PolyPublie
Polytechnique Montréal

POLYTECHNIQUE MONTRÉAL
UNIVERSITÉ D'INGÉNIERIE

Titre: Title:	Méthode de recherche adaptative sur le web avec utilisation de Wikipedia pour l'expansion de requêtes
Auteur: Author:	Amir Masoud Oveissian
Date:	2006
Type:	Mémoire ou thèse / Dissertation or Thesis
Référence: Citation:	Oveissian, A. M. (2006). Méthode de recherche adaptative sur le web avec utilisation de Wikipedia pour l'expansion de requêtes [Master's thesis, École Polytechnique de Montréal]. PolyPublie. https://publications.polymtl.ca/7898/

Document en libre accès dans PolyPublie
Open Access document in PolyPublie

URL de PolyPublie: PolyPublie URL:	https://publications.polymtl.ca/7898/
Directeurs de recherche: Advisors:	Michel Gagnon, & Michel C. Desmarais
Programme: Program:	Unspecified

Ce fichier a été téléchargé à partir de PolyPublie, le dépôt institutionnel de Polytechnique Montréal
This file has been downloaded from PolyPublie, the institutional repository of Polytechnique Montréal

https://publications.polymtl.ca

UNIVERSITÉ DE MONTRÉAL

MÉTHODE DE RECHERCHE ADAPTATIVE SUR LE WEB AVEC UTILISATION DE WIKIPEDIA POUR L'EXPANSION DE REQUÊTES

AMIR MASOUD OVEISSIAN
DÉPARTEMENT DU GÉNIE INFORMATIQUE
ÉCOLE POLYTECHNIQUE DE MONTRÉAL

MÉMOIRE PRÉSENTÉ EN VUE DE L'OBTENTION
DU DIPLÔME DE MAÎTRISE ÈS SCIENCES APPLIQUÉES
(GÉNIE INFORMATIQUE)
AÔUT 2006

© Amir Masoud OVEISSIAN, 2006.

Library and Archives Canada

Published Heritage Branch

395 Wellington Street
Ottawa ON K1A 0N4
Canada

Bibliothèque et Archives Canada

Direction du Patrimoine de l'édition

395, rue Wellington
Ottawa ON K1A 0N4
Canada

Your file Votre référence
ISBN: 978-0-494-19318-1
Our file Notre référence
ISBN: 978-0-494-19318-1

NOTICE:
The author has granted a non-exclusive license allowing Library and Archives Canada to reproduce, publish, archive, preserve, conserve, communicate to the public by telecommunication or on the Internet, loan, distribute and sell theses worldwide, for commercial or non-commercial purposes, in microform, paper, electronic and/or any other formats.

The author retains copyright ownership and moral rights in this thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without the author's permission.

AVIS:
L'auteur a accordé une licence non exclusive permettant à la Bibliothèque et Archives Canada de reproduire, publier, archiver, sauvegarder, conserver, transmettre au public par télécommunication ou par l'Internet, prêter, distribuer et vendre des thèses partout dans le monde, à des fins commerciales ou autres, sur support microforme, papier, électronique et/ou autres formats.

L'auteur conserve la propriété du droit d'auteur et des droits moraux qui protège cette thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

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

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

Conformément à la loi canadienne sur la protection de la vie privée, quelques formulaires secondaires ont été enlevés de cette thèse.

Bien que ces formulaires aient inclus dans la pagination, il n'y aura aucun contenu manquant.

Canada

UNIVERSITÉ DE MONTRÉAL

ÉCOLE POLYTECHNIQUE DE MONTRÉAL

Ce mémoire intitulé:

MÉTHODE DE RECHERCHE ADAPTATIVE SUR LE WEB AVEC UTILISATION DE WIKIPEDIA POUR L'EXPANSION DE REQUÊTES

présenté par : OVEISSIAN Amir Masoud

en vue de l'obtention du diplôme de : Maîtrise ès sciences appliquées

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

M. PESANT Gilles, Ph.D., président

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

M. DESMARAIS Michel, Ph.D., membre et codirecteur de recherche

Mme. DA SYLVA Lyne, Ph.D., membre

A l'âme du Désert

REMERCIEMENTS

J'aimerais d'abord remercier mon directeur de recherche, *Michel GAGNON*, dont la vision, les conseils, les critiques et les encouragements ont fortement teinté l'ensemble de ma démarche de recherche et de production de ce mémoire.

Je tiens à remercier tout particulièrement *Michel DESMARAIS*, mon codirecteur de recherche pour son soutien et ses encouragements dans le contexte du sujet et du validation de l'architecture de mon projet.

J'aimerais également remercier ma mère *Leyla*, mon père *Djamshid*, et mon frère *Arash* pour leurs encouragements et leur soutien dans tous les moments de ma vie et de mes études.

Je souhaite aussi remercier :

- *Ramin DEBAN* et *Alexandro ZANARINI* pour leurs participation à la validation de l'architecture de mon projet.

- *Kavé SALAMATIAN* qui m'a conseillé sur le choix de sujet de recherche et qui m'a appris comment de faire une recherche scientifique.

- les membres de mon laboratoire de travail pour le climat de travail et leur entraide quotidienne.

Enfin à toutes autres personnes qui, de près ou de loin, ont contribué à la réussite de ce travail, je dis grand merci.

RÉSUMÉ

Malgré les grandes avancées réalisées durant la dernière décennie en matière de recherche d'information, plusieurs difficultés font encore obstacle à la facilité de trouver l'information désirée pour l'utilisateur moyen. Parmi ces difficultés, citons notamment le faible nombre de termes utilisés dans la majorité des requêtes et l'absence d'information quant au contexte de la recherche, comme les intérêts de l'utilisateur, qui pourraient contribuer à mieux interpréter et préciser les requêtes d'information.

Plusieurs moteurs de recherche ont été proposés durant les dernières années pour tenter de contourner ces obstacles. Ces projets appliquent principalement les techniques de filtrage de l'information, d'extraction de l'information, d'expansion de la requête, ou finalement le recours aux mécanismes du web sémantique pour améliorer le résultat de la recherche d'information. Parmi ces travaux, il y a des architectures qui sont adaptatives selon soit l'intérêt spécifique d'un seul utilisateur, soit l'intérêt plus générique de son groupe d'appartenance.

Parmi ces limites, on retrouve la difficulté de s'adapter à tous les types de recherche dans Internet (exploratoire, informationnelle et navigationnelle), la difficulté de s'adapter aux changements d'intérêt de l'utilisateur à travers les différents sujets de recherche et finalement la nécessité d'obtenir un grand nombre de documents pré-évalués par l'utilisateur quant à son degré d'intérêt.

Nous proposons une architecture, ARIIA (Adaptive Reinforcement Iteration-based Internet Agent), qui vise à repousser les limites évoquées précédemment. ARIIA est un agent de recherche du côté de l'utilisateur qui est indépendant de la langue et qui étend

la requête de recherche d'utilisateur selon ses intérêts personnels ou professionnels à long terme, tout en pouvant s'adapter à un changement de l'intérêt de l'utilisateur. Pour réaliser la fonctionnalité d'ARIIA, nous avons employé une approche d'extraction d'ontologie à partir d'une encyclopédie en ligne, Wikipedia[1]. Cette approche est basée sur les hyperliens de Wikipedia et nous garantit une évolution progressive du vocabulaire utilisé pour les recherches de l'utilisateur et en temps réel. La fonctionnalité d'ARIIA est basée sur plusieurs aspects :

1. ARIIA emploie, dans son choix de document, un mécanisme de rétroaction[2] basé sur la pertinence des documents.
2. Il emploie la structure d'hyperlien pour mesurer la pertinence des termes et étendre les requêtes de recherche.
3. Il repose principalement sur l'infrastructure et les services existants disponibles pour sa recherche, à savoir Wikipedia, qui est une encyclopédie universelle, et un moteur de recherche existant tel que Google (ou AltaVista).
4. Il crée une requête booléenne avec l'ensemble de termes qui sont ajoutés par une sélection basée sur les hyperliens encyclopédiques et le feedback de l'utilisateur sur le niveau d'importance des liens.
5. Il se fonde principalement sur les itérations d'expansion de requête et le *relevance feedback* de l'utilisateur.
6. Il crée un profil des mots clefs de l'utilisateur qui est basé sur le contenu historique des requêtes de recherche.

[1] Une encyclopédie disponible à : www.wikipedia.org
[2] *relevance feedback*

7. Il peut être employé comme un agent de recherche en plusieurs langues. ARIIA peut être appliqué aux recherches avec n'importe quelle langue qui existe dans le Wikipedia.

Pour l'évaluation du fonctionnement d'ARIIA, nous définissons quelques indicateurs de mesure en considérant le contenu et l'ordre des pages retournées, et nous testons ARIIA selon ces indicateurs. Les résultats retournés d'ARIIA sont composés à ceux de Google et sont, de manière générale, meilleurs: Les résultats d'ARIIA nous montrent que dans les pires cas on note une amélioration de 4% à 6%, selon les types de recherche réalisés. Dans la majorité des cas, cette amélioration se stabilise à environ 10% et parfois nous avons une augmentation jusqu'à 20% par rapport à Google.

ABSTRACT

In spite of the remarkable advances in information retrieval technology which are retrieved in the recent decade, some difficulties still exist as the obstacles to find the desired information for the searching users. Among these difficulties, we can name the limited number of terms which may be given in a search query, the lack of ways to better interpret the exact user desires, and the impossibility to choose the users' common interest as a robust basis for each particular user's query interpretation. These difficulties are the potential aspects of information search which argue to the needs to find some accurate measures to have a better precision and a better interpretation of each particular user's interest in each one of his searching subjects.

There are several user side search agents or global search engines that have been proposed in recent years. These projects mainly apply Information Filtering (IF), Information Retrieval (IR), query expansion, or semantic web techniques to improve the search results through the user's specialized long or short term interest or a group of user's public general interests. Most of them have some shortcomings. Some are not really supporting the different general purposed user searching types. Some are not adaptive according to each of the user's different interest subjects and his shifts in interest. Some others need a large set of user examples and a time-consuming batch pre-processing over the user interested documents. Finally, some of them are not clearly applicable to other languages than English.

This project is a crossroad to this issue in which we will introduce the architecture of ARIIA (Adaptive Reinforcement Iteration-based Internet Agent); a language independent user-side search agent that expands the user search queries according to his long time personal or professional interests and can adapt itself according to the user shifts in interest. To realize the functionality of ARIIA, we have used a Wikipedia[3] ontology extraction approach based on the Wikipedia hyperlinks that guaranties us a progressive evolution in user personal vocabulary during his continuous and real-time searches. ARIIA's functionality is based on several aspects as:

1. ARIIA uses the *relevance feedback* in its document selection.
2. It uses the hyperlink structure to measure the relevance of the terms and expands the search queries.
3. It mainly relies on the available existing infrastructure and services for its search. Talking in more details, it uses the Wikipedia free online encyclopedia and a general purposed search engine such as Google (or AltaVista) for its searches.
4. It creates a Boolean query with disjunctive expanded set of terms based on an encyclopedia hyperlink and the user feedback over the importance of the links.
5. It mainly relies on the iterations over the expanded queries and the relevance feedback.
6. It creates a user profile based on the query history and user interesting document contents.

[3] A free encyclopedia available at: www.wikipedia.org

7. It can be used as a Multilanguage search agent. ARIIA can be applied to searches with in any language which is used in Wikipedia.

To evaluate ARIIA's functionality, we define several measuring indicators considering the contents and the order of the resulted pages. The ARIIA's results are based on the Google's results and, in general, are almost better than the Google's: in the worst case, ARIIA shows us an improvement about at least 4 to 6 percent depending to the search type. In most of the cases this improvement stabilizes on 10 percent compared to the Google's result, and in its best situation, ARIIA improves the Google's result up to 20 percent and more.

