciation pour l'aide et le temps qu'ils ont su m'accorder. Finalement, je tiens à remercier parents, frère, amis et anciennes blondes pour l'écoute, l'encouragement, la confiance, la compréhension et l'intérêt dont ils ont fait preuve envers moi et mes travaux.

RÉSUMÉ

Les systèmes de gestion d'une flotte de véhicules automatiques visent une réduction des coûts d'opérations, ainsi qu'une amélioration de la productivité. Ce mémoire présente une étude du problème de choix des routes pour l'automatisation des véhicules de manutention opérant en souterraine.

Le problème consiste à assigner de meilleures routes pour un ensemble de véhicules évoluant dans un réseau minier souterrain. Le réseau comporte des segments bidirectionnels à une seule voie. Le problème présente aussi des contraintes de conflit entre les véhicules, ainsi qu'une contrainte par rapport à l'orientation des véhicules. Deux approches permettent de résoudre le problème. Une méthode véhicule par véhicule permet de déterminer la route d'un véhicule selon les déplacements préétablis des autres véhicules du système. Une méthode globale quant à elle, réassigne les routes pour l'ensemble des véhicules à chacune des requêtes nécessaire au plan de production. Ce mémoire présente un algorithme d'énumération par programmation dynamique développé pour résoudre le problème d'une manière globale.

Le projet permet d'élaborer deux modèles. Le premier modèle représente la position des véhicules sur les segments de route ou aux intersections, alors que le deuxième modèle crée des états selon la position des véhicules uniquement sur ces segments. Pour chaque modèle, des propositions et des procédures sont proposées pour assurer un enchaînement admissible, c'est-à-dire sans conflits, entre deux états.

L'algorithme global permet d'analyser l'écart des solutions entre les deux approches de résolution. De plus, l'étude de différents critères de recherche permet

d'améliorer le temps de résolution en vue d'une application en temps réel. Les résultats montrent l'efficacité de résolution des méthodes véhicules par véhicules dans un environnement minier.

ABSTRACT

The objectives of automated guided vehicles systems are to reduce operation cost and to improve productivity. This master thesis is a study of a route selecting problem regarding hauling vehicles in underground mine.

The problem consists in assigning the best routes to a set of vehicles in an underground mine road network. The network segments are bidirectional on a single path. The problem includes vehicles conflict constraints, as well as vehicles orientation constraints. The problem can be resolved by two approaches. A vehicle route can be assessed from a vehicle by vehicle approach taking into account the existing routes of the other vehicles in the system. A global approach re-assigns the routes to all the vehicles upon every request needed by the schedule production. In this thesis, a dynamic programming enumeration algorithm is presented which can resolve the problem by a global approach.

Two models are developed. For the first model, states are represented by vehicles position on arcs and nodes of the network. The second model only use arcs to represent vehicles position. Procedures and propositions are presented to allow states progression.

The algorithm allows us to study the variation that occurs between the solutions coming from global and vehicle by vehicle approaches. Moreover, the algorithm resolving time can be reduced by different research criteria through the enumeration tree. The results show the effectiveness of the vehicle by vehicle methods for mining operation.

TABLE DES MATIÈRES

LISTE DES TABLEAUX

TABLE DES FIGURES

CHAPITRE 1 : DESCRIPTION DU PROJET

Les opérations minières visent la mise en valeur d'un gisement. Les études de faisabilités ainsi que les travaux de développement pour mettre en oeuvre une telle entreprise requierent un investissement en capitaux de taille. Il faut ajouter à ces montants fixes les coûts d'opération requis pour extraire le minerai et pour le concentrer en vue de lui attribuer une valeur économique. Cependant, sous des risques élevés, les revenus générés par ces exploitations permettent souvent de réaliser des profits. De par la taille de ces opérations et des capitaux investis, l'industrie minière peut retirer de nombreux avantages des méthodes développées en recherche opérationnelle. Plusieurs systèmes d'optimisation servent déjà les exploitations minières. La planification de la récupération des réserves minières, la maintenance des équipements, le transport du minerai, ainsi que tout l'aspect financier d'une entreprise minière sont tous des problèmes propices à des méthodes d'optimisation.

Les travaux résumés par ce document traitent des possibilités d'optimisation du transport du minerai en milieu souterrain. Le projet vise principalement à développer un modèle mathématique pour la gestion d'une flotte de véhicules typiques aux opérations minières.

Ce premier chapitre se divise en cinq sections. Tout d'abord, la première section expose les caractéristiques importantes des opérations de transport du minerai pour les exploitations minières souterraines. Les deux sections suivantes servent à définir les aspects et les contraintes du problème. Une revue de la littérature permet ensuite de résumer l'ensemble des méthodes de résolution utilisées pour des problèmes similaires. Finalement, la cinquième section présente les objectifs des travaux de recherche.

1.1 Description des opérations minières souterraines

On peut caractériser les propriétés minières par leur méthode d'exploitation. Dans certains cas, le gisement est peu profond et il est possible d'extraire les minéraux recherchés par des méthodes d'exploitation de surface, généralement appelées mines à ciel ouvert. Dans d'autres cas, le gisement est trop profond (entre 750 et 2000 mètres sous la surface) et il faut alors utiliser des méthodes dites souterraines. Généralement, ces deux types d'exploitation gèrent le transport du minerai, d'un point de soutirage vers les étapes de concentration, à l'aide d'une flotte de véhicules. Ces opérations de transport présentent des caractéristiques propices à l'optimisation des opérations. Cependant, l'environnement typique de chacune de ces méthodes d'exploitation ne permet pas de résoudre les problèmes par les mêmes modèles mathématiques. Ce mémoire présente des modèles mathématiques applicables aux opérations de transport en milieu souterrain.

Il existe plusieurs méthodes d'exploitation pour les gisements souterrains. La méthode retenue pour une exploitation minière dépend entre autres des caractéristiques du gisement : les dimensions, la forme et le pendage. Bien qu'il en existe plusieurs, ces méthodes possèdent de nombreuses caractéristiques similaires. De manière générale, les méthodes souterraines nécessitent la construction d'un puits de production, ainsi que d'un réseau de galeries afin d'accéder aux réserves minières et permettre leur manutention jusqu'à la surface. Usuellement, les étapes de production et de manutention se déroulent comme suit. Tout d'abord, une section du gisement est foudroyé grossièrement par dynamitage. Cette section est alors vidée à l'aide de véhicules mécanisés, conçus pour ces fonctions. Un système de convoyeur est rarement envisagé dû aux dimensions

excessives des blocs de minerai créés par le dynamitage. Les véhicules de type LHD (Load-Haul-Dump) sont les plus souvent utilisés pour acheminer le minerai du lieu de dynamitage vers une chute à minerai. Ces chutes à minerai débouchent tous vers un concasseur situé sous les niveaux exploitables du gisement. Ainsi, tout au cours de la durée de vie de la mine, les installations permettent de diriger le minerai exploité au même endroit, où on le réduit de nouveau afin de permettre une remontée efficace par un système de treuil à l'intérieur du puits de production.

1.1.1 Le réseau minier

Le réseau se représente par l'ensemble de tunnels excavés pour les besoins d'opération de la mine. Ces galeries permettent l'accès aux points de soutirage, aux chutes à minerai, au puits de production, ainsi qu'à tout autre lieu nécessaire à l'exploitation, tel un garage ou une salle électrique. Généralement, les voies d'accès sont construites à l'extérieur des zones minéralisées et leur excavation représente un coût important des travaux de développement. Certaines tentatives démontrent qu'il est non rentable de construire les infrastrutures d'une opération minière souterraine à même le gisement. Pour la majorité des méthodes d'exploitation, une perte importante sur le recouvrement des ressources ne justifie pas les coûts sauvés par la roche stérile conservée in situ. De plus, dans certains cas, la roche minéralisée n'offre pas toujours la stabilité nécessaire pour y construire des installations permanentes. Une roche stérile ne représente aucune valeur économique. Le coût associé à son extraction est élevé. Cependant, ces excavations sont nécessaires pour accéder au gisement.

Dans la majorité des cas, les caractéristiques suivantes sont typiques à un réseau tunnelier minier. Dans un premier temps, le degré d'une intersection est généralement inférieur à quatre. On peut définir le degré d'une intersection comme

étant le nombre de galeries permettant l'accès à une intersection. Ainsi un véhicule se présentant à une intersection possède souvent au maximum quatre choix pour la quitter.

Deuxièmement, les segments routiers sont souvent bidirectionnels à une seule voie. Cette caractéristique provient du coût associé à la construction du réseau minier. Pour minimiser les coûts de développement, le volume des galeries de l'ensemble du réseau minier se fixe selon la taille maximale des machineries. Puisqu'ils sont généralement les plus volumineux, la taille des véhicules de transport du minerai fixe les dimensions d'une galerie. Ainsi, pour les véhicules de manutention du minerai, les segments du réseaux routiers représentent un chemin à une seule voie et bidirectionnelle.

Une troisième caractéristique des réseaux miniers découle du coût important d'excavation de la roche stérile. Un réseau contient peu ou pas de cycle. C'est à dire, qu'il existe souvent qu'un seul chemin entre deux intersections du réseau.

Finalement, la longueur de chaque segment routier est souvent de l'ordre des centaines de mètres. Puisque le nombre d'intersections est petit et que l'étendue d'un gisement est grand, la longueur totale des segments peut représenter plusieurs kilomètres.

Peu importe la méthode d'exploitation, les réseaux miniers possèdent généralement toutes ces caractéristiques. Pour certaines méthodes d'exploitation telle la méthode chambres et piliers, une section du réseau se retrouve à l'intérieur du gisement et présente de nouvelles caractéristiques. Cependant le réseau d'accès à ces sections particulières répond aux caractéristiques mentionnées précédemment.

1.1.2 La manutention du minerai

On exploite les ressources minières à partir de chantiers. Tel que mentionné précédemment, le réseau minier permet d'accéder aux différents lieux de l'exploitation. Peu importe sa conception, un chantier permet de foudroyer le minerai in situ. Il doit contenir l'expansion du matériel fragmenté et offrir un point de soutirage. À partir de ce point de soutirage, un procédé doit être en mesure d'acheminer le minerai fragmenté vers les prochaines étapes de production. Les chutes à minerai servent à diriger la totalité des réserves exploitées vers un même point afin de le hisser à la surface. Pour plusieurs exploitations, on retrouve aussi des chutes pour la roche stérile.

Les chargeuse-navettes (LHD), voir figure 1.1, sont les véhicules le plus souvent utilisés par les mines souterraines pour la manutention primaire. On retrouve également des procédés de transport par train et locomotive, ainsi que par convoyeur. Cependant, au cours des dernières années, les opérations minières délaissent ces deux dernières méthodes devant l'efficacité des chargeuses-navettes. Ces véhicules représentent une machinerie spécialisée pour les opérations minières souterraines. Leur conception caractéristique permet de manoeuvrer efficacement parmi l'environnement restreint.

Ces véhicules sont équipés d'une pelle qu'ils utilisent pour le chargement et le transport du minerai. Les plus gros véhicules ont une capacité d'environ 10 mètres cube. Chaque chargement représente, selon le matériel transporté, environ de 10 à 15 tonnes. Ces véhicules voyagent à une vitesse d'environ trente kilomètres par heure et sont généralement alimentés par diesel. Dans le cas d'opérations minières souterraines à grande production, une flotte d'environ 20 véhicules est en opération chaque jour. Usuellement, des petits groupes de véhicules travaillent sur un même chantier sans interagir avec les autres véhicules de la flotte.

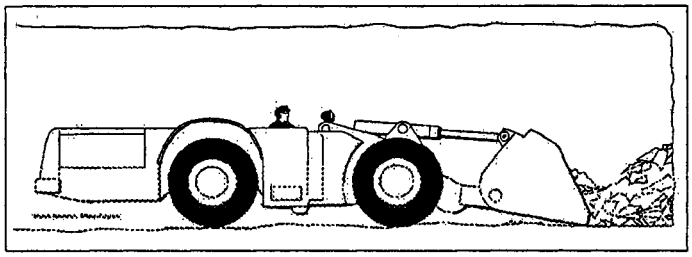

Figure 1.1: Représentation d'une chargeuse-navette

Leur forme longitudinale permet de minimiser la largeur et la hauteur des excavations nécessaires à leur utilisation. Puisque l'environnement minier ne permet pas un changement d'orientation facile pour un véhicule, une bonne manoeuvrabilité en marche avant ou arrière est un avantage. Grâce à son axe de rotation situé au centre de l'appareil, une chargeuse-navette est capable de circuler à vitesse maximale aussi bien en marche avant qu'en marche arrière. Ainsi, tout au long de son trajet, l'orientation du véhicule importe peu. Cependant, elle s'avère importante lorsque le véhicule arrive à destination. Lorsqu'un véhicule atteint un lieu de chargement ou de déchargement, il doit toujours se présenter pelle première. Ces lieux se situent souvent à l'extrémité d'un segment routier et ne débouchent sur aucun autre segment. Ils représentent des culs-de-sac et le véhicule doit faire marche arrière pour en sortir. Pour effectuer sa tâche le véhicule doit pouvoir manoeuvrer sa pelle. C'est pourquoi, l'opérateur doit s'assurer d'utiliser les intersections du réseau routier, pour changer d'orientation, avant d'emprunter le dernier segment de son parcours en marche avant.

Présentement, on retrouve dans l'industrie minière des véhicules équipés d'un système de téléguidage. Ces systèmes permettent à un individu d'opérer son véhicule à distance. Certains chantiers représentent une chambre ouverte où peu de soutènement retiennent les parois. Pour éviter une chute de roches sur un ouvrier, ce dernier opère son véhicule à distance. Ainsi, lorsqu'un opérateur

de véhicule arrive à un lieu de chargement, il débarque de son véhicule en un endroit sécuritaire. Avec le système de téléguidage, il manoeuvre le véhicule à l'intérieur du chantier afin de charger la pelle du véhicule. Ainsi, les opérations de chargement et de déchargement peuvent être effectuées sans opérateur sur le véhicule. Actuellement, on retrouve des système de téléguidage avec caméra. Un opérateur peut alors manoeuvrer le véhicule à partir de la surface. Cependant, bien qu'il puisse téléguider le trajet du véhicule vers une chute à minerai, un téléguidage complet des véhicules est rarement mis en place pour la totalité d'un cycle de production. La difficulté de manoeuvrabilité par système téléguidé affecte le temps de cycle des véhicules et, par le même fait, la rentabilité de l'exploitation.

1.1.3 Systèmes de gestions d'une flotte de véhicules automatiques

Comparativement à d'autres industries, ainsi qu'aux exploitations à ciel ouvert, le transport du minerai en milieu souterrain n'exploite pas les avantages associés aux processus d'optimisation et d'automatisation des tâches. Par exemple, il est fréquent de retrouver des chariots automatiques pour le transport du matériel en milieu manufacturier. Actuellement, les systèmes de gestion d'une flotte de véhicules peuvent répondre à certaines contraintes reliées au transport automatique du minerai en milieu souterrain. Cependant, des contraintes, telle la robustesse du milieu empêchent l'utilisation directe des technologies développées par le domaine manufacturier. Par exemple, la poussière présente dans les excavations de la mine empêchent l'utilisation efficace du matériel de guidage des véhicules. Il est donc nécessaire de développer de nouveaux outils afin de concevoir un système de gestion d'une flotte de véhicules automatiques pour le transport du minerai en milieu souterrain.

Un système de gestion d'une flotte de véhicules oeuvrant dans un réseau de galeries d'une exploitation minière requiert principalement trois éléments pour permettre son bon fonctionnement. Premièrement, un véhicule automatique doit être en mesure de se déplacer entre deux points de manière autonome. C'est-à-dire qu'aucune ressource humaine est nécessaire pour piloter le véhicule. En milieu manufacturier, les manoeuvres des véhicules à guidage automatique (automated guided vehicles-AGV) s'obtiennent souvent grâce à un tracé peinturé au sol de l'aire d'utilisation de ces équipements. Dans un milieu souterrain, il est difficile de conserver un plancher suffisamment propre pour permettre au véhicule automatique de se guider selon un tracé au sol. Au cours des dernières années, les technologies associées à la reconnaissance d'images permettent un guidage précis des véhicules oeuvrant en milieu robuste. L'information obtenue par un système de reconnaissance d'images permet un guidage efficace et améliore la capacité d'intelligence artificielle du véhicule automatique.

Dans un deuxième temps, un système de gestion d'une flotte de véhicules automatiques demande un protocole de communication entre les véhicules et un contrôleur. Ce système de communication permet de suivre les manoeuvres des véhicules et de les corriger en cas d'imprévus. Encore une fois, le milieu souterrain offre un certain défi sur cet aspect du problème. La présence du massif rocheux empêchent la propagation d'ondes de hautes fréquences. Cependant, les ressources technologiques actuelles permettent de gérer adéquatement cette contrainte. On retrouve présentement des mines souterraines utilisant les systèmes de positionnement global.

Dans un troisième temps, un contrôleur ou répartiteur doit attribuer les différentes tâches aux véhicules ainsi que toutes les informations nécessaires pour la réaliser. Un individu est en mesure de réaliser les opérations de répartiteur. Toutefois, des erreurs sur les décisions d'affectations génèrent bien souvent des

pertes en production. L'utilisation des techniques de la recherche opérationnelle permet de maximiser l'ouvrage accompli par les véhicules. Grâce aux méthodes mathématiques et à la puissance de calcul des outils informatiques, il est possible de calculer les meilleures routes ainsi que les meilleures affectations aux tâches pour les véhicules. Selon la complexité du problème, une solution demande un certain temps pour être calculée. Dans certains cas, on calcule préalablement les meilleures affectations et les meilleures routes pour une journée entière de travail. Les véhicules utilisent alors cet horaire préétabli. En milieu souterrain, de nombreux incidents peuvent se produire. Par exemple, un tronçon de route peut devenir inutilisable suite à la chute d'une roche. Le contrôleur, individu ou machine, doit alors tenir compte de cette nouvelle information pour rétablir les routes et tâches des véhicules. Des méthodes en temps réel permettent d'assurer une bonne interaction entre les véhicules et le contrôleur. Ainsi, suite à un incident affectant l'optimalité d'un horaire préétabli, le contrôleur est en mesure de prendre une décision rapide afin de minimiser l'arrêt de travail. Pour obtenir de nouveau un horaire optimal, sous contrôleur machine, il est nécessaire d'obtenir de nouvelles solutions par l'entremise d'un ordinateur. Pour agir en temps réel et pour minimiser le temps d'arrêt du système, le temps de calcul des algorithmes de résolution se doit d'être court.

Les travaux exposés dans ce mémoire s'intéressent plus particulièrement au troisième type de technologies nécessaires pour l'implantation d'un système de gestion d'une flotte de véhicules pour l'industrie des mines souterraines. Les travaux visent l'étude des modèles appliqués au contrôle des véhicules automatiques en milieu souterrain, ainsi que ceux appliqués à des domaines connexes. De plus, ce document propose une approche par programmation dynamique pour résoudre le routage des véhicules.

1.2 Description du problème

Les procédures de manutention primaire du minerai se déroulent en quatre étapes. Un cycle de production comprend le chargement d'un véhicule, le transport vers une chute à minerai, le déchargement, puis le déplacement du véhicule vers un nouveau chantier. Les travaux de recherche se concentrent sur le choix des routes pour les véhicules, ainsi que sur la prévention de conflits entre les véhicules. De manière générale, un système doit prendre en charge la circulation des véhicules automatiques sur un réseau à voie unique et bidirectionnelle, afin d'éviter toutes les collisions possibles entre les véhicules et d'assurer que les véhicules arrivent pelle première à un point de chargement ou de déchargement. De plus, dues aux conditions de travail en milieu souterrain, un système de gestion des véhicules automatiques doit permettre une adaptation efficace selon les imprévus rencontrés durant une journée d'exploitation.

Trois sous-problèmes sont associés au choix des routes pour les véhicules. Dans un premier temps, un problème de répartition se présente pour le choix des destinations des véhicules. Lorsqu'un véhicule termine une étape de chargement ou de déchargement, le système lui affecte alors une nouvelle destination selon l'état du véhicule et selon l'utilisation des chantiers. Inévitablement, un véhicule plein doit se diriger vers une chute, tandis qu'un véhicule vide a pour destination un chantier. L'affectation à un chantier pour un véhicule dépend généralement de la philosophie de gestion de la mine. On peut vouloir affecter les véhicules afin de minimiser l'attente à un chantier ou encore pour minimiser le trajet d'un véhicule. Pour certaines exploitations, l'affectation des véhicules se fait selon un problème de mélange. Puisque, la teneur de chaque chantier est différente, il est possible d'homogénéiser la teneur du minerai remonté à la surface. Une teneur constante du minerai permet d'améliorer les étapes de concentration. Ainsi avec

deux ou plusieurs chantiers de teneurs différentes, il est possible d'affecter les tâches des véhicules afin de répondre au problème de mélange.

Un deuxième sous-problème se pose pour le choix des routes des véhicules. Ce problème consiste à évaluer le meilleur chemin afin qu'un véhicule puisse se rendre à destination. Idéalement, on affecte le plus court chemin pour un véhicule. Cependant, le choix d'une route pour un véhicule dépend fortement des routes empruntées par les autres véhicules. Il faut donc assurer qu'une route ne représente aucun conflit avec les routes des autres véhicules du système.

Un troisième sous-problème, la gestion des conflits entre les véhicules, doit assurer la construction d'une route sans collision pour un véhicule. Afin d'éviter une collision entre véhicules, il est souvent nécessaire d'imposer une attente à un véhicule. Le problème de gestion des conflits consiste à évaluer les meilleures routes sans collision et sans impasse pour l'ensemble des véhicules. Ainsi, pour deux ou plusieurs chemins conflictuels, on doit évaluer le meilleur point d'attente pour un véhicule afin d'éviter le conflit ou encore obtenir un chemin différent pour un des véhicules.

Le problème principal se constitue donc de trois sous-problèmes : l'affectation aux véhicules des tâches de chargement et de déchargement, la résolution d'un plus court chemin pour chaque véhicule et la gestion des conflits entre les véhicules de la flotte.

1.3 Les contraintes liées au réseau

La complexité et les particularités de trouver des routes sans conflits pour un ensemble de véhicules sont largement reliées à la nature du réseau. Plus le nombre

de chemins entre deux extrémités est petit, plus le risque de conflits entre les véhicules est grand. Pour éviter tous conflits, un algorithme de résolution doit permettre et gérer l'attente d'un véhicule pour obtenir l'ensemble des meilleurs chemins des véhicules.

Plusieurs conflits peuvent survenir lors de l'élaboration d'une route pour un véhicule et leurs natures se divisent en deux groupes :

Les collisions sur un segment

Dans un premier temps, l'algorithme doit générer des solutions sans collision entre les véhicules. On retrouve les collisions frontales et les collisions de rattrapage. Les collisions frontales se produisent lorsque deux véhicules empruntent un même segment en sens opposé. Si on définit la position du véhicule, à un instant donné, par (k, O, D), soit k le numéro du véhicule, O le noeud de provenance du véhicule et D le noeud de destination du véhicule, on remarque qu'il est impossible que les états $(1, 5, 6)$ et $(2, 6, 5)$ soient réalisable au même moment, (voir la figure 1.2).

Pour les collisions de rattrapage, il est nécessaire de prendre en compte l'aspect du temps. Si tous les véhicules circulent à la même vitesse, il est impossible qu'un véhicule en rattrape un autre. Cependant, il est utile d'intégrer cet aspect à un algorithme de résolution. En effet, pour des raisons mécaniques, il est possible qu'un véhicule soit ralenti ou immobilisé, ce qui pourrait résulter en un conflit de rattrapage avec un autre véhicule. De plus, les processus d'attente des véhicules peuvent causer des conflits de rattrapage.

L'unicité des véhicules au noeud

Un seul véhicule peut occuper une intersection à une période de temps donnée sur l'horaire d'affectation des véhicules. Cette contrainte est reliée à la nature

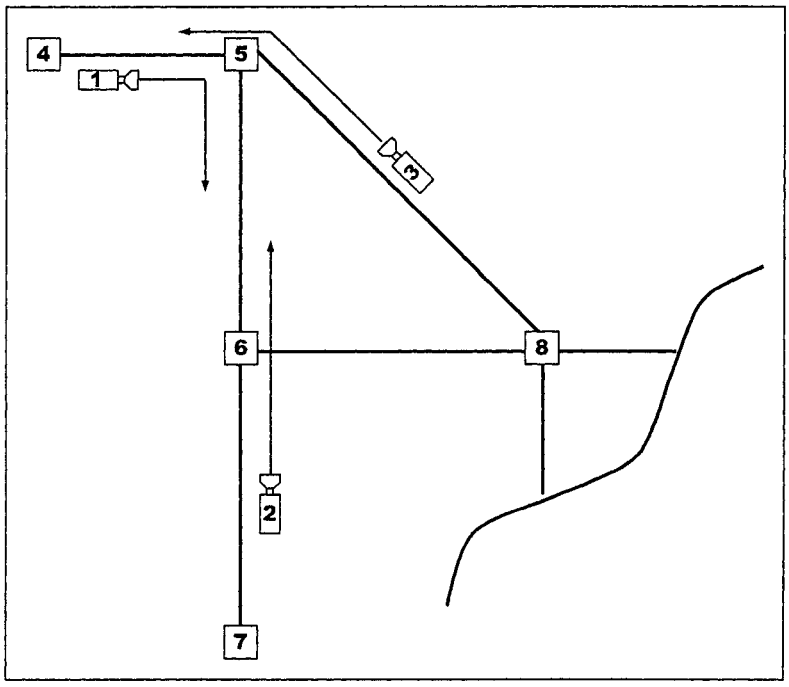

Figure 1.2: Exemple de conflits

physique du réseau. L'excavation n'est pas assez large pour permettre à deux véhicules d'y être présents en même temps. De plus, la gestion des véhicules serait beaucoup plus complexe si l'on permettait à deux véhicules de franchir une intersection simultanément. Ainsi, lorsque deux véhicules se dirigent vers la même intersection, l'algorithme doit prévoir l'attente d'un des véhicules à l'embouchure du segment.

1.4 Revue de la littérature

Peu d'articles traitent du problème de gestion d'une flotte de véhicules en milieu souterrain. Cependant, les problèmes d'affectation des véhicules pour une mine à ciel ouvert, ainsi que les problèmes de gestions de véhicules automatiques en milieu manufacturier, présentent des méthodes de résolution répondant à des contraintes similaires au problème étudié.

1.4.1 Mines à ciel ouvert

Dans le cas de méthodes d'optimisation pour la répartition des véhicules d'un mine à ciel ouvert, on cherche à maximiser le taux de production pour les pelles et les camions. Les pelles se situent en différents endroits à l'intérieur de la fosse. Les véhicules ont alors la tâche de transporter le minerai à l'extérieur de la fosse. Dans un premier temps, on peut chercher à minimiser l'attente des véhicules aux pelles. Puisque, le chargement d'un camion requiert une certaine période de temps et qu'il y a plus de camions que de pelles, on peut retrouver deux camions à la même pelle à un instant donné. Un des véhicules doit nécessairement attendre que la pelle se libère. Cette attente résulte en une perte de

production. De même, lorsque aucun camion n'est affecté à une pelle, celle-ci génère une perte en production. Un deuxième critère d'optimisation peut être le respect d'une contrainte de mélange pour la teneur du minerai. Les procédés pour la concentration des matériaux à valeur économique requièrent l'ajustement de certains paramètres selon la teneur du minerai. Lorsque la teneur en minerai varie par rapport à celle escomptée, elle entraîne alors une mauvaise récupération des matériaux recherchés. Puisque la teneur du minerai varie selon chaque chantier, il est préférable d'effectuer un mélange du minerai excavé avant de l'acheminer vers les étapes de concentration. Ainsi, un répartiteur doit s'assurer qu'en tout instant, les séquences de déchargement des camions fournissent une teneur constante.

De par la fréquence des imprévus rencontrés, les opérations minières souterraines et à ciel ouvert demandent des systèmes de gestion de véhicules en temps réel. On retrouve au sein de la littérature deux types d'approches pour résoudre le problème de répartition pour les exploitations en fosse. Devant la complexité du problème, des méthodes heuristiques visent à obtenir de bonnes solutions en un temps de calcul rapide, tandis que d'autres méthodes s'adaptent aux imprévus selon un plan de production optimal calculé avant le début des opérations d'un quart de travail. D'autres méthodes heuristiques, ou exactes, permettent d'ajuster le plan de production selon les imprévus rencontrés. Les travaux de Munirathinam et Yingling présentent les différents critères d'optimisation permettant de résoudre le problème de gestion des véhicules pour les opérations en fosse. Les ouvrages de Alarie et Gamache résument l'ensemble des stratégies de résolution pour le problème.