TABLE DES MATIÈRES

DÉDICACE iv

REMERCIEMENTS v

RÉSUMÉ vi

ABSTRACT ix

TABLE DES MATIÈRES xii

LISTE DES TABLEAUX xv

LISTE DES FIGURES xvi

LISTE DES SIGLES ET DES ABREVIATIONS xvii

INTRODUCTION 1

CHAPITRE 1: Présentation du problème de recherche sur Internet et d'indexation des documents 4

1.1. Evolution du domaine 4

1.1.1. Naissance de moteurs de recherche plus avancés 8

1.2. Brève revue des caractéristiques d'Internet : un environnement complexe et non structuré de recherche 9

1.3. Comportement de l'utilisateur web dans une tâche de recherche sur le web 12

1.3.1. Différent types de recherche web faite par l'utilisateur 14

CHAPITRE 2 : Revue critique des architectures d'indexation et de recherche des documents sur Internet 16

2.1 Moteurs de recherche: Présentation des travaux de ce domaine 16

2.1.1 Google (1998) 17

2.1.2 SWoogle 21

2.1.3 Tap 24

2.1.4 KartOO 28

2.2 Agent adaptatif d'utilisateur: Présentation des travaux de ce domaine 30

2.2.1 Nootropia 30

2.2.2 Sifter 32

2.2.3 Résumé des approches de ce chapitre 33

CHAPITRE 3 : Revue des méthodes d'expansion de requête de recherche 34

CHAPITRE 4 : Définition du problème étudié et cadre du projet 38

4.1 Méthodologie et approche utilisées 38

4.2 Le cadre de conception 43

CHAPITRE 5 : Description de ARIIA 45

5.1 Algorithme de fonctionnement de base de ARIIA 45

5.2 Structure détaillée des composants 49

5.2.1 Unité *User Request Analyzer (URA)* 49

5.2.2 Unité *Independent ENcyclopedia (IEN)* 51

5.2.3 Unité *Web Request Maker (WRM)* 52

5.2.4 Unité *Web page Evaluating and Scoring(WES)* 54

5.2.5 Unité *Interest PRofile (IPR)* 56

5.3 Implémentation 60

CHAPITRE 6 : Stratégies de validation et résultats obtenus 63

6.1 Choix des exemples de test 64

6.2 Les résultats de la première itération 69

6.2.1. Recherche exploratoire 70

6.2.2. Analyse des scores de recherche exploratoire pure (les sous questions "Other") 72

6.2.3. Recherche informationnelle (les sous questions sans "Other") 73

6.3 Les résultats pour les deuxième et troisième itérations 74

6.3.1. Recherche exploratoire (toutes les sous-questions contenant "Other"). 74

6.3.2. Recherche exploratoire pure (les sous-questions "Other") 79

6.3.3. Recherche informationnelle (les sous-questions sans "Other") 80

CHAPITRE 7 : Conclusion et travaux futurs 82

BIBLIOGRAPHIE 85

LISTE DES TABLEAUX

Tableau 1: Descriptions des messages 46

Tableau 2 : Format du message Page_score 57

Tableau 3 : L'ensemble des exemples de TREC2004 choisis pour le test 65

Tableau 4 : Les votes d'utilisateur sur les 10 premières pages retournées d'ARIIA 67

Tableau 5 : les coefficients exponentiels de pondération des pages de resultants 68

Tableau 6 : Les résultats obtenus pour la recherche exploratoire 71

Tableau 7 : Les résultats obtenus pour la recherche exploratoire (les sous-questions "Other") 72

Tableau 8 : Les résultats obtenus pour la recherche informationnelle 73

Tableau 9 : Les résultats des itérations de recherche obtenus pour la recherche exploratoire 75

Tableau 10 : Statistiques sur les nombres des extensions de chacun des titres des questions 77

Tableau 11 : Les résultats de recherche exploratoire 80

Tableau 12 : Les résultats de recherche exploratoire 81

LISTE DES FIGURES

Figure 1: L'architecture de Google (1998) 19

Figure 2 : L'architecture de SWoogle 22

Figure 3 : (a) La structure actuelle de recherche d'Internet (b) la structure de recherche d'Internet envisagée par (ANTONIOU G. et al 2004) 39

Figure 4 : Les composants d'ARIIA 45

Figure 5 : Unité User Request Analyzer (URA) 50

Figure 6 : INDEPENDENT ENCYCLOPEDIA (IEN) 52

Figure 7 : Unité WEB REQUEST MAKER (WRM) 53

Figure 8 : (en haut) WEB PAGE EVALUATING & SCORING (WES) (En bas) Interface de vote d'utilisateur 54

Figure 9 : Le schéma logique de Base de donnée de profile d'utilisateur 56

Figure 10 : L'unité Interest Profile (IPR) : la phase de vérification des extensions 58

Figure 11 : L'unité Interest Profile (IPR) : la phase d'enregistrement des termes 60

Figure 12 : Différence entre les valeur d'indicateur SE pour les thèmes généraux des questions de tableau 3 76

Figure 13 : Différence entre les valeur d'indicateur MS pour les thèmes généraux des questions de tableau 3 79

LISTE DES SIGLES ET DES ABREVIATIONS

ARIIA	Adaptive Reinforcement Iteration-based Internet Agent
CSS	Complete Search Set
CWE	Checked Word and its Extension
DAML	DARPA agent markup language
DES	Data Encryption Standard
E	$\bigcup \{e_i \mid (t_i, e_i, s_i)\ USS\}$
e_i	un terme constituant une extension de t
EMS	Efficacité de Moyenne des Scores
ESE	Efficacité de Score Exponentiel
EXT(t)	$\{(e_i, s_i) \mid e_i$ est un terme constituant une extension de t et $s_i = score(e_i)\}$
EXWO	EXtension Word
HTML	HyperText Markup Language
IBM	International Business Machines
IEN	Independent Encyclopedia
IPR	Interest PRofile
ISEQ	Internet Search Engine Query
MS	Moyenne des Scores
OIL	Ontology Inference Layer ou Ontology Interchange Language
OWL	Web Ontology Language
PACO	PAge COntents
PDF	Portable Document Format
P_i	Une page web
PS	Page Score
RDF	Resource Description Framework
SE	Score Exponentiel
SEA	Internet Search Engine Answer

s_i	Score de e_i
T	$\{ t_i \mid (t_i, e_i, s_i) \in USS \}$
t_i	Un terme de la requête
TREC	Text REtrieval Conference
UDBQ	User Data Base Query
URA	User Request Analyser
USQ	User Search Query
USQW	User Search Query Word
USS	User Search Set
UV	User Vote
$V(P_i)$	Score attribué par l'utilisateur à la page P_i
WES	Web Page Evaluating and Scoring
WHP	Wikipedia HTML Pages
WRM	Web Request Maker
WSQ	Wikipedia Search Query
W_{tk}	$\{ t_i \mid t_i$ est un terme associé à un hyperlien dans une page $P_k \in WHP \}$
WWWW	World Wide Web Worm
X_i	Un terme
XML	Extensible Markup Language
Y_i	Un terme

INTRODUCTION

Nos recherches sur Internet se soldent souvent par la frustration d'obtenir une grande quantité de réponses qui ne sont pas en relation avec ce que nous recherchons vraiment. Il est étonnant que nous puissions envoyer des navettes dans l'espace et des hommes sur la lune mais que nous ne puissions pas inventer une manière simple d'organiser l'information de l'Internet.

Une des raisons de ce problème est la mauvaise interprétation de la requête de recherche par les moteurs de recherche sur Internet. En d'autres mots, puisque une grande partie du processus de manipulation de requête de recherche utilisé par les moteurs de recherche est basé sur deux artefacts - le contenu des documents de l'Internet et la question de l'utilisateur - le jugement final des moteurs de recherche est toujours relié à l'interprétation la plus probable de la phrase de la requête plutôt qu'à la véritable intention de l'utilisateur. L'interprétation de la requête ne tient pas compte du contexte de la requête et des changements temporels d'intérêt de l'utilisateur.

Notre projet de recherche aborde directement ce sujet. Nous verrons comment l'utilisation de données encyclopédiques comme Wikipedia permet de créer un modèle de l'intérêt de l'utilisateur et, ultimement, une requête plus précise. Plus spécifiquement, nous présenterons l'architecture d'un agent personnalisé de recherche appelé ARIIA (Adaptive Reinforcement Iteration-based Internet Agent) qui peut adapter la requête de l'utilisateur pour mieux tenir compte de ses vraies intentions, au cours d'une session de recherche dans Internet qui s'étend sur plusieurs requêtes. Cela

se fait en effectuant de manière itérative quelques modifications sur la requête, pour améliorer les résultats selon ses changements d'intérêt pour les sujets recherchés.

En bref, ARIIA, le système que nous avons développé, est un agent personnalisé de recherche qui est applicable à plusieurs langues et qui peut être installé comme un outil complémentaire de recherche soit sur un poste de travail personnel, soit sur la machine d'un groupe d'utilisateurs. Il améliore, selon leurs intérêts personnels ou professionnels à long terme, les requêtes qui sont transmises aux moteurs de recherche publics d'Internet.

ARIIA utilise une méthode simple et efficace basée sur l'extraction de données encyclopédiques qui sont utilisées pour étendre la requête de l'utilisateur et ainsi aider à mieux atteindre son objectif final. Cette approche permet d'obtenir de meilleurs résultats de recherche que ceux des moteurs de recherche actuels.

Le fonctionnement global d'ARIIA peut être divisé en deux étapes principales : expansion de la requête de recherche de l'utilisateur, et élimination d'informations non pertinentes (filtrage d'informations) dans les pages retournées par le moteur de recherche sur le web. Dans l'étape d'expansion, l'agent étend la requête en y ajoutant un ensemble de termes. Plus précisément, l'expansion de la requête est faite en cherchant dans Wikipedia les termes entrés par l'utilisateur et en ajoutant des termes se retrouvant dans les définitions trouvées dans Wikipedia. L'étape de filtrage de l'information d'ARIIA est une étape dans laquelle l'agent essaie d'éliminer les termes non pertinents dans les documents retournés par le moteur de recherche, et de classifier les termes conservés selon leur degré de pertinence. Comme nous le verrons plus loin, les termes

conservés ont un impact dans l'itération suivante de la recherche menée par l'utilisateur, puisqu'ils fournissent plus de spécifications sur la requête de recherche. En fait, cette dernière étape est un processus qui essaie d'éliminer les résultats retournés non favorables selon un avis de l'utilisateur sur leur contenu et enregistre les résultats ainsi que les jugements de l'utilisateur dans un historique de vocabulaire qui sera utilisé pour les recherches futures.

Naturellement, l'atteinte de cet objectif n'est pas aussi simple qu'elle le semble à première vue, puisque quelques ajustements de paramètres, les limitations de l'environnement, et des détails fonctionnels doivent être respectés pendant le temps d'opération de l'agent. Nous avons testé d'abord chacun de ces aspects dans l'élaboration des composants internes d'ARIIA.

Après une présentation de l'architecture, nous montrerons quelques stratégies possibles de validation et détaillerons celles que nous avons choisies pour évaluer les performances d'ARIIA en comparaison avec l'utilisation d'un moteur de recherche classique comme Google.

Dans les chapitres 1, 2 et 3, nous présentons la problématique de la recherche d'information sur Internet par une introduction critique de plusieurs projets récents dans ce domaine. Le chapitre 4 définit le cadre du projet de recherche. Puis, la structure détaillée d'ARIIA est illustrée dans le chapitre 5. Ensuite, dans le chapitre 6, nous présentons plusieurs des stratégies de validation et de test, suivie des résultats obtenus pour chacune des stratégies choisies. En conclusion, quelques avenues possibles pour les travaux futurs sur l'approche sous-jacente à ARIIA seront présentées.

CHAPITRE 1: Présentation du problème de recherche sur Internet et d'indexation des documents

1.1. Evolution du domaine

Internet est le résultat d'une imagination visionnaire de gens au début des années 1960. Il permet aujourd'hui à des ordinateurs de partager l'information sur le développement de la recherche dans un grand nombre de domaines. La croissance d'Internet, qui est passée de centaines d'utilisateurs privés en 1990 à plus de 500 millions de personnes et plus de 96 millions de sites webs en 2006[4], montre le taux d'intérêt remarquable et l'importance des outils pour partager et rendre accessible l'information. De nos jours, le nombre croissant de sites web, et l'obtention des milliers de résultats pour chaque requête de recherche, font que la difficulté initiale de "comment partager l'information ?" a évolué vers un nouveau type de difficulté: "comment trouver l'information la plus appropriée parmi les résultats de recherche obtenus?". Une des raisons principales de ce problème est que depuis 1990 le nombre de documents dans les index a augmenté de plusieurs ordres de grandeur, alors que la capacité de lire les documents n'a pas augmenté pour les utilisateurs! Les gens voulaient (et veulent toujours) seulement regarder quelques dizaines de documents pour satisfaire leurs besoins. La croissance de taille de la collection de documents sur Internet a donc entraîné l'apparition de **moteurs de recherche.**

[4] http://www.netvalley.com/intvalstat.html.