Pour le problème de gestion des véhicules en milieu souterrain, un même véhicule est souvent responsable des étapes de chargement et de déchargement, ainsi que celles de transport. Dans certain cas, on utilise un système de pelle et

camion pour les opérations en souterrain. Pour ces cas, le chargement des camions est rapide comparativement aux opérations de transport et l'attente des équipements de chargement est inévitable. Les situations conflictuelles entre les véhicules représentent une perte de production beaucoup plus importante que celle associée aux opérations de chargement. De plus, on remarque qu'un réseau minier souterrain se sature rapidement. C'est à dire que l'accessibilité de ses voies d'accès diminue selon le nombre de véhicule présent dans le système. Pour un certain nombre de véhicules, la congestion présente sur le réseau empêche toute amélioration au niveau de production. Le réseau de transport est moins restrictif pour les opérations à ciel ouvert et conséquemment un plus grand nombre de véhicules opèrent simultanément. De par leur fonction, les chargeuses-navettes permettent de maximiser l'efficacité des opération de transport primaire du minerai. Ainsi, l'aspect d'attente d'équipement à un lieu de chargement importe peu pour une amélioration des tâches de production. D'autre part, l'élaboration d'un plan de production ainsi que le problème de mélange des teneurs du minerai excavé, représentent des problèmes similaires retrouvés par les opérations minières souterraines.

1.4.2 Mines souterraines et milieu manufacturier

On retrouve au sein de la littérature de nombreux articles traitant de la gestion d'une flotte de véhicules en milieu manufacturier. Des algorithmes efficaces permettent de résoudre les problèmes d'affectation des tâches et de routage sans conflit en milieu manufacturier. Le contexte et l'utilité des différentes entreprises engendrent plusieurs travaux répondant aux différentes contraintes d'une manufacture. Dans certains cas, les véhicules automatiques circulent sous des conditions semblables au domaine minier. Plus particulièrement, les méthodes de routages de véhicules sous un réseau bidirectionnelle se rapprochent des contraintes

retrouvées en milieu souterrain. Parmi l'ensemble des méthodes développées, il est possible de classer les différents algorithmes sous deux types : le routage véhicule par véhicule et le routage globale de la flotte de véhicules.

Méthode d'optimisation véhicule par véhicule

Le routage véhicule par véhicule consiste à obtenir une nouvelle route pour un véhicule lorsque celui-ci complète un cycle de production. Cette route se construit selon l'itinéraire des véhicules déjà en marche sur le réseau routier. Une fois un chemin trouvé, cette route devient inviolable et les véhicules demandant une affectation subséquemment considèrent cette route de manière fixe. Le problème consiste à trouver un parcours sans conflit de coût minimum sans avoir à modifier les chemins déjà réservés par d'autres véhicules.

Parmi les approches véhicule par véhicule, Broadbent et al. proposent une méthode heuristique pour résoudre le problème. Leur approche consiste à trouver le plus court chemin avec l'algorithme de Dijkstra. Il détermine ensuite si le chemin est sans conflit. Autrement, des procédures permettent de réparer le chemin. Dans les cas de conflits de rattrapage, le véhicule à router est ralenti. Dans le cas de collisions sur un segment, le segment devient tabou et on recommence la procédure. Cette approche ne garantit pas une solution optimale.

Huang et al. proposent quant à eux un algorithme pour résoudre le problème à l'aide d'un graphe de fenêtres de temps.Pour établir le chemin d'un véhicule, l'algorithme procède en trois étapes. La première étape consiste à déduire les fenêtres de temps où les intersections et les voies sont libres en fonction de celles réservées par les autres véhicules. Ensuite, un graphe est construit selon les fenêtres de temps réalisables pour un véhicule. Finalement, on résout un plus court chemin sur le graphe obtenu.

Implanté adéquatement, cette méthode procure une complexité de l'ordre de $O(D^2 log_d D)$ où D est le nombre total de fenêtres de temps pour un problème et d représente le nombre de fils pour chaque noeud d'un monceau. Chaque véhicule réserve des chemins de longueur proportionnelle au nombre de noeuds du graphe. Chaque noeud du graphe représente une fenêtre de temps pouvant être réservée ou non par un véhicule. Ainsi, dans le pire cas, un véhicule réserve $O(N)$ fenêtres temps où N représente l'ensemble des fenêtres de temps. Donc, on peut écrire la complexité de l'algorithme sous la forme $O(V^2 N^2 log_d VN)$ où V est le nombre de véhicules du système.

Cette approche implique une limitation majeure. Une voie de circulation ne peut être occupée que par un seul véhicule à la fois. Ainsi, deux véhicules ne peuvent se suivre sur un même tronçon routier. Dans le cas des opérations minières, les segments routiers représentent de longue distance et il est impensable d'empêcher deux véhicules de circuler sur un même segment. Le milieu manufacturier offre des voies de transport de très courte distance. Le modèle de Huang et al. peut être toutefois efficace pour un milieu souterrain si on discrétise un long segment routier. Toutefois, cette transformation génère un plus grand nombre de fenêtre de temps à évaluer et par conséquent affecte la taille de l'ensemble N.

Les travaux de Vagenas exposent les problèmes d'automatisation en milieu souterrain et proposent une méthode de résolution pour la gestion des chargeuses-navettes. L'algorithme proposé par Vagenas s'effectue véhicule par véhicule et se déroule en trois étapes. La première étape consiste à évaluer les plus courts chemins pour un véhicule vers un ensemble de destinations. Ces destinations correspondent aux différents chantier en productions et aux points de déchargement de la mine. L'algorithme de Dijkstra permet d'obtenir les plus courts chemins. La deuxième étape consiste à valider la réalisabilité des routes pour un véhicule.

Aucun conflit ne doit survenir avec les véhicules déjà en route. Trois algorithmes gèrent les différents types de conflits : les conflits de rattrapages sur un arc, les conflits sur les arcs bidirectionnels et les conflits de trafic aux intersections. Ces algorithmes permettent de construire une série de parcours sans conflit pour un véhicule nécessitant l'affectation d'une nouvelle tâche. Pour certains cas, il est impossible d'obtenir une route pour un véhicule vers une destination. Les déplacements des véhicules déjà en route peuvent empêcher l'élaboration d'un chemin réalisable. Les algorithmes de gestion de conflits tentent alors d'établir une route pour le véhicule en demande d'une destination en modifiant la route des véhicules et les destinations des véhicules déjà en marche. Après avoir créé un ensemble de routes vers chacune des destinations admissibles pour un véhicule, la troisième étape de la procédure de Vagenas consiste à sélectionner la meilleure affectation possible pour le véhicule. Le critère de sélection pour la destination d'un véhicule est le temps minimum requis pour l'atteindre. Vagenas spécifie qu'il est également possible d'utiliser d'autres critères d'optimisation, telles la disponibilité d'un chemin ou la priorité des chantiers. Son modèle est restrictif et ne s'applique qu'à certains types de réseaux. La construction des routes se fait selon des méthodes heuristiques et peut générer des parcours qui s'éloignent de l'optimatlité.

Entre les années 1984 à 1991, les travaux dirigés par Tanchoco permettent de publier une série de quatre articles traitant des caractéristiques et du potentiel des systèmes de chariots automatiques. Egbelu et Tanchoco démontrent que l'utilisation de segments bidirectionnels permet d'améliorer le taux de production des véhicules. Toutefois, ils constatent aussi que le problème de routage s'avère plus complexe et ils proposent différentes stratégies pour résoudre les conflits entre les véhicules.

En 1991, Kim et Tanchoco établissent une méthode de résolution permettant d'obtenir le plus court chemin sans conflit, pour un véhicule. De plus, leur ap-

proche est valide autant pour des segments bidirectionnels que des segments unidirectionnels. Une solution optimale par un algorithme d'une complexité de l'ordre $O(V^4N^2)$ où V est le nombre de véhicules et N le nombre d'intersections du réseau routier. La méthode de Kim et Tanchoco propose de résoudre les problèmes à l'aide d'un graphe formé de fenêtre de temps libres et réservées. Contrairement à la méthode de Huang et al., les fenêtres de temps servent seulement à représenter la disponibilité d'une intersection. La procédure permet de déduire les conflits potentiels sur les voies de circulation, à partir du temps de passage des véhicules à une intersection.

La méthode d'affectation locale de Kim et Tanchoco utilise l'information des fenêtres de temps réservées par les véhicules du système pour construire un graphe. Ce graphe permet d'obtenir les fenêtres libres du système et une route sans conflit pour un véhicule demandant une affectation. Chaque noeud du graphe représente les intervalles de temps où l'intersection est libre. Un arc relie deux fenêtres libres s'il existe un lien physique entre les deux intersections et si, pour le meilleur cas, le véhicule peut atteindre le noeud avant la borne supérieure de l'intervalle de temps représenté par le noeud. Suite à la construction de ce graphe, l'algorithme détecte les arcs représentant un conflit sur les voies de circulation et les retire du graphe. Le graphe possède alors un ensemble de chemins sans conflit pour un véhicule. À l'aide de l'algorithme du plus court chemin de Dijkstra, le graphe temporel permet d'obtenir le meilleur chemin sans conflit pour le véhicule.

Les travaux de Grimard appliquent la méthode d'affectation et de gestion de conflits telles que développée par Kim et Tanchoco à des réseaux routiers typiques aux opérations minières. Il propose, de plus, une adaptation au modèle pour assurer l'orientation d'un véhicule lors de l'affectation d'une route à un véhicule. Ces résultats démontrent que plusieurs impasses surviennent pour des

réseaux aussi pauvres en chemins disjoints, tels que ceux retrouvés en milieu minier souterrain. Bigras reprend les travaux de Grimard et développe des procédures heuristiques pour gérer les situations d'impasses et améliorer la qualité de la solution initiale. Il développe également un simulateur pour évaluer l'efficacité de l'algorithme de construction de chemins sans conflits. Le simulateur permet de tenir compte des aspects stochastiques des opérations minières en souterrain. C'est-à-dire que le simulateur génère des interruptions de façon aléatoire. De plus, les temps de chargement et de déchargement des véhicules est aléatoire. Leurs résultats permettent de démontrer l'efficacité d'un système de gestion d'un flotte de véhicule en contexte minier sousterrain.

Méthode d'optimisation globale

La construction d'intinéraires sous une méthode véhicule par véhicule possède des caractéristiques statiques ; c'est-à-dire, qu'un chemin pour un véhicule se construit selon le parcours fixe des autres véhicules du système. Ainsi, lors de la construction d'un parcours pour un véhicule, l'itinéraire des véhicules déjà en route demeure fixe. Les méthodes globales permettent d'évaluer l'ensemble des routes empruntées par les véhicules à chaque demande d'affectation. Lorsqu'un véhicule arrive à destination, un algorithme global doit pouvoir modifier les routes des véhicules déjà en marche afin d'assurer que tous les véhicules du système possèdent les meilleurs chemins pour accomplir leurs tâches respectives. Contrairement aux méthodes véhicule par véhicule, les méthodes globales se caractérisent par leur aspect dynamique. Pour chaque prise de décision, un algorithme optimise la route de tous les véhicules ou d'une partie de la flotte. Les méthodes globales offrent de meilleures solutions. Cependant, la complexité du problème d'affectation de routes sans conflit est plus lourde sous des considérations globales. Il s'agit d'un problème difficile et un algorithme efficace doit pouvoir le résoudre en peu de temps pour être applicable à un système de gestion d'une flotte de véhicule en temps réel.

Tout comme pour les méthodes véhicule par véhicule, les ouvrages sur le problème de routage global d'une flotte de véhicules proviennent du domaine manufacturier. Fujii et al. proposent de résoudre une méthode globale en évaluant les k plus courts chemins pour l'ensemble des véhicules. Pour chaque demande d'assignation, la méthode consiste à évaluer le meilleur horaire global selon un ensemble de routes. Des méthodes linéaires et combinatoires permettent d'obtenir les meilleures routes sans conflit pour les véhicules. La méthode de Fuji et al. favorise les systèmes avec peu d'attente et d'interférence. Leur algorithme cherche à établir des chemins sans interférence, plutôt que de réparer les conflits possibles entre les véhicules.

Langevin et al. proposent un algorithme de programmation dynamique pour résoudre simultanément l'affectation et le routage des véhicules évoluant en milieu manufacturier flexible. Toutefois, leur algorithme est efficace en temps réel seulement pour deux véhicules.

Krishnamurthy et al. proposent de résoudre le problème par une méthode de génération de colonnes. L'objectif du problème maître est de minimiser le temps de parcours du véhicule le plus lent. Les colonnes du problème maître représentent les différentes routes pour les véhicules. Trois contraintes associées au problème maître permettent d'évaluer des routes sans conflit. Le sous-problème est un plus court chemin avec contraintes de ressources de temps. À partir des variables duales du problème maître, le sous-problème cherche à déterminer la route, pour un véhicule, de coût réduit maximum. Leurs résultats démontrent qu'une fois sur quatre la solution initiale est optimale. Cette solution initiale s'obtient généralement en moins de dix secondes à l'aide d'un processeur SUN 4/490 pour un problème de quatre à neuf véhicules évoluant sur un graphe de plus de 300 arcs. Cependant, de 10 à 500 secondes sont nécessaire pour ajuster les solutions initiales non optimales.

Villeneuve formule un modèle mathématique sous un réseau multi-flots pour résoudre simultanément et de façon exacte les problèmes de répartition et de routages sans collision. Son modèle nécessite une solution en nombre entier et comporte des contraintes non linéaires. Il utilise alors la décomposition de Dantzig-Wolfe pour résoudre le problème par génération de colonnes. Le problème maître vise à minimiser le délais des véhicules et comprend trois types de contraintes : les contraintes de couverture des tâches, les contraintes d'évitement des collisions et les contraintes d'ordonnancement. Ce dernier type de contraintes assure l'ordre des différentes tâches effectuées par un véhicule. Le sous-problème correspond à un problème de plus court chemin avec contraintes de recouvrement et de chargement sur un graphe acyclique. Le graphe acyclique s'obtient d'une transformation à partir du réseau multi-flots. Chaque sous-problème représente les contraintes propres à chaque chariot. Une méthode de séparation et d'évaluation progressive permet d'obtenir une solution entière. La méthode développée par Villeneuve permet d'obtenir la solution exacte pour un système comprenant de 2 à 4 véhicules en moins de dix minutes.

1.5 Objectifs du mémoire

Ce mémoire à pour objectif d'évaluer l'écart entre les solutions de types véhicules par véhicules et celles obtenues par méthode globale sur des réseaux caractéristiques aux opérations minières souterraines. Un algorithme efficace, développé par Bigras permet d'obtenir une solution sous une approche véhicule par véhicule. Ce mémoire propose une méthode de résolution afin d'obtenir un routage simultané des véhicules. Plusieurs méthodes de modélisation peuvent permettre de résoudre le problème. Il est possible de formuler le problème par programmation dynamique et tirer avantages de certaines caractéristiques des

opérations minières souterraines. Le petit nombre de chemins entre les destinations, la fréquence des interférences entre les véhicules, le petit nombre de véhicules en opérations simultanément, les processus d'attente et l'orientation des véhicules entraînent peu de déplacements possibles pour les véhicules du systèmes. L'algorithme par programmation dynamique permet d'évaluer des séquences de déplacement réalisables pour l'ensemble des véhicules et d'en constituer un horaire global.

Ce mémoire expose un algorithme de résolution par programmation dynamique et démontre les résultats obtenues sur des réseaux caractéristiques aux opérations minières. De plus, il présente des méthodes pour en accélérer le temps de résolution, afin de le rendre applicable à des processus en temps réel. Certaines adaptations utilisent l'information obtenue par une résolution véhicule par véhicule. Finalement, des expériences permettent d'évaluer l'efficacité de la méthode développée par Grimard et Bigras.

CHAPITRE 2 : FORMULATION DU MODÈLE MATHÉMATIQUE

Les travaux présentés dans ce chapitre visent la création d'un algorithme capable d'obtenir les meilleurs chemins pour un ensemble de véhicules. Une approche par programmation dynamique permet d'obtenir une solution optimale du problème. La programmation dynamique est une méthode d'optimisation qui transforme un problème complexe en une séquence de problèmes plus simples. Le principe de l'algorithme est de construire séquentiellement un parcours de coût minimal pour l'ensemble des véhicules. L'algorithme réalise une affectation globale, car il construit simultanément les routes de tous les véhicules du système.

Le contenu de ce chapitre se divise en neuf sections. En premier lieu, on effectue un bref survol des principes developpés en programmation dynamique. Par la suite, on présente une nomenclature définissant les aspects du modèle mathématique. Elle permet de définir le graphe physique, les états ainsi que les fonctions reliées au modèle. Puis, on développe les différentes contraintes du problème et on définit les règles pour la propagation des états. Ensuite, on présente des structures de résolution et on démontre une règle de dominance entre deux états. En dernier lieu, une présentation sommaire des résultats permet d'orienter les travaux futurs.

2.1 Principes généraux de programmation dynamique

La programmation dynamique est une technique mathématique conçue pour prendre une suite de décisions mutuellement reliées entre elles en vue d'opti-

miser un objectif donné. Comme les méthodes de séparation et d'évaluation progressive, elle constitue une exploration intelligement structurée de l'espace des solutions réalisables d'un problème d'optimisation. Elle permet de résoudre des problèmes caractérisés par des décisions interdépendantes et séquentielles.

Définissons d'abord les termes et symboles qui sont nécessaires à la formulation d'un problème quelconque comme programme dynamique.

État Un état est une configuration d'un système.

Étape Un intervalle durant lequel le système passe d'un état à un autre.

Coût Le coût est un nombre ou une fonction exprimant les valeurs engendrées pour cheminer d'un état à l'autre.

Action Chaque nouvel état engendre un ensemble d'actions réalisables. L'action choisie détermine l'état du système à la prochaine étape et le coût engendré par le fait même.

Valeur d'un état La valeur d'un état est une fonction des coûts engendrés pour atteindre cet état à partir de l'état initial.

Relation de récurrence Une relation de récurrence est une expression mathématique qui exprime la valeur d'un état en fonction du coût engendré par l'action immédiate adoptée et de la valeur de l'état précédent.

2.2 Représentation du graphe physique

Le réseau minier se représente par un graphe orienté $G = (N, A)$. La figure 2.1 illustre un modèle de graphe utilisé pour la résolution du problème. Chaque noeud représente une intersection, alors que les arcs correspondent aux différentes voies de transport du réseau minier. Les voies de transport sont bidirectionnelles et ne peuvent accueillir deux véhicules circulant en sens opposés. On

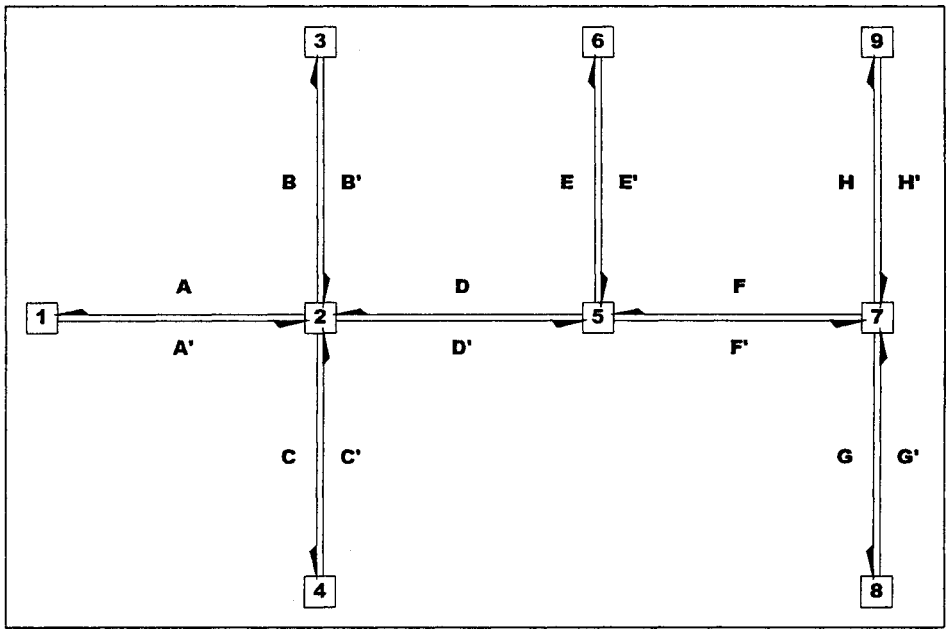

Figure 2.1: Représentation d'un réseau minier

utilise une combinaison de deux arcs pour représenter chaque voie. Ces doublons d'arcs permettent d'obtenir efficacement le sens de déplacement d'un véhicule. On associe un coût $c_{\alpha\beta}$ à l'arc $(\alpha, \beta) \in A$ pour représenter le temps nécessaire pour parcourir la distance entre les intersections α et β. Il est possible de modéliser un graphe où $c_{\alpha\beta} \neq c_{\beta\alpha}$. On associe un coût c_γ aux temps requis pour libérer une intersection $\gamma \in N$. Les différentes positions exprimées par le graphe $G = (N, A)$ servent pour la formulation des états et permettent la résolution implicite de certaines contraintes lors du processus d'optimisation.

2.3 Caractéristiques du modèle

2.3.1 Les états

Les états du modèle représentent, pour un instant donné, un agencement réalisable du positionnement de m véhicules sur le graphe. Soit V l'ensemble de ces m véhicules. On définit un état S_i par

$$S_i = (\mathbf{p_i}; \mathbf{t_i})$$

où

$$\mathbf{p_i} = (p_i^1, p_i^2, \ldots, p_i^k, \ldots, p_i^m)$$
$$\mathbf{t_i} = (t_i^1, t_i^2, \ldots, t_i^k, \ldots, t_i^m)$$

où p_i^k est la position $p \in N \cup A$ du véhicule k à l'état i et t_i^k le temps restant au véhicule k pour parcourir l'arc ou le noeud associé à la position p_i^k. Si le véhicule k est en attente $t_i^k = 0$. L'attente ne peut s'effectuer que sur des $p_i^k \in A$. Si d'autres véhicules attendent, alors on pose $t_i^k = 0 + q\delta$, où q est le nombre de véhicules en attente devant le véhicule k et δ un facteur de sécurité pour la distance entre les véhicules.

Il faut noter qu'il existe un nombre fini d'états pour représenter tous les chemins de tous les véhicules. Dans un premier temps, on est en mesure de constater qu'il existe un nombre de positions fini qu'un véhicule peut emprunter. Ce nombre de positions fini permet d'affirmer qu'il existe alors un nombre fini de combinaisons réalisables entre les différentes positions pouvant être empruntées par un ensemble de véhicules. De plus, pour chaque véhicule k, il existe un chemin de sa position initiale jusqu'à une position p^k et il existe un coût fixe t^k pour atteindre p^k. Il est donc possible de discrétiser le parcours continu d'un véhicule en une suite d'événements finis. Il existe donc un nombre fini d'états pour représenter le parcours des véhicules.

2.3.2 Les étapes

Les étapes caractérisent une suite d'évènements discrets sur un horizon de planification continu.

2.3.3 Les coûts

On exprime le coût engendré lors de la progression entre deux états par $c(S_i, S_j)$. On obtient la valeur du coût par

$$c(S_i, S_j) = \max_{k \in V} T_k$$

où

$$T_k = \begin{cases} t_i^k & \text{si } p_i^k \neq p_j^k\,; \\ 0 & \text{sinon.} \end{cases} \qquad k = 1, 2, \ldots, m$$

Il est à noter que pour chaque couple (S_i, S_j), il existe au moins un k tel que $p_i^k \neq p_j^k$.

2.3.4 Les actions

D'un état S_i, on peut atteindre un ensemble d'états réalisables, noté Γ_{S_i}. Cet ensemble d'états, contenant tous les successeurs immédiats et réalisables à partir de S_i, se définit en fonction d'un ensemble d'actions admissibles qui permettent de gérer le traffic sur le réseau.

2.3.5 La valeur des états

Nous définissons la valeur d'un état S_i par le temps total et minimal nécessaire pour que l'ensemble des véhicules puissent atteindre le positionnement décrit par le vecteur $\mathbf{p_i}$. Ainsi, on définit par $h(S_i)$ la valeur de l'état S_i. Cette valeur représente l'horloge du système et nous informe sur la progression de la solution.

2.3.6 La relation de récurrence

Pour construire séquentiellement une solution au problème d'affectation de routes à une flotte de véhicules, on utilise une relation de récurrence. Pour la formulation présentée dans ce mémoire, la valeur d'un état S_j se calcule à partir des valeurs des états $S_i, i \in \Gamma_{S_j}^{-1}$. On exprime la fonction de récurrence par

$$h(S_j) = \min_{S_i \in \Gamma_{S_j}^{-1}} \{h(S_i) + c(S_i, S_j)\}$$

où $\Gamma^{-1}_{S_j}$ est l'ensemble des prédécesseurs de S_j, i.e. les états S_i admissibles pour atteindre S_j.

2.4 Règles de propagation des états

À partir d'un état initial, il est possible de générer un ensemble d'états subséquents. Pour démontrer la propagation des états à une étape, utilisons un exemple pour une flotte de trois véhicules. Soit le graphe physique illustré à la figure 2.1, avec un coût de 5 unités pour les arcs et de 2 unités pour les noeuds, ainsi que l'état initial

$$S_0 = (B', D, 2; 5, 0, 0)$$

Cet état signifie que le véhicule 1 vient tout juste de quitter le noeud 3 et que le véhicule 2 est à l'embouchure du segment D, prêt à accéder au noeud 2 ou en situation d'attente. Le véhicule 3, quant à lui, se retrouve au noeud 2 et il a écoulé le temps nécessaire pour libérer l'intersection pour la prochaine étape. Individuellement, on remarque que chacun des véhicules fait face à un ensemble de choix pour sa prochaine action.

$$k = 1 : \{B', 2\}$$
$$k = 2 : \{D, 2\}$$
$$k = 3 : \{2, B, B', C, C', D, D', A, A'\}$$

La combinaison de ces trois ensembles forme un ensemble d'états qu'il est possible d'atteindre à partir de $S_0 = (B', D, 2; 5, 0, 0)$

$$\Gamma_{S_0=(B',D,2;5,0,0)} =$$

$$\{(B', D, B; 5, 0, 5), (B', D, B'; 5, 0, 0), (B', D, C; 5, 0, 5), (B', D, C'; 5, 0, 0),$$
$$(B', D, D; 5, 0, 0), (B', D, D'; 5, 0, 5), (B', D, A; 5, 0, 5), (B', D, A'; 5, 0, 0),$$
$$(B', 2, 2; 5, 2, 0), (B', 2, B; 5, 2, 5), (B', 2, B'; 5, 2, 0), (B', 2, C; 5, 2, 5),$$
$$(B', 2, C'; 5, 2, 0), (B', 2, D; 5, 2, 0), (B', 2, D'; 5, 2, 5), (B', 2, A; 5, 2, 5),$$
$$(B', 2, A'; 5, 2, 0), (2, D, 2; 2, 0, 0), (2, D, B; 2, 0, 0), (2, D, B'; 2, 0, 0),$$
$$(2, D, C; 2, 0, 0), (2, D, C'; 2, 0, 0), (2, D, D; 2, 0, 0), (2, D, D'; 2, 0, 0),$$
$$(2, D, A; 2, 0, 0), (2, D, A'; 2, 0, 0), (2, 2, 2; 2, 0, 0), (2, 2, B; 2, 0, 0),$$
$$(2, 2, B'; 2, 0, 0), (2, 2, C; 2, 0, 0), (2, 2, C'; 2, 0, 0), (2, 2, D; 2, 0, 0),$$
$$(2, 2, D'; 2, 0, 0), (2, 2, A; 5, 0, 0), (2, 2, A'; 2, 0, 0)\}$$

Tel que présenté sous cette forme, le nombre d'états générés pour chaque étape paraît très grand. Cependant, un grand nombre de ces états ne sont pas réalisables. Dans la section suivante, on élabore des procédures afin de détecter les états non réalisables de l'ensemble Γ_{S_0}.

2.5 Gestion des conflits

Lors de l'agencement séquentiel d'une solution $S_0, \ldots, S_i, \ldots, S_{fin}$, où S_{fin} est un état dont le vecteur position $\mathbf{p_{fin}}$ représente les destinations finales désirées, l'algorithme gère les conflits entre les véhicules à l'aide d'un ensemble de contraintes. Ces contraintes sont appliquées lors de la création de chaque ensemble Γ_{S_i}. Tel que décrit au chapitre 1, deux types de conflits peuvent se produire entre les véhicules du systèmes. Il est possible de détecter ces conflits par l'entremise de contraintes logiques. Les sous-sections suivantes formulent les procédures utilisées pour chacun des types de conflits.