Les moteurs de recherche doivent avoir une précision très élevée, c'est-à-dire qu'ils doivent trouver, parmi les dizaines de milliers de documents, la dizaine de documents les plus pertinents pour l'utilisateur. En effet, pour les utilisateurs d'Internet la notion de "pertinent" implique seulement *les dix ou vingt meilleurs documents*.

Au début de l'année 1990, la technologie des moteurs de recherche a dû s'améliorer dramatiquement pour suivre la croissance de l'information sur le web. En 1994, un des premiers moteurs de recherche web, le World Wide Web Worm (WWWW) avait un index contenant 110,000 pages web (selon SearchEngineWatch[5]). En novembre 1997, les moteurs de recherche plus performants tels que WebCrawler, ont prétendu pouvoir de classer de 2 millions à plus de 100 millions de documents web.

Auparavant, en mars et avril 1994, le World Wide Web Worm a reçu une moyenne d'environ 1500 requêtes de recherche par jour. En novembre 1997, le moteur de recherche AltaVista[6] a prétendu manipuler approximativement 20 millions de requêtes par jour. En 2000, un index très détaillé du web comportait des milliards de documents et, en même temps, le nombre de requêtes manipulées par les moteurs de recherche s'est accru incroyablement. Avec le nombre croissant d'utilisateurs du web, et les moteurs de recherche automatiques et gratuitement disponibles sur le web, il était prévisible que les moteurs de recherche de haute qualité traiteraient des centaines de millions de requêtes par jour en plus de classer un nombre énorme et croissant de documents web.

Compte tenu de cette croissance fulgurante, il n'est donc pas surprenant que même les internautes les plus expérimentés doivent toujours composer avec des requêtes qui

[5] http://www.searchenginewatch.com
[6] www.altavista.com

retournent un nombre considérable de documents non pertinents (ou de moindre pertinence) après une première ou seconde recherche sur un sujet.

Comme mentionné dans l'introduction, la première raison de ce problème vient de la nature des méthodes de balayage[7] et des algorithmes de classification choisis par les moteurs de recherche ainsi que la croissance exponentielle du nombre de documents. En outre, il faut rappeler l'incapacité de l'utilisateur de bien anticiper la formulation de la requête qui amènera les documents les plus pertinents. Mais y a-t-il une manière de s'assurer que le moteur de recherche a bien interprété et bien compris la requête et ce que l'utilisateur cherche vraiment ?

Pour mieux répondre à cette question, considérons le fait que, en même temps que le nombre des documents web augmente et que les moteurs de recherche améliorent leurs techniques d'indexation et de balayage des documents, l'intérêt des utilisateurs pour avoir des interfaces de recherche le plus évoluées se développe aussi. Ceci risque d'empirer encore le problème des mauvaises interprétations des requêtes par les moteurs de recherche. Aujourd'hui on peut rarement trouver un moteur de recherche qui ne fournit pas la possibilité d'utiliser des opérateurs logiques dans les requêtes. D'ailleurs, la limite du nombre des termes par requête est relativement grande. Par exemple, il y a toujours une possibilité d'avoir 32 termes distincts dans une requête de recherche lancée dans le moteur de recherche Google[8] et pour chacun des termes il y a la possibilité d'avoir une opération logique.

---

[7] Crawling methods
[8] www.google.com

Aussi, chaque mot peut avoir plusieurs significations et, en conséquence, plusieurs interprétations selon les différents contextes. D'ailleurs, dès que la requête s'étend à plus d'un terme, le nombre d'interprétations peut devenir aussi grand que le nombre de sous-ensembles ordonnés des termes fournis par l'utilisateur !

Dans ce travail nous ne considérons pas les diverses interprétations possibles en employant les opérateurs logiques. Nous ne considérons pas non plus la cardinalité des termes de la requête.

Par exemple, un ensemble des trois termes {"étudiant", "professeur", "guide"} peut avoir au moins trois interprétations différentes pour l'humain: "le professeur guide l'étudiant", " l'étudiant guide le professeur", "l'étudiant et le professeur sont guidés)", etc. Et plus de 15 interprétations possibles (le nombre est calculé par 3 ! x 1+ 2 ! x 3 + 3 = 15) telles que "professeur étudiant", "étudiant professeur", "professeur guide", "professeur guide étudiant", etc., pour la machine, puisque les moteurs de recherche actuels ne peuvent pas chercher selon les sens des phrases. Ils sont quand même censés accepter et traiter les erreurs commises par les utilisateurs lorsqu'ils soumettent leurs requêtes[9].

Ce dernier problème est assez sérieux et il y a peu de travail qui lui a été consacré. Ce problème doit être résolu du côté de l'utilisateur et ne peut pas être traité par les moteurs de recherche, car pour chaque question il peut y avoir plusieurs interprétations

[9] Par exemple aujourd'hui, les moteurs de recherche comme Google corrigent nos requêtes et nous proposent leurs corrections de l'orthographe.

possibles pour un même terme avec une gamme aussi étendue que le nombre d'intérêts des utilisateurs d'Internet.

1.1.1. Naissance de moteurs de recherche plus avancés

La plupart des moteurs de recherche actuels tels qu'AltaVista et Yahoo! utilisent deux étapes principales pour mettre à jour leur information: le balayage ("*web spider*" est proposé comme traduction de balayage, alors que "*web crawler*" avait été utilisé précédemment), et l'indexation de documents. La première opération consiste à consulter les documents du web en recherchant leurs adresses URL et les hyperliens. La deuxième étape consiste à analyser et indexer des documents selon leurs contenus. Certains moteurs de recherche, comme SWoogle (*DING L. et al. 2004)* et Tap (*GUHA R.V. et al. 2006*) utilisent les techniques du web sémantique pour avoir une bonne interprétation de la requête. Cependant, un des problèmes des moteurs de recherche basés sur le web sémantique est qu'il y a toujours une grande partie des documents d'Internet qui ne sont pas encore indexés et classés par ces moteurs. Nous fournissons une analyse plus détaillée de ces types de moteurs de recherche dans le prochain chapitre. Dans ce projet, nous nous intéressons plus particulièrement à l'interprétation de la requête de l'utilisateur. En d'autres mots, comment mieux cerner l'intérêt de l'utilisateur afin de mieux interpréter sa requête. Nous essayons de l'analyser en utilisant de nouvelles techniques basées sur l'utilisation d'ontologies dans la recherche d'information sur le web. Une *ontologie* est une spécification d'une conceptualisation

d'un domaine de connaissance. Dans le reste de ce chapitre, nous verrons d'abord quelques architectures existantes de moteur de recherche sur le web. Puis dans la section 1.3, nous tenterons d'identifier quelques comportements d'utilisateurs ainsi que différents types de recherche.

1.2. Brève revue des caractéristiques d'Internet : un environnement complexe et non structuré de recherche

Il y a six problèmes principaux concernant la recherche de données sur Internet :

1. **Diversité des significations d'un terme ou d'une expression:** cette caractéristique rend la tâche plus difficile pour les moteurs de recherche, qui ont du mal à se retrouver parmi les différentes significations possibles pour une même requête. Par exemple, la présence de l'abréviation "MVP" dans une requête peut confondre le moteur de recherche par ses différentes interprétations, telles que (1) " Microsoft Most Valuable Professional", qui est une récompense annuelle donnée par Microsoft, (2)"Minimum Viable Population", un terme scientifique utilisé en écologie, (3) " Missouri and Valley Park Railroad", qui est une forme régionale d'utilisation pour cette abréviation, (4) "Mitral Valve Prolapse", un terme médical spécifique pour une maladie cardiovasculaire, (5) "Most Valuable Player", qui est un terme généralement employé aux États Unis dans le domaine du sport, ou (6) le "Model-view-

presenter", qui est un type d'architecture de logiciel. Il y a un grand nombre d'exemples similaires dans toutes les langues : abréviations, argot, termes scientifiques, expressions idiomatiques, etc. La deuxième difficulté, comme mentionné à la sous-section 1.1 pour l'exemple de requête "professeur étudiant guide", est de trouver la meilleure interprétation pour un ensemble de plusieurs termes afin de trouver et renvoyer les résultats désirés. Rappelons ici qu'il y a trois façons de mieux spécifier la requête de recherche: soit ajouter des termes à la requête, ou en éliminer, soit remplacer les termes de la requêtes avec les termes plus significatifs d'une façon qui peut permettre de surmonter ce problème, mais ces méthodes ne fonctionne pas toujours car il n'y a aucune règle définitive pour cette spécification.

2. **Diversité dans les formats de fichiers de données:** Ceci est dû au manque de contrôle sur les types des documents disponibles sur Internet, qui exige donc que les moteurs de recherche puissent localiser tous les sujets désirés par l'utilisateur dans différents formats standards ou non standards : HTML, PDF, XML, etc.
3. **Volume élevé des documents pertinents:** Par exemple, pour la simple requête "HTML", Google renvoie plus de 4 milliards de résultats!
4. **Variation du nombre du documents ainsi que de leurs sujets:** Ce changement varie selon le rythme de croissance et de changement de l'information dans chacun des domaines auxquels l'utilisateur s'intéresse. Ce comportement rend impossible que les utilisateurs soient à jour par rapport aux changements qui se

produisent à chaque moment dans les documents de leurs domaines d'intérêt. Il y a beaucoup de sites web avec des mises à jour périodiques très rapides qui forcent les moteurs de recherche à faire de même.

5. **Ambiguïté dans les titres des pages web:** Cette caractéristique rend impossible le classement de pages web en se basant seulement sur leurs titres ou de manière générale sur quelques étiquettes HTML. D'autre part, le contenu d'un même document peut être intéressant pour différents utilisateurs qui l'interprètent chacun de manière différente. Par exemple, le contenu de l'article intitulé " Extracting a 3DES key from an IBM 4758" peut être intéressant pour deux utilisateurs différents, un désirant trouver quelques informations sur le sujet de l'algorithme de codage DES, et un autre qui veut découvrir quelques informations sur les machines d'opérations bancaires. La pertinence de ce titre n'est pas la même pour les deux utilisateurs, puisqu'elle ne montre pas clairement le sujet "banking" par rapport au mot technique DES qui est le nom de l'algorithme de codage. Ce problème peut être généralisé pour la plupart des articles sur Internet: leur contenu couvre souvent plusieurs sujets et leur titre ne couvre pas entièrement leur contenu.
6. **Difficulté pour calibrer l'opération de recherche par deux techniques classiques; filtrage d'information[10] et l'extraction d'information[11]:** Ces deux stratégies de base se présentent comme les techniques de recherche d'information pour un grand ensemble de données et leur importance varie selon

[10] Information Filtering
[11] Information Retrieval

le contexte et le type de recherche effectué (*MOSTAFA J. et al.1997*). L'extraction d'information est principalement employée pour répondre aux besoins de recherche en temps réel, en essayant d'accéder à un ensemble de documents déjà stockés et classifiés à l'aide de méthodes d'indexation. Dans cette technique, l'extracteur de l'information doit bien interpréter, ou adapter le plus grand nombre possible de réponses pour chaque requête au cours d'une période minimale de temps. Le but final d'extraction d'information est de maximiser la quantité de documents pertinents retournés comme réponse. Au contraire, le filtrage d'information est une technique dans laquelle le but principal est l'élimination des données moins pertinentes d'un grand ensemble partiellement pertinent. Cette technique est principalement employée lorsqu'on est confronté à un changement dynamique du nombre de documents dans un entrepôt des documents et nécessite généralement une longue durée de temps. Ces deux techniques sont nécessaires pour une recherche sur Internet et les deux doivent être justifiées et appliquées pour l'extraction de données selon le but de la recherche.

1.3. Comportement de l'utilisateur web dans une tâche de recherche sur le web

L'utilisateur qui effectue une recherche d'information sur le web a des comportements qui dépendent de sa connaissance et de ses besoins sur le sujet de

recherche (*BHARAT K. 2005*). Il y a une variabilité d'intérêts pour chaque sujet. L'utilisateur exécute plusieurs recherches en parallèle pour un ou plusieurs sujets en même temps. Il varie ses requêtes pour un même sujet. Chacune de ses recherches peut s'étendre sur plusieurs sessions et il redémarre parfois son fureteur pendant les périodes de recherche. Après avoir obtenu les premières informations désirées, il n'est pas toujours sûr d'avoir la meilleure réponse et en conséquence il désire souvent trouver plus de ressources liées au sujet recherché. Il commence la recherche avec un minimum de mots-clés, Mais surtout, son intérêt peut évoluer selon les résultats obtenus ou simplement selon le temps.

1.3.1. Différent types de recherche web faite par l'utilisateur

Outre les intérêts, on peut aussi caractériser le comportement de l'utilisateur par trois différents types de recherche sur Internet:

1. **La recherche navigationelle (*JOACHIMS Th. et al. 2005*)** : L'utilisateur désire trouver une adresse URL précise d'une page web. En d'autres mots, il cherche une page précise et ne connaît pas son URL exacte. Comme exemple générique de ce type de recherche, mentionnons la recherche de la page personnelle officielle d'une personne ou de la page officielle d'une organisation particulière.
2. **La recherche informationnelle (*JOACHIMS Th. et al. 2005*) :** L'utilisateur désire une réponse à sa requête sans se préoccuper des adresses ou des types de pages web qui sont retournées. Comme exemples de ce type de recherche, on peut citer les requêtes générales comme "qui était le mieux payé dans les années 60 aux États Unis ?" ou "quelle est la plus haute montagne du Québec?". Rappelons que selon (*MISHNE G. et al. 2005*), environ 13 pourcent de 7 millions des résultats des requêtes échantillon de recherche retournés par AltaVista contiennent des réponses que l'on peut retrouver dans l'encyclopédie Wikipedia. Les pages de Wikipedia se retrouvent parmi les dix premiers résultats retournés par ce moteur de recherche.
3. **La recherche exploratoire :** Ce type de recherche est défini par nous. Dans la recherche exploratoire, l'utilisateur cherche tous les documents pertinents à un sujet particulier. Ce type de recherche journalistique est toujours un sous-ensemble d'un

des deux types de recherche mentionnés ci-dessus qui se comporte d'une façon continue sans interruption. Par exemple, des requêtes telles que "trouver tous les événements politiques qui se sont produits dans les deux derniers mois de la vie de John F. Kennedy", "trouver les dernières nouvelles de la tornade Katrina", ou "trouvez tous les URLs des pages web officielles d'une entreprise particulière" correspondent à ce type de recherche.

CHAPITRE 2. Revue critique des architectures d'indexation et de recherche des documents sur Internet

Dans cette section nous présentons une revue critique de plusieurs projets dans le domaine de la recherche d'information.

2.1 Moteurs de recherche: Présentation des travaux de ce domaine

On peut identifier environ sept différents types de moteurs de recherche classifiés selon *SearchEngineWatch*[12] :

- moteurs de recherche universels
- méta balayeurs[13]
- moteurs de recherche de nouvelles
- moteurs de recherche de vente et de commerce électroniques
- moteurs de recherche multimédias
- moteurs de recherche spécifiques
- moteurs de recherche régionaux (spécifiquement définis pour les pays)

Selon le type de recherche effectuée par un utilisateur, ces moteurs de recherche peuvent être classifiés en deux catégories principales. Le premier type comprend les

[12] http://www.searchenginewatch.com
[13] Meta crawlers

moteurs qui sont plutôt utilisés pour surfer sur le web en utilisant les hyper-liens des pages web. L'opération de recherche effectuée par ces moteurs commence souvent par le choix des mots clés d'un index de haute qualité préparé par des humains, comme dans Yahoo!. Le deuxième type comprend les moteurs de recherche qui donnent la possibilité de lancer une requête dans des interfaces texte telles que Google et AltaVista. Dans ce projet, nous nous attardons plutôt sur les moteurs de recherche universels dont la recherche n'est pas limitée à un critère particulier de recherche. Dans ce chapitre nous présentons brièvement certains d'entre eux, qui sont généralement employés comme outils de recherche universels ou universitaires.

2.1.1 Google (1998)

Actuellement, Google[14] est l'un des moteurs de recherche les plus utilisés. C'est un moteur de recherche de grande échelle qui utilise la structure d'hypertexte des pages web. Il a été créé en 1998 comme un prototype de moteur de recherche en milieu universitaire. Il n'accepte aucune commandite commerciale pour influencer la pertinence des liens et est gratuitement disponible pour tous les utilisateurs du web. Son architecture initiale (*BRIN S. et al.1998*) emploie deux techniques générales: **Page Rank** et **Anchor Text** :

- L'algorithme **Page Rank** est employé pour classer les pages en utilisant une mesure quantitative d'importance pour chaque page. Cette mesure est basée

[14] Le nom est venu de la prononciation du terme anglais googol, qui signifie 10^{100}

sur le nombre de pages qui y réfèrent. Cet algorithme se calcule selon la formule suivante:

$$PR(A) = (1-d) + d(PR(t_1)/C(t_1) + ... + PR(t_n)/C(t_n)) \quad (2.1)$$

Dans cette formule d est une constante, souvent fixée à 0.85. t_1 à t_n sont des pages qui réfèrent à la page A. $C(t_i)$ est le nombre des liens extrants de la page t_i. Dans cet algorithme, les références peuvent parfois former un cycle entre les hyperliens, l'algorithme doit donc tenir compte pour éviter les boucles infinies.

- L'algorithme **Anchor Text (*appelé ancre*)** examine le texte des hyperliens dans les pages HTML. Selon (*BRIN S. et al.1998*), le nombre d'ancres dans les pages web est en moyenne dix fois plus grand que le nombre de pages. L'algorithme Anchor Text est une méthode qui, pour toutes les pages balayées dans un entrepôt de documents, crée un graphe entre ces liens qui existe dans les pages. L'avantage de cette méthode est de profiter des informations plus précises des descriptions des ancres qui fournissent les sujets de URL. Par exemple, les pages qui n'ont aucune information textuelle, telles que les images, peuvent être mieux classées par les ancres qui réfèrent à elles que leur URLs.

La première architecture de Google est illustrée à la figure 1 inspirée de (*BRIN S. et al.1998*):

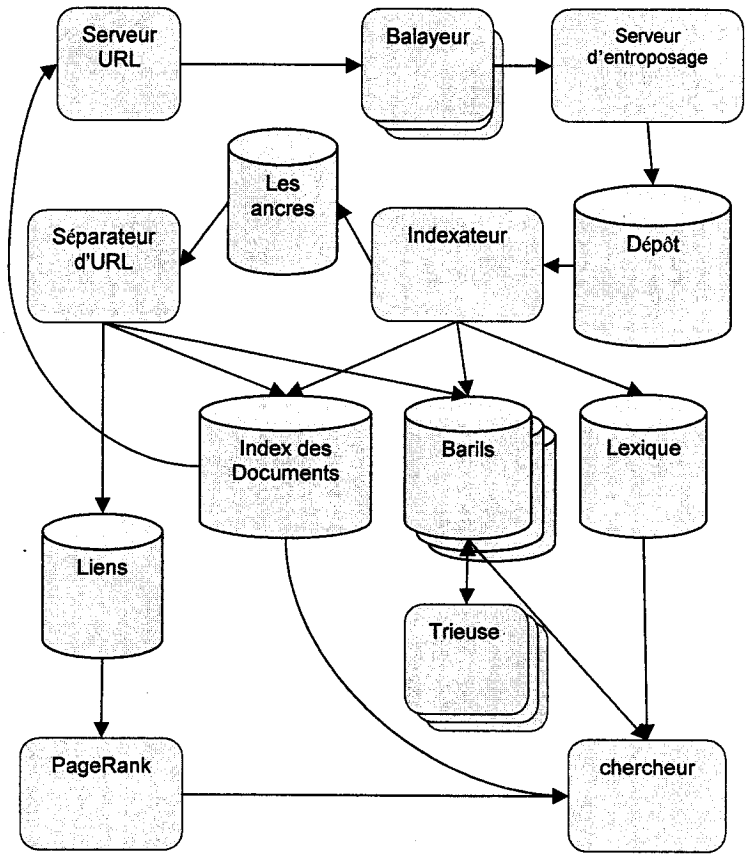

Figure 1: L'architecture de Google (1998)

En bref, les étapes du processus sont les suivantes:

- Le serveur URL envoie la liste des URLs aux balayeurs distribués.
- Les pages webs seront téléchargées par plusieurs balayeurs distribués.
- Les pages web cherchées sont envoyées au serveur d'entreposage, qui les compresse et les stocke de telle manière à ce qu'elles puissent être consultées à l'avenir par un code d'identification.
- Puis l'indexateur analyse les mots et les liens dans les pages stockées et les classifie selon leur ensemble des mots, leur taille de police, et leurs positions

dans les documents et distribue les documents classés à un ensemble d'entrepôts de données[15]. L'indexateur stocke aussi les informations importantes sur les hyperliens des pages dans un fichier d'ancres.

- Les pages sont ensuite triées selon plusieurs critères comme leurs tailles, leurs mots clefs, etc.
- Le séparateur de URL[16] lit le fichier d'ancres et convertit les URL relatifs en URL absolus. Par exemple un URL commençant par les caractères "../" ou "./" sera converti en URL complet[17]. Il met aussi les ancres liées à l'URLs dans un index[18] qui permet de trouver un URL en utilisant les mots de la page. En plus, il crée une base de données des hyperliens.
- La base de données des hyperliens est employée pour l'algorithme Page Rank.
- Le module de tri emploie les codes d'identification des documents pour créer un index inversé qui a la même structure que l'index direct. Cette indexation permet de trouver les pages désirées selon leurs éléments textuels comme leurs mots clefs. Et ceux-ci permettent le fusionnement rapide des différentes listes de documents).
- Le module de tri crée aussi une liste des codes d'identifications pour les mots.
- Un programme appelé DumpLexicon prend toutes ces listes ainsi que le lexique produit par l'indexateur et produit un nouveau lexique à être employé par l'unité de recherche.

---

[15] barrel
[16] URL resolver
[17] Consultez à http://www.webreference.com/html/tutorial2/3.html pour plus de détails
[18] Forward index

- Enfin, le module de recherche est lancé par le serveur web et emploie l'index inversé, et la valeur de Page Rank pour répondre aux requêtes de l'utilisateur.

La principale lacune de Google est que les pages web retournées comme résultats de recherche ne sont pas personnalisées selon les intérêts des utilisateurs spécifiques. L'interprétation des requêtes sera la même pour les utilisateurs qui décalent ou changent complètement leur domaine d'intérêt sur un sujet au cours du temps. Un autre problème de Google est la vulnérabilité de l'algorithme de Page Rank à ce qu'on appelle la bombe de Google. La bombe de Google est une tentative d'influencer le rang d'une page dans les résultats retournés par le moteur de recherche de Google. Dans l'algorithme Page Rank, les membres d'une grande communauté en ligne peuvent affecter les résultats des recherches de Google en liant leurs sites webs à une page web spécifique. Actuellement, il n'y a aucune solution définitive à ce problème.

2.1.2 SWoogle

SWoogle (*DING L. et al. 2004*) est un moteur de recherche qui balaye le web de page en page et qui analyse et indexe le contenu des documents RDF et OWL selon leurs structures sémantiques et les indexe selon les ontologies. SWoogle a été créé en 2004 par une équipe de recherche à l'université Maryland, Baltimore. Pour son indexation et son extraction d'information, SWoogle extrait les méta-données pour

chaque document découvert, et calcule des relations sémantiques[19] entre différents documents web enregistrés dans sa base des documents. Une des propriétés intéressantes de SWoogle est ***Ontology Rank*** qui classifie les ontologies selon une mesure d'importance inspirée de l'algorithme ***Page Rank*** de Google. La figure 2 (*DING L. et al. 2004*) illustre les quatre composants de l'architecture de SWoogle.