2.5.1 Conflits aux intersections

Dans un premier temps, on peut établir aisément que tous les états dont le vecteur $\mathbf{p_i}$ possède deux éléments identiques et appartenant à l'ensemble N du graphe G sont non réalisables. Il est alors possible de retirer un certain nombre d'états d'un ensemble Γ_{S_0}. À titre d'exemple, utilisons l'ensemble construit à la section 2.4. Les états soulignés sont ainsi retirés de l'ensemble.

$$\Gamma_{S_0=(B',D,2;5,0,0)} =$$

$$\{(B',D,B;5,0,5),(B',D,B';5,0,0),(B',D,C;5,0,5),(B',D,C';5,0,0),$$

$$(B',D,D;5,0,0),(B',D,D';5,0,5),(B',D,A;5,0,5),(B',D,A';5,0,0),$$

$$\underline{(B',2,2;5,2,0)},(B',2,B;5,2,5),(B',2,B';5,2,0),(B',2,C;5,2,5),$$

$$(B',2,C';5,2,0),(B',2,D;5,2,0),(B',2,D';5,2,5),(B',2,A;5,2,5),$$

$$(B',2,A';5,2,0),\underline{(2,D,2;2,0,0)},(2,D,B;2,0,0),(2,D,B';2,0,0),$$

$$(2,D,C;2,0,0),(2,D,C';2,0,0),(2,D,D;2,0,0),(2,D,D';2,0,0),$$

$$(2,D,A;5,0,0),(2,D,A';5,0,0),\underline{(2,2,2;2,0,0)},\underline{(2,2,B;2,0,0)},$$

$$\underline{(2,2,B';2,0,0)},\underline{(2,2,C;2,0,0)},\underline{(2,2,C';2,0,0)},\underline{(2,2,D;2,0,0)},$$

$$\underline{(2,2,D';2,0,0)},\underline{(2,2,A;5,0,0)},\underline{(2,2,A';5,0,0)}\}$$

La contrainte d'unicité des véhicules à un noeud peut se formuler de la façon suivante.

Proposition 2.1 : L'action $S_i \rightarrow S_j$ est non valide si au moins deux éléments $p_j^k, p_j^l \in N$ tel que $p_j^k = p_j^l$ où $l \neq k$.

Dans un deuxième temps, une suite d'états peut générer un conflit à une intersection. Il est donc nécessaire de vérifier la progression de chaque état avec les caractéristiques de son prédécesseur. Lors de la propagation des états, un conflit peut survenir lorsqu'un couple de véhicules permutent leurs positions respectives d'un état S_i à un état S_j. De plus, on doit retrouver une permutation impliquant un changement d'un noeud vers un arc pour un des véhicules et inversement pour le deuxième véhicule. Toujours avec le même exemple, la procédure retire les éléments soulignés.

$$\Gamma_{S_0=(B',D,2;5,0,0)} =$$
$$\{(B',D,B;5,0,5),(B',D,B';5,0,0),(B',D,C;5,0,5),(B',D,C';5,0,0),$$
$$(B,D,D;5,0,0),(B',D,D';5,0,5),(B',D,A;5,0,5),$$
$$(B',D,A';5,0,0),(B',2,B;5,2,5),(B',2,B';5,2,0),(B',2,C;5,2,5),$$
$$(B',2,C';5,2,0),\underline{(B',2,D;5,2,0)},\underline{(B',2,D';5,2,5)},(B',2,A;5,0,5),$$
$$(B',2,A';5,0,0),\underline{(2,D,B;2,0,0)},\underline{(2,D,B';2,0,0)},(2,D,C;2,0,0),$$
$$(2,D,C';2,0,0),(2,D,D;2,0,0),(2,D,D';2,0,0),(2,D,A;5,0,0),$$
$$(2,D,A';5,0,0)\}$$

On formule cette contrainte de la façon suivante.

Proposition 2.2 : L'action $S_i \rightarrow S_j$ est non valide si on a pour

$$p_i^k \in N \text{ et } p_i^l \in A$$
$$p_i^k = p_j^l \text{ et } (p_j^k = p_i^l \text{ ou } p_j^k = (p_i^l)^{-1})$$

où $(p_i^l)^{-1}$ est l'arc inverse de p_i^l, tel que défini par le graphe $G = (N, A)$.

2.5.2 Conflits sur les arcs

On remarque un tel type de conflit lorsque deux véhicules, ou plus, se retrouvent sur le même segment. On détecte les conflits par les intervalles de temps associés à l'état initial et son successeur. Trois types de succession d'états peuvent générer des conflits entre les véhicules sur un même segment.

Premièrement, lorsque deux véhicules traversent le même segment dans le même sens, il faut s'assurer qu'il n'y ait aucune permutation entre les deux véhicules. Ainsi, il est d'abord nécessaire de déterminer le véhicule le plus avancé sur le segment. Ceci peut s'effectuer en comparant les vecteurs position et temps des états S_i et S_j.

La proposition 2.3 définit cette contrainte.

Proposition 2.3 : L'action $S_i \rightarrow S_j$ est non valide si pour

$$\alpha \in A \text{ et}$$
$$p_i^k = p_i^l = \alpha \text{ et } t_i^k > t_i^l \text{ (ou } t_i^k < t_i^l)$$
on a
$$p_j^k = p_i^l = \alpha \text{ et } t_j^k < t_j^l \text{ (ou } t_j^k > t_j^l)$$

Un autre type de conflit peut être généré lorsque deux véhicules se trouvent sur un même segment. Ces propagations d'états non admissibles surviennent lorsque deux véhicules se suivent sur le même arc et qu'à un état subséquent le deuxième véhicule dépasse le premier pour se retrouver sur un noeud. Pour illustrer ce phénomène, utilisons l'état suivant :

$$S_0 = (B', B'; 5, 0)$$

On remarque alors que les véhicules 1 et 2 se retrouvent sur le même arc. Le véhicule 1 est au début de l'arc B', tandis que le deuxième véhicule est à la fin du même arc. Parmi, l'ensemble des actions subséquentes, on retrouve l'état :

$$S_i = (2, B'; 2, 0)$$

Cet état se doit d'être éliminé, car il représente une collision entre les deux véhicules. Le véhicule 1 dépasse le second. On formule donc une contrainte pour les dépassements à un noeud sous la forme suivante.

Proposition 2.4 : L'action $S_i \rightarrow S_j$ est non valide si pour

$\alpha \in A$ et

$p_i^k = p_i^l = \alpha$ et $t_i^k < t_i^l$ (ou $t_i^k > t_i^l$)

on a

$p_j^k = \alpha \ \ et \ \ p_j^l \in N$ (ou $p_j^l = \alpha \ \ et \ \ p_j^k \in N$)

À première vue, il peut sembler intéressant de gérer ce type de conflits en empêchant la création d'états dont deux éléments du vecteur position sont identiques. Cependant, ce type de modélisation est trop restrictif car il rejette certaines séquences tout à fait légales en réalité. Alors, il faut permettre aux véhicules de se suivre sur un même segment.

Deuxièmement, on ne veut pas empêcher l'attente de plusieurs véhicules sur un même arc. La séquence d'états suivante doit être réalisable. Il faut toutefois faire reculer le véhicule positionné initialement à l'extrémité de l'arc B.

$$(2, B'; 0, 0) \rightarrow (B', B'; 0, 1)$$

La procédure 2.1 permet de gérer ce type de situation.

Procédure 2.1 : Soit

$$p_i^k \in N \ , \ p_i^l = p_j^k = p_j^l = \alpha \ \text{ où } \ \alpha \in A$$
$$\text{et l'arc } p_i^l \text{ est incident au noeud } p_i^k$$

l'action $S_i \rightarrow S_j$ est valide si on a

$$t_j^k = 0 \text{ et } (t_j^k + \delta) < t_j^l$$

sinon, on rend l'action valide en posant

$$t_j^l = (t_j^k + \delta)$$

où δ représente un facteur de sécurité pour la distance entre les véhicules.

Dans un troisième temps, lorsque les deux véhicules se retrouvent sur le même segment, mais dans des directions opposées, la situation s'avère beaucoup plus complexe. Généralement, la présence de deux véhicules en direction opposées et sur le même segment résulte en une collision frontale. Cependant, il n'est pas souhaitable d'empêcher la présence de deux véhicules sur un couple d'arcs (p_i^k, p_i^l), où $p_i^k = (p_i^l)^{-1}$. En effet, dû à la contrainte d'orientation des véhicules, il est parfois nécessaire d'utiliser un segment où un véhicule est déjà présent. Par exemple, l'enchaînement d'états suivant peut représenter une bonne solution.

$$(D, 2, B; 0, 1, 5) \rightarrow (2, B', B; 1, 0, 4) \rightarrow (C, 2, B; 5, 1, 3) \rightarrow (C, D', B; 4, 5, 2)$$

Ainsi, le véhicule 2 et 3 se retrouvent en même temps sur le segment B' et B. Pour que cet enchaînement soit réalisable, il faut permettre, lorsqu'un véhicule

traverse un segment, qu'un autre véhicule puisse occuper l'extrémité opposée à celle vers laquelle le premier véhicule se dirige. De plus, le premier véhicule ne doit pas se trouver au début de sa trajectoire. Lors de la génération d'états, seules les combinaisons (p_j^k, p_j^l), où $p_j^k = (p_j^l)^{-1}$ et au moins un des deux véhicules se retrouve à un point d'attente, sont acceptées. La proposition 2.5 nous permet de détecter ce type de conflit.

Proposition 2.5 : Soit

$$p_i^k \in N; p_i^l, p_j^k, p_j^l \in A$$
$$\text{où } p_i^l = p_j^l \text{ et } p_j^k = (p_j^l)^{-1}$$

L'action $S_i \rightarrow S_j$ est valide si

$$t_j^l < c_{p_j^l} \text{ et l'arc } p_j^k \text{ est incident au noeud } p_i^k$$

où $c_{p_j^l}$ est le coût pour traverser l'arc p_j^l tel que défini par le graphe $G = (N, A)$.

À l'aide de ces principes de sélection, il est alors possible pour l'algorithme de construire des séquences d'états. Ces séquences permettent d'obtenir un plus court chemin pour un ensemble de véhicules à l'aide d'un arbre de résolution.

2.6 Schéma de résolution

La méthode d'optimisation consiste à énumérer une suite d'états représentant les parcours minimaux pour les véhicules du système. À l'aide d'un arbre d'énumération, il est possible de construire un chemin sans conflit pour l'ensemble des véhicules. Chaque noeud de l'arbre représente un état. Un noeud engendre

un certain nombre de noeuds fils équivalent aux états de son ensemble Γ. Ainsi, à partir d'un état initial, il est possible de générer un certain nombre de noeuds fils. Ces noeuds fils deviennent à leur tour père en créant leurs ensembles Γ respectifs et permettent à de nouveaux états se greffent à l'arbre. Ainsi, l'arbre se prolonge jusqu'à l'obtention de l'état final désiré.

Pour un état donné S_i, l'algorithme génère l'ensemble de ses états successeurs noté Γ_{S_i}. Les propositions 2.1 à 2.5 et les procédures 2.1 et 2.2 filtrent l'ensemble Γ_{S_i} d'un état S_i. Ces propositions et procédures servent à éliminer les états non valides ou à rétablir la validité d'un état. Par exemple, un état possèdant plus d'un véhicules à la même intersection est nécessairement non valide. Toutefois, pour un état représentant un conflit de rattrapage sur un arc, il est possible de modifier les éléments de l'état afin de ralentir un véhicule et ainsi conserver sa validité.

À une itération donnée, l'algorithme doit choisir un état S_i, parmi les feuilles de l'arbre. Supposons dans un premier temps que le critère de sélection consiste à choisir parmi les feuilles l'état S_i dont la valeur de $h(S_i)$ est la plus petite. Notons cet état S_i^*. Pour S_i^*, l'algorithme génère l'ensemble $\Gamma_{S_i^*}$. Chaque solution partielle représente des chemins sans conflit pour l'ensemble de la flotte de véhicules à partir de leur point d'origine respectif. L'ordre selon lequel les états sont générés s'apparente à une recherche en largeur d'abord.

Similairement à l'algorithme développé par Dijkstra pour la résolution d'un problème de plus court chemin, l'algorithme utilise deux listes d'états. Une liste active permet de retrouver tous les états atteints, mais dont les successeurs n'ont pas encore été générés. Ces noeuds sans successeurs forment l'ensemble des feuilles de l'arbre à une itération donnée. Inversement, un état dont les successeurs sont connus s'insère dans la liste des états traités. Si Q représente la liste des états traités et P la liste des états traités, alors sous forme schématique, on définit l'algorithme de résolution par :

Procédure de Programmation Dynamique

1. $Q = \emptyset$ et $P = \emptyset$
2. $Q \leftarrow S_0$
3. Tant que $Q \neq \emptyset$
4. Choisir $S_i = \min\{h(S_i) \mid S_i \in Q\}$
5. Créer Γ_{S_i}
6. Filtrer Γ_{S_i}
7. $\forall S_j \in \Gamma_{S_i}$
8. si $h(S_j) \succ h(S_i) + c(S_i, S_j)$
9. $h(S_j) = h(S_i) + c(S_i, S_j)$
10. $pred(S_j) = S_i$
11. $Q \leftarrow Q \cup \{S_j\}$
12. $Q \leftarrow Q \backslash \{S_i\}$
13. $P \leftarrow P \cup \{S_i\}$

La ligne 8 de l'algorithme implique la définition de l'opérateur $\succ$ sur un état. Les particularités de l'opérateur se retrouvent à la section suivante.

L'algorithme génère des suites d'états jusqu'à l'obtention des conditions de l'état final. Il est alors possible de retracer les mouvements nécessaires pour passer de l'état initial à l'état final. Il est important de noter que l'algorithme termine lorsque l'ensemble des états etiquetés est vide. Ainsi, il est possible d'obtenir un état équivalent à l'état final. On appele alors la solution représentée par cet état : la premiére solution. La solution optimale est obtenue lorsque lorsqu'on assurre que tous les états etiquetés ne représentent pas une meilleure solution.

Il peut sembler que la génération d'états soit passablement grande. Toutefois, un même état peut être accédé par plusieurs états antécédents. Pour diminuer la taille de l'information accumulée par l'arbre, il suffit de ne conserver qu'un seul prédécesseur pour les fins de la solution finale. Par exemple, l'état $(D', 5, F'; t, t, t)$ s'obtient par $(2, E, 5; t, t, t)$, par $(D', 5, F'; t, t, t)$, ou encore par $(2, 5, F'; t, t, t)$. Cette redondance entre les prédécesseurs de chaque état permet une grande simplification pour la recherche de solution. En effet, si deux états dont les vecteurs $\mathbf{p_i}$ sont identiques et de noeud père différent, il est alors possible d'en conserver qu'un seul au sein de l'arbre d'énumération. En établissant, des règles de dominances entre les états identiques, il est possible de retrancher des noeuds de l'arbre, ainsi que les branches qui en découlent. La section suivante établit les régles de dominance entre les états identiques.

2.7 Règle de dominance

Afin d'éviter la génération d'un trop grand nombre d'états, une règle de dominance est mise en oeuvre. Dans certains cas, une équivalence entre deux états,

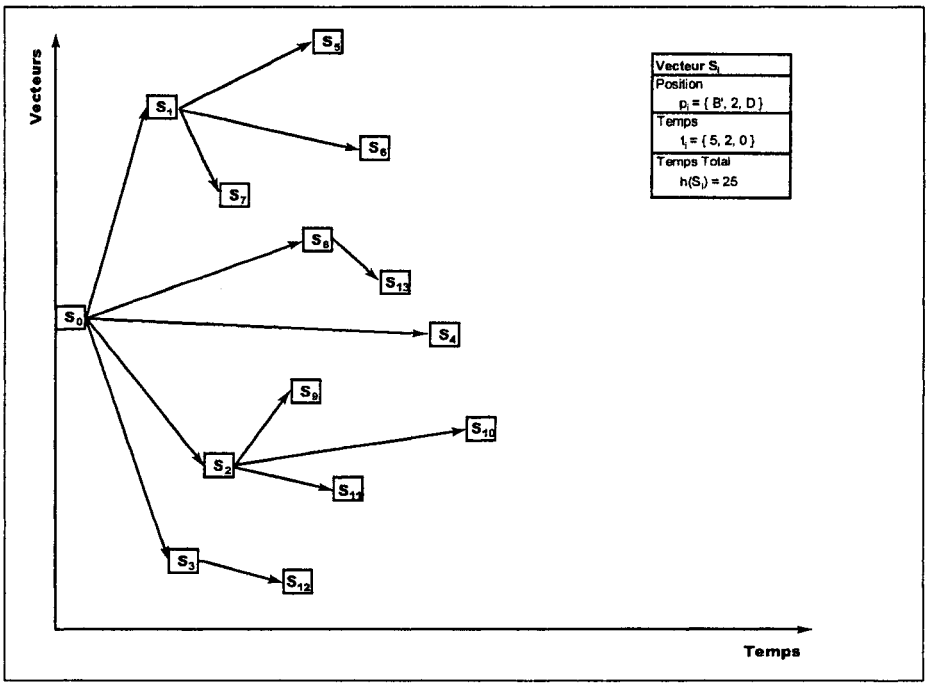

Figure 2.2: Arbre d'énumération

dont les éléments du vecteur $\mathbf{p_i}$ sont identiques, permet de retirer un état de l'arbre de recherche et les nombreux successeurs qu'il implique. Par contre, pour certains cas, deux états de vecteurs $\mathbf{p_i}$ identiques peuvent procurer deux solutions différentes. Il faut alors conserver les deux états.

La règle de dominance implique les principes suivant :

Soit S_i et S'_i deux états dont les vecteurs $\mathbf{p_i}$ sont égaux.

1. *Si la relation $h(S_i) + t_i^k \leq h(S'_i) + t_i'^k$ est vraie pour tous les éléments $k = 1, 2, \ldots, m$, alors l'état S_i représente un chemin plus court de S_0 à S_i que S_0 à S'_i. Ainsi, il n'est plus nécessaire de conserver S'_i, car les véhicules peuvent atteindre la même configuration en un temps égal ou moindre. On dit alors que la solution représentée par S_i domine celle de S'_i.*

2. *Lorsque la relation $h(S_i) + t_i^k \leq h(S'_i) + t_i'^k$ est fause pour un ou plusieurs éléments $(< m-1)$ $k = 1, 2, \ldots, m$, alors S_i et S'_i représentent deux solutions réalisables. Dans un tel cas, les états représentent deux possibilités de parcours. Pour chacun des parcours, le temps d'arrivée d'un véhicule, différent pour les deux états, est favorisé. Par exemple, sur la figure 2.1, les états $S_i = (A', F; 10, 0)$ et $S'_i = (B', E'; 0, 10)$, et $h(S_i) = h(S'_i)$, peuvent tous deux générer l'état $S_j = (2, 5; x, x)$. Ainsi il faut conserver les suites d'états $(A', F; 10, 0) \rightarrow (2, 5; 2, 0)$ et $(B', E'; 0, 10) \rightarrow (2, 5; 0, 2)$, car elles représentent deux solutions différentes et potentiellement optimales.*

Lors du processus de résolution, un filtrage de l'ensemble Γ_{S_i} permet de retirer les états dominés. Si un état traité S_p domine un état etiqueté S_q, alors S_q est

retiré de l'ensemble Γ de son prédéscesseur. D'autre part, si un état etiqueté S_q domine un état traité S_p, alors S_q est conservé et S_p ainsi que tous ses successeurs sont retirés de l'arbre. Il n'est pas possible de réaffecter les successeurs d'un état dominé sans réévaluer le potentiel de ceux-ci. Pour le cas où des états ne peuvent se dominer entre-eux, ils sont tous conservés et forment des branches distinctes. Bien souvent, l'arbre conserve un seul noeud pour représenter une configuration admissible des véhicules. Cependant, il peut être nécessaire de conserver jusqu'à $m!$ états pour une configuration d'un vecteur $\mathbf{p_i}$, où m est le nombre total de véhicules du système. En fait, il faut conserver un état identique pour chaque ordre d'arrivée des véhicules.

2.8 Recherche de solutions

Un système de gestion d'une flotte de véhicules cherche à maximiser un horaire de production. L'algorithme résout un demi-cycle de production, c'est-à-dire le transport du minerai vers une chute à minerai ou le retour vers un chantier. Dans le cadre des travaux, l'objectif d'optimisation est la minimisation du parcours pour le véhicule le plus lent. Cet objectif permet d'enchaîner une suite de demi-cycles de production de manière minimale.

L'objectif de l'algorithme est de minimiser le temps d'arrivée pour l'ensemble des véhicules. Une solution est optimale lorsque le temps d'arrivée du véhicule le plus lent est minimal. En cas d'égalité sur ce critère d'optimisation, les états sont retenus afin de procurer un temps global minimal. Le schéma de résolution, présenté à la section 2.6, présente une méthode de solution dont la procédure est équivalente à une recherche en largeur. L'état sélectionné à chaque itération est celui dont la valeur de la fonction $h(S_i)$ est la plus petite. Lorsque S_{fin} est

l'élément dont $h(S_{fin})$ est le plus petit de l'ensemble des états traités, alors la suite d'états permettant d'obtenir S_{fin} représente le plus court chemin sans conflit de S_0 à S_{fin}. L'algorithme atteint le critère d'optimalité lorsque tous les états etiquetés S_i avec $h(S_i) \leq h(S_{fin})$ sont traités. Cependant, une recherche en largeur entraîne le prolongement de plusieurs états inutiles. Par exemple, sur la figure 2.1, pour un véhicule ayant une origine au noeud 4 et une destination vers le noeud 6, l'algorithme de résolution génère tout de même l'enchaînement des états pour se rendre aux noeuds 1, 3 et 7. Certaines stratégies peuvent être mises en place pour améliorer l'efficacité de l'algorithme. Les sous-sections suivantes présentent les concepts développés.

2.8.1 Critère de sélection des états

Une solution obtenue par une recherche en largeur implique que tous les états dont la valeur de $h(S_i)$ est inférieure à $h(S_{fin})$ se retrouvent au sein de l'arbre de recherche. Puisque l'ordre de sélection des états influence l'enchaînement des solutions partielles, il est possible d'atteindre une première solution, avec un effort moindre. Pour créer une recherche par profondeur d'abord, une modification du critère de sélection permet de visiter les états les plus prometteurs en premier. Ainsi, la valeur d'un état S_i, c'est-à-dire $(h(S_i))$, se remplace par une nouvelle fonction :

$$g(S_i) = \sum_{k=1}^{m} PCC(S_i, S_{fin})$$

où $PCC(p_i^k, p_{fin}^k)$ est le plus court chemin à partir de

la position du véhicule k jusqu'à sa destination finale.

Cette nouvelle valeur permet à l'algorithme de sélectionner en premier les états les plus proches de l'état final. La valeur du plus court chemin ne tient pas compte d'un conflit éventuel entre les véhicules. Ainsi, l'algorithme favorise les états les plus près de la solution finale. Pour obtenir une progression plus rapide des véhicules vers leurs destinations finales, le plus court chemin est calculé à partir du prochain noeud intercepté. Ce processus permet d'avantager les véhicules en route sur un segment comparativement au véhicule en attente. En cas d'égalité sur $g(S_i)$ entre deux états, la sélection se fait selon la valeur minimale des fonctions $h(S_i)$. Si l'égalité persiste, la sélection de l'état suivant se détermine selon l'ordre de création.

2.8.2 Évaluation des bornes

Pour tenter d'améliorer l'efficacité de l'algorithme et analyser l'impact des méthodes de recherche, il est possible d'évaluer séquentiellement les états selon une recherche par meilleur d'abord. Une fois la première solution obtenue, il est nécessaire de vérifier tous les états pour assurer qu'il n'y a pas d'autres chemin représentant un temps de parcours moindre. À l'aide d'un concept de bornes, deux règles permettent de couper certaines branches de l'arbre de recherche et de s'assurer que la solution est optimale.Une première règle peut se développer en utilisant le temps de parcours du véhicule le plus lent de l'état S_{fin} comme borne supérieure. Il est alors possible d'éliminer les états ne pouvant pas atteindre l'état final en un temps inférieur à cette borne fixée par le véhicule. Évidemment, il n'est pas possible de savoir le temps exact pour qu'un véhicule atteigne sa destination. Cependant, le plus court chemin d'un véhicule, sans considération des conflits éventuels, fournit le temps minimal nécessaire pour se retrouver à la position souhaitée. Cette approximation permet de connaître le

temps restant à parcourir dans le meilleur des cas. Si en utilisant son meilleur chemin un véhicule ne peut atteindre sa position finale dans un meilleur temps, alors il n'est pas utile de prolonger l'état qui le représente. Mathématiquement, cette règle se formule par :

Règle 2.1 : L'état S_i est conservé au sein de l'arbre de recherche si

$$\max_{k=0,1,\ldots,m} h(S_i) + t_i^k + PCC(p_i^k, p_{fin}^k) < \max_{k=0,1,\ldots,m} h(S_{fin}) + t_i^k$$

Dans un deuxième temps, il est possible de restreindre le nombre d'états selon le temps global. Le temps global de la solution se définit par la somme de temps nécessaire pour que chacun des véhicules atteignent leur destination respective. Lorsque l'estimation d'un prolongement d'un état S_i vers S_{fin} représente un temps égal au temps du véhicule le plus lent, le temps global agit comme deuxième critère. Ainsi, dans une situation où un état représente une possibilité de parcours en un temps égal à la meilleure solution évaluée jusqu'à ce point par l'algorithme, le temps global sert à discriminer les états non prometteurs. Cette règle se formule par :

Règle 2.2 : L'action S_i est conservée au sein de l'arbre de recherche si

$$\max_{k=0,1,\ldots,m} h(S_i) + t_i^k + PCC(p_i^k, p_{fin}^k) = \max_{k=0,1,\ldots,m} h(S_{fin}) + t_i^k$$

et

$$\sum_{k=0}^{m} h(S_i) + t_i^k + PCC(p_i^k, p_{fin}^k) \leq \sum_{k=0}^{m} h(S_{fin}) + t_i^k$$

La section suivante présente les test et résultats développés à l'aide de ces différentes stratégies de recherche.

2.9 Résultats sommaires

Les travaux visent, entre autres, à évaluer l'affectation globale des chemins pour les véhicules de transport du minerai en contexte souterrain. Tel que mentionné précédemment, les méthodes véhicule par véhicule comportent certaines limites. Leurs solutions n'atteignent pas toujours l'optimalité globale ou résultent parfois en impasse. Cette section présente une évaluation sommaire de l'algorithme développé et appuie les travaux futurs. Elle permet d'évaluer les forces et les faiblesses de la méthode de résolution. L'algorithme testé n'évalue cependant pas l'orientation des véhicules au lieu de chargement et de déchargement.

2.9.1 Méthodologie

Les résultats escomptés doivent permettre d'obtenir les meilleurs chemins globaux pour un ensemble de véhicules. Pour évaluer son optimalité, ainsi que son efficacité, on propose de résoudre des demi-cycles de production sous différents réseaux pour une flotte de trois véhicules. Chaque réseau comporte les particularités retrouvées en milieu souterrain. La figure 2.3 illustre les trois réseaux utilisés pour les expérimentations. Le premier réseau possède 16 arcs et 9 noeuds. De plus, il ne contient aucun cycle, mis à part les cycles créés par un arc et son inverse. Le deuxième réseau contient le même nombre d'arcs que le premier. Cependant, il contient un cycle et ne possède que 8 noeuds. Le troisième réseau est plus complexe. Plusieurs chemins permettent d'atteindre les 12 noeuds du réseaux. Il contient 26 arcs.

Pour évaluer l'efficacité de l'algorithme sur un demi-cycle de production, chacun des véhicules possède un point de départ, ainsi qu'une destination fixe. Les

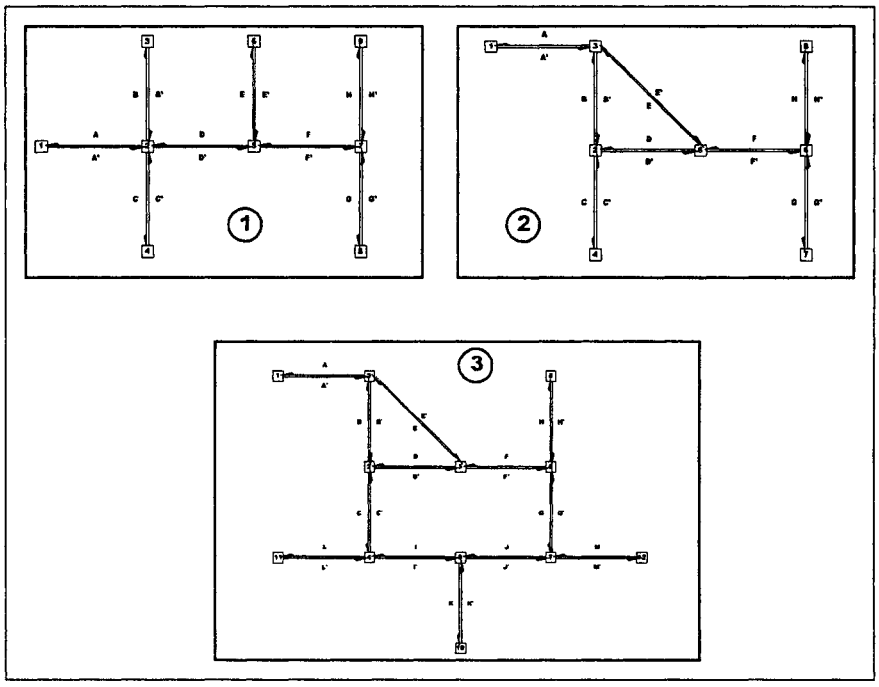

Figure 2.3: Réseaux miniers

solutions représentent alors le parcours utilisé par chacun des véhicules pour se déplacer vers sa destination. Afin d'évaluer le plus grand nombre de situations conflictuelles, des temps de départ aléatoires permettent de modifier les conditions du système. On conserve la même séquence de nombres aléatoires pour évaluer différents scénarios de résolution sur un même réseau.