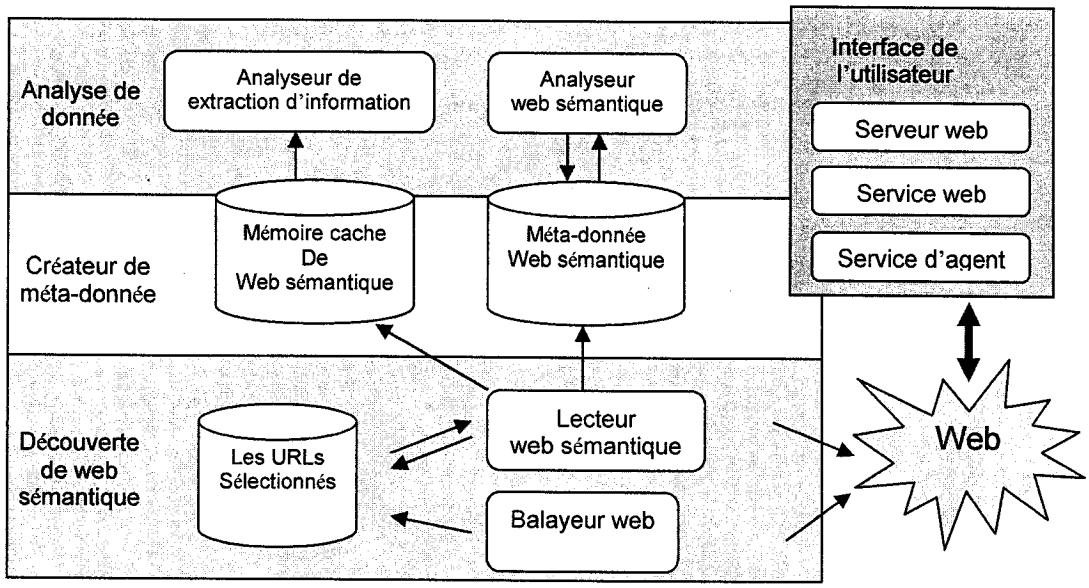

Figure 2 : L'architecture de SWoogle

SWoogle effectue quatre opérations pour augmenter la performance dans une session de recherche. Ces opérations sont encapsulées dans chacune des quatre composants de son architecture:

1. Le composant d'interface de l'utilisateur qui donne la possibilité d'entrer la requête de recherche sur le web.

[19] La relation sémantique signifie les termes qui sont définis par les autres termes, ainsi que la priorité entre les différentes versions de définitions des ontologies, l'inclusion d'une ontologie dans une autre ontologie, la compatibilité entre les différentes versions des ontologies.

2. Le premier niveau de l'engin est une connexion entre l'unité d'analyse des documents et l'interface utilisateur sur le web qui est employée pour interpréter les requêtes en utilisant la relation sémantique entre les termes. Cette couche donne la possibilité de chercher pour les diversités des significations des termes, et permet de choisir chacune des ontologies par l'utilisateur. Cette dernière possibilité tire profit de l'évaluation de l'utilisateur pour les résultats de recherche et classe les ontologies selon leur popularité chez les utilisateurs de SWoogle.
3. La deuxième couche est employée pour enregistrer les relations sémantiques qui existent entre les différentes ontologies qui sont découvertes par SWoogle.
4. La troisième couche aide le moteur de recherche à trouver automatiquement les documents du web sémantique et à les classer pour les recherches futures.

La couche inférieure de SWoogle se divise en deux parties: le balayeur de Google[20] et le balayeur focalisé[21]. Le premier balayeur recherche le format sémantique des documents web de manière ad hoc en utilisant Google et le deuxième se concentre sur des documents sémantiques de certains sites web spécifiques qui sont choisis par l'administrateur de SWoogle. Cette couche peut détecter plusieurs codages de document tels que XML, RDF, et des langages d'ontologie tel que OWL, OIL, et DAML etc.

[20] Google crawler
[21] Focalized crawler

Un des problèmes majeur de SWoogle est sa lenteur pour chercher et classer les contenus sémantiques des documents, car après deux ans, SWoogle n'a balayé que deux millions de documents environ. Un autre problème de SWoogle est que sa recherche et les résultats retournés ne sont pas personnalisés; il généralise toujours les requêtes et essaie de retourner les résultats de toutes les ontologies correspondant aux termes de la requête. De plus, il y a toujours un manque d'adaptabilité face aux besoins spécifiques d'un utilisateur particulier, car tous les utilisateurs reçoivent les mêmes résultats.

2.1.3 Tap

Tap est un moteur de recherche de méta-données qui est développé au département d'informatique de l'université Stanford. C'est un moteur de recherche qui crée un ensemble sur demande de segments d'information pour améliorer la recherche des sujets par l'utilisateur chercheur web. Le code source de Tap[22] est présenté dans (*GUHA R.V. et al. 2003*) et est disponible sur le site web de ses créateurs. Il a quelques méthodes uniques d'extraction d'ontologie qui le différencient de SWoogle. Tap emploie l'intégration de la connaissance en utilisant plusieurs relations sémantiques distribuées et séparées d'ontologie sous la forme de graphes sémantiques entre les mots dans lequel les nœuds de ce graphe sont les mots clefs et les arcs sont les relations entre les mots clefs.

[22] Site web de Tap se trouve en http://tap.stanford.edu

Cette architecture emploie un type de requête appelé Getdata qui crée, pour chaque terme de la requête, un graphe représentant ses extensions sémantiques. Pour répondre à la requête de l'utilisateur, il crée un graphe global en reliant ces graphes selon les relations sémantiques existantes entre leurs nœuds. L'architecture de Tap est basée sur quatre éléments principaux. D'abord, les catégories des contextes de recherche pour les graphes sémantiques crées par Tap sont des sujets de recherche déjà classifiés et indexés par les humains. La taille de cette indexation est limitée et basée sur quelques sites web contenant de l'information de haute qualité. Par exemple les contextes de recherche de Tap englobent les catégories de finance, jeux logiciels, cinéma, etc. Deuxièmement, chaque base de connaissance sous forme de graphe est formée localement est indépendamment des autres graphes locaux. Troisièmement, la complexité d'intégration du graphe global se base sur les relations entre les graphes locaux de bases de connaissances et dépend fortement de la taille des tous les graphes locaux qui participent à l'intégration. Et finalement, pour les requêtes de recherche d'utilisateur, il n'y a aucune opération batch afin de créer le graphe sémantique intégrée de Tap.

Tap a quelques inconvénients. Comme avec SWoogle et Google, les résultats retournés pour une requête ne sont pas personnalisés et en conséquence ne sont pas adaptatifs selon les besoins personnels et la variation des intérêts de l'utilisateur. D'ailleurs, Tap généralise la requête. Cela signifie que chaque utilisateur reçoit toutes les dénotations possibles (par exemple les diverses significations possibles) pour les termes de sa requête de recherche tandis que son vrai désir correspond à une dénotation

unique et spécifique qui se limite à un petit sous-ensemble de tous les dénotations diverses retournées par Tap. Comme c'est le cas avec SWoogle, ceci oblige l'utilisateur à effectuer une autre recherche dans les résultats retournés par Tap. Par exemple avec la requête "Java", l'utilisateur doit d'abord vérifier les diverses significations de terme "Java" et ensuite il doit choisir son chemin désiré à propos de ce terme. Ce problème se généralise pour tous les termes, même pour les termes qui sont moins ambigus que le terme "Java". D'autre part, compte tenu du fait que le lexique de Google (1998) contenait environ 14 millions de termes différents aux premiers jours de démarrage de Google (*BRIN S. et al.1998*) et qu'à cette époque le nombre des pages balayées par Google était environ de 24 millions, et qu'actuellement le nombre de pages dépasse le milliard, on peut facilement conclure qu'aujourd'hui le nombre de termes dans lexique d'un moteur de recherche comme Google est beaucoup plus élevé que 14 millions (plus que le nombre des termes de toutes les langues d'une encyclopédie telle que Wikipedia). Comparé à cela, la rapidité de balayage, d'indexation, et de création de la base de connaissance de Tap est trop lente ; on voit que après trois ans son index contient moins de 20% de tous les termes d'un dictionnaire anglais[23]. D'autre part, pour chaque nouveau terme qui apparaît et qui n'est pas encore enregistré dans la base de connaissances de Tap, tous les sites déjà indexés par Tap doivent être encore recherchés afin de trouver les nouvelle relations entre ce terme et tous les mots clefs des bases de connaissances locales déjà existantes. Cette opération nécessite énormément de ressources puisque Tap crée un graphe complet des relations sémantiques des termes.

[23] Actuellement la base de connaissance de Tap possède 1,000,000 termes contenant toutes les formes grammaires et linguistiques.

De plus, Tap effectue également l'identification de tous les types de relation existant entre eux. Par exemple entre deux nœuds "étudiant" et "professeur" il peut exister plusieurs relations sémantiques comme "enseigner", "diriger", etc. Si on considère que les deux ontologies "étudiant" et "professeur" soient les nœuds d'un graphe et les relations sémantiques entre les ontologies soient les arcs reliant les nœuds du graphe, cette opération revient à ajouter un nœud à un graphe entièrement connecté qui a plus d'un arc entre chaque paire de noeuds!

Un autre inconvénient de Tap (et aussi de SWoogle) est que ce moteur de recherche est fortement basé sur la fiabilité des utilisateurs d'Internet pour assigner des titres consistants à leurs propres documents puisque il n'y a aucune vérification globale de l'exactitude des choix des noms des objets et des pages dans les documents web. Par exemple : le nom "tornade" peut être assigné à un ouragan ou à un site de musique. Autre exemple, les noms des gens qui ont leurs propres sites web personnels augmentent l'ambiguïté des noms des personnes et complique encore le processus de balayage de Tap puisque chaque personne est différente des autres qui sont nommées par le même titre. En conséquence, pour chaque nom de personne qui s'ajoute, il y a une nouvelle ambiguïté qui doit être reconnue par Tap. En général, Tap n'est utilisable que pour une catégorie limitée des recherches navigationelles ou informationnelles.

Finalement, l'autre inconvénient de Tap (et aussi de SWoogle) est que sa recherche est limitée à la langue anglaise puisque Tap a besoin de sa propre base de connaissance pour chaque langue.

2.1.4 KartOO

KartOO est un site Web qui propose deux nouveautés: sa classification ontologique de résultats de recherche et sa façon de présenter les résultats de recherche. Il semble que malgré sa nature commerciale, il a plusieurs aspects qui peuvent être intéressants même pour une recherche académique.

Anecdote intéressante, nous n'avons pas trouvé le nom de Google parmi les 14 moteurs de recherche qui sont énumérés comme principales sources d'informations de KartOO. De plus, comme mentionné par son développeur, ce robot ne supporte pas les méta-étiquettes des documents en XML.

En employant KartOO, on obtient une présentation graphique qui illustre plusieurs classifications des différents aspects d'un simple mot tel que "étudiant". Cependant, cette approche n'est pas adéquate pour une recherche d'un article tel que "User Assessments in Visual Web Genre Classifier", car dans ce cas l'utilisateur doit exécuter au moins trois navigations dans les icônes de KartOO pour obtenir l'adresse exacte du document pdf, alors que la même recherche dans Google retournera le document recherché parmi les 2 premiers résultats.

Ceci suggère que KartOO n'est utile que pour les recherches imprécises qui englobe une vaste gamme des contextes et où l'utilisateur est disposé à (et suffisamment patient pour le faire!) fureter parmi les résultats retournés. En autres mots, c'est un outil de

recherche pour visualiser les structures des liens entre les documents mais pas leurs contenus. Voici les possibilités de KartOO:

1. Au niveau des aspects linguistiques, KartOO a :

   - la possibilité d'identifier automatiquement le format des documents, tel que XML, pdf, etc.

2. Au niveau du moteur de balayage de sites Web et de la base de données de URL, KartOO offre:

   - une recherche périodique des sites relatifs au sujet d'un domaine;
   - l'utilisation d'un analyseur et d'un interpréteur pour analyser la signification des mots en utilisant les dictionnaires;
   - la classification des mots selon leur position dans le document;

3. Au niveau de l'analyse des mots et de l'heuristique appliquée, KartOO offre:

   - un thésaurus de synonymes pour les mots;
   - un support des expressions booléennes dans les requêtes;
   - un support de l'appariement de formes[24] pour les combinaisons des mots

4. Au niveau du filtrage de résultats[25] et de l'adaptation du temps de recherche, KartOO peut:

---

[24] Pattern matching

- offrir un filtrage tel que ceux utilisés pour le contrôle parental;
- limiter les résultats recherchés à un certain nombre de pages;
- ajuster le temps moyen de recherche par le biais d'un outil statistique et le nombre des pages par recherche.