Pour un même scénario, une série de vingt instances permet d'évaluer l'efficacité de l'algorithme sous différentes conditions. Lorsque le temps de résolution pour un scénario est trop grand, on résout alors seulement cinq instances. Pour chaque instance, on conserve l'information sur le temps de résolution, le nombre d'itérations avant d'obtenir une première solution, le nombre d'itérations nécessaires pour minimiser la première solution réalisable et le nombre de noeuds de l'arbre une fois la solution optimale obtenue. De plus, l'écart entre la première solution et finale se représente sous forme de pourcentage. Les colonnes *gap d'optimalité* servent à représenter les résultats associées aux écarts avec la première solution. La première colonne permet d'évaluer l'erreur commise par rapport au temps du véhicule le plus lent. La deuxième valeur évalue l'écart sur le temps global de la solution.

2.9.2 Présentation des scénarios et résultats

Les deux méthodes d'exploration de l'arbre d'énnumération, ainsi que les réseaux, présentés à la section précédente, permettent de créer différents scénarios. Dans un premier temps, des solutions s'obtiennent grâce à la méthode de recherche en largeur. Les tableaux 2.1, 2.2 et 2.3 présentent les résultats obtenues pour chacun des trois réseaux.

Le premier scénario présente une recherche en largueur pour les trois réseaux. Cette technique ne permet pas de résoudre efficacement le réseaux 3. Cependant,

Tableau 2.1: Résultats réseau 1 pour 3 véhicules, recherche en largeur

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	225,9	15858	905	17287	0,0	0,0
1	222,1	16531	1811	18071	0,0	0,0
2	234,5	16156	2564	18585	0,0	0,0
3	178,4	14969	2167	16805	0,0	0,0
4	216,3	16128	1266	18001	3,0	-6,7
5	238,3	16792	1887	18402	0,0	0,0
6	210,3	15787	1293	17744	20,6	13,3
7	222,3	15367	1941	17009	0,0	0,0
8	198,1	14808	1613	16944	0,0	0,0
9	246,9	15572	1200	17267	6,7	-1,9
10	227,3	15017	2093	16840	0,0	0,0
11	233,0	15593	1914	17175	0,0	0,0
12	222,8	14612	2031	16325	0,0	0,0
13	212,7	17255	2664	19753	0,0	0,0
14	223,0	16268	1927	17853	0,0	0,0
15	239,7	16867	1869	18497	0,0	0,0
16	212,2	15485	1832	17227	0,0	0,0
17	213,7	15182	1625	16777	0,0	0,0
18	211,3	14905	1780	16536	0,0	0,0
19	239,7	17074	1962	18729	0,0	0,0
moyenne	221,4	15811,3	1817,2	17591,4	1,5	0,2

Tableau 2.2: Résultats réseau 2 pour 3 véhicules, recherche en largeur

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	153,2	16114	769	16363	0,9	-6,6
1	159,4	16054	681	16247	10,4	0,8
2	180,1	15927	709	16215	0,0	0,0
3	162,1	15661	785	15946	0,0	0,0
4	170,4	15666	725	15881	1,8	-6,1
5	166,7	15957	695	16122	4,2	-3,5
6	179,6	15868	822	16161	1,7	0,0
7	182,4	15605	714	15820	0,0	0,0
8	173,8	15860	759	16073	4,1	-3,6
9	195,4	14358	1198	15040	0,0	0,0
10	170,7	14509	1498	15493	0,0	0,0
11	200,5	15907	640	16042	10,4	1,4
12	173,6	14541	1472	15488	0,0	0,0
13	182,0	16148	828	16548	0,0	0,0
14	190,7	15976	729	16199	7,3	-0,7
15	195,3	15832	703	15999	3,2	-4,1
16	204,5	16014	673	16140	0,0	0,0
17	182,6	14258	1521	15235	0,0	0,0
18	181,3	14516	1472	15433	0,0	0,0
19	196,2	15886	743	16058	6,2	-2,3
moyenne	180,0	15532,9	906,8	15925,2	2,5	-0,5

Tableau 2.3: Résultats réseau 3 pour 3 véhicules, recherche en largeur

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	1378,9	46566	8592	54030	0,0	0,0
1	1817,0	46443	7288	53162	0,0	0,0
2	2027,3	57811	4452	62503	0,0	0,0
3	1968,1	54587	6958	60040	0,0	0,0
4	3738,8	52230	4487	55486	0,0	0,0
moyenne	2186,0	51527,4	6355,4	57044,2	0,0	0,0

les résultats obtenues sur les réseaux 1 et 2 permettent de tirer les conclusion suivantes. Les temps de résolutions, le nombre d'états avant d'obtenir une première solution et nécessaire pour la minimiser, ainsi que le nombre de noeuds contenus par l'arbre de recherche une fois la solution optimale obtenue, sont relativement constants. La taille de l'arbre de recherche comparativement au nombre d'états générés laisse croire que peu d'états sont retirés de l'arbre au cours du processus de recherche de solution. Aussi, la majorité du temps requis pour la complétion de l'algorithme se retrouve dans le processus de recherche d'une première solution. Le gap d'optimalité démontre que la première solution est souvent optimale. Dans certains cas, la première solution représente un temps global plus court. Ces cas possèdent un gap d'optimalité global négatif. De plus, on retrouve des erreurs de l'ordre de 20 % sur le temps du véhicule le plus lent. Toutefois, ces instances ne représentent pas les problèmes les plus longs à résoudre. Si on compare les résultats pour chacun des réseaux, on remarque que le réseau 1 semble plus difficile à résoudre ; pourtant il ne contient aucun cycle. Il existe donc peu de chemins entre deux noeuds, ce qui engendre un grand nombre de conflits entre les véhicules. Les véhicules doivent forcément attendre au cours de leurs trajets et ceci provoque la génération d'un plus grand nombre d'états, car il existe un grand nombre de lieux d'attente valide. Aussi, le réseau 1 possède un noeud de plus que le réseau 2. De plus, il contient une intersection à quatre voies. Ces facteurs influencent tous l'efficacité de l'algorithme.

Dans un deuxième temps, on modifie le réseau 2 afin d'évaluer l'efficacité de l'algorithme pour des segments de longueurs différentes. Un premier réseau, le réseau 2a, s'obtient en réduisant de moitié les coûts associés à chaque segment du réseau 2. Similairement, d'autres résultats s'obtiennent en multipliant par un facteur de 5 les coût associés au réseau 2 pour obtenir le réseau 2b . Finalement, un troisième réseau, le réseau 2c, peut être obtenu en conservant la même configuration et en appliquant un coût aléatoire pour chacun des segments. Les valeurs générées aléatoirement varient entre 20 et 200 unités. Les tableaux 2.4, 2.5 et 2.6 présentent les résultats obtenus pour les trois réseaux créés.

Tableau 2.4: Résultats réseau 2a pour 3 véhicules, recherche en largeur

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	151,7	15622	649	15763	8,2	-2,8
1	149,2	15688	416	15594	0,0	0,0
2	163,6	15451	636	15671	0,0	0,0
3	157,5	15611	415	15510	0,0	0,0
4	160,6	15544	656	15647	8,2	-2,8
5	161,7	15693	692	15906	9,4	-1,3
6	163,2	15610	644	15754	8,2	-2,7
7	168,7	15401	532	15449	0,0	0,0
8	167,0	15615	675	15793	9,2	-1,3
9	172,0	15490	677	15619	0,0	0,0
10	167,8	14316	1132	14976	0,0	0,0
11	176,1	15755	415	15675	0,0	0,0
12	179,8	14067	1090	14621	0,0	0,0
13	189,3	15689	519	15780	0,0	0,0
14	180,1	15797	415	15777	0,0	0,0
15	182,7	15635	699	15837	9,1	-1,3
16	180,6	15318	690	15587	0,0	0,0
17	180,9	14079	1084	14636	0,0	0,0
18	193,1	14067	1090	14621	0,0	0,0
19	190,5	15821	431	15802	0,0	0,0
moyenne	171,8	15313,5	677,9	15500,9	2,6	-0,6

Le deuxième scénario présente l'efficacité de l'algorithme selon la longueur des segments. Pour les réseaux 2, 2a, 2b et 2c il existe peu de différences entre

Tableau 2.5: Résultats réseau 2b pour 3 véhicules, recherche en largeur

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	165,2	16999	646	17077	10,5	2,7
1	167,3	17023	629	17123	12,9	4,6
2	172,3	16700	686	16817	0,0	0,0
3	171,8	16748	837	17009	0,0	0,0
4	175,9	16936	646	17024	10,7	2,9
5	178,9	16984	635	17095	11,2	3,4
6	181,8	17025	646	17093	10,7	2,9
7	187,0	16551	682	16658	0,0	0,0
8	183,9	16940	635	17052	11,2	3,4
9	190,0	16952	643	17046	8,4	1,1
10	181,5	15563	1381	16361	0,0	0,0
11	197,5	16996	629	17105	12,9	4,7
12	184,6	15512	1410	16327	0,0	0,0
13	204,4	16949	836	17214	0,0	0,0
14	200,0	16968	629	17058	12,0	4,1
15	202,9	16969	634	17034	10,9	3,2
16	211,9	16555	653	16570	0,0	0,0
17	191,0	15525	1403	16322	0,0	0,0
18	194,4	15513	1410	16335	0,0	0,0
19	215,4	16979	629	17053	11,8	3,8
moyenne	187,9	16619,4	815,0	16868,7	6,2	1,8

Tableau 2.6: Résultats réseau 2c pour 3 véhicules, recherche en largeur

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	187,0	13042	955	14172	0,0	0,0
1	188,9	13171	914	14281	0,0	0,0
2	197,8	13476	844	14547	0,0	0,0
3	192,5	13186	954	14296	0,0	0,0
4	192,4	13046	969	14185	0,0	0,0
5	195,6	13085	968	14207	0,0	0,0
6	194,6	13075	945	14197	0,0	0,0
7	125,6	13185	945	14321	0,0	0,0
8	130,5	13078	960	14188	0,0	0,0
9	150,5	12087	1164	13338	0,0	0,0
10	134,0	13295	1345	14373	29,0	10,4
11	132,3	13333	875	14390	0,0	0,0
12	144,8	13303	1345	14388	27,5	9,7
13	137,5	13892	1209	14917	0,0	0,0
14	137,2	13144	973	14280	0,0	0,0
15	136,6	13074	966	14193	0,0	0,0
16	137,8	13212	921	14310	0,0	0,0
17	149,2	13342	1346	14420	26,8	9,3
18	154,0	13324	1345	14401	27,5	9,7
19	143,4	13098	964	14217	0,0	0,0
moyenne	158,1	13172,4	1045,4	14281,1	5,5	2,0

les résultats obtenus. Il est possible de conclure que le modèle permet de créer correctement des événements discrets peu importe la longueur des segments. Dans ces cas-ci, la première solution est souvent optimale. Malheureusement, tout comme pour les résultats précédents, cet avantage ne procure rien au temps de résolution.

Une méthode de recherche par profondeur d'abord, appliquée aux réseaux 1, 2 et 3, permet d'obtenir les résultats présentés aux tableaux 2.7, 2.8 et 2.9.

Tableau 2.7: Résultats réseau 1 pour 3 véhicules, recherche par profondeur d'abord

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	68,3	31	13239	980	9,0	24,7
1	35,8	31	7168	794	2,0	11,7
2	13,0	26	2680	435	4,0	5,0
3	75,2	32	14442	1020	1,8	17,0
4	166,5	31	30396	1008	6,9	22,5
5	76,2	31	14245	1286	1,8	26,2
6	111,7	31	21399	800	6,9	22,3
7	26,6	31	5150	804	1,8	10,9
8	109,8	31	19814	1093	7,4	36,3
9	82,5	31	15926	674	13,3	29,4
10	15,1	33	2889	683	2,3	12,3
11	63,3	31	11591	953	1,9	11,4
12	4,6	30	916	642	2,0	10,7
13	16,1	26	3233	442	4,1	5,1
14	79,1	31	14528	1042	1,7	13,3
15	77,8	31	14193	1257	2,6	26,4
16	9,4	28	1836	648	12,8	7,5
17	13,3	33	2590	507	2,0	11,0
18	24,1	26	4710	620	2,0	11,2
19	161,9	31	29265	1412	1,9	16,3
moyenne	61,5	30,3	11510,5	855,0	4,4	16,6

Une méthode de recherche par profondeur d'abord sur les réseaux 1, 2 et 3 constitue le troisième scénario. Cette méthode de recherche vise à trouver une première solution rapidement, puis utiliser l'information qu'elle contient pour borner les états générés subséquemment. Les résultats démontrent que la sélection des états permet d'obtenir une première solution rapidement (moins de 35 itérations). Toutefois, le processus de minimisation s'avère coûteux et génère un très grand nombre d'états. Par contre en comparant le nombre d'états retenus par l'arbre de recherche, pour une méthode en largeur et un méthode meilleur d'abord, on remarque un écart significatif. Les bornes appliquées à la recherche

Tableau 2.8: Résultats réseau 2 pour 3 véhicules, recherche par profondeur d'abord

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	801,9	22	165721	2069	0,0	13,2
1	154,8	22	30446	1391	0,0	10,2
2	25,0	25	4918	874	0,0	0,0
3	658,8	22	125503	1596	0,0	15,6
4	905,3	34	168537	1833	8,8	12,1
5	1800,8	22	315814	1667	0,0	14,1
6	312,6	22	53627	1459	0,0	10,8
7	110,0	23	18120	1333	0,0	3,4
8	2381,0	22	363686	1782	0,0	17,4
9	2637,0	22	307317	1094	6,8	29,4
10	144,0	23	21283	1082	3,1	7,5
11	285,6	22	36518	1357	0,0	10,0
12	215,8	23	30710	1095	4,7	8,8
13	47,0	27	6820	880	0,0	1,1
14	922,5	22	112627	1382	0,0	12,0
15	2776,0	22	316931	1785	0,0	13,9
16	75,2	24	9781	982	2,9	3,1
17	419,6	24	49832	1070	6,5	11,4
18	604,5	24	71840	1087	4,7	9,2
19	1844,8	22	190546	1517	0,0	14,8
moyenne	856,1	23,5	120028,9	1366,8	1,9	10,9

Tableau 2.9: Résultats réseau 3 pour 3 véhicules, recherche par profondeur d'abord

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	1711,3	26	339715	956	29,9	44,6
1	579,4	26	115057	586	32,7	31,2
2	522,4	26	103815	458	25,2	22,4
3	674,4	27	122961	1907	19,4	20,3
4	2523,7	26	430115	902	30,2	47,6
moyenne	1202,2	26,2	222332,6	961,8	27,5	33,2

par meilleur d'abord empêchent l'insertion de nombreux états inutiles. On remarque aussi que le temps de résolution est beaucoup moins constant pour une recherche avec profondeur d'abord. Pour l'ensemble des réseaux étudiés par ce scénario, les solutions initiales ne sont jamais optimales. Les temps de résolutions obtenues sur le réseau 1 sont généralement courts. Le réseau 2 provoque un très long temps de résolution pour certaines instances. Les instances formulées sur le réseau 3 ne permettent pas d'obtenir un temps de résolution efficace. Cependant on remarque qu'une première solution s'obtient en peu d'étapes. Le processus de minimisation est toutefois inefficace et requiert beaucoup d'étapes.

Dans un quatrième temps, comme il est difficile de trouver une première solution par le principe de recherche en largeur, on suggère d'appliquer une borne initiale plus sévère que l'infini. Ainsi, pour tous les problèmes analysés, une borne supérieure empêche de créer certains états ne menant vers aucune solution. Cette borne équivaut à une valeur environ 20 % supérieure au temps maximal rencontré. Les tableaux 2.10, 2.11 et 2.12 présentent les résultats obtenus par cette méthode.

Tableau 2.10: Résultats réseau 1 pour 3 véhicules, recherche largeur bornée

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	70,8	9434	150	9288	0,0	0,0
1	87,2	10922	213	10765	0,0	0,0
2	88,3	10041	240	9996	0,0	0,0
3	61,4	8263	144	8138	0,0	0,0
4	79,9	9940	134	9751	3,0	-6,7
5	69,3	8837	112	8665	0,0	0,0
6	75,0	9235	173	9105	20,6	13,3
7	62,6	7789	118	7649	0,0	0,0
8	66,8	8586	130	8447	0,0	0,0
9	62,7	7398	161	7323	6,7	-1,9
10	116,1	11845	592	12063	0,0	0,0
11	77,4	8796	107	8603	0,0	0,0
12	88,8	9926	395	9996	0,0	0,0
13	99,5	11311	340	11333	0,0	0,0
14	60,7	7656	103	7560	0,0	0,0
15	62,9	7803	103	7685	0,0	0,0
16	115,1	11800	276	11676	0,0	0,0
17	89,6	9781	380	9820	0,0	0,0
18	91,7	9908	395	9954	0,0	0,0
19	89,3	9851	158	9674	0,0	0,0
moyenne	80,8	9456,1	221,2	9374,6	1,5	0,2

Le quatrième scénario permet d'évaluer l'efficacité de l'algorithme lorsqu'une recherche en largeur utilise une meilleure borne supérieure. Les résultats démontrent des temps de résolutions relativement courts, ainsi qu'une diminution du nombre d'états conservés par l'arbre de recherche. De plus, cette méthode permet de résoudre efficacement les instances associées au réseau 3.

Tableau 2.11: Résultats réseau 2 pour 3 véhicules, recherche largeur bornée

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	37,5	5451	29	5338	0,9	-6,6
1	42,1	6586	85	6479	10,4	0,8
2	34,6	5314	79	5245	0,0	0,0
3	20,7	3558	4	3464	0,0	0,0
4	35,9	5697	13	5536	1,8	-6,1
5	28,2	4589	6	4464	4,2	-3,5
6	33,0	5226	35	5111	1,7	0,0
7	19,3	3234	4	3146	0,0	0,0
8	27,9	4453	19	4333	4,1	-3,6
9	22,4	3747	24	3656	0,0	0,0
10	56,1	7610	229	7593	0,0	0,0
11	28,3	4677	10	4543	10,4	1,4
12	37,7	5845	46	5705	0,0	0,0
13	40,5	6199	136	6151	0,0	0,0
14	17,0	2962	3	2900	7,3	-0,7
15	21,8	3793	6	3697	3,2	-4,1
16	54,7	7538	83	7373	0,0	0,0
17	36,4	5490	68	5392	0,0	0,0
18	38,5	5847	46	5691	0,0	0,0
19	36,2	5636	26	5478	6,2	-2,3
moyenne	33,4	5172,6	47,6	5064,8	2,5	-1,2

Tableau 2.12: Résultats réseau 3 pour 3 véhicules, recherche largeur bornée

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	87,3	11542	152	11409	0,0	0,0
1	130,0	13379	277	13316	0,0	0,0
2	92,8	9832	186	9783	0,0	0,0
3	21,2	3587	61	3578	0,0	0,0
4	116,8	13104	70	12892	0,0	0,0
5	48,6	6281	36	6182	0,0	0,0
6	87,9	10704	52	10520	0,0	0,0
7	15,1	2670	17	2634	0,0	0,0
8	43,6	5604	36	5540	0,0	0,0
9	71,2	8774	117	8744	0,0	0,0
10	220,7	16457	487	16568	0,0	0,0
11	63,4	6879	35	6788	0,0	0,0
12	72,1	7829	12	7704	0,0	0,0
13	112,2	12107	275	12110	0,0	0,0
14	21,6	3350	22	3324	0,0	0,0
15	37,5	4754	36	4730	0,0	0,0
16	132,7	12210	205	12130	0,0	0,0
17	77,5	7682	4	7555	0,0	0,0
18	67,8	8021	39	7942	0,0	0,0
19	110,1	11652	98	11480	0,0	0,0
moyenne	81,5	8820,9	110,9	8746,5	0,0	0,0

Pour l'ensemble des scénarios étudiés, les temps requis pour la génération, la validation et l'insertion des états est similaire. Malgré qu'il soit difficile des les évaluer avec précision, on peut tout-de-même affirmer que ces étapes de l'algorithme sont rapides. On remarque un écart significatif entre les temps moyens et les temps maximum. Ces écarts ne sont pas associés à un certain type d'instances et sont des événements rares. La fréquence des états générés, validés et insérés pour chaque itération s'avère une mesure exacte. Une analyse des résultats démontrent que le temps maximal retrouvé pour les étapes de générations et de validation ne concordent pas avec les fréquences maximales pour ces processus.

CHAPITRE 3 : ADAPATATION POUR L'ORIENTATION DES VÉHICULES

Tel que défini au chapitre 1, les véhicules utilisés dans un contexte minier possèdent certaines particularités. Une chargeuse-navette effectue ses manoeuvres grâce à un axe de rotation situé au centre de l'appareil. Ainsi, ce véhicule se contrôle tout aussi bien en marche avant ou arrière. Toutefois, le véhicule doit toujours se présenter en marche avant lorsqu'il emprunte le dernier arc de son parcours. Ainsi, l'algorithme doit évaluer le meilleur moment pour changer l'orientation d'un véhicule.

Ce chapitre couvre les ajustement effectués au modèle afin d'inclure les contraintes d'orientation.

Dans un premier temps, on présente une formulation pour un état orienté. Une deuxième section expose les cas où un véhicule change d'orientation. La section suivante illustre une problématique reliée au modèle orienté et présente les modifications nécessaires pour le rendre valide de nouveau.

3.1 Modélisation de l'orientation

Puisqu'il existe deux possibilités d'orientation pour un véhicule sur un arc (marche avant et marche arrière), l'ajout d'un vecteur binaire aux états permet d'inclure cette nouvelle contrainte au modèle. Ainsi, on représente un état orienté S_i par :

$$S_i = (\mathbf{p_i}; \mathbf{t_i}; \mathbf{o_i})$$

où

$$\mathbf{p_i} = (p_i^1, p_i^2, \ldots, p_i^k, \ldots, p_i^m)$$
$$\mathbf{t_i} = (t_i^1, t_i^2, \ldots, t_i^k, \ldots, t_i^m)$$
$$\mathbf{o_i} = (o_i^1, o_i^2, \ldots, o_i^k, \ldots, o_i^m)$$

où, o_i^k est une variable binaire pour représenter l'orientation du véhicule k ; i.e. $o_i^k = 1$ si le véhicule est en marche avant, sinon $o_i^k = 0$.

L'ajout des éléments $\mathbf{o_i}$ implique la création de 2^m nouveaux états pour chaque état défini à la sous-section 2.3.1. Ainsi on a pour un état sans orientation, 2^m états avec orientation.

Par exemple, pour un problème à deux véhicules, on a :

$$S_i = (p_i^1, p_i^2; t_i^1, t_i^2) \qquad S_i' = (p_i^1, p_i^2; t_i^1, t_i^2; 0, 0)$$
$$(p_i^1, p_i^2; t_i^1, t_i^2; 1, 0)$$
$$(p_i^1, p_i^2; t_i^1, t_i^2; 0, 1)$$
$$(p_i^1, p_i^2; t_i^1, t_i^2; 1, 1)$$

La résolution d'un problème orienté implique un plus grand nombre de possibilités pour l'algorithme. Cependant, pour chaque étape, la taille de l'ensemble

des actions Γ_{S_i} d'un état demeure équivalente pour un problème orienté ou non. Puisqu'un changement d'orientation découle d'une suite d'actions, il est impossible d'obtenir deux orientations différentes pour deux états identiques à partir du même prédécesseur. Pour chaque action, aucun des états admissibles possèdent une configuration identique sur $\mathbf{p_i}$. L'orientation des éléments de l'ensemble des actions Γ_{S_i} s'obtient à l'aide de S_i et des prédécesseurs de celui-ci. La section suivante traite des procédures pour établir l'orientation d'un état.

3.2 Changement d'orientation

Un véhicule change d'orientation que sous deux conditions. Dans un premier temps, un véhicule modifie son orientation s'il passe de marche avant à marche arrière sur un même arc. Ce phénomène se produit lorsqu'un véhicule atteint un point de chargement ou de déchargement. Ainsi le véhicule arrive en marche avant par un arc (a, b) et quitte en marche arrière par l'arc inverse (b, a). Le modèle permet un changement d'orientation de ce type que pour les arcs incidents à un noeud de fin de parcours ; c'est-à-dire les noeuds de degré 2, où le véhicule ne peut utiliser qu'une seule voie physique pour s'y rendre ou en sortir. Le degré du noeud est égal à 2 car chaque voie de transport se représente par deux arcs, soit un arc émergent et un arc incident au noeud. Les cas où un véhicule change de sens sur un même arc, outre ceux de fin de parcours, ne sont pas valides. Ces déplacements représentent un allez-retour sur un même arc. Ils ne pemettent pas d'obtenir de meilleure solution. Ils sont équivalents à faire attendre un véhicule à l'embouchure d'une intersection. Ainsi, l'algorithme doit changer l'orientation d'un véhicule que lorsque celui-ci arrive à destination.

Dans un deuxième temps, il y a changement d'orientation lorsqu'un véhicule traverse deux fois le même noeud consécutivement. Ces modifications aux sens

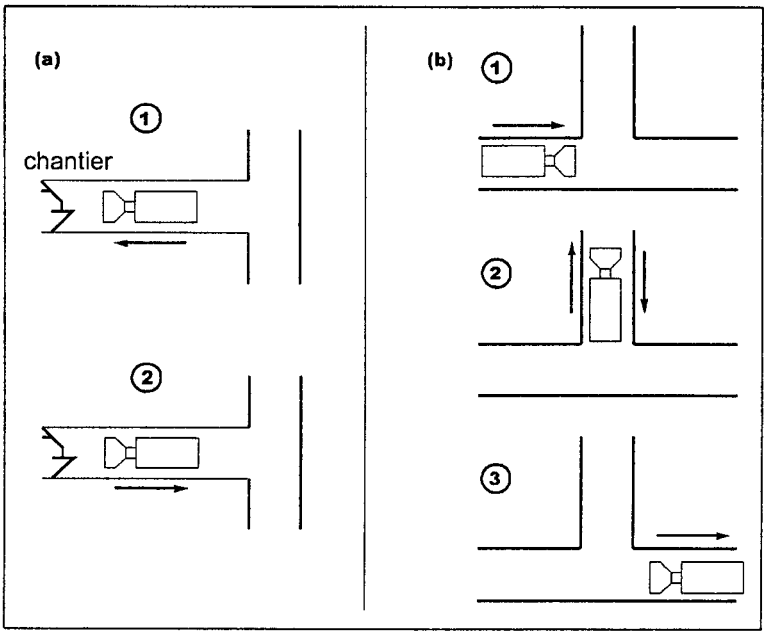

Figure 3.1: Types de changement d'orientation. (a)Changement d'orientation sur un arc. (b) Changement d'orientation sur un noeud.

des véhicules sont plus complexes à modéliser et affectent grandement la validité d'une solution. D'une manière générale, les travaux de Bigras définissent qu'un changement d'orientation se produit lorsqu'un cycle $i - j - i$ est élément du parcours d'un véhicule. La figure 3.1 (b) illustre les déplacements effectués par un véhicule afin d'inverser son orientation. Cependant, la nature du graphe ne permet pas d'utiliser cette condition sous cette forme. Elle est valide pour le graphe utilisé dans les travaux de Bigras puisque chaque noeud du graphe représente un lieu d'attente ; par exemple, trois noeuds servent à représenter une intersection à trois voies. Le véhicule de la figure 3.1(b) emprunte chacun de ces noeuds pour changer son orientation. Pour le modèle présenté dans ce document, les points d'attente sont aux extrémités des arcs. Les procédures de détections de cycle, élaborées par les travaux de Bigras, ne peuvent pas être appliquées directement. Cependant, il est possible d'utiliser des principes semblables pour établir l'orientation d'un véhicule à partir des configurations qui le précèdent.

Selon la structure de la méthode de résolution, les cas de chagements d'orientation caractérisés par la figure 3.1 (a) surviennent lorsqu'un véhicule quitte un noeud à l'aide de l'arc inverse utilisé pour atteindre le noeud. Dans un deuxième temps, on peut détecter un changement d'orientation tel qu'illustré par la figure3.1 (b) lorsqu'un véhicule se déplace d'un noeud vers un arc incident à ce noeud pour ensuite emprunter un arc émergent à ce même noeud. Toutefois un cas particulier se présente quand un véhicule emprunte les arcs d'où il provenait afin de se retourner. Ces conditions rendent le changement d'orientation invalide.

La figure 3.2 permet d'illustrer ces propos. L'exemple simule le changement d'orientation pour trois véhicules. Le parcours de chacun des trois véhicules représente les trois cas proposés précédemment. Le premier véhicule du système effectue des manoeuvres ne résultant pas en un changement d'orientation. Il

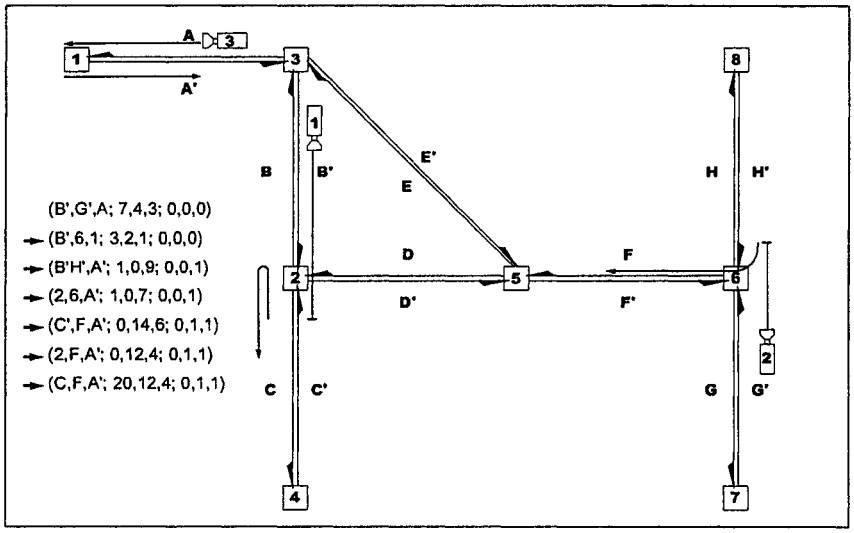

Figure 3.2: Exemples de changements d'orientation

utilise l'arc incident C' et l'arc émergent C consécutivement. Le véhicule 1 est en marche avant lorsqu'il voyage sur l'arc B'. Son parcours démontre qu'il est toujours en marche avant lorsqu'il atteint l'arc C. Le trajet emprunté par le deuxième véhicule caractérise un changement d'orientation tel qu'illustré par la figure 3.1 (b). Il traverse donc l'arc H' incident au noeud 6, pour ensuite traverser de nouveau le noeud 6 et aboutir sur l'arc émergent F. Finalement, le troisième véhicule change d'orientation en emprutant les arcs A et A', ainsi que le noeud 1. On remarque que le parcours du premier véhicule contient aussi une séquence de deux arcs inverses. Il faut alors vérifier le cas particulier pour les deux types de revirement.

Dans un premier temps, une procédure pour détecter un changement d'orientation sur deux arcs inverse se formule par :

Procédure 3.1 (a) : Pour l'action $S_i \rightarrow S_j$

$$p_i^k \in N; p_j^k \in A,$$
$$\text{il y a changement d'orientation si}$$
$$p_h^k = (p_j^k)^{-1}$$
$$p_g^k \neq p_i^k$$

où, $p_h^k \in A$ est le prédécesseur de p_i^k et $p_g^k \in N$ est le prédécesseur de p_h^k.

En imposant $p_g^k \neq p_i^k$, l'orientation du véhicule 1 selon l'exemple de la figure 3.2 demeure la même. Dans un deuxième temps, la procédure 3.1 (b) sert à détecter les chagements d'orientation effectué sur des arcs différents.

Procédure 3.1 (b) : Pour l'action $S_i \rightarrow S_j$

$$p_i^k \in N; p_j^k \in A,$$
$$\text{où } p_j^k \text{ est émergent à } p_i^k$$
$$\text{il y a changement d'orientation si}$$
$$p_h^k \text{ est incident à } p_g^k$$
$$\text{et } \ p_g^k \neq p_i^k$$

où, $p_h^k \in A$ est le prédécesseur de p_i^k et $p_g^k \in N$ est le prédécesseur de p_h^k.

Cependant, cette modélisation s'avère insuffisante pour obtenir une solution optimale du problème. Le principe de dominance n'est plus applicable car la validité d'un état dépend maintenant d'une suite de prédécesseurs. Il en découle que la dominance entre les états n'est plus exacte. La section suivante présente la problématique de la dominance lorsque le modèle utilise des états orientés, ainsi que les adaptations apportées pour rendre le modèle valide de nouveau.

3.3 Problématique d'orientation sur un noeud

Lorsqu'un véhicule se positionne à un noeud, son orientation est indéterminée. En effet, à l'étape suivante, le véhicule change, ou non, d'orientation. Toutefois, l'orientation d'un véhicule à une étape dépend de l'information contenue par l'état avant que le véhicule traverse le noeud. Ainsi, l'état représentant un véhicule à un noeud contient de l'information biaisée par son prédécesseur. Cette caractéristique résulte en la création d'ensembles d'actions Γ différents pour un même état et compromet l'optimalité de la solution.

Le modèle présenté jusqu'ici ne peut être utilisé pour résoudre un problème avec orientation. Cependant, les résultats des travaux démontrent qu'il est possible de modifier le modèle pour le rendre exact de nouveau.

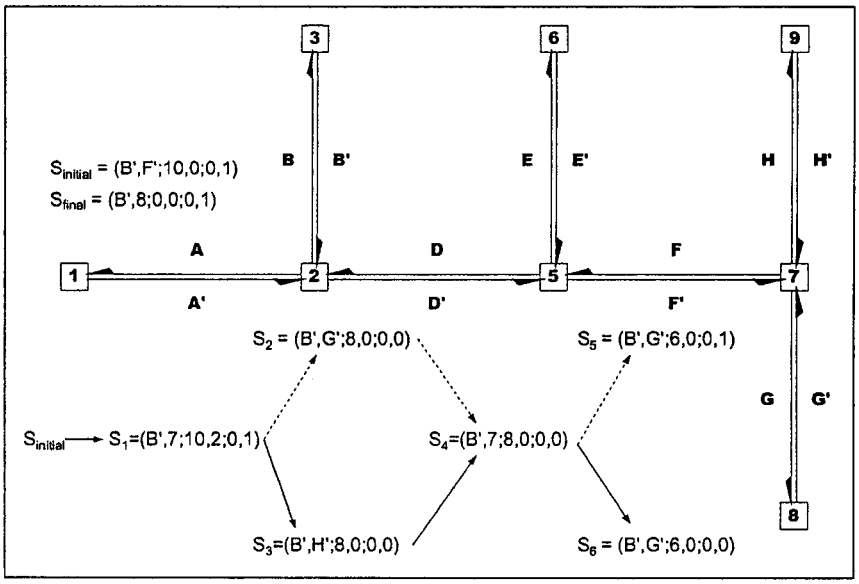

Figure 3.3: Exemple d'un état biaisé

L'exemple suivant sert à démontrer la fausse équivalance qui existe entre deux états orientés. Soit un système composé de deux véhicules. Tel qu'illustré à la

figure 3.3, à un instant donné, les véhicules occupent la configuration exprimée par $S_{initial}$ et recherche à atteindre S_{final}. Le deuxième véhicule du système ne possède pas la bonne orientation et doit effectuer un revirement avant d'accéder au noeud 8. Ainsi, le véhicule 2 se présente au noeud 7, via l'arc F', et a deux possibilités pour changer de direction : les arcs G' et H'. Ces deux actions distinctes génèrent toutes deux le même état S_4. Cependant, les actions découlant de S_4 sont différentes selon que si S_2 ou S_3 est le prédécesseur. Ainsi, le sens d'un véhicule est incertain tant qu'il n'a pas quitté le noeud utilisé pour son changement d'orientation.

3.3.1 Modification à la modélisation des états

À partir du constat de l'orientation incertaine des véhicules, il faut remettre en question la pertinence de l'information sur un noeud. Il est important de savoir le moment où un véhicule se trouve à un noeud. Cette information permet de valider la réalisabilité d'un état par rapport à son successeur. Par ailleur, le parcours d'un véhicule, d'un arc vers un autre arc, implique nécessairement le passage sur un noeud. De plus, le noeud en question est unique pour chaque combinaison d'arcs réalisable. Ainsi, l'information contenue sur une étiquette à un noeud est liée aux arcs de son prédécesseur et de son successeur.

Un nouveau modèle, découlant de celui proposé pour des problèmes sans orientation, permet de résoudre un problème orienté. Puisqu'il est possible d'obtenir implicitement le temps de passage à un noeud, il n'est pas nécessaire de conserver l'état représentant le passage d'un véhicule à un noeud. Le temps de passage à un noeud s'obtient grâce aux positions représentées par le prédécesseur à un état S_j. Il est possible de construire une solution optimale à partir de séquences d'états uniquement constituées d'arcs. Ainsi, pour un état S_i le vecteur $\mathbf{p_i} \in A$.

L'ensemble Γ des actions réalisables de S_i constitue toutes les combinaisons réalisables d'arcs adjacents aux noeuds par lesquels les arcs du vecteurs $\mathbf{p_i}$ sont incidents. Pour un véhicule se trouvant sur l'arc B' sur la figure 3.3, ces possibilités sont :

$$B' \rightarrow \{A, A', B', D, D'\}$$

Une suite d'états représente le déplacement des véhicules uniquement sur les arcs du problème. Le noeud d'origine d'un élément p_i^k s'obtient par sa position précédente. Il est alors possible de connaître le temps de passage d'un véhicule à un noeud et d'en déduire les conflits éventuels. Cependant, la formulation des propositions et des procédures doit être adaptée pour cette nouvelle démarche.

3.3.2 Modification aux contraintes

Pour le déplacement des véhicules d'arc en arc, les types de conflits demeurent les mêmes. Des collisions peuvent se produire aux intersections, ainsi que sur les voies de transports. Les propositions 3.1 à 3.6 permettent de gérer les conflits entre les successions d'états représentés seulement par les arcs du réseau.

Tout d'abord, il faut empêcher la présence de deux véhicules à une même intersection au même moment. Lors du processus de résolution, un ensemble Γ_{S_i} ne doit pas contenir des états créant ce type de conflits. L'indentification des états non conformes requiert la connaissance de l'information sur les prédécesseurs d'un état. Il faut d'abord connaître le noeud prédécesseur par lequel un véhicule k arrive sur l'arc p_j^k. Cette information s'obtient à l'aide de l'arc parcouru précédemment par le véhicule k. Ainsi, il faut retrouver le prédécesseur de S_j

où $p_i^k \neq p_j^k$. Puisque cette information découle directement des prédécesseurs d'un état et qu'elle ne représente pas un critère de décison pour le modèle, elle se conserve pour chaque noeud de l'arbre sans alourdir la résolution du problème. Le vecteur $\bar{\mathbf{p}}_\mathbf{i}$ contient l'identification des intersections du réseau utilisés par les véhicules d'un système pour obtenir la configuration représentée par un état S_i. Similairement, chaque état conserve un vecteur $\bar{\mathbf{t}}_\mathbf{i}$ pour représenter le temps d'arrivée aux intersections. Cette deuxième information permet de valider ou non un conflit lorsqu'un état représente le parcours de deux véhicules ayant empruntés la même intersection. En effet, un même état peut contenir deux éléments $\vec{p}_i^k$ et $\vec{p}_i^l$ identiques sans représenter un état conflictuel. Un conflit se produit lorsque le temps d'arrivée à un noeud pour un véhicule est en conflit avec la fenêtre de temps reservée par un autre véhicule ($[\vec{t}_i^k, \vec{t}_i^l$ + F.S.]).

On formule la contrainte d'unicité aux intersections par :

Proposition 3.1 : L'action $S_i \rightarrow S_j$ est non valide si

$$\vec{p}_j^k = \vec{p}_j^l$$
$$\vec{t}_j^k \geq \vec{t}_j^l \text{ et } \vec{t}_j^k < \vec{t}_j^l + F.S. \quad (\vec{t}_j^l \geq \vec{t}_j^k \text{ et } \vec{t}_j^l < \vec{t}_j^k + F.S.)$$

Pour un état admissible, tous les véhicules possèdent une fenêtre de temps unique sur l'intersection utilisée pour atteindre leurs positions décrites par $\mathbf{p_i}$. Ainsi, pour une suite d'états, représentant une solution complète ou partielle, les véhicules empruntent les noeuds de leurs parcours selon un horaire défini. Grâce à la proposition 3.1, l'horaire contient une fenêtre de temps réservée pour chaque noeud utilisé par les véhicules. De plus, aucune intersection entre les fenêtres de temps d'un même noeud n'est permise. Il en résulte le respect de l'unicité des véhicules à un noeud pour un parcours représenté par une suite d'états.

Toutefois, un second type de conflit peut se produire lorsque deux véhicules traversent le même noeud, mais aboutissent sur des voies différentes. Un conflit peut survenir à une intersection même si chacun des véhicules l'empruntent à des temps différents, car la progression des états se réalise uniquement sur les arcs. Lorsque deux véhicules changent d'arc et traversent la même intersection, le premier véhicule à franchir le noeud ne peut se retrouver sur l'arc, ou l'arc inverse, d'où provient le deuxième véhicule. Par exemple, à partir du graphe de la figure 3.3, soit un état $S_i = (A', D; 0, 5)$ et $h(S_i) = 20$. Alors, le temps de passage pour les véhicules 1 et 2 est respectivement de $[20, 22]$ et de $[25, 27]$ pour un facteur de sécurité égale à deux unités de temps. Bien que les fenêtres de temps ne se croisent pas, l'état successeur $S_j = (D, B; 0, 10)$ représente un conflit sur le noeud 2. Le premier véhicule traverse le noeud 2 pour se retrouver sur l'arc D. Cependant, lorsque le premier véhicule arrive sur l'arc D, le deuxième véhicule occupe l'arc pour encore trois unités de temps avant de se retrouver sur l'arc B. De plus, on peut retrouver un croisement des véhicules lorsqu'ils se situent sur le meme arc et que le dernier véhicule sur l'arc quitte avant le premier. Par exemple, la suite d'état $(A', A'; 0, 5) \rightarrow (A', D'; 0, 10)$ occasionne un conflit. Ainsi, lors de la progression des états, il faut éliminer les états dont le passage à une même intersection résultent en un conflit sur un arc. La proposition 3.2 permet de détecter ce type de conflits selon l'ordre d'arrivée des véhicules à l'intersection conflictuelle.

Proposition 3.2 : Soit

$$\vec{p}_j^k = \vec{p}_j^l \text{ et } \vec{t}_j^k < \vec{t}_j^l \text{ (ou } \vec{t}_j^k > \vec{t}_j^l)$$

L'action $S_i \rightarrow S_j$ est non valide si

$$p_j^k = p_i^l \text{ ou } (p_i^l)^{-1} \quad (p_j^l = p_i^k \text{ ou } (p_i^k)^{-1})$$

Soit

$$p_i^k = p_i^l \text{ et } t_i^k < t_i^l \text{ (ou } t_i^k > t_i^l)$$

et $p_i^k = p_j^k$ (ou $p_i^l = p_j^l$)

L'action $S_i \rightarrow S_j$ est non valide si

$$p_j^l \neq p_i^l \quad (p_j^k \neq p_i^k)$$

Il est important de noter que la proposition évalue seulement les états, et leurs successeurs directs, pour lesquels au moins deux véhicules changent d'arcs de l'étape i à j tout en passant par le même noeud. Pour une suite d'états $(A', D; 5, 0) \rightarrow (A', B'; 3, 10) \rightarrow (D', B'; 10, 5)$, obtenue à partir du graphe de la figure 3.3, la propositon 3.2 n'évalue aucune suite de deux états. Pour la suite d'états présentés ci-haut, chaque progression s'avère exacte pour la proposition 3.2, car la césure du déplacement des véhicules entre les états assure l'ordre de passage à un noeud. La progression des états implique nécessairement le déplacement d'un arc vers un autre arc pour au moins un des véhicules à chaque étape.

Pour une suite d'états $S_i \rightarrow S_j$, sous les conditions $p_i^k = p_j^k$ et $p_i^l \neq p_j^l$ et $\bar{p}_j^k = \bar{p}_j^l$, un conflit éventuel ne peut que découler des conditions reliées à un conflit sur un arc. Le véhicule k traverse l'intersection $\bar{p}_i^k$ avant l'étape j et puisque sa position ne change pas de l'étape i à j, $\bar{p}_i^k = \bar{p}_j^k$ et $\bar{t}_i^k = \bar{t}_j^k$. Ainsi, la valeur de $\bar{t}_i^k$ est nécessairement inférieure à la valeur de $h(S_j)$ car le véhicule traverse le noeud à l'étape i ou à une étape précédente. Donc, si le véhicule k, qui est le premier à traverser l'intersection, se positionne sur un même segment que celui où se déplace le véhicule l, alors il y a conflit sur l'arc à l'étape i. Ainsi, pour accélérer le processus, la proposition 3.2 évalue seulement les suites d'états où au moins deux véhicules executent un déplacement, par la même intersection, de l'étape i vers j. Les cinq propositions suivantes servent à détecter tous les types de conflits sur les arcs pour une suite d'états.

Les collisions frontales et les collisions de rattrapage représentent les deux types de conflits qui peuvent se produire sur un arc. Aucune suite d'états ne doit résulter en une collision frontale. Cependant, il est possible d'ajuster les éléments d'un état pour valider une action préalablement sous les conditions d'un conflit de rattrapage. Sans ces ajustements sur les états, le modèle ne peut permettre l'attente de plusieurs véhicules sur le même arc pour des cas particuliers et la solution n'est plus optimale dans tout les cas. Par exemple, pour un état $S_i = (F', F', H'; 5, 0, 10)$ et pour une destination $S_{fin} = (G, H, E; 0, 0, 0)$ obtenuent à partir de la figure 3.3, aucune suite d'états mène vers une solution optimale. Pour obtenir une solution optimale, les deux premiers véhicules doivent attendre que le troisième véhicule libère l'arc H'. L'ensemble Γ_{S_i} contient un successeur se générant sous la forme $S_j = (F', F', G'; 0, 0, 0)$. Ce successeur n'est pas valide car deux véhicules occupent la même position sur un arc. Toutefois, cette configuration est un élément du trajet minimal pour les véhicules du système. De plus, elle ne peut s'obtenir de manière équivalente par les successeurs

des états qui sont éléments de Γ_{S_i}. C'est-à-dire qu'il est impossible d'obtenir S_j sous sa forme minimale. Pour conserver l'optimalité de la solution, les propositions permettent d'ajuster les valeurs d'un état pour le rendre valide. Ainsi, S_j devient $(F', F', G'; F.S., 0, 0)$, où F.S. est le facteur de sécurité du système. Un tel ajustement sur les états peut toutefois générer un autre type de conflit : la saturation en véhicules sur un même arc. Cependant, cette problématique peut facilement être contournée en limitant le nombre de véhicules sur un arc et son inverse. Selon les caractéristiques d'un graphe, il est possible de remarquer que le nombre maximal de véhicules sur un même segment est le modulo de $\frac{c_{\alpha\beta}}{F.S.}$. Ainsi, une contrainte fixe le nombre maximal de véhicules pouvant se retrouver sur un même segment. L'algorithme demeure optimal et réalisable avec un ajustement sur les états.

L'ajustement des états, qui est réalisable tout en conservant l'optimalité de la solution, permet de développer des propositions pour identifier et modifier tous les états menant vers un conflit sur un arc. Deux informations sont nécessaires pour identifier correctement un conflit éventuel. Dans un premier temps, la nature du conflit n'est pas identique pour deux véhicules provenant du même noeud ou lorsqu'ils proviennent de deux noeuds différents. Dans un deuxième temps, la gestion des conflits s'opère différemment si un véhicule emprunte un arc incident ou émergent au noeud de sa provenance. Comme premier cas, la proposition 3.3 cherche à détecter un conflit de rattrapage sur un même arc lorsque deux véhicules proviennent du même noeud et que leur arc de destination est émergent du noeud de provenance. Par exemple, la suite d'états $(D', E', H'; 0, 0, 10) \rightarrow (D', F', H'; 0, 5, 8) \rightarrow (F', F', H'; 5, 3, 6) \rightarrow (F', F', G'; 0, 0, 0)$, tirés à partir du graphe de la figure 3.3, possède un état conflituel. Le dernier état de la suite doit être ajusté afin de conserver la réalisabilité de la solution. L'ajustement se fait selon l'ordre d'arrivée des véhicules sur l'arc conflituel. Pour l'exemple, le

véhicule 1 arrive après le véhicule 2. Il est donc nécessaire d'ajuster le temps du véhicule 1 pour éviter un conflit. Il faut remarquer que le véhicule 2 a priorité car les deux véhicules s'engagent sur un arc émergent au noeud 7. Ainsi, puisque les véhicules voyagent déjà à leur vitesse maximale, la seule option est de faire ralentir un véhicule. Dans ce cas-ci, le véhicule 2 doit ralentir afin de conserver une distance sécuritaire entre les véhicules. La proposition 3.3 ajuste les éléments de l'état sous ces conditions de façon à le conserver au sein de l'arbre de recherche. La proposition 3.3 se formule par :

Proposition 3.3 : Soit

$$p_j^k = p_j^l \text{ et } \bar{p}_j^k = \bar{p}_j^l$$
$$\text{et } p_j^k \text{ est émergent à } \bar{p}_j^k$$

$$\text{L'action } S_i \rightarrow S_j \text{ est ajustée si}$$

$$\bar{t}_j^k < \bar{t}_j^l$$
$$\text{et } t_j^k + F.S. > t_j^l$$

$$\text{alors,}$$

$$t_j^l = t_j^k + F.S.$$

Il est important de noter que lorsque plus de deux véhicules se retrouvent en conflit sur un même arc, il faut tout d'abord trier les éléments selon leur temps de passage au noeud. Ainsi, le véhicule ayant le temps de passage au noeud le plus tôt possède alors un temps de parcours exact. Le deuxième véhicule

ajuste alors son temps de parcours selon celui du premier véhicule. Puisque le temps du deuxième véhicule à franchir le noeud s'avère exact selon celui du premier, le temps de parcours pour le troisième véhicule peut s'obtenir à partir des informations du second véhicule. Ainsi de suite, l'algorithme ajuste les temps de chaque véhicule selon le temps de parcours du véhicule qui le précède au noeud. Cette proposition implique tout d'abord un filtrage des états d'un ensemble Γ_{S_i} par les propositions 3.1 et 3.2. De plus, tel que discuté, un triage des éléments du vecteur $\mathbf{p_i}$ selon l'ordre d'arrivée le plus court au noeud est nécessaire. Puisque la taille du vecteur $\mathbf{p_i}$ est égale au nombre de véhicules dans le système et puisque que ce nombre est petit, il n'est pas avantageux de greffer un algorithme de tri performant. Cependant, pour l'implantation relative aux travaux, un triage par monceau trie les éléments.

Lorsque des véhicules empruntent un même segment émergent (a, b) à un noeud a, l'ordre de priorité des véhicules s'établit selon le temps de passage le plus tôt au noeud a. Cet ordonnancement des véhicules assure que le premier véhicule qui accède un arc soit le premier à en sortir. Cependant, lorsque plusieurs véhicules utilisent un arc incident (b, a) au noeud a, la priorité entre les véhicules s'inverse. En effet, puisque les véhicules empruntent un arc incident à un noeud, ils devront nécessairement traverser de nouveau ce même noeud. Par exemple, la suite d'états$(F', F', G; 4, 0, 10) \rightarrow (F', H', G; 2, 0, 8) \rightarrow (H', H', G; 0, 0, 4) \rightarrow (H', H', F; 0, 0, 10)$, tirés à partir du graphe de la figure 3.3, représente un tel type de conflits. Il est possible de remarquer que l'état $(H', H', G; 0, 0, 4)$ génère le conflit car deux véhicules occupent la même position au même temps. De plus, il faut aussi noter que le véhicule 1 doit quitter l'arc H' avant le véhicule 2. Ainsi, en ordonnant les véhicules selon le temps de passage le plus tard sur le noeud et en utilisant le principe de la proposition 3.3, il est possible de développer la propostion 3.4 pour le cas où plusieurs véhicules se retrouvent sur

le même arc incident au noeud par lequel ils l'ont accédé. Ainsi, pour l'exemple précédent, la proposition doit apporter les modifications nécessaires pour obtenir l'état $(H', H', G; 0, 2, 4)$. Dans ce cas-ci, le premier véhicule accède au segment en dernier et le quitte obligatoirement en premier. Pour rendre l'état valide, le véhicule 2 doit reculer afin de créer un espace suffisant entre les véhicules. La proposition 3.4 se formule par :

Proposition 3.4 : Soit

$$p_j^k = p_j^l \text{ et } \bar{p}_j^k = \bar{p}_j^l$$
$$\text{et } p_j^k \text{ est incident à } \bar{p}_j^k$$

L'action $S_i \rightarrow S_j$ est ajustée si

$$\bar{t}_j^k > \bar{t}_j^l$$
$$\text{et } t_j^k + F.S. > t_j^l$$

alors,

$$t_j^l = t_j^k + F.S.$$

Les propositions 3.3 et 3.4 sont semblables. Elles ne diffèrent que par la priorité accordée à chacun des véhicules selon l'orientation des arcs conflictuels. Cependant, elles assument que les véhicules empruntent le même noeud pour atteindre l'arc causant un conflit. Une deuxième situation se présente lorsque deux véhicules se retrouvent sur un même arc et que les noeuds d'arrivée sont différents pour les véhicules en confits. Par exemple, à partir de la figure 3.4,

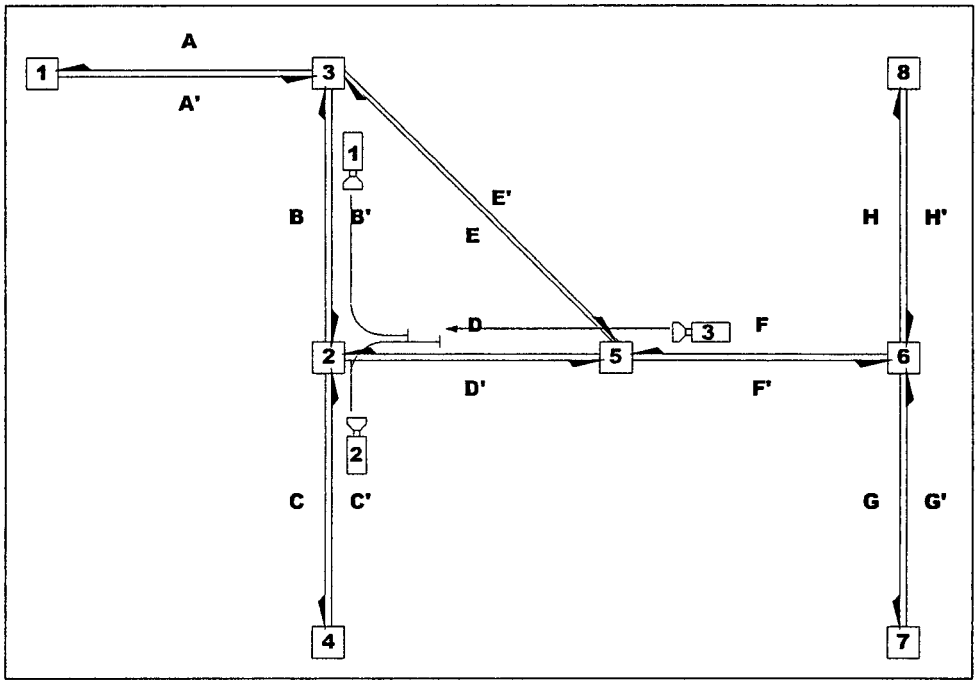

Figure 3.4: Exemple d'un conflit sur un même arc

la suite d'états $(B', C', F; 7, 3, 2) \rightarrow (D, D, D; 0, 0, 1)$ se doit d'être réalisable. Tout comme pour les deux propositions précédentes, il est nécessaire d'ajuster les valeurs d'un état pour conserver sa validité. Cependant, il n'est plus possible d'ajuster le temps des véhicules selon leur ordre d'arrivée au noeud d'entrée sur l'arc conflictuel. Par contre, le sens accordé à chacun des arcs permet d'établir la priorité entre les véhicules. Puisque la situation demande que les véhicules voyagent sur le même arc et que leur noeud d'accès soit différent, il en découle qu'un véhicule utilise un noeud émergent, tandis qu'un deuxième véhicule utilise nécessairement un noeud incident. De plus puisqu'un arc incident à un noeud signifie un point d'attente ou de revirement pour un véhicule, celui-ci est toujours le premier à quitter l'arc. Pour ajuster les états non conformes, la proposition 3.5 se formule comme suit :

Proposition 3.5 : Soit

$$p_j^k = p_j^l \text{ et } \bar{p}_j^k \neq \bar{p}_j^l$$

L'action $S_i \rightarrow S_j$ est ajustée si

$$\begin{array}{ll} p_j^k \text{ est incident à } \bar{p}_j^k & (p_j^l \text{ est incident à } \bar{p}_j^l) \\ \text{et } t_j^k + F.S. > t_j^l & (t_j^l + F.S. > t_j^k) \end{array}$$

alors,

$$t_j^l = t_j^k + F.S. \qquad (t_j^k = t_j^l + F.S.)$$

Ces trois dernières propositions servent à ajuster le temps de certains états occasionnant des conflits. Ces ajustements sont nécessaires à l'obtention d'une solution optimale. Elles réparent les conflits en établissant la priorité par comparaison sur deux véhicules à la fois. Lorsque plusieurs véhicules sont en conflits sur un même arc, plusieurs propositions peuvent être nécessaires pour ajuster correctement l'état. Pour l'exemple précédent, l'état $S_j = (D, D, D; 0, 0, 1)$ occasionne deux types de conflits : un conflit sur un arc incident à un noeud utilisé par deux véhicules différents et un conflit sur un arc par deux véhicules, ou plus, provenant de noeuds différents. Les proprositions 3.4 et 3.5 permettent de gérer ce type de conflit. Cependant, si la proposition 3.5 traite l'état avant la proposition 3.4, alors un conflit sur l'état est toujours présent. Pour l'état $S_j = (D, D, D; 0, 0, 1)$, la proposition 3.5 ajuste les éléments du vecteur $\mathbf{t_i}$ afin de conserver la priorité des deux premiers véhicules, car ils proviennent d'un noeud incident. L'état devient $S_j = (D, D, D; 0, 0, 2)$. La proposition 3.4, quant à elle, ajuste les temps afin de conserver la priorité entre les deux véhicules provenant du même noeud. L'état devient $S_j = (D, D, D; 0, 2, 2)$ et représente toujours un conflit. Ainsi, une importance se rattache à l'ordre utilisé pour ajuster les états. Les propositions 3.3 à 3.5 demandent que tous les états représentant un conflit sur un noeud soit préalablement retirés de l'ensemble des actions réalisables. Lors du processus de filtrage d'un ensemble d'actions Γ, les propositions 3.1 et 3.2 agissent en premier. Ensuite, la proposition 3.4 permet d'ajuster les conflits survenant sur un même arc et incident au noeud d'arrivée. Les véhicules atteignant un arc à partir d'un noeud incident impliquent qu'ils doivent traverser de nouveau le même noeud pour sortir du segment emprunté. Alors que les véhicules provenant d'un arc émergent à un noeud quittent nécessairement l'arc après le départ de tous les véhicules préalablement en attente sur le même arc. Ainsi, après avoir traité l'ensemble Γ à l'aide de la proposition 3.4, chaque état conserve un facteur de sécurité entre les véhicules en attente sur un arc conflictuel. Grâce à la connaissance de la position du véhicule en attente le plus éloigné du noeud de sortie de

Tableau 3.1: Exemple d'ajustement sur un état représentant plusieurs conflits de rattrapage

Propositions	Ajustements
3.4	$(D, D, D, D; 0, 2, 1, 4)$
3.5	$(D, D, D, D; 0, 2, 4, 4)$
3.3	$(D, D, D, D; 0, 2, 4, 6)$

l'arc, la propositon 3.5 est alors en mesure d'ajuster correctement le temps des véhicules provenant du noeud dont l'arc est émergent. Finalement, la proposition 3.3 est mise en oeuvre pour établir la priorité entre les véhicules provenant d'un même noeud et se retrouvant sur un arc émergent à ce noeud. Par exemple, pour une suite détats $(B', C', F, E'; 7, 3, 2, 4) \rightarrow (D, D, D, D; 0, 0, 1, 3)$, tirés de la figure 3.4, les propositions ajustent l'état selon l'ordre résumé au tableau 3.1.