2.2 Agent adaptatif d'utilisateur: Présentation des travaux de ce domaine

D'après nos recherches, il y a au moins neuf projets d'interfaces de recherche d'information personnalisées qui diffèrent selon leur type d'architecture ou leur performance pour répondre aux différents types de recherche d'information (*MOSTAFA J. et al. 1997, NANAS N. et al. 2004, MISHNE G. et al 2005, DING L. et al. 2004, SHEN X. et al. 2005, SHEN X. et al. 2005, GUHA R.V. et al. 2003, BHARAT K. et al. 2005, IWAYAMA M. 2000*). Parmi ces projets il y a deux architectures (*MOSTAFA J. et al. 1997, NANAS N. et al. 2004*) qui semblent pertinentes par rapport aux objectifs de ce projet. Nous en donnons une revue détaillée dans les deux sous-sections suivantes.

2.2.1 Nootropia

Nootropia (*NANAS N. et al. 2004*) est un agent pour filtrage adaptatif des documents. Les concepteurs de Nootropia mettent l'accent sur l'habilité d'auto-

[25] Output filtering

organisation de ce système qui détecte les intérêts multiples des utilisateurs pour s'adapter aux décalages et changements d'intérêts de l'utilisateur. Dans ce système, le profil de l'utilisateur est un réseau hiérarchique de termes qui est construit en quatre phases: d'abord, pour chaque utilisateur, le système extrait les termes dans chaque document et leur assigne un poids selon leurs fréquences de répétition dans les documents. La deuxième étape met à jour les poids des termes du profil et enlève les termes du profil quand leur poids devient plus petit qu'un seuil prédéfini. Ensuite, le système ajoute dans le profil les nouveaux termes recueillis. Finalement, il établit une corrélation entre les termes qui sont enregistrés dans la base de données de profils. Dans Nootropia, plus une hiérarchie des corrélations des termes est profonde, plus efficace sera le rôle de cette hiérarchie pour classifier les nouveaux documents.

Nootropia est basé sur le contenu entier des documents déjà classifiés et enregistrés dans le profil de chaque utilisateur. Il ne peut donc pas être employé pour la recherche dans les environnements dynamiques d'information comme Internet puisque, pour chaque sujet d'intérêt qui n'est pas en relation avec les autre sujets enregistrés dans le profil, il a besoin d'un ensemble de documents pré-évalués et pré-classifiés pour établir un jugement sur la valeur de nouveaux documents. En d'autres mots, ceci signifie que pour chaque requête de l'utilisateur, tous les intérêts possibles de cet utilisateur doivent être connus et tous les documents semblables et pertinents doivent être recueillis avant que l'utilisateur envoie sa requête vers le moteur de recherche. Quand les résultats du moteur de recherche sont retournés, avant que l'utilisateur ne les ouvre, Nootropia doit exécuter un long prétraitement et regroupement par similarité de tous les résultats

retournés, qui peuvent représenter une grande quantité de documents. En plus, ce système n'utilise pas un extracteur de requête pour manipuler l'ambiguïté[26] de chaque terme ou groupes de termes dans sa hiérarchie.

2.2.2 Sifter

Sifter (*MOSTAFA J. et al.1997*) est un agent de recherche dont l'infrastructure présente certaines ressemblance avec l'architecture proposée dans ce projet. Ce système est un outil de filtrage d'information qui classifie un ensemble statique de documents par une méthode d'apprentissage non supervisée et puis filtre les documents selon le contenu d'un thesaurus et un modèle de l'intérêt de l'utilisateur qui est créé par une méthode d'apprentissage basée sur le renforcement.

Sifter classifie les documents en se servant d'un feedback facultatif de l'utilisateur et recueille les informations pertinentes du profil d'intérêt de l'utilisateur selon le contenu entier des documents qui sont intéressants pour lui. Il classifie d'abord les documents par une méthode de traitement par lot, en utilisant un thesaurus. Il les filtre ensuite en utilisant la base de données contenant les termes du profil d'intérêt de l'utilisateur. Ce système doit accéder à tous les documents et les classifier avant la soumission de la requête par l'utilisateur. D'un autre côté, Sifter utilise la similarité du contenu entier des documents pour comparer deux documents, ce qui nécessite donc beaucoup de temps pour le prétraitement des documents. Dans le prochain chapitre, nous verrons que cette

[26] Ambiguïté peut correspond a diversité de signification des termes par exemple les différents sens possibles pour le terme "Java"

méthode a plusieurs similarités avec la nôtre, mais diffère aussi sur plusieurs points. Par exemple l'architecture que nous proposerons dans le chapitre 5 étend d'abord la requête de l'utilisateur et ensuite lance la requête vers le moteur de recherche. Cette expansion joue également le rôle d'une classification intégrée qui nous permet de l'employer sur les documents en ligne.

2.2.3 Résumé des approches de ce chapitre

Nous avons vu dans ce chapitre un résumé des deux types, global et personnalisé, d'outils de recherche d'information sur le web. Dans le prochain chapitre nous verrons des approches qui nous aident à mieux adapter les requêtes de l'utilisateur par une technique d'expansion de sa requête originale. Ce que nous désirons est d'intégrer une des méthodes d'expansion de requête de recherche avec une architecture de personnalisation de la recherche et ainsi améliorer les résultats.

CHAPITRE 3. Revue des méthodes d'expansion de requête de recherche

Le choix d'une méthode appropriée pour l'expansion d'une requête peut avoir un rôle important dans le bon fonctionnement d'un système comme celui que nous proposons. Une requête bien étendue peut mieux cerner les besoins de l'utilisateur. Le but principal de l'expansion de requête est de lui ajouter quelques termes additionnels de telle manière que les résultats soient plus en accord avec les vraies intentions de l'utilisateur. Dans ce chapitre nous présentons quelques approches utilisées pour l'expansion des requêtes.

Selon (*LITA VIAD L. et al. 2004*), il y a six différentes méthodes d'expansion de requête d'utilisateur qui varient selon le type de lexique employé pour la sélection des termes additionnels. Ces méthodes sont les suivantes :

- **Emploi de synonymes, d'hyperonymes, et d'hyponymes :** Dans cette méthode, les termes de la requête sont étendus en employant le lexique de WordNet[27] pour des synonymes, des hyperonymes, et des hyponymes[28]. La méthode est employée notamment par (*MOLDOVAN D. et al. 2001*). Cette

[27] Un lexique sémantique disponible à http://wordnet.princeton.edu/

[28] L'**hyperonymie** est la relation sémantique hiérarchique d'un lexème à un autre selon laquelle l'extension du premier terme, plus général, engloble l'extension du second, plus spécifique. Le premier terme est dit **hyperonyme** de l'autre, ou superordonné par rapport à l'autre. C'est le contraire de l'hyponymie. [Wikipedia]

méthode est peu adéquate pour répondre aux questions d'une recherche informationnelle, dans laquelle on retrouve souvent des noms de personnes, des entités nommées[29] (par exemple Paris, IBM, etc.), ou des expressions plus compliquées comme des expressions idiomatiques, des proverbes, etc. On ne retrouve généralement pas ces éléments dans WordNet. D'ailleurs, le lexique de WordNet est limité à l'ensemble des mots d'un dictionnaire.

- **Emploi de la racine des termes de la requête:** Il s'agit de trouver, pour chaque terme de la requête, les mots qui ont la même racine. Ce type de lexique ne peut évidemment pas être employé pour les recherches universelles pour les différents sujets (comme les noms tels que "Paris" et les abréviations telles que "IBM"). En d'autres mots, elle n'est utilisable que pour les recherches basées sur l'aspect linguistique des textes des documents. En plus, comme la méthode ci-dessus, cette méthode ne peut pas répondre aux questions qui concernent les entités nommées.
- **Emploi d'une association générale entre des mots:** Cette méthode emploie une base de données de termes qui contient quelques statistiques sur la probabilité d'association entre deux mots. Par exemple la probabilité d'association des deux mots "java" et "platform" semble être plus élevée que celle de "java" et "apple". La base de données "South Florida Word Association database" (*NELSON D. L. et al. 1998*) en est un exemple. Cette base de données

[29] Une entité nommée est un objet qui comporte un nom choisi par l'humain. Parfois les règles grammaticales ne sont pas respectées pour ce choix du nom.

contient 5019 mots avec leurs probabilités d'association avec 72176 autres mots. Le problème principal de cette méthode est qu'elle requiert une base de données complète de tous les mots d'une langue.

- **Emploi de statistique sur la co-occurrence des termes:** Cette méthode emploie un ensemble de termes qui se retrouvent dans un corpus de documents pré-évalués, qui sont tous liés à un sujet particulier et qui sont intéressants pour l'utilisateur. Le système trouve la statistique sur l'existence et la co-occurence des mots dans les documents qui sont intéressants pour lui. Le défaut principal de cette méthode est qu'il faut beaucoup de temps pour construire un bon corpus de documents et les évaluer par chacun des utilisateurs pour chaque sujet de recherche.
- **Emploi de lissage en arrière-plan[30]:** Cette méthode est plutôt utilisée dans les approches comme *relevance feedback.* Elle emploie une probabilité de l'existence de la relation entre n'importe quel mot simple et tous les autres mots dans un contexte. Comme l'approche précédente, cette approche a besoin de beaucoup de documents déjà évalués par l'utilisateur sur un sujet donné pour que le système soit capable de rassembler une base de donnée robuste contenant les probabilités d'existence des mots dans le contexte de recherche. Considérant que l'ensemble de mots existants sous chacune de leurs formes grammaticales dans une langue particulière a au moins la taille du nombre de mots d'un dictionnaire, la création de cet ensemble de probabilités des relations entre les

[30] Background smoothing

mots peut être très coûteuse en temps puisque cela nécessite de créer un graphe complet dans lequel, pour chaque nœud, on a besoin de centaines d'évaluations dans le corpus de documents utilisés.

- **Emploi de statistique sur la co-occurrence des termes dans un corpus universel et volumineux sur le Web:** Cette méthode est basée sur les descriptions qu'on retrouve dans une encyclopédie, comme Wikipedia, un corpus d'environ 900 000 articles anglais de Wikipedia (en date de janvier 2006). Il y a plusieurs méthodes pour trouver des termes appropriés de requête dans les pages de définition d'une encyclopédie. Par exemple, on peut employer les hyperliens de chaque page de description de Wikipedia. Le problème principal de cette méthode est le bruit causé par les termes qui peuvent être trouvés dans toutes les pages de description de Wikipedia. Notre travail est basé sur cette approche et nous la verrons plus en détail dans les prochains chapitres.

CHAPITRE 4 : Définition du problème étudié et cadre du projet

Dans la première section de ce chapitre, nous présentons le but principal du projet, les motivations et les besoins d'un tel système, et ce qui en fait un système de recherche adaptatif permettant d'apprendre les intérêts de l'utilisateur par un algorithme de renforcement. Ensuite nous présentons la méthode d'expansion de la requête adoptée pour ce projet. Enfin, nous présentons les caractéristiques d'un prototype de l'architecture proposée.

4.1 Méthodologie et approche utilisées

Avant de voir les caractéristiques du projet, nous verrons d'abord la structure de la recherche sur Internet envisagée par (*ANTONIOU G. et al. 2004*) et nous la comparerons à la structure actuelle de recherche.

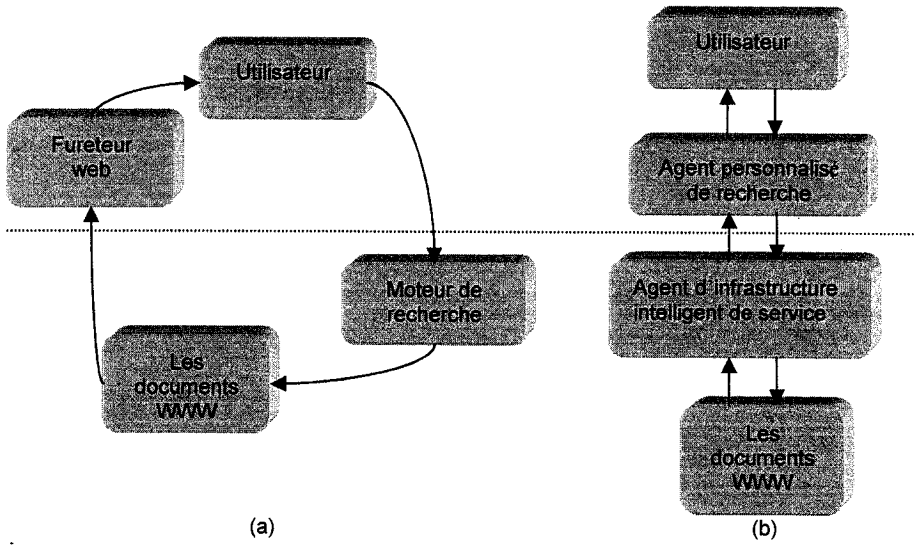

Figure 3 : (a) La structure actuelle de recherche d'Internet (b) la structure de recherche d'Internet envisagée par (ANTONIOU G. et al 2004)

La figure 3 montre deux approches différentes de recherche. Le schéma (a) montre la structure traditionnelle et celui de droite (b) montre la structure désirée pour la recherche sur Internet. La ligne pointillée montre les frontières entre les interfaces d'application du côté de l'utilisateur et les moteurs universels de recherche sur Internet.