Finalement, un dernier cas de conflits sur un arc se présente lorsque deux véhicules, ou plus, occupent deux arcs inverses. Encore une fois, il faut analyser les situations selon si les véhicules empruntent le même noeud, ou non, pour accéder aux arcs. Contrairement aux cas où plusieurs véhicules se retrouvent sur le même arc, lorsque des véhicules circulent sur des arcs inverses l'action ne peut qu'être valide ou non. Il n'est pas possible d'ajuster un tel type d'état pour en conserver sa validité.

Tout d'abord, lorsque deux véhicules accèdent deux arcs inverses par le même noeud, il est possible de conclure qu'un des véhicules occupe un arc émergent au noeud, tandis que l'autre véhicule voyage sur un arc incident. Par exemple la suite d'états $(E', F; 3, 7) \rightarrow (D', D; 0, 6)$, tirés du graphe de la figure 3.4, représente ce type de conflit. Pour $h(S_i) = 20$, la fenêtre de temps réservée pour le passage du premier véhicule sur le noeud 5 du graphe est [23,25]. Pour le deuxième véhicule, la fenêtre de temps réservée est de [27,29]. Ainsi, le premier

véhicule accède l'arc D' en premier. Il est alors impossible pour le deuxième véhicule de parcourir l'arc D. Les seules situations acceptées sont lorsque tous les véhicules empruntant l'arc incident traversent l'intersection avant que les véhicules se positionnent en attente sur l'arc inverse. Ainsi, pour un conflit de véhicules circulant sur deux arcs inverses et provenant d'un même noeud, la proposition 3.6 détecte les états non valide.

Proposition 3.6 : Soit

$$p_j^k = (p_j^l)^{-1} \text{ et } \bar{p}_j^k = \bar{p}_j^l$$

L'action $S_i \rightarrow S_j$ est non valide si

p_j^k est émergent à $\bar{p}_j^k$
et $\bar{t}_j^k > \bar{t}_j^l$

Dans un deuxième temps, il est nécessaire d'identifier les conflits se produisant sur deux arcs inverses atteints par les véhicules selon deux noeuds différents. Lorsqu'une telle situation se produit, les véhiucules peuvent soit se diriger un vers l'autre ou se retrouver en attente. La figure 3.5 illustre les deux situations. Dans le cas de la figure 3.5 (a), les véhicules provoquent une collision frontale. Il est donc nécessaire d'éliminer un état représentant un tel type de conflits. Toutefois, la situation illustrée par la figure 3.5 (b) est valide. On remarque alors qu'un conflit survient lorsque les véhicules voyagent sur un arc émergent à leur noeud d'entrée sur l'arc. De plus, puisque les véhicules proviennent de noeuds différents, si un véhicule emprunte un arc émergent, alors le deuxième

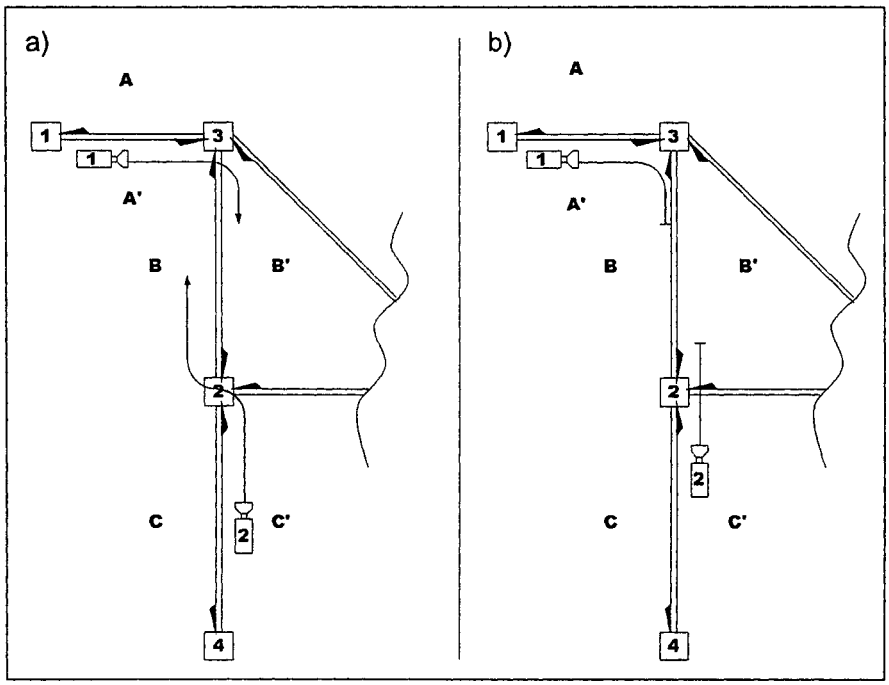

Figure 3.5: Exemples de conflits sur des arcs inverses. (a) Les véhicules traversent leur arc respectif. (b) Les véhicules se positionnent en attente.

véhicule circule obligatoirement sur un arc émergent aussi, car les deux véhicules empruntent des arcs inverses. La proposition 3.7 permet de gérer ce type de situation :

Proposition 3.7 : Soit

$$p_j^k = (p_j^l)^{-1} \text{ et } \bar{p}_j^k \neq \bar{p}_j^l$$

L'action $S_i \rightarrow S_j$ est non valide si

p_j^k est émergent à $\bar{p}_j^k$

Ces deux dernières propositions permettent de gérer les conflits survenant sur des arcs inverses. Puisque ces deux propositions n'effectuent aucune modification aux états, elles doivent nécessairement traiter les éléments d'un ensemble d'actions Γ avant les trois propositions associées aux conflits sur un même arc. Ainsi, pour les propositions 3.1, 3.2, 3.6 et 3.7 l'ordre du traitement des données n'influence pas la validité des états. Cependant, la validité des propositons 3.3, 3.4 et 3.5 requiert que seuls les conflits sur un même arc soient encore présents dans l'ensemble Γ. De plus, une hiéarchie s'établit entre ces trois dernières propositions. Le tableau 3.2 résume l'ordre des propositions nécessaire pour le traitement valide des états.

À l'aide de ces sept propositions, il est possible d'obtenir une solution globale et optimale. Cette solution répond à trois critères d'optimisation. Dans un premier temps, la solution indique un chemin sans conflits pour tous les véhicules du système. Dans un deuxième temps, la solution assure que le parcours du véhicule

Tableau 3.2: Ordre entre l'ensemble des propositions

Ordre de traitement	Propositions
1	3.1,3.2,3.6 ou 3.7
2	3.1,3.2,3.6 ou 3.7
3	3.1,3.2,3.6 ou 3.7
4	3.1,3.2,3.6 ou 3.7
5	3.4
6	3.5
7	3.3

le plus lent est le plus court chemin sans conflit pour la paire origine/destination du véhicule. De plus, la solution obtenue implique le temps minimun global pour l'ensemble des parcours des autres véhicules du système. Finalement, cette méthode de résolution permet d'obtenir un horaire global pour une flotte de véhicule, tout en respectant la contrainte sur l'orientation des véhicules.

CHAPITRE 4 : ANALYSE DES RÉSULTATS

Ce quatrième chapitre étudie les différents résultats obtenus avec l'algorithme de programmation dynamique. Ces résultats permettent d'élaborer de nouvelles méthodes de recherche, ainsi que d'évaluer et d'améliorer l'efficacité de l'algorithme.

Cinq sections servent à analyser les résultats obtenus. On compare tout d'abord les résultats d'une modélisation avec les noeuds (modèle A) à celle sans noeuds (modèle B). Ensuite, on applique la contrainte d'orientation au modèle B et on évalue l'efficacité de l'algorithme sur différents scénarios. La section suivante présente des approches pour améliorer l'efficacité de l'algorithme à l'aide de l'information obtenue par une méthode véhicule par véhicule. Par la suite, une section présente les résultats pour une simulation sur horaire de travail complet. Finalement, une discussion permet de cibler les forces et les faiblesses de l'algorithme.

4.1 Comparaison avec la modélisation incluant les noeuds

Le chapitre 3 présente un modèle où les états contiennent seulement les déplacements sur les arcs, alors que le modèle A du chapitre 2 représente les déplacements sur les noeuds et les arcs du réseau. Cette section vise à évaluer le comportement de l'algorithme sous ces deux modèles. Le modèle du chapitre 2 ne permet pas de résoudre un problème orienté. Toutefois les deux modèles permettent de traiter les problèmes non orientés. Pour ce faire, il faut retirer les

procédures 3.1 a) et b) du modèle B et attribuer la même valeur sur le vecteur $\mathbf{o_i}$ pour tous les états.

Pour évaluer les deux modèles, les scénarios développés à la section 2.9 permettent de les comparer. Les tableaux 4.1, 4.2 et 4.3 présentent les résultats obtenus sur les réseaux 1, 2 et 3 à partir du modèle B :

Tableau 4.1: Résultats réseau 1 pour 3 véhicules, modèle sans noeuds, recherche largeur bornée

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	38,0	1205	17	1673	0,0	12,8
1	48,7	1348	8	2341	0,0	0,8
2	30,9	941	14	1810	0,0	1,5
3	30,0	1049	7	1208	0,0	0,7
4	41,4	1214	108	1869	0,0	3,1
5	34,4	1172	12	1417	0,0	10,4
6	36,0	1132	90	1708	0,0	3,0
7	24,6	848	8	1052	0,0	0,7
8	36,8	1194	18	1432	6,1	16,2
9	31,7	1115	4	1403	0,0	14,9
10	61,3	1338	7	2483	0,0	0,9
11	34,4	1118	7	1490	0,0	0,7
12	38,9	1111	7	1616	0,0	0,8
13	38,4	1067	6	2196	0,0	1,6
14	29,7	1107	8	1288	0,0	0,7
15	34,3	1164	12	1385	1,0	10,8
16	53,8	1333	7	2461	0,0	0,8
17	39,6	1106	7	1645	0,0	0,8
18	39,9	1101	7	1606	0,0	0,8
19	50,0	1459	7	2037	0,0	0,8
moyenne	38,6	1156,1	18,1	1706,0	0,4	4,1

On obtient les résultats présentés aux tableaux 4.1, 4.2 et 4.3 à l'aide d'une recherche par largeur bornée. Les tableaux 2.10, 2.11 et 2.12 présentent eux aussi les résultats obtenus par ce même type de recherche avec le modèle A. Les instances sont identiques sur chacun des réseaux et il est possible de comparer les résultats entre les deux modèles.

Tableau 4.2: Résultats réseau 2 pour 3 véhicules, modèle sans noeuds, recherche largeur bornée

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	49,0	1109	66	1769	2,3	1,8
1	53,4	1075	72	1943	2,6	15,4
2	37,0	695	62	1567	2,6	1,8
3	32,2	855	88	1270	2,2	21,3
4	48,8	1044	81	1708	2,3	1,9
5	42,2	1039	81	1548	2,2	1,7
6	47,4	1073	83	1702	2,3	1,7
7	33,9	793	84	1223	2,2	13,9
8	39,9	1034	71	1580	2,1	1,7
9	33,0	813	216	1227	2,1	0,8
10	56,3	959	69	1872	2,9	2,2
11	39,8	885	84	1483	2,3	14,6
12	45,6	951	70	1603	2,5	1,9
13	41,6	685	66	1692	2,7	1,8
14	28,8	713	77	984	2,1	16,7
15	31,4	845	81	1199	2,1	1,6
16	51,3	988	67	2070	2,7	10,1
17	40,3	976	77	1585	2,5	1,9
18	40,6	984	70	1654	2,5	1,9
19	44,2	1069	71	1848	2,4	1,9
moyenne	41,8	929,3	81,8	1576,4	2,4	5,8

Tableau 4.3: Résultats réseau 3 pour 3 véhicules, modèle sans noeuds, recherche largeur bornée

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	87,3	1815	198	3643	2,3	1,9
1	152,6	3084	9	4754	0,0	0,0
2	103,1	2227	26	3649	0,0	13,9
3	33,4	1203	22	1432	0,0	30,5
4	112,8	2495	174	4424	1,1	29,4
5	61,0	1734	39	2230	0,0	26,1
6	86,3	1897	176	3568	1,1	0,9
7	22,9	867	22	1000	0,0	25,5
8	53,4	1562	39	2029	0,0	30,2
9	73,3	1685	518	2899	2,1	28,0
10	230,0	3597	347	5854	11,4	42,7
11	61,7	1665	26	2490	0,0	23,1
12	101,4	2294	253	3079	3,7	34,8
13	132,7	2494	20	4259	0,0	12,7
14	37,0	1094	28	1351	0,0	24,3
15	45,3	1281	39	1678	0,0	25,3
16	166,9	3009	346	4510	13,0	10,5
17	104,1	2450	152	3064	3,6	37,5
18	108,3	2379	258	3147	3,7	36,0
19	122,3	2690	39	4068	0,0	27,8
moyenne	94,8	2076,1	136,6	3156,4	2,1	23,1

Les résultats démontrent une réduction du nombre d'étapes pour atteindre une première solution lorsqu'utilisé avec le modèle B comparativement aux modèles A. De plus, une réduction équivalente affecte la taille de l'arbre de recherche. Ainsi, on peut conclure que le modèle B génère un moins grand nombre de combinaisons réalisables ce qui a un impact significatif sur le nombre d'étapes et d'états nécessaires pour construire l'arbre de recherche. Cependant le modèle sans noeud procure peu d'amélioration sur le temps de résolution. Bien que moins d'étapes soient nécessaires pour résoudre une instance du modèle sans noeud, chaque étape requiert un temps plus élevé. L'augmentation du temps de résolution des étapes s'explique par l'ajustement effectué sur les états par les propositions 3.3, 3.4 et 3.5. Ces propositions ajustent le vecteur temps des états pour les rendre valides. Ainsi, pour chaque étape, on retrouve un plus grand nombre d'états valides et les étapes d'insertions requirent une consommation de temps plus élevée.

L'ensemble des scénarios étudiés jusqu'ici permet de remarquer l'influence du critère de recherche sur le temps de résolution. Pour améliorer l'efficacité de la résolution de l'algorithme, un nouveau critère de recherche est mis en oeuvre. Il permet de construire l'arbre de recherche à partir des états répondant le mieux aux objectifs d'optimisation. La sélection des états se fait selon le temps du véhicule le plus lent. Ainsi, l'algorithme sélectionne en premier tous les états dont le véhicule le plus lent n'attend pas. L'algorithme selectionne ensuite les états offrant le moins d'attente sur le véhicule le plus lent. Pour trier les états, l'algorithme peut seulement considérer l'attente jusqu'à la position exprimée par l'état en question. Un plus court chemin sans conflit offre une borne inférieure pour le restant du parcours vers la position finale des véhicules. Ainsi, la recherche de solutions se fait selon les états offrant le moins d'attente. Puisque plusieurs états réprésentent le meme temps d'attente, un deuxième critère permet de trier les états selon le temps de parcours global de tous les véhicules.

Les tableaux 4.4, 4.5 et 4.6 présentent les résultats obtenus à partir de cette méthode.

Tableau 4.4: Résultats réseau 1 pour 3 véhicules, modèle sans noeuds, recherche selon le véhicule le plus lent

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	26,6	316	14	1339	0,0	0,0
1	16,8	240	65	1567	0,0	0,8
2	10,2	172	27	1137	0,0	0,0
3	24,8	378	56	1475	0,0	1,5
4	22,7	327	14	1355	0,0	0,7
5	24,3	363	98	1438	0,0	0,0
6	13,6	202	14	1292	0,0	0,0
7	14,3	236	117	1260	0,0	0,0
8	23,8	334	40	1376	0,0	0,7
9	3,8	65	64	596	0,0	0,0
10	13,4	201	74	1184	0,0	0,0
11	17,5	257	146	1463	0,0	0,9
12	10,6	172	101	868	0,0	0,7
13	10,1	160	17	1234	0,0	0,8
14	23,1	367	56	1481	0,0	1,6
15	24,4	363	46	1435	0,0	0,7
16	8,0	126	42	973	0,0	0,0
17	12,8	184	46	1120	0,0	0,8
18	12,4	184	46	1120	0,0	0,8
19	25,7	362	56	1515	0,0	0,8
moyenne	16,9	250,5	57,0	1261,4	0,0	0,5

Ce nouveau critère de recherche permet d'améliorer le temps de résolution pour l'ensemble des réseaux étudiés. On remarque une réduction importante du temps de calcul pour les problèmes nécessitant peu d'attente pour le véhicule le plus lent. Les résultats obtenus sur le réseau 3 reflètent bien ce phénomène. Les instances associées à ce réseau génèrent peu de conflits, car peu de véhicules circulent sur un réseau vaste. Les résultats démontrent aussi une réduction du nombre d'étapes nécessaires pour atteindre une première solution, ainsi que dans certain cas une réduction du nombre d'étapes pour atteindre la solution optimale. Ces réduction reflètent l'efficacité du critère de recherche. La construction

Tableau 4.5: Résultats réseau 2 pour 3 véhicules, modèle sans noeuds, recherche selon le véhicule le plus lent

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	25,9	422	83	1218	0,0	0,0
1	27,2	441	65	1289	0,0	0,0
2	11,4	118	77	866	0,0	0,0
3	24,3	460	83	1153	0,0	0,0
4	26,6	446	91	1290	0,0	0,0
5	24,9	416	103	1210	0,0	0,0
6	25,1	419	89	1182	0,0	0,0
7	19,9	331	158	982	0,0	0,0
8	26,3	454	83	1215	0,0	0,0
9	3,0	29	118	409	0,0	0,0
10	24,2	309	89	1392	0,0	0,0
11	25,3	418	83	1180	0,0	0,0
12	20,3	281	101	1156	0,0	0,0
13	11,1	121	59	872	0,0	0,0
14	23,6	416	83	1108	0,0	0,0
15	25,3	449	83	1180	0,0	0,0
16	18,3	239	65	1061	0,0	0,0
17	15,7	195	65	1079	0,0	0,0
18	22,0	320	65	1157	0,0	0,0
19	26,9	414	83	1277	0,0	0,0
moyenne	21,4	334,9	86,3	1113,8	0,0	0,0

Tableau 4.6: Résultats réseau 3 pour 3 véhicules, modèle sans noeuds, recherche selon le véhicule le plus lent

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	2,6	6	58	256	0,0	0,0
1	2,3	6	19	263	0,0	0,0
2	3,1	6	56	331	0,0	0,0
3	2,3	6	70	255	0,0	0,0
4	2,4	7	53	260	0,0	0,0
5	2,3	6	35	257	0,0	0,0
6	2,3	6	58	256	0,0	0,0
7	2,4	6	70	255	0,0	0,0
8	2,3	6	35	257	0,0	0,0
9	2,4	7	65	262	0,0	0,0
10	2,8	6	70	313	0,0	0,0
11	2,2	5	37	246	0,0	0,0
12	2,6	6	59	283	0,0	0,0
13	2,8	6	28	323	0,0	0,0
14	2,2	6	36	245	0,0	0,0
15	2,3	6	35	257	0,0	0,0
16	2,4	6	65	264	0,0	0,0
17	2,4	6	58	267	0,0	0,0
18	2,5	7	55	275	0,0	0,0
19	2,3	6	34	262	0,0	0,0
moyenne	2,4	6,1	49,8	269,4	0,0	0,0

de l'arbre de recherche se fait selon les états les plus prometteurs en fonction de l'objectif de résolution ce qui explique la réduction de la taille de l'abre de recherche. La meilleure propagation des états permet d'éviter la création de branches inutiles.

L'ensemble des scénarios étudiés jusqu'à présent évaluent l'algorithme sur des problèmes non orientés pour une flotte comprenant trois véhicules. Les scénarios résumés aux tableaux 4.4, 4.5 et 4.6 permettent de suivre le comportement de l'algorithme sur des problèmes non orientés à quatre véhicules. De nouvelles instances sont développées pour les réseaux 1, 2 et 3.

Tableau 4.7: Résultats réseau 1 pour 4 véhicules, modèle sans noeuds, recherche selon le véhicule le plus lent

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	463,8	3109	232	7906	0,0	0,6
1	84,1	264	558	2370	2,3	1,8
2	208,3	2052	168	3379	0,0	0,0
3	673,6	3334	2779	7772	0,0	0,0
4	214,7	1931	499	3543	0,0	1,8
moyenne	328,9	2138,0	847,2	4994,0	0,5	0,8

Le nombre de véhicules affecte exponentiellement la taille des états réalisables. Les résultats exprimés aux tableaux 4.7, 4.8 et 4.9 réflètent bien ce facteur. Les instances des réseaux 1 et 2 peuvent s'avérer difficiles à résoudre. On constate pour certains cas des temps de résolution élevés. Les temps de résolutions pour les instances du réseau 3 augmentent comparativement aux problèmes résolus pour trois véhicules. Cependant, cette variation est moindre que pour les deux réseaux précédents. Le petit nombre de conflits générés par le réseau 3 explique l'efficacité de l'algorithme sur ce réseau.

Tableau 4.8: Résultats réseau 2 pour 4 véhicules, modèle sans noeuds, recherche selon le véhicule le plus lent

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	924,7	5302	688	9359	0,0	0,0
1	58,9	209	166	1852	0,0	0,0
2	46,6	211	85	2350	0,0	0,0
3	772,9	4145	959	8600	0,0	0,0
4	121,2	533	148	5290	0,0	0,0
moyenne	384,9	2080,0	409,2	5490,2	0,0	0,0

Tableau 4.9: Résultats réseau 3 pour 4 véhicules, modèle sans noeuds, recherche selon le véhicule le plus lent

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Gap d'optimalité (%)	
		Initiale	Finale		Temps total	Global
0	11,8	6	240	1086	0,0	0,0
1	36,5	35	574	2404	0,0	0,0
2	19,2	7	342	1692	0,0	0,0
3	19,5	15	115	1868	0,0	0,0
4	18,0	7	69	1967	0,0	0,0
5	10,5	6	67	1199	0,0	0,0
6	9,6	6	107	1087	0,0	0,0
7	18,9	6	177	1900	0,0	0,0
8	16,4	7	235	1565	0,0	0,0
9	11,6	6	89	1323	0,0	0,0
10	11,3	6	145	1195	0,0	0,0
11	11,6	6	174	1262	0,0	0,0
12	22,6	10	404	1898	0,0	0,0
13	10,5	6	97	1182	0,0	0,0
14	14,7	7	170	1537	0,0	0,0
15	10,7	6	57	1167	0,0	0,0
16	11,3	6	69	1201	0,0	0,0
17	15,2	7	87	1697	0,0	0,0
18	12,0	6	130	1303	0,0	0,0
19	14,9	8	120	1600	0,0	0,0
moyenne	15,3	8,5	173,4	1506,7	0,0	0,0

4.2 Évaluation de l'agorithme sur des problèmes orientés

Un vecteur binaire permet de respecter les contraintes d'orientation sur les véhicules. Ce vecteur entraine toutefois une augmentation du nombre d'états réalisables. Pour évaluer l'impact des contraintes d'orientation, les réseaux 1, 2 et 3 servent pour la création de nouvelles instances. Comme le modèle A ne peut traiter les problémes avec orientation, toute autre comparaison avec ce modèle est maintenant impossible.

Les procédures d'insertion représentent un pourcentage élevé du temps requis pour obtenir la solution optimale. L'insertion d'un noeud à l'arbre nécessite la recherche d'états identiques, la comparaison entre les états identiques et l'ajustement de l'arbre de recherche selon les états retenus. L'ajout de colonnes aux tableaux des résultats permet de mieux analyser les étapes de résolution. Un premiére colonne, intitulé *Nombre d'états détruits*, sert à représenter la fréquence des manipulations où un état etiqueté domine un autre état etiqueté ou traité. Un état etiqueté représente un noeud dont les successeurs n'ont pas encore été explorés. Un état traité possède des successeurs et lorsque dominé par un état etiqueté, les procédures d'insertion doivent éliminer tous les fils de l'état rendu non-valide. La septième colonne des tableaux, *Nombre d'états équivalents* indique la fréquence où l'algorithme insère un état alors qu'un état identique est déjà présent au sein de l'arbre de recherche et qu'aucune dominance n'est possible entre les deux états. Les colonnes *Nombre total d'états générés*, *Nombre total d'états admissibles* et *Nombre total d'états non dominés* servent à suivre l'élimination des états au cours des différentes phases de l'algorithme. Elles informent sur la taille des éléments traités par l'algorithme. Les tableaux 4.10, 4.11 et 4.12 fournissent les résultats obtenus sur les problèmes orientés.

Tableau 4.10: Résultats réseau 1 pour 4 véhicules, problème orienté, modèle sans noeuds, recherche selon le véhicule le plus lent

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Nombre d'états détruits	Nombre d'états équivalents	Nombre total d'états générés	Nombre total d'états admissibles	Nombre total d'états non dominés	Gap d'optimalité (%)	
		Initiale	Finale							Temps total	Global
0	155,8	1211	336	6005	700	1063	231743	71915	6705	0,0	0,0
1	114,1	1263	319	5161	378	596	205418	79131	5539	0,0	0,0
2	122,9	1168	135	6231	385	515	185403	84899	6616	0,0	0,0
3	284,9	2584	166	7916	1269	2063	429976	169213	9185	0,0	0,0
4	164,6	1544	35	7266	732	969	207479	102179	7998	0,0	0,0
5	89,8	897	226	4514	184	226	141933	66695	4698	0,0	0,0
6	72,1	674	222	4029	488	680	113308	48475	4517	0,0	0,0
7	49,7	451	37	3436	242	291	59464	30560	3678	0,0	0,0
8	264,5	2403	258	7213	985	1489	416071	158200	8198	0,0	0,0
9	224,2	2364	162	6508	1038	1571	398436	125803	7546	0,0	0,0
10	69,5	563	73	5891	396	952	94328	44084	6287	0,0	0,0
11	173,0	2403	155	5548	689	969	394078	110309	6237	0,0	0,0
12	50,7	590	121	4499	233	297	93753	38345	4732	0,0	0,0
13	62,3	686	109	4101	453	622	104493	49886	4554	0,0	0,0
14	82,9	897	226	4381	179	221	141933	65012	4560	0,0	0,0
15	107,8	1246	172	5475	428	640	172438	83518	5903	0,0	0,0
16	54,0	652	62	4850	227	540	98424	39077	5077	0,0	0,0
17	94,1	1041	312	5356	273	444	166883	65628	5629	0,0	0,0
18	163,8	2661	167	4687	980	1408	449018	111737	5667	0,0	0,0
19	59,9	685	66	5341	314	631	100849	42678	5655	0,0	0,0
moyenne	123,0	1299,2	168,0	5420,4	528,7	809,4	210271,4	79367,2	5949,1	0,0	0,0

Tableau 4.11: Résultats réseau 2 pour 4 véhicules, problème orienté, modèle sans noeuds, recherche selon le véhicule le plus lent

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Nombre d'états détruits	Nombre d'états équivalents	Nombre total d'états générés	Nombre total d'états admissibles	Nombre total d'états non dominés	Gap d'optimalité (%)	
		Initiale	Finale							Temps total	Global
0	100,7	2018	353	3985	1008	1418	285153	64320	4993	0,0	0,0
1	32,3	174	335	3363	136	326	58027	8252	3499	0,0	0,0
2	59,7	771	251	4143	364	664	124128	34456	4507	0,0	0,0
3	123,9	1821	405	5592	1305	1970	270828	72815	6897	0,0	0,0
4	53,8	789	290	3789	206	344	131053	30551	3995	0,0	0,0
5	71,1	831	439	4882	454	1039	152156	35891	5336	0,0	0,0
6	19,5	104	218	2330	30	83	36572	4738	2360	0,0	0,0
7	63,0	1453	264	3466	349	459	208363	41909	3815	0,0	0,0
8	112,2	1392	455	5001	706	984	221453	59577	5707	0,0	0,0
9	45,4	454	657	2815	91	261	130713	19084	2906	0,0	0,0
10	124,7	1624	441	5338	1164	1614	251599	76437	6502	0,0	0,0
11	52,4	942	487	2827	207	287	168215	27450	3034	0,0	0,0
12	120,5	1789	373	5586	1318	1768	264564	75212	6904	0,0	0,0
13	89,1	1166	326	5394	660	1095	182078	48736	6054	0,0	0,0
14	74,3	831	439	4827	452	1031	152156	35746	5279	0,0	0,0
15	67,9	702	266	4754	656	1327	116298	31111	5410	0,0	0,0
16	132,9	2319	376	4678	1753	2463	330933	81303	6431	0,0	0,0
17	83,0	946	431	5306	519	861	167343	40454	5825	0,0	0,0
18	119,0	2073	71	4046	842	1103	255298	62619	4888	0,0	0,0
19	159,3	2148	236	5837	1764	2573	293758	92669	7601	0,0	0,0
moyenne	85,2	1217,4	355,7	4398,0	699,2	1083,5	190034,4	47166,5	5097,2	0,0	0,0

Tableau 4.12: Résultats réseau 3 pour 4 véhicules, problème orienté, modèle sans noeuds, recherche selon le véhicule le plus lent

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Nombre d'états détruits	Nombre d'états équivalents	Nombre total d'états générés	Nombre total d'états admissibles	Nombre total d'états non dominés	Gap d'optimalité (%)	
		Initiale	Finale							Temps total	Global
0	60,1	359	506	4916	345	509	97187	25161	5261	0,0	0,0
1	67,5	346	266	6033	53	168	67642	25328	6086	0,0	0,0
2	71,7	388	1074	4488	99	109	164796	28458	4587	0,0	0,0
3	33,5	156	213	3624	21	46	38387	12389	3645	0,0	0,0
4	131,9	1024	677	8549	356	553	204187	64129	8905	0,0	0,0
5	60,9	262	1063	4083	21	24	154367	19529	4104	0,0	0,0
6	37,4	130	999	2905	6	50	128639	9163	2911	0,0	0,0
7	89,1	835	338	6611	373	562	142734	48001	6984	0,0	0,0
8	43,5	315	442	4729	207	309	82847	22782	4936	0,0	0,3
9	13,4	107	234	2419	17	26	33187	8320	2436	0,0	0,0
10	9,7	82	181	2415	21	33	29513	4771	2436	0,0	0,0
11	57,1	408	382	5938	204	303	90772	24959	6142	0,0	0,0
12	74,4	622	222	5313	129	227	101853	42287	5442	0,0	0,0
13	32,3	105	878	2582	12	20	111257	7672	2594	0,0	0,0
14	61,8	262	1063	4083	21	24	154367	19529	4104	0,0	-0,3
15	59,6	284	349	5505	131	339	71727	20903	5636	0,0	0,0
16	52,9	267	556	4683	251	404	94313	17195	4934	0,0	0,0
17	28,8	122	78	3264	69	188	22358	7938	3333	0,0	0,0
18	13,5	76	342	1594	3	13	35380	3544	1597	0,0	2,8
19	65,5	503	104	4757	451	548	72949	33048	5208	0,0	0,0
moyenne	53,2	332,7	498,4	4424,6	139,5	222,8	94923,1	22255,3	4564,1	0,0	0,1

Les colonnes ajoutées aux tableaux démontrent l'efficacité de l'algorithme à restreindre le nombre d'états réalisables. La colonne intitulée *Nombre total d'états générés* indique le nombre total d'états crées valides ou non. La colonne suivante, *Nombre total d'états admissibles*, exprime le nombre d'éléments suite au retrait des états conflictuels, ainsi que ceux résultant avec une horloge supérieure aux bornes fixées. De manière générale, ces procédures réduisent d'environ 75% la taille des états générés. La colonne *Nombre total d'états non dominés* indique le nombre d'états conservés dans l'arbre de recherche. Il est généralement possible d'observer un écart de 75% entre la taille des états admissibles et des états insérés à l'arbre de recherche. Cet écart reflète le grand nombre d'états identiques comparés entre eux.

La taille des états détruits représente les états retirés de l'arbre de recherche. On retire un noeud lorsqu'il est possible de générer un meilleur état pour représenter cette position. La différence entre le nombre d'états non dominés et le nombre d'états détruits équivaut au nombre de noeud de l'arbre.

Les résultats obtenus à la colonne intitulée *Nombre total d'états insérés* permet de démontrer que l'algorithme insère généralement moins de 15% d'états dont leur vecteur position est identique à un autre noeud de l'arbre. Les valeurs indiquent la fréquence d'utilisation des procédures pour insérer deux éléments de position identique à l'arbre. Il faut noter que le nombre d'états identiques se retrouvant sur l'arbre final est généralement moindre que la valeur associée pour une instance sur les tableaux de résultats. En effet, il est possible que deux états identiques soient dominés et tous deux retirés lors d'une étape de l'algorithme.

Les résultats permettent de constater que l'algorithme obtient généralement une première solution proche ou égale à l'optimalité. Cependant, un nombre élevé d'étapes sont nécessaires pour atteindre une première solution. Bien que les

instances ne soient pas les mêmes, l'augmentation globale des valeurs du temps de résolution et du nombre d'étapes, ainsi que du nombre de noeuds à l'arbre de recherche se voient à l'aide des tableaux 4.4, 4.5 et 4.6. Dans son ensemble, les contraintes d'orientation triplent la taille et le temps des résultats.

L'algorithme véhicule par véhicule développé par Bigras permet de comparer les deux techniques de résolution. Les mêmes instances servent à la création d'itinéraires sans conflits pour un demi-cycle de production. Une approche véhicule par véhicule entraine un comportement glouton selon l'ordre d'affectation aux véhicules. Le premier véhicule à obtenir une route n'attend jamais. Pour améliorer les solutions, l'algorithme calcule l'ensemble des horaires selon les différents ordres possibles entre les véhicules. Parmi ces solutions, l'algorithme retient la meilleure selon les critères d'optimisation établis. De plus, il faut segmenter les arcs du réseaux pour permettre l'attente de plusieurs véhicules sur un même arc. Le modèle de Bigras permet l'attente des véhicules que sur des noeuds. Les tableaux 4.13, 4.14 et 4.15 montrent l'erreur comise entre les meilleures solutions obtenues par une approche véhicule par véhicule comparativement aux solutions d'une méthode globale.

Tableau 4.13: Erreur commise par l'algorithme véhicule par véhicule sur le réseau 1

Problème	Gap d'optimalité (%)	
	Temps du véhicule le plus lent	Temps global
0	0,0	0,0
1	4,2	-0,8
2	4,1	-0,8
3	0,0	0,0
4	0,0	0,0
5	4,3	-0,8
6	4,3	-0,8
7	0,0	0,0
8	0,0	0,0
9	1,8	-2,3
10	26,7	7,0
11	1,7	1,3
12	4,3	-0,8
13	4,2	-0,8
14	4,2	-0,8
15	4,5	-0,8
16	20,4	10,7
17	4,3	-0,8
18	3,4	-0,7
19	14,0	3,4

Tableau 4.14: Erreur commise par l'algorithme véhicule par véhicule sur le réseau 2

Problème	Gap d'optimalité (%)	
	Temps du véhicule le plus lent	Temps global
0	0,0	0,9
1	16,8	10,0
2	0,0	0,0
3	0,0	1,0
4	0,0	0,0
5	13,6	10,6
6	0,0	0,0
7	5,8	4,8
8	0,0	2,6
9	0,0	0,6
10	1,8	0,0
11	0,0	0,0
12	5,3	4,6
13	10,1	8,1
14	13,3	10,4
15	14,1	11,1
16	1,6	2,5
17	11,4	9,1
18	0,8	-3,8
19	2,6	3,9

Tableau 4.15: Erreur commise par l'algorithme véhicule par véhicule sur le réseau 3

Problème	Gap d'optimalité (%)	
	Temps du véhicule le plus lent	Temps global
0	0,0	0,0
1	0,0	0,0
2	0,0	0,0
3	0,0	0,0
4	0,0	0,0
5	0,0	0,0
6	0,0	0,9
7	0,0	0,0
8	0,0	0,0
9	0,0	0,0
10	0,0	1,3
11	0,0	0,0
12	5,4	-0,3
13	0,0	0,0
14	0,0	0,0
15	0,0	0,0
16	0,0	0,0
17	0,0	0,0
18	0,0	2,9
19	3,6	2,2

Les résultats démontrent qu'une approche véhicule par véhicule ne fournit pas la solution optimale en tout temps. Bien que la meilleure solution s'obtient à partir de tous les horaires réalisables selon l'ordre d'affectation des routes aux véhicules, certaines erreurs se produisent comparativement aux solutions obtenues par une méthode globale. Cependant, sur l'ensemble des résultats, il est possible de remarquer que pour un certain nombre d'instances l'algorithme atteint la solution optimale. L'erreur sur les solutions est généralement faible. Seulement certains cas procurent un écart de plus de 20% sur le temps requis pour le véhicule le plus lent. De plus, certaines solutions obtenues par l'approche véhicule par véhicule présentent une meilleure solution selon le deuxième critière d'optimisation : les temps de parcours globaux. Cependant, la valeur pour le parcours du véhicule le plus lent est suppérieure à celle obtenue par une approche globale. L'erreur sur le temps global se présente alors un pourcentage négatif.

Tableau 4.16: Fréquences des erreurs sur les instances

Réseaux	Succès		
	Égalité sur le véhicule le plus lent	Gap d'optimalité inférieur a 5%	Solution égale a l'optimalité
1	5	17	5
2	8	12	14
3	18	19	15

Le tableau 4.16 permet d'illustrer le taux de réussite pour l'agorithme véhicule par véhicule. Le pire scénario résulte en une erreur sur le véhicule le plus lent inférieure à 5% pour 12 cas sur 20.

L'algorithme véhicule par véhicule fournit généralement une bonne solution. Cette solution requiert peu de temps à obtenir et peut servir à améliorer la recherche de solution pour l'algorithme global.

4.3 Amélioration de l'algorithme

La solution obtenue par la construction d'itinéraires véhicule par véhicule répréSente la meilleure solution réalisable parmi celles générées selon tous les ordres d'affectation entre les véhicules. Ainsi, plus le nombre de véhicules impliqués est grand, plus le temps de résolution est lent, car l'algorithme doit construire un plus grand nombre d'horaires. Cependant, les opérations de transports en souterrain impliquent des flottes de petite taille. Ainsi, il est possible d'obtenir une solution rapidement avec la méthode véhicule par véhicule. Les solutions obtenues avec une méthode véhicule par véhicule peuvent être utilisées alors comme borne.

Un premier scénario fixe un borne supérieure sur le temps du véhicule le plus lent grâce à une solution véhicule par véhicule. Cette borne peut être qualifiée de borne dynamique, car sa valeur est adaptée pour chacun des problèmes.

Pour chaque instance, l'algorithme véhicule par véhicule fournit un temps maximum sur l'itinéraire des véhicules. L'algorithme global utilise ensuite cette borne comme point de départ pour la recherche de la solution optimale.

Les tableaux 4.17, 4.18 et 4.19 montrent les résultats pour les réseaux 1, 2 et 3.

Tableau 4.17: Résultats réseau 1 pour 3 véhicules, problème orienté, modèle sans noeuds, recherche selon le véhicule le plus lent, borne obtenue par l'algorithme véhicule par véhicule

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Nombre d'états détruits	Nombre d'états équivalents	Nombre total d'états générés	Nombre total d'états admissibles	Nombre total d'états non dominés	Gap d'optimalité (%)	
		Initiale	Finale							Temps total	Global
0	39,5	1209	299	1509	97	214	222428	20702	1606	0,0	0,0
1	54,4	1263	319	2140	223	281	205418	41620	2363	0,0	0,0
2	44,3	1168	135	1908	114	132	185403	39841	2022	0,0	0,0
3	101,1	2548	105	2653	214	512	416341	75463	2867	0,0	0,0
4	41,9	1542	35	1580	101	197	207229	32108	1681	0,0	0,0
5	37,3	897	226	1586	27	45	141933	32416	1613	0,0	0,0
6	25,7	674	222	1234	131	227	113308	20973	1365	0,0	0,0
7	7,6	449	37	489	24	26	59214	5447	513	0,0	0,0
8	100,3	2403	159	2562	181	394	402046	74008	2743	0,0	0,0
9	100,2	2364	162	2679	192	413	398436	73273	2871	0,0	0,0
10	69,6	563	73	4746	287	730	94328	36857	5033	0,0	0,8
11	98,4	2403	155	2766	371	500	394078	76845	3137	0,0	0,0
12	20,1	590	121	1213	96	118	93753	13417	1309	0,0	0,0
13	28,6	686	109	1366	138	210	104493	22574	1504	0,0	0,0
14	39,7	897	226	1586	27	45	141933	32416	1613	0,0	0,0
15	48,7	1246	172	2020	220	266	172438	40107	2240	0,0	0,0
16	51,9	652	62	3525	197	421	98424	31463	3722	0,0	0,0
17	42,8	1041	312	1805	163	197	166883	31791	1968	0,0	0,0
18	127,5	2661	167	3432	815	1205	449018	94283	4247	0,0	0,0
19	36,8	685	66	2352	225	333	100849	23416	2577	0,0	0,0
moyenne	55,8	1297,1	158,1	2157,6	192,2	323,3	208397,7	40951,0	2349,7	0,0	0,0

Les tableaux 4.10, 4.11 et 4.12 présentent les résultats lorsqu'une borne fixe est utilisée pour l'ensemble des problèmes, i.e. une borne 20% supérieure au trajet le plus long entre l'ensemble des origines-destinations du réseau. L'algorithme véhicule par véhicule permet d'obtenir une borne plus adaptée à chaque

Tableau 4.18: Résultats réseau 2 pour 3 véhicules, problème orienté, modèle sans noeuds, recherche selon le véhicule le plus lent, borne obtenue par l'algorithme véhicule par véhicule

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Nombre d'états détruits	Nombre d'états équivalents	Nombre total d'états générés	Nombre total d'états admissibles	Nombre total d'états non dominés	Gap d'optimalité (%)	
		Initiale	Finale							Temps total	Global
0	67,7	2018	269	2510	576	742	275053	40799	3086	0,0	0,0
1	23,0	174	335	3117	131	316	58027	7852	3248	0,0	0,0
2	14,2	758	183	991	32	155	114003	8413	1023	0,0	0,0
3	52,3	1821	309	2241	411	647	259228	34707	2652	0,0	0,0
4	17,2	781	232	1015	81	101	122803	11324	1096	0,0	0,0
5	59,9	831	439	3944	418	907	152156	31810	4362	0,0	0,0
6	3,1	103	211	329	3	4	35572	831	332	0,0	0,0
7	55,4	1453	264	3048	318	404	208363	38796	3366	0,0	0,0
8	41,0	1389	351	1740	204	259	208478	27871	1944	0,0	0,0
9	24,2	448	657	1200	49	121	129963	10385	1249	0,0	0,0
10	46,9	1620	308	1928	140	299	234474	37242	2068	0,0	0,0
11	24,2	926	408	1334	49	51	156940	15089	1383	0,0	0,0
12	69,7	1789	373	3126	464	705	264564	48591	3590	0,0	0,0
13	61,6	1166	326	3876	353	681	182078	36229	4229	0,0	0,0
14	61,5	831	439	3944	418	907	152156	31810	4362	0,0	0,0
15	54,6	702	266	3839	585	1157	116298	27092	4424	0,0	0,0
16	84,3	2319	372	3176	781	1253	330433	58944	3957	0,0	0,0
17	59,8	946	431	3874	368	649	167343	31961	4242	0,0	0,0
18	76,2	2065	71	2738	472	622	254298	43207	3210	0,0	0,0
19	78,9	2148	232	3198	381	806	293258	56019	3579	0,0	0,0
moyenne	48,8	1214,4	323,8	2558,4	311,7	539,3	185774,4	29948,6	2870,1	0,0	0,0

Tableau 4.19: Résultats réseau 3 pour 3 véhicules, problème orienté, modèle sans noeuds, recherche selon le véhicule le plus lent, borne obtenue par l'algorithme véhicule par véhicule

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Nombre d'états détruits	Nombre d'états équivalents	Nombre total d'états générés	Nombre total d'états admissibles	Nombre total d'états non dominés	Gap d'optimalité (%)	
		Initiale	Finale							Temps total	Global
0	15,2	359	506	874	98	123	97187	5287	972	0,0	0,0
1	6,8	334	257	592	2	2	65017	3150	594	0,0	0,0
2	32,4	388	1066	1483	59	60	163796	10089	1542	0,0	0,0
3	3,4	156	213	378	4	7	38387	1507	382	0,0	0,0
4	15,1	907	0	906	27	49	113375	9559	933	0,0	0,0
5	27,3	262	1038	1301	8	11	151242	7146	1309	0,0	0,0
6	21,0	130	999	1172	3	27	128639	4288	1175	0,0	0,0
7	12,4	707	0	706	9	11	88375	9666	715	0,0	0,0
8	10,1	315	441	757	48	62	82722	4121	805	0,0	0,0
9	3,2	107	234	347	3	5	33187	1355	350	0,0	0,0
10	2,3	79	173	254	3	5	28138	698	257	0,0	0,0
11	1,8	180	0	180	0	2	22500	814	180	0,0	0,0
12	29,2	622	222	1828	75	95	101853	21081	1903	0,0	0,0
13	16,4	105	869	975	12	19	110132	3541	987	0,0	0,0
14	27,2	262	1038	1301	8	11	151242	7146	1309	0,0	2,7
15	8,5	284	345	637	17	31	71227	3855	654	0,0	0,0
16	11,1	262	536	885	13	25	91188	3745	898	0,0	0,0
17	1,4	115	52	168	0	2	18333	435	168	0,0	1,9
18	3,7	76	340	470	1	3	35130	1089	471	0,0	0,0
19	21,0	503	104	1384	128	188	72949	14663	1512	0,0	0,0
moyenne	13,5	307,7	421,7	829,9	25,9	36,9	83231,0	5661,8	855,8	0,0	0,2

problème. En comparant les résultats aux tableaux 4.17, 4.18 et 4.19, il est possible de remarquer une diminution du temps de résolution pour la majorité des problèmes. Un écart plus important entre le temps de résolution des deux scénarios se remarque pour certains problèmes. Pour ces problèmes, la borne fixe est souvent très éloignée de la valeur de la solution optimale. Le gap d'optimalité représente l'écart entre la première solution et la solution optimale obtenues par l'algorithme global. Les tableaux 4.13, 4.14 et 4.15 présentent l'écart entre les bornes initiales et le temps requis par la solution finale pour les instances des trois réseaux.

Il est également possible de remarquer que l'application d'une borne dynamique influence peu le nombre d'étapes à réaliser. Il est généralement inférieur pour le cas d'une borne dynamique. Toutefois, son écart est peu significatif.

Les résultats montrent une diminution sur le nombre d'états conservés par l'arbre de recherche entre les deux scénarios. De plus, la borne dynamique accroît le taux de restriction entre les étapes de génération, de validation et d'insertion de l'algorithme. Finalement, on remarque une réduction du nombre d'états équivalents conservés, ainsi qu'une diminution du nombre d'appels aux procédures pour la destruction d'états déjà insérés à l'arbre. Toutes ces constatations reflètent l'efficacité de la borne dynamique.

Dans un deuxième temps, il est possible d'utiliser une borne dynamique pour valider un gap d'optimalité pour une solution véhicule par véhicule. En fixant une borne inférieure à la solution obtenue par la méthode véhicule par véhicule, il est possible de vérifier s'il existe une solution inférieure au pourcentage de réduction de la borne dynamique. Ainsi, lorsque l'algorithme global ne trouve pas de solution avec une borne réduite de 5%, il est possible d'affirmer que la solution véhicule par véhicule représente un écart inférieure à 5% de la solution optimale.

L'utilisation d'une borne réduite devrait résulter en un réduction du nombre d'étapes et du temps de résolution. Elle vise à pouvoir démontrer rapidement qu'une solution obtenue véhicule par véhicule se situe à l'intérieur d'un certain pourcentage par rapport à la solution optimale. Les tableaux 4.20, 4.21 et 4.22 montrent les résultats pour les réseaux 1, 2 et 3.

Tableau 4.20: Résultats réseau 1 pour 3 véhicules, problème orienté, modèle sans noeuds, recherche selon le véhicule le plus lent, borne 5% inférieure à la solution obtenue véhicule par véhicule

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Nombre d'états détruits	Nombre d'états équivalents	Nombre total d'états générés	Nombre total d'états admissibles	Nombre total d'états non dominés	Gap d'optimalité (%)	
		Initiale	Finale							Temps total	Global
0	0,012	1	0	1	0	0	175	0	0	-	-
1	45,980	1201	0	1201	83	118	145055	31871	1283	-	-
2	36,525	970	0	969	10	11	141150	26597	978	-	-
3	51,325	1483	0	1482	65	211	229395	22536	1546	-	-
4	15,512	603	0	602	55	103	70075	6209	656	-	-
5	32,297	841	0	840	10	11	109755	24933	849	-	-
6	18,503	652	0	652	73	138	85140	10038	724	-	-
7	5,290	238	0	237	16	17	32250	2232	252	-	-
8	50,897	1430	0	1429	62	143	218750	22097	1490	-	-
9	51,106	1432	0	1431	51	146	221800	22369	1481	-	-
10	60,560	563	73	3123	278	629	94328	26087	3399	0,0	0,8
11	76,153	2024	0	2024	119	228	300030	42345	2142	-	-
12	12,102	472	0	471	28	29	63500	6216	498	-	-
13	9,652	396	0	396	56	93	54600	4269	451	-	-
14	33,559	841	0	840	10	11	109755	24933	849	-	-
15	43,132	1179	0	1179	90	117	142185	30643	1268	-	-
16	47,330	652	62	2526	181	336	98424	23363	2706	0,0	0,0
17	24,400	856	0	855	27	28	108740	13983	881	-	-
18	93,668	2524	0	2523	184	529	409510	59334	2706	-	-
19	19,042	685	66	1346	186	276	100849	13574	1531	0,0	0,0
moyenne	36,4	952,2	10,1	1206,4	79,2	158,7	136773,3	20681,5	1284,5	0,0	0,3

L'utilisation d'une borne réduite génère trois types de résultats. Pour certains cas, on ne peut générer aucun état. Dans ces cas une borne supérieure initiale est fixée à une valeur qui est inférieure au plus court chemin sans attente pour

Tableau 4.21: Résultats réseau 2 pour 3 véhicules, problème orienté, modèle sans noeuds, recherche selon le véhicule le plus lent, borne 5% inférieure à la solution obtenue véhicule par véhicule

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Nombre d'états détruits	Nombre d'états équivalents	Nombre total d'états générés	Nombre total d'états admissibles	Nombre total d'états non dominés	Gap d'optimalité (%)	
		Initiale	Finale							Temps total	Global
0	0,008	1	0	1	0	0	125	0	0	-	-
1	25,946	174	335	2543	126	288	58027	6829	2668	0,0	0,0
2	3,620	279	0	279	24	32	34875	1464	302	-	-
3	14,169	715	0	715	90	119	87575	6297	804	-	-
4	6,850	419	0	419	19	24	52375	3754	437	-	-
5	45,838	831	439	2873	290	635	152156	24443	3162	0,0	0,0
6	0,006	1	0	1	0	0	125	0	0	-	-
7	32,590	1452	214	1666	156	183	202088	22292	1821	0,0	0,0
8	13,484	765	0	764	51	70	94025	5302	814	-	-
9	0,007	1	0	1	0	0	125	0	0	-	-
10	23,882	1130	0	1129	80	205	141250	15902	1208	-	-
11	0,007	1	0	1	0	0	125	0	0	-	-
12	44,436	1788	292	2080	210	327	254414	29666	2289	0,0	0,0
13	40,591	1166	326	2301	227	479	182078	24898	2527	0,0	0,0
14	47,331	831	439	2873	290	635	152156	24443	3162	0,0	0,0
15	33,875	702	266	2502	269	611	116298	17157	2770	0,0	0,0
16	40,529	1609	0	1609	449	671	201125	29128	2057	-	-
17	44,889	946	431	2663	254	505	167343	24331	2916	0,0	0,0
18	0,006	1	0	1	0	0	125	0	0	-	-
19	45,801	1762	0	1762	207	359	220250	32021	1968	-	-
moyenne	23,2	728,7	137,1	1309,2	137,1	257,2	105833,0	13396,4	1445,3	0,0	0,0

Tableau 4.22: Résultats réseau 3 pour 3 véhicules, problème orienté, modèle sans noeuds, recherche selon le véhicule le plus lent, borne 5% inférieure à la solution obtenue véhicule par véhicule

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Nombre d'états détruits	Nombre d'états équivalents	Nombre total d'états générés	Nombre total d'états admissibles	Nombre total d'états non dominés	Gap d'optimalité (%)	
		Initiale	Finale							Temps total	Global
0	0,008	1	0	1	0	0	125	0	0	-	-
1	0,009	1	0	1	0	0	125	0	0	-	-
2	0,008	1	0	1	0	0	125	0	0	-	-
3	0,008	1	0	1	0	0	125	0	0	-	-
4	0,008	1	0	1	0	0	125	0	0	-	-
5	0,009	1	0	1	0	0	125	0	0	-	-
6	0,008	1	0	1	0	0	125	0	0	-	-
7	2,884	142	0	142	3	4	17750	695	144	-	-
8	0,007	1	0	1	0	0	125	0	0	-	-
9	0,008	1	0	1	0	0	125	0	0	-	-
10	0,008	1	0	1	0	0	125	0	0	-	-
11	0,008	1	0	1	0	0	125	0	0	-	-
12	17,045	618	209	828	46	53	99728	13235	873	0,0	0,0
13	0,007	1	0	1	0	0	125	0	0	-	-
14	0,007	1	0	1	0	0	125	0	0	-	-
15	0,007	1	0	1	0	0	125	0	0	-	-
16	0,007	1	0	1	0	0	125	0	0	-	-
17	0,007	1	0	1	0	0	125	0	0	-	-
18	0,007	1	0	1	0	0	125	0	0	-	-
19	3,975	292	0	292	38	71	36500	2195	329	-	-
moyenne	1,2	53,5	10,5	64,0	4,4	6,4	7805,2	806,3	67,3	0,0	0,0

le véhicule ayant le plus long parcours. Ainsi, selon le critère de sélection, l'algorithme n'admet aucun état fils à l'état initial. Un temps de résolution de l'odre des centième de secondes est nécessaire pour vérifier un tel type de cas. Cependant, ce temps de résolution rapide ne présente aucun avantage car il est possible de conclure qu'il n'existe aucune solution lorsque la borne utilisée est inférieure au parcours du véhicule le plus lent sans attente.