Dans cette figure, la structure traditionnelle est principalement basée sur des itérations de recherche : l'utilisateur envoie sa requête de recherche vers le moteur de recherche, le moteur de recherche analyse la requête et renvoie les documents pertinents à son sujet recherché. Dans cette approche, le fureteur joue le rôle d'une interface qui aide à mieux percevoir le contenu des pages retournées, soit dans un mode graphique, soit dans une mode texte. L'utilisateur parcourt alors les pages ouvertes de son fureteur, raffine sa requête et la soumet à nouveau pour que le moteur de recherche retourne de meilleurs résultats.

En considérant maintenant la seconde approche, deux idées intéressantes ressortent. Premièrement, cette structure essaie d'éliminer ou de réduire le nombre d'itérations faites par l'utilisateur dans une recherche sur Internet. Deuxièmement, elle contient deux éléments évolués de recherche : *l'agent personnalisé de recherche (pour l'utilisateur)* et *l'agent intelligent d'infrastructure de service.* L*'agent personnalisé de recherche* remplace le fureteur et raffine la requête de l'utilisateur pour la rendre plus conforme au but désiré. L*'agent intelligent d'infrastructure de service* remplacera les moteurs de recherche actuels afin de mieux répondre aux questions à l'aide des méthodes d'indexation plus sophistiquées qui peuvent être basées sur le principe du web sémantique.

Afin de mieux raffiner les requêtes de l'utilisateur, selon ses caractéristiques personnelles, *l'agent personnalisé* pourrait éventuellement utiliser des informations spécifiques de chaque utilisateur, tels que ses différents noms d'utilisateur, ses mots de passe, ses intérêts, ses qualifications, son âge, sa profession, son niveau d'éducation, etc.

Dans ce projet, nous nous intéressons plutôt à l'interface de recherche du côté de l'utilisateur (*l'agent personnalisé de recherche*). L'architecture proposée doit être capable de communiquer directement avec les moteurs de recherche universels existants tels que Google et AltaVista.

Une des raisons expliquant ce choix est que jusqu'à aujourd'hui il n'y a pas beaucoup de travail accompli dans le domaine des agents personnalisés de recherche. Par contre, il y a plusieurs moteurs de recherche universels commerciaux ou

académiques, tels que Google, qui sont bien connus par les utilisateurs d'Internet. Mais nous ne connaissons aucun agent personnalisé de recherche qui est globalement utilisé par les utilisateurs d'Internet. et qui a la même importance que les moteurs de recherche pour les utilisateurs d'Internet. L'autre raison est que en même temps que le nombre de moteurs de recherche se développe, il viendra un moment où les utilisateurs d'Internet auront beaucoup de difficulté à qualifier et choisir le moteur de recherche le plus approprié, et en conséquence ils auront besoin d'une application intermédiaire qui peut les aider à communiquer avec plusieurs moteurs de recherche pour chacune de leurs simples sessions de recherche. Une des originalités de KartOO, qui a été présenté dans le chapitre 2, est qu'il donne aux utilisateurs la possibilité de choisir plusieurs moteurs de recherche pour chaque session de recherche.

Dans ce travail, l'architecture proposée utilise une méthode basée sur l'adaptation itérative des résultats de la recherche pour mieux s'adapter aux changements dans les intérêts de l'utilisateur. En d'autres mots, il s'agit d'une architecture qui calibre les résultats de recherche avec les vrais besoins de l'utilisateur, en employant les méthodes d'expansion de la requête et en profitant d'une rétroaction de l'utilisateur à chaque itération afin d'améliorer les résultats des futures itérations de recherche. Dans ce contexte, nous notons cinq caractéristiques générales:

1. **Environnement partiellement observable:** l'agent ne connaît pas les besoins spécifiques de l'utilisateur. Par contre, à chaque itération c'est le feedback de l'utilisateur qui permet de mieux cerner ses besoins. En plus, l'agent n'a pas d'accès direct à tous les documents du web.

2. **Besoin d'exploration:** il est impossible de formuler les règles qui décrivent la relation entre les besoins de l'utilisateur, les actions effectuées par l'agent, et les documents du web. L'agent ne peut donc qu'approximer ces règles de manière implicite.

3. **Récompense retardée pour les actions:** notre agent ne peut pas savoir de manière immédiate s'il fait un bon choix d'action. Ce n'est que lorsque l'utilisateur fournit un feedback sur les résultats retournés que l'agent peut évaluer son action.

4. **Résultats d'action simples mais pas définitivement interprétables:** nous ne pouvons pas avoir une classification préalable des contenus de tous les documents en ligne afin d'estimer la classe la plus pertinente pour la recherche actuelle. Mais, il est toujours possible d'avoir une classification en fonction l'historique des résultats et selon les feedbacks obtenus de l'utilisateur sur chacun des résultats.

5. **Longue durée du processus d'apprentissage:** Les vrais besoins de l'utilisateur peuvent changer selon le temps. L'architecture doit donc apprendre de manière continue.

Selon (*MITCHEL T. M. 1997*), une des méthodes d'apprentissage appropriées pour des situations dynamiques, comme c'est le cas avec les besoins informationnels de

l'utilisateur, est la méthode d'apprentissage par renforcement, qui est basée sur l'obtention interactive d'une récompense après chaque étape d'action de recherche. Nous utilisons donc un agent de recherche basé sur l'apprentissage par le renforcement, le feedback de l'utilisateur sur les pages retournées, et un processus itératif pour ajuster les requêtes selon l'intérêt de l'utilisateur, indépendamment du type de moteur de recherche utilisé. Ce système s'appelle ARIIA - une abréviation de "Adaptive Reinforcement Iteration-based Internet Agent" - et sera illustré en détail dans le prochain chapitre.

4.2 Le cadre de conception

Pour la conception et l'implémentation d'ARIIA, nous nous sommes efforcé de respecter les caractéristiques d'utilisateur présentées dans la section 1.3. En plus des recherches à longue durée qui couvrent naturellement les intérêts plus stables de l'utilisateur, ARIIA doit aussi s'appliquer aux recherches de courte durée et s'adapter aux variations dans les intérêts de l'utilisateur. En d'autres mots, après plusieurs itérations de recherche, ARIIA doit éliminer les résultats pour lesquels l'utilisateur a perdu de l'intérêt, et retourner de nouveaux résultats qui sont plus appropriés aux nouveaux intérêts de l'utilisateur sur un sujet.

Nous croyons important de prendre en compte le feedback explicite de l'utilisateur pour évaluer les résultats retournés par ARIIA. Le calibrage des résultats sera donc basé sur une simple évaluation de l'utilisateur pour chacune des pages web retournées.

Idéalement, il faudrait plutôt utiliser un feedback implicite, mais il s'agit là d'un domaine de recherche qui est encore peu avancé.

Nous verrons plus en détails dans le chapitre 5, la méthode d'expansion de requête utilisée dans notre architecture. Nous avons choisi une approche basée sur le lexique de Wikipedia, pour plusieurs raisons. Actuellement le lexique de Wikipedia évolue beaucoup plus rapidement que les autres lexiques de la même catégorie. Deuxièmement, l'évolution de Wikipedia est en plusieurs langues en même temps. Troisièmement, par rapport aux ressources utilisables d'expansion de requête mentionnées dans le chapitre 3, Wikipedia semble être plus approprié pour notre architecture, car c'est une encyclopédie très générale qui englobe beaucoup de domaines de recherche et en même temps est disponible gratuitement sur le web.

Finalement, le facteur du temps de réponse sera ignoré dans notre évaluation puisque nous nous sommes plutôt intéressé à la logique de fonctionnement de l'architecture qu'à sa performance temporelle. Cependant, la durée d'exécution d'une itération de recherche par ARIIA ne dépasse pas une trentaine de secondes.

CHAPITRE 5 : Description de ARIIA

5.1 Algorithme de fonctionnement de base de ARIIA

Dans cette section nous présentons les composants d'ARIIA. L'architecture d'ARIIA contient cinq composants principaux et un composant optionnel (qui n'est pas illustré dans la figure ci-dessous). Chacun d'eux communique avec les autres composants à l'aide d'un ou plusieurs messages. En d'autres mots, chaque composant est responsable de la transformation du message reçu en un autre type de message qui doit être convenable pour le prochain composant dans l'ordre de communication. La figure 4 illustre l'architecture d'ARIIA.

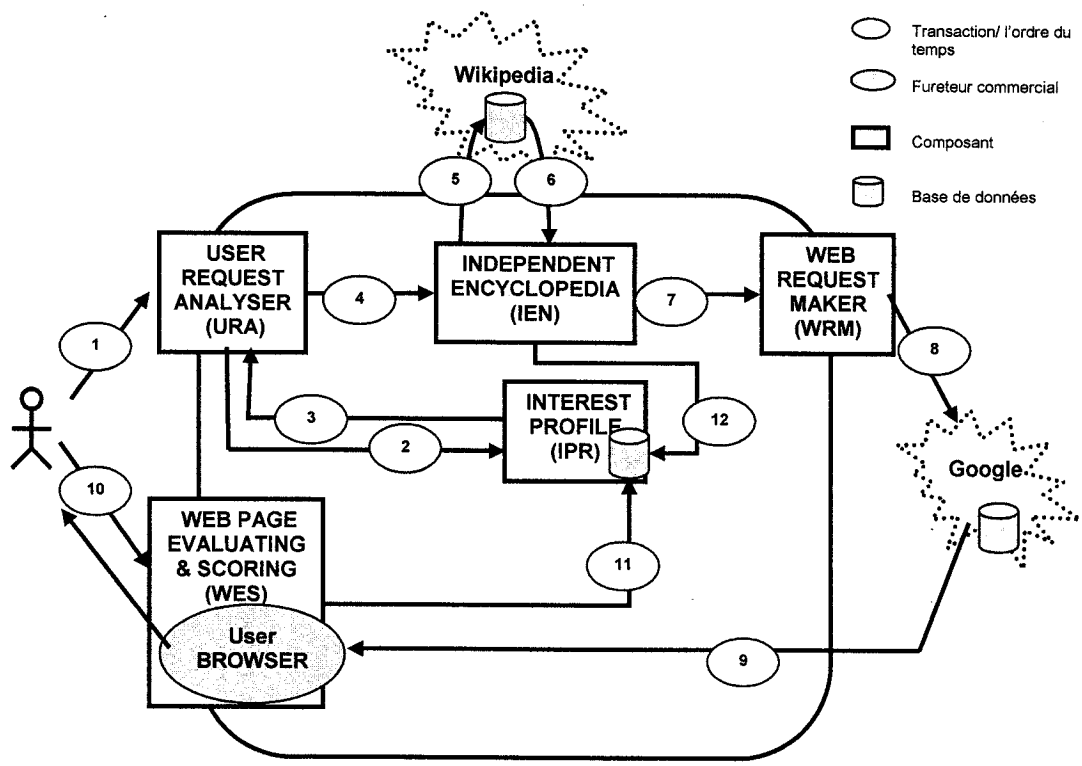

Figure 4 : Les composants d'ARIIA

Dans cette figure les ovales numérotés sur chaque flèche dénotent le nom ainsi que l'ordre chronologique de transfert des messages. Le tableau 1 montre les noms et les structures des messages échangés entre chacun des composants. Dans les sous-sections suivantes, tous ces éléments sont décrits dans le cadre du fonctionnement global et de l'ordre chronologique des messages échangés. Le tableau 1 montre la liste des messages.