Les cas, pour lesquels l'algorithme peut générer des successeurs à l'état initial se divise en deux catégories. Certains problèmes peuvent atteindre la solution optimale tandis que d'autres génèrent un certain nombre d'états sans atteindre une solution. Lorsque la borne réduite est inférieure à la solution optimale, mais supérieure au plus court chemin sans attente du véhicule le plus lent, l'algorithme ne peut pas atteindre la solution finale. Ces résultats permettent de conclure que la solution obtenue par l'algorithme véhicule par véhicule représente un écart inférieur à 5% de la solution optimale. Pour tous ces cas, il est possible de remarquer une réduction du temps de résolution par rapport au scénario utilisant une borne dynamique. Cependant, certains cas demandent un temps de résolution supérieur à 60 secondes.

Finalement, certains cas permettent de trouver et d'améliorer une solution. Le temps de résolution lorsqu'il existe une solution inférieure à la réduction de la borne initiale est généralement équivalent ou inférieur par rapport aux résultats des tableaux 4.17, 4.18 et 4.19. L'amélioration du temps de résolution est peu significative car l'algorithme obtient une première solution généralement égale à l'optimalité pour les scénarios utilisant une borne dynamique.

Outre l'utilisation d'une borne, la solution obtenue pas la méthode véhicule par véhicule permet d'autres possibilités. La solution représente un itinéraire particulier. Il est possible de construire les états associés à cet itinéraire pour fournir

une solution initiale à la méthode globale. Cette solution initiale représente une branche au sein de l'arbre. Les procédures de l'algorithme global sont mis en oeuvre pour valider l'optimalité de la solution.

Pour construire la branche de l'arbre, l'algorithme doit transformer la solution véhicule par véhicule en états. Des fenêtres de temps sur les intersections servent à représenter les parcours avec l'approche véhicule par véhicule. Il faut combiner ces différents parcours et représenter le déplacement des véhicules sur les arcs pour permettre de générer les états de la branche initiale. Cette transformation peut s'avérer difficile, car les réseaux soumis à l'algorithme véhicule par véhicule possèdent un grand nombre de noeuds. Cette discrétisation des arcs permet à plusieurs véhicules d'attendre sur le même segment, car ceux-ci ne peuvent s'arrêter qu'à un noeud. Les solutions obtenues alors par l'algorithme véhicule par véhicule contiennent un grands nombre de noeuds. Pour simplifier les étapes de transformation de la solution en branche initiale, on retire les noeuds de discrétisation des arcs. Il peut en résulter un dégradation de la solution initiale.

Les résultats obtenus à partir du réseau 3 diffèrent peu des résultats préalables et ne sont pas présentés. Les tableaux 4.23 et 4.24 montrent les résultats obtenues sur les réseaux 1 et 2.

Les résusltats montrent que l'utilisation d'une branche comme solution initiale influence peu le temps de résolution de façon à l'améliorer. La comparaison entre les résultats des tableaux 4.17 et 4.18 avec les résultats des tableaux 4.23 et 4.24 permet de constater que pour certains cas l'utilisation d'une branche peut réduire le temps de résolution d'une dizaine de secondes. Cependant, pour d'autres cas, l'utilisation d'une branche accroit le temps de résolution. Toutefois, la simplification du graphe donné à l'algorithme véhicule par véhicule entraîne la dégradation de la solution initiale servant à construire la branche. Ainsi, pour

Tableau 4.23: Résultats réseau 1 pour 3 véhicules, problème orienté, modèle sans noeuds, recherche selon le véhicule le plus lent, construction à partir d'une branche

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Nombre d'états détruits	Nombre d'états équivalents	Nombre total d'états générés	Nombre total d'états admissibles	Nombre total d'états non dominés	Gap d'optimalité (%)	
		Initiale	Finale							Temps total	Global
0	53,0	1546	2223	9	84	0	231328	30904	2236	3,7	2,8
1	-	-	-	-	-	-	-	-	-	-	-
2	49,3	1302	2128	7	108	0	185108	41574	2127	23,5	8,6
3	-	-	-	-	-	-	-	-	-	-	-
4	-	-	-	-	-	-	-	-	-	-	-
5	39,3	1122	1771	0	17	0	141638	33883	1671	25,8	9,7
6	35,4	897	1907	8	246	0	113163	27093	2008	29,8	12,6
7	8,1	486	543	0	23	0	59094	6245	509	4,0	3,1
8	100,8	2600	2698	0	173	0	405846	75109	2778	3,7	15,3
9	-	-	-	-	-	-	-	-	-	-	-
10	23,7	636	1699	0	115	0	94208	13600	1734	28,9	10,9
11	96,3	2557	2774	0	362	0	393663	76802	2996	5,2	4,6
12	-	-	-	-	-	-	-	-	-	-	-
13	23,7	796	1181	9	90	0	104448	19716	1149	28,1	11,5
14	40,8	1122	1771	0	17	0	141638	33883	1671	25,3	9,5
15	-	-	-	-	-	-	-	-	-	-	-
16	-	-	-	-	-	-	-	-	-	-	-
17	52,3	1351	2586	0	183	0	166363	38351	2627	16,1	10,8
18	-	-	-	-	-	-	-	-	-	-	-
19	-	-	-	-	-	-	-	-	-	-	-
moyenne	23,2	728,7	137,1	1309,2	137,1	257,2	105833,0	13396,4	1445,3	0,0	0,0

Tableau 4.24: Résultats réseau 2 pour 3 véhicules, problème orienté, modèle sans noeuds, recherche selon le véhicule le plus lent, construction à partir d'une branche

Problème	Temps de résolution (sec.)	Étapes pour atteindre la solution		Noeuds de l'arbre	Nombre d'états détruits	Nombre d'états équivalents	Nombre total d'états générés	Nombre total d'états admissibles	Nombre total d'états non dominés	Gap d'optimalité (%)	
		Initiale	Finale							Temps total	Global
0	59,6	0	2066	2068	336	0	247408	30906	2313	0,0	0,0
1	13,5	0	511	1039	33	0	58037	2791	987	3,7	0,0
2	-	-	-	-	-	-	-	-	-	-	-
3	50,6	0	1878	1880	255	0	227928	26891	2045	0,0	0,0
4	-	-	-	-	-	-	-	-	-	-	-
5	26,0	0	1204	1290	74	0	143666	11078	1304	1,9	0,7
6	6,1	0	329	491	11	0	36667	1147	449	2,8	-0,7
7	-	-	-	-	-	-	-	-	-	-	-
8	38,5	0	1440	1442	138	0	171558	21948	1517	0,0	0,0
9	-	-	-	-	-	-	-	-	-	-	-
10	79,6	0	2095	2987	314	0	255229	48967	3240	5,4	2,0
11	35,9	0	1380	1662	82	0	161850	18087	1651	1,6	1,2
12	53,6	0	2104	2147	197	0	256514	31047	2288	0,9	0,7
13	29,7	0	1424	1459	68	0	172798	14379	1452	0,9	0,3
14	26,4	0	1204	1290	74	0	143666	11078	1304	1,9	0,7
15	19,5	0	913	1069	78	0	109183	7759	1099	2,0	0,7
16	82,3	0	2571	2582	559	0	315093	52453	3059	0,8	0,6
17	18,5	0	955	957	21	0	116793	8210	906	0,0	0,0
18	-	-	-	-	-	-	-	-	-	-	-
19	71,9	0	2271	2282	181	0	279193	47222	2408	0,9	0,6
moyenne	1,2	53,5	10,5	64,0	4,4	6,4	7805,2	806,3	67,3	0,0	0,0

les cas ayant une branche représentant une moins bonne borne que pour les cas identiques des tableaux 4.17 et 4.18, l'algorithme peut nécessiter un temps de résolution plus long. Si on compare les résultats ayant une borne identique pour les deux scénarios, il est possible de remarquer une amélioration du temps de résolution pour tous les cas.

Les résultats des tableaux 4.23 et 4.24 permettent aussi de constater l'importance de la discrétisation des segments sur les graphes de l'algorithme véhicule par véhicule. L'impossibilité de faire attendre plusieurs véhicules sur un même arc engendre l'irréalisabilité d'un certain nombre de solutions. Pour l'ensemble des 40 cas testés, on retrouve 14 instances n'ayant pu trouver une solution avec l'algorithme véhicule par véhicule. Pour le réseaux 1, l'absence de cycle entraine l'irréalisabilité de la solution 9 fois sur 20. Pour les cas où le graphe fourni à l'algorithme véhicule par véhicule contient une discrétisation sur les segments, il est possible de trouver une solution pour tous les cas.

4.4 Simulations

Les processus de simulations à l'aide d'un algorithme véhicule par véhicule génèrent des situations d'impasses. Bigras propose une méthode pour réparer les situations d'impasse de manière heurestique, ainsi que pour simuler une période de travail. Le caractère dynamique de la méthode globale permet de gérer les situations d'impasses. Entre les demi-cycles de production, l'absence de routes fixes pour certains véhicules permet de gérer les impasses facilement. Cependant, il faut noter que l'absence de routes fixes lors de l'évaluation d'un horaire pour un demi-cycle de production peut entraîner l'inefficacité de certains véhicules au cours de la période de simulation. Sous ces procédures dynamiques, l'horaire

d'un véhicule peut constamment être modifié par une pertubation causée par un autre véhicule du système et ne jamais atteindre sa destination. Toutefois, le petit nombre de conflits entre les véhicules laisse croire qu'il est peu probable que cette situation se produise.

Différents objectifs peuvent être appliqués pour évaluer la productivité d'une période de travail. Bigras fait une étude des simulations selon un routage véhicule par véhicule et par routage simultané pour des affectations fixes et des affectations selon le temps de début de service le plus tôt. La méthode heurestique de routage simultané avec un critère d'affectation pour minimiser le véhicule le plus lent offre les meilleurs résultats.

L'algorithme global requiert un état initial, ainsi qu'un état final afin d'être capable d'obtenir une solution. Un état initial est toujours fixe car il représente les conditions réelles du systèmes. Les destinations des véhicules ne sont toutefois pas fixes. Pour permettre d'évaluer les meilleures affectations entre chaque demi-cycle, il faut évaluer chaque possibilité de configuration de l'état final selon les chantiers et les chutes à minerai du réseau, puis choisir la meilleure solution. Une telle simulation s'avère difficile à résoudre à l'aide du matériel informatique utilisé pour ce projet. Ainsi, les résultats obtenus par les méthodes développées par Bigras servent à créer un plan de production pour l'algorithme global.

Les simulations s'effectuent sur les réseaux 2 et 3. Pour le réseaux 2, trois véhicules sont en production. Quatre véhicules oeuvrent sur le réseau 3. Le plan de production alloue un temps de chargement aléatoire pour les véhicules. La valeur du nombre alléatoire est équivalente au simulateur développé par Bigras et suit un loi normale N(10,1). Le temps de simulation est de 5 000 secondes et représente un quart de travail. Un ajustement sur le coût des arcs du réseau permet de simuler des conditions semblables aux opérations minières. La simulation

permet d'évaluer la productivité des véhicules. Les figures 4.1 et 4.2 permettent d'évaluer la fréquence des demi-cycles complétés lors de la simultation avec l'algorithme global et l'algorithme véhicule par véhicule.

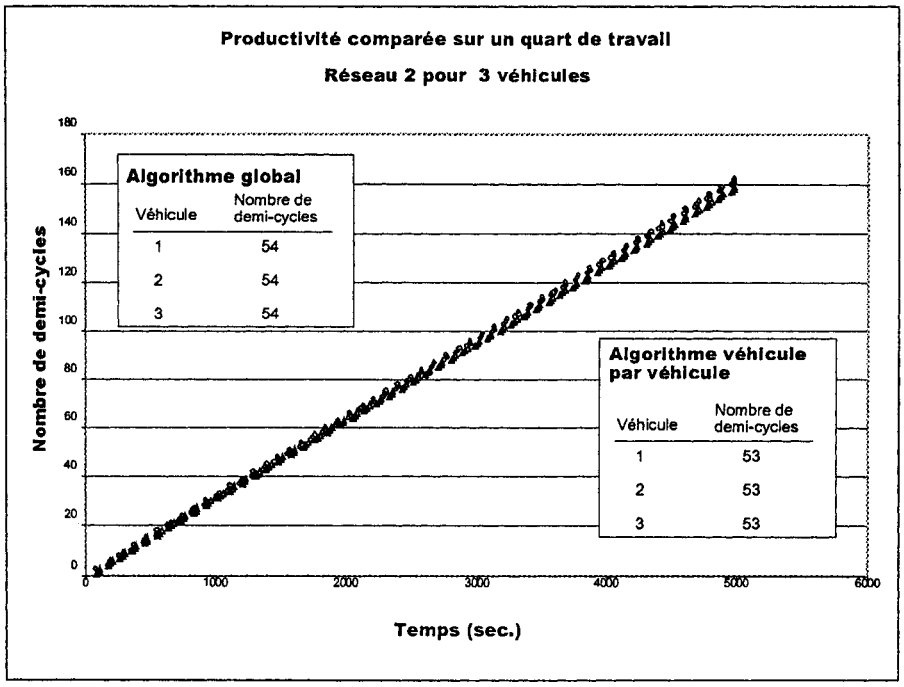

Figure 4.1: Productivité comparée sur un quart de travail, réseau 2 pour 3 véhicules

L'algorithme global fournit de meilleurs résultats pour les deux simulations. Selon le plan de production proposé à l'aide de la méthode véhicule par véhicule, l'algorithme global trouve l'horaire optimal pour les véhicules. Il est aussi possible de remarquer que la différence entre le nombre cycles complétés pour les deux méthodes est petite. Pour le réseau 2, la solution obtenue par l'algorithme global améliore la productivité d'environ 2%. On peut constater une amélioration de 6 % sur la productivité des quatre véhicules du réseau 3. De plus, les résultats démontrent que la répartition du travail entre les véhicules est uniforme.

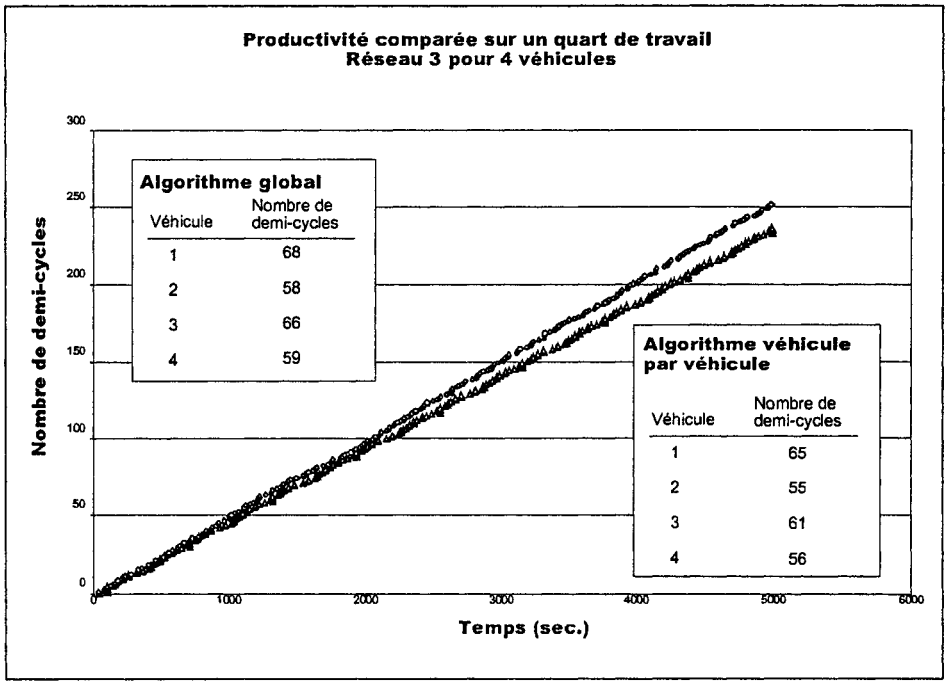

Figure 4.2: Productivité comparée sur un quart de travail, réseau 3 pour 4 véhicules

Un deuxième scénario de simulation permet d'évaluer l'efficacité des méthodes avec un plus grand nombre de véhicules. Les réseau 2 et 3 sont combinés. Un arc est ajouté entre le noeud 5 du réseau 2 et le noeud 3 du réseau 3. Son coût est de 10 unités. Il est possible d'effectuer une simulation pour 7 véhicules avec l'algorithme véhicule par véhicule. Il est difficile d'obtenir un temps de résolution efficace pour des flottes de plus de quatre véhicules avec l'algorithme global. Cependant, il est possible de combiner les résultats des simulations pour obtenir une solution pour 7 véhicules. Toutefois, cette solution combinée ne représente pas une solution optimale. Elle sert uniquement à évaluer le comportement de l'algorithme véhicule par véhicule sur le réseau combiné. La figure 4.3 montre les résultats obtenue avec la méthode véhicule par véhicule sur le réseau 2 et 3 combiné, ainsi que la somme des solutions obtenues sur les deux réseaux du scénario précédent à l'aide des deux méthodes.

Les résultats obtenues sur le réseau combiné montrent qu'un réseau présentant un plus grand nombre de chantiers et de chutes à minerai permet d'accroitre le nombre de demi-cycles effectués par les véhicules. Pour la méthode véhicule par véhicule, la combinaison des réseaux permet d'effectuer 30 demi-cycles de plus pour la période de simulation. Les résultats démontrent également que la méthode véhicule par véhicule est plus efficace sur le réseau combiné que la métode globale sur les réseaux séparés.

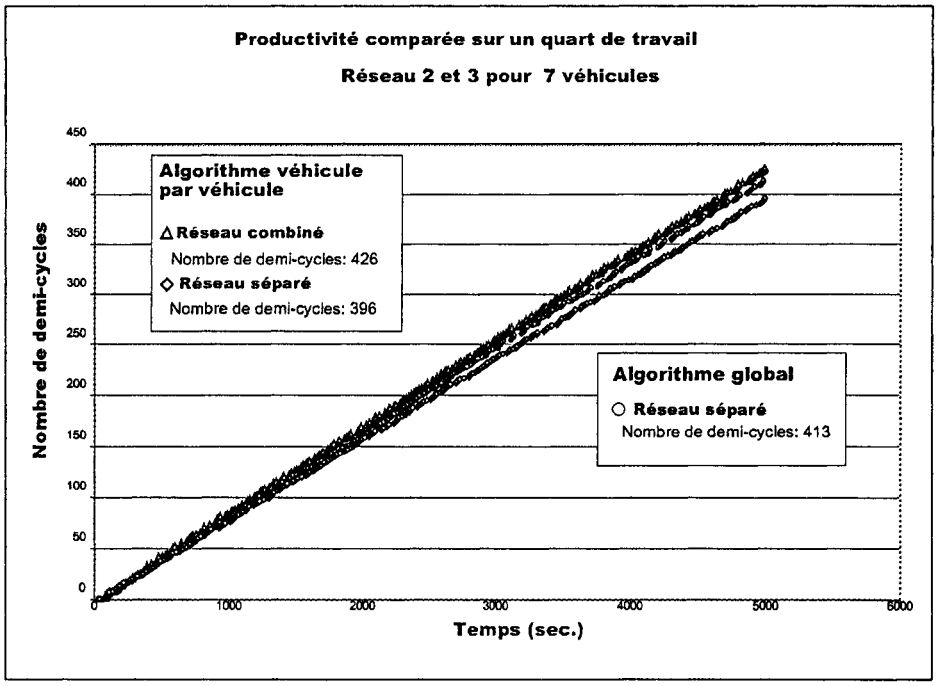

Figure 4.3: Productivité comparée sur un quart de travail, réseau 2 et 3 pour 7 véhicules

CONCLUSION

Les systèmes de véhicules automatiques procurent généralement des gains en productivité. L'industrie minière de surface et l'industrie manufacturière tirent déjà avantages des systèmes de véhicules automatiques. Les opérations de transport de minerai pour les mines souterraines présentent des caractéristiques propices à l'appliquation de l'automatisation des véhicules. Cependant, on retrouve peu d'exploitations minières en souterrain utilisant une flotte de véhicules automatiques. Trois aspects sont nécessaire pour assurer le bon fonctionnement d'un système de véhicules automatiques en milieu souterrain : la robotisation des véhicules, un système de gestion des routes des véhicules et un protocole de communication entre le système de gestion et les véhicules.

Les travaux décris par ce mémoire se concentrent sur l'établissement d'un système de gestion des routes des véhicules. L'objectif est d'établir un horaire optimal afin de maximiser la productivité des opérations de manutention des véhicules. La construction des routes présente des contraintes de conflits et d'orientation sur les véhicules. Deux approches permettent de résoudre le problème. Les méthodes véhicules par véhicules résolvent le problème en établissant la route optimale pour un véhicule selon les routes fixes des autres véhicules du système. Un méthode globale permet de trouver une solution représentant les meilleures routes sans conflits pour l'ensemble des véhicules.

Ce mémoire propose un algorithme d'énumération par programmation dynamique pour la gestion d'une flotte de véhicules. Le modèle A est présenté pour résoudre de manière globale les problèmes sans contrainte d'orientation. Les états servent à représenter la position de tous les véhicules, ainsi que le temps

nécessaire aux véhicules pour atteindre les positions respectives selon un état prédécesseur. Un arbre d'énumération permet de construire des séquences d'états vers les destinations désirées et des procédures, ainsi que des critères de dominance, servent à vérifier l'admissibilité des états selon les contraintes spécifiées. L'algorithme tire avantage des aspects opérationnelles d'une mine en souterrain. Les configurations particulières des réseaux et le petit nombre de véhicules impliqués permettent de restreindre le nombre d'états admissibles. Des méthodes de recherche en largeur et profondeur permettent d'abord d'analyser différents aspects de l'algorithme. Une méthode de recherche en largeur avec l'utilisation d'un borne suppérieure offre les meilleurs résultats. Les tests effectués sur différents réseaux permettent de vérifier la stabilité de l'algorithme. L'efficacité de l'algorithme à résoudre une instance semble être grandement influencée par la congestion du réseau. Le modèle A permet de résoudre de manière globale l'affectation des routes aux véhicules sans contrainte d'orientation. De plus, avec les critères de recherche développés, le temps de résolution est court pour un système à trois véhicules.

L'application de la contrainte d'orientation cause des erreurs pour l'évaluation de certains états à l'aide du modèle A. Le modèle B comporte des modifications aux états et aux procédures afin de rendre l'algorithme valide de nouveau. La contrainte d'orientation détériore le temps de résolution des problèmes. De nouveau critères de recherche sont développés pour améliorer le temps de résolution de l'algorithme. Une méthode de recherche selon le véhicule le plus lent et utilisant une borne suppérieure permet de résoudre le problème avec orientation pour trois véhicules généralement à l'intérieur de 200 secondes. L'ensemble des méthodes globales présentées à la section 1.4 *Revue de la littérature* demandent généralement des temps de résolutions similaires ou supérieurs pour des problèmes non-orientés.

L'utilisation d'une méthode véhicule par véhicule permet de générer de meilleures bornes pour l'algorithme global. De plus, les résultats démontrent qu'une méthode véhicule par véhicule, où la solution provient de l'ensemble des solutions pour chaque ordre d'affectation entre les véhicules, n'atteint pas toujours l'optimalité.

Les modèles A et B permetent de gérer efficacement les situations d'attente pour les véhicules uniquement à l'aide des arcs du réseau. Plusieurs modèles utilisent le temps de passage à une intersection pour évaluer l'horaire des véhicules et admettent l'attente d'un véhicule qu'à des endroits précis du réseau. Dans certains cas, un véhicule ne peut trouver de destination, car l'ensemble des points d'attente et instersections du réseau sont occupés par d'autres véhicules. Pour éviter ces situations conflictuelles, les auteurs proposent une discrétisation des segments du réseau de façon à créer plusieurs noeuds d'attente. Cependant, la complexité des algorithmes de résolution est généralement fonction du nombre de noeuds du réseau. La modélisation sur les arcs permet de gérer les situations d'attente de manière optimale, sans affecter la complexité de l'algorithme selon le nombre de points d'attente du réseau.

L'implantation de l'algorithme global à un simulateur permet de vérifier l'efficacité de la méthode heurestique développée par Bigras. L'écart de productivité entre les deux méthodes est peu significatif sur une période de travail. L'algorithme heuristique génère d'excellentes solutions avec un temps de résolution qui permet l'application de cette méthode en temps réel.

L'algorithme global permet de construire de manière optimale les routes sans conflits pour une flotte de véhicules. Avec l'implantation actuelle de l'algorithme, il est possible de résoudre des problèmes pour trois véhicules sans orientation en temps réel. Toutefois, les meilleurs critères de recherche sur les problèmes

orientés montrent que les temps de calculs peuvent atteindre 100 secondes pour certains cas et empêchent une utilisation efficace de l'algorithme en vue d'application en temps réel. De plus, l'ordinateur utilisé pour les tests possède une capacité en mémoire de 64 Mega octet et les problèmes de plus de trois véhicules occasionnent des débordements de mémoire. L'implantation de l'algorithme sur un ordinateur destiné aux calculs devrait améliorer le temps de résolution de l'algorithme et pouvoir traiter des problèmes de plus grande taille.

Il serait également intéressant de développer des méthodes heurestiques à partir de l'algorithme d'énumération par programmation dynamique. Par exemple, pour les situations où un véhicule doit attendre, l'algorithme génère souvent plusieurs états pour différents points d'attente admissibles pour le véhicule. Toutefois, ces états représentent souvent des solutions équivalentes. Des procédures heuristiques pourraient restreindre le nombre d'états générés et ainsi améliorer l'efficacité de résolution de l'algorithme et obtenir une solution proche de l'optimalité.

Finalement, puisque la méthode véhicule par véhicule et heureustique développée par Bigras procure de bons résultats, il serait intéressant d'établir une méthode utilisant les résultats des deux algorithmes pour obtenir la meilleure solution.

BIBLIOGRAPHIE

ALARIE,GAMACHE(2002). *Overview of solution strategies used in truck dispatching systems for open pit mines.* International Journal of Surface Mining, Reclamation and Environment.

BIGRAS(2001). *Gestion en temps réel d'une flotte de véhicules automatisés pour une mine souterraine.* Mémoire de maîtrise, École Polytechnique de Montréal.

BROADBENT, BESANT, PREMI, WALKER(1985). *Free randing AGV systems : promises, problems and pathways.* Proc. 2^{nd}. Int. Conf. on automated materials handling, IFS (Publ) Ltd.

EGBELU, TANCHOCO (1986) *Pontentials for bi-directional guide-path for automated guided vehicle systems.* International Journal of Production Research.

FUJII, SANDOH, HOHZAKI(1988) *Routing control of automated guided vehicles in FMS.* Proceedings of the USA-JAPAN symposium on flexible automation.

GRIMARD(2001). *Développement d'un système de gestion et de répartition des véhicules automatisés pour les mines souterraine.* Mémoire de maîtrise, École Polytechnique de Montréal.

HUANG, PALEKAR, KAPOOR(1993). *A labelling algortihm for the navigation of automated guides vehicles.* Journal of egineering for industry.

KIM, TANCHOCO (1991) *Conflict-free shortest-time AVG routeing.* International Journal of Production Research.

KRISHNAMRTHY, BATTA, KARWAN *Developing conflict-free routes for automated guided vehicles.* Operations Research.

LANGEVIN, LAUZON, RIOPEL(1985). *Dispatching, routing and scheduling of two automated guided vehicles in a flexible manufacturing system.* Les cahiers du Gerad.

MUNIRATHINAM,YINGLING *A review of computer-based truck dispatching strategies for surface mining orperations.* International Journal of Surface Mining, Reclamation and Environment.

VAGENAS(1991). *Dispatch control of a fleet of remote-controlled/automatic load-haul-dump (LHD) vehicles in underground hard rock mines.* International Journal of Production Research.

VILLENEUVE(2000) *Répartition et routage d'un système automatique de chariots à l'aide d'une méthode de génération de colonnes.* Mémoire de maîtrise, École Polytechnique de Montréal.