Tableau 1: Descriptions des messages

Ordre du Transaction/ temps	Nom du Message	Description
1	***User search query*** *(USQ)*	$USQ = \{t_1, t_2, ..., t_n\}$ t_i est un terme de la requête
2	***User Data Base query*** *(UDBQ)*	Soit $USQ^M = \{ t_i' \mid t_i \in USQ$ et t_i' est la forme majuscule de $t_i \}$, $UDBQ = \wp(USQ) \cup \wp(USQ^M)$
3	***Checked word and its extension*** *(CWE)*	*Soit* $EXT(t) = \{ (e_i, s_i) \mid e_i$ est un terme constituant une extension de t et $s_i = score(e_i)\}$, $CWE = \{(t_i, e_i, s_i) \mid t_i \in UDBQ$ et $(e_i, s_i) \in EXT(t_i)\}$

4	***User search set*** *(USS)*	$USS \subseteq CWE$ La cardinalité de *USS* est limité à n dont la valeur est paramétrable. Si l'utilisateur a demandé l'utilisation de scores négatifs : $USS = \{ (t_i, e_i, s_i) \mid (t_i, e_i, s_i) \in CWE$ et s_i est parmi les $\frac{n}{2}$ plus élevés $\} \cup \{ (t_i, e_i, s_i) \mid (t_i, e_i, s_i) \in CWE$ et s_i est parmi les $\frac{n}{2}$ moins élevés $\}$ Sinon : $USS = \{ (t_i, e_i, s_i) \mid (t_i, e_i, s_i)$ *CWE* et s_i est parmi les n plus élevés $\}$
5	***Wikipedia search query*** *(WSQ)*	$T = \{ t_i \mid (t_i, e_i, s_i) \in USS \}$ $E = \{ e_i \mid (t_i, e_i, s_i) \in USS \}$ $WSQ \subseteq T \cup E$, *tel que WSQ soit choisi aléatoirement*
6	***Wikipedia HTML pages*** *(WHP)*	$WHP = \{ P_i \mid P_i$ est la page retournée par Wikipedia pour le terme $t_i \}$
7	***Complete Search set*** *(CSS)*	$WSQ' \subseteq WSQ$, $WSQ' = \{ t_i \mid t_i \in WSQ, t_i$ est sélectionné aléatoirement$\}$ $W_{t_k} = \{ t_i \mid t_i$ est un terme associé à un hyperlien dans une

		page $P_k \in WHP\}$ $W_{t_k}' \subseteq W_{t_k}$, *tel que WSQ soit choisi aléatoirement* $CSS = USQ \cup WSQ' \cup W'_{t_1} \cup .... \cup W'_{t_n}$
8	***Internet search engine query*** *(ISEQ)*	*ISEQ* est une sequence: $X_1 \ldots X_n \quad Y_1 \quad OR \quad Y_2 \; OR \ldots Y_n$ *où* $X_i \in USQ$ *et* $Y_i \in CSS / USQ$
9	***Internet search engine answer*** *(SEA)*	$SEA = \{P_i \mid P_i$ est une page retournée de moteur de recherche$\}$
11	***User Vote*** *(UV)*	Soit $V(P_i)$ le score attribué par l'utilisateur à la page P_i, $UV = \{(P_i, S_i) \mid S_i = V(P_i)$ et $P_i \in SEA\}$
12	***Page Score*** *(PS)*	Soit $F_{t_i}(P_j)$ la fréquence du terme t_i dans la page P_j, $R_{t_i} = \sum_{p_j \in SEA} F_{t_i}(P_j) \quad V(P_j)$ $PS = \{(t_i, R_{t_i}) \mid t_i \in CSS\}$ Chaque entrée $EXT(t_i)$ de IPR est transformée en : (e_i', s_i')

		où pour chaque $t_i \in USQ$ chaque item (e_i', s_i') de $EXT(t_i)$ est transformé $(e, s+R_{e_i})$ si $(e, R_{e_i}) \in PS$ et pour chaque paire $(e, R_{e_i}) \in PS$ telle que e_i ne se trouve pas déjà dans $EXT(t_i)$ on ajoute à $EXT(t_i)$

5.2 Structure détaillée des composants

5.2.1 Unité *User Request Analyzer (URA)*

La figure ci-dessous montre de manière simple la coopération entre les sous-composants du URA. La première tâche du URA est de recevoir la requête initiale de l'utilisateur qui s'appelle User Search Query (USQ) et de chercher dans son IPR les extensions existantes pour chaque terme du message. D'autre part, le URA transforme cette requête en un message User Data Base Query (UDBQ) et l'envoie au IPR. Cette opération consiste à créer tous les sous-ensembles de combinaisons possibles des termes de User Search Query pour vérifier leur existence dans le profil d'utilisateur. Par exemple à partir de la requête {"student", "professor", "book"}, ce composant créera l'ensemble de termes {{"student"}, {"professor"}, {"book"}, {"professor student"}, {"student professor"}, {"book student"}, {"book professor"}, {"student book"},

{"professor book"}, {"book professor student"}, {"book student professor"}, {"student book professor", etc.} qui sera utilisé pour la recherche dans le IPR. Le IPR est présenté plus en détails à la section 5.2.5; Essentiellement, le IPR recherche chaque terme de cet ensemble dans la base de données du profil d'utilisateur et renvoie un message indiquant pour chacun s'il existe, ainsi que son score, tel que calculé préalablement par WES. En plus, le IPR doit trouver et retourner toutes les extensions de termes dont les votes assignés par l'utilisateur sont élevés. Le URA élimine ensuite les termes répétés de l'ensemble retourné, choisit les termes dont le score est le plus élevé et les envoie au IEN. Mentionnons que s'il n'y a aucune extension pré-existante dans le IPR, le URA enverra simplement le message initial User Database Query au IEN.

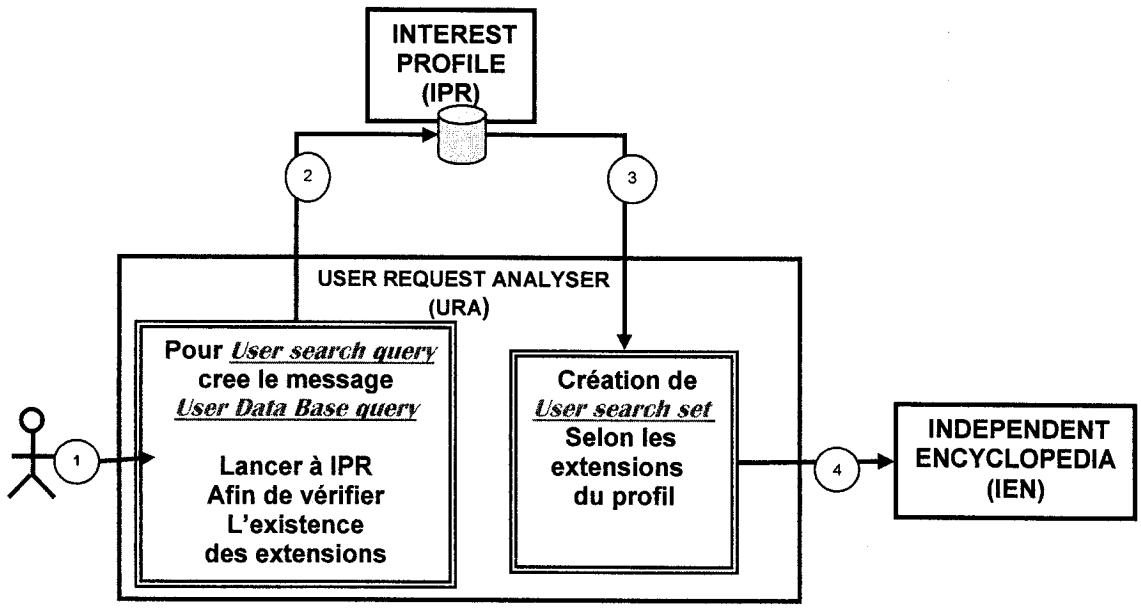

Figure 5 : Unité User Request Analyzer (URA)

5.2.2 Unité *Independent ENcyclopedia (IEN)*

Ce composant contient deux sous-composants principaux : il reçoit du URA l'ensemble de termes du message User Search Set (USS) et les transforme en un ensemble de termes composant le message Complete Search Set (CSS) qui doit être envoyé au WRM. Par exemple, ayant reçu le message USS {"student", "book", "student book", "book student"}[31], cette unité créera le message Wikipedia Search Query (WSQ) et consultera le Wikipedia pour tous les termes de l'ensemble {"student", "book", "student_book", "book_student", "student-book", "book-student", "book'student", "student'book"}[32], ainsi que leurs formes en majuscule {"STUDENT", "BOOK", "STUDENT_BOOK", "BOOK_STUDENT", "STUDENT-BOOK", "BOOK-STUDENT", "BOOK'STUDENT", "STUDENT'BOOK" pour tenir compte des abréviations aussi. Le message WSQ est une URL qui est dynamiquement créée selon la langue de l'utilisateur. On obtient en retour un ensemble de pages possiblement existantes dans Wikipedia (Wikipedia HTML Pages). Le IEN cherche alors les termes reliés aux hyperliens dans les pages retournées et enlève certains de ces termes, comme ceux qui sont répétitifs, ou qui contiennent des nombres ou des caractères spéciaux comme le tiret, l'espace, etc. Finalement, le IEN crée le message Complete Search Set (CSS) et l'envoie au WRM. Un aspect important est la limite du nombre de termes que nous pouvons envoyer à Google ou AltaVista. Cette limite force le IEN à sélectionner

---

[31] Pour simplifier le message nous avons éliminé les extensions et les scores.

[32] Avoir cette forme avec un trait d'union est parfois nécessaire pour qu'une combinaison des termes se trouve dans le Wikipedia. Par exemple le "Club Med" se trouve en forme de "Club_med" en Wikipedia.

un nombre très limité de termes parmi les centaines d'extensions trouvées dans Wikipedia. L'interface de l'ARIIA doit donc avoir un dispositif qui permet de déterminer cette limite en choisissant le ratio entre les termes préenregistrés de profil d'utilisateur et des nouveaux termes dérivés de Wikipedia.

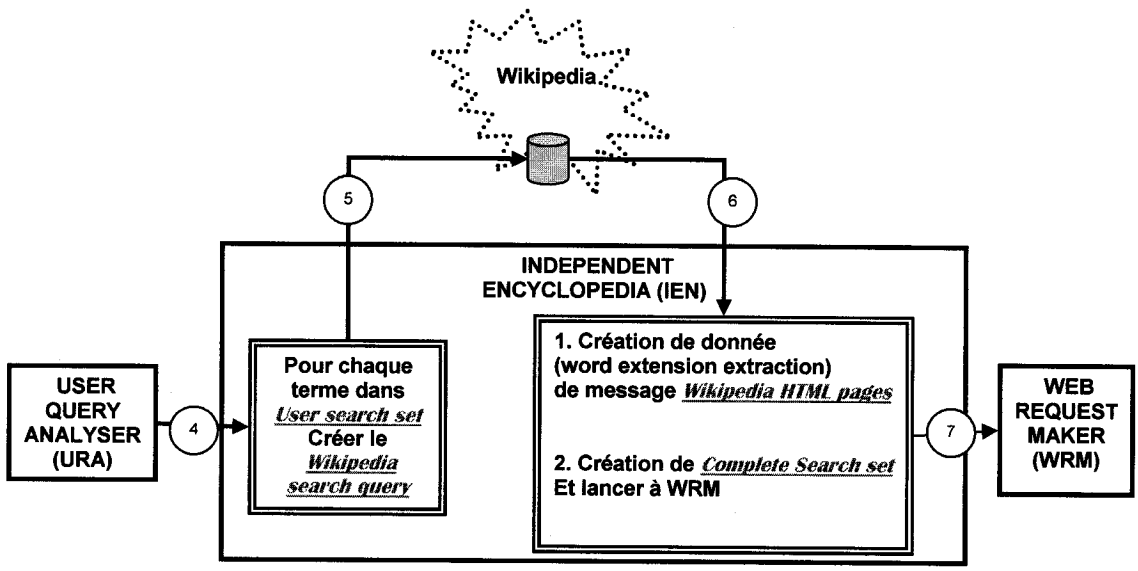

Figure 6 : INDEPENDENT ENCYCLOPEDIA (IEN)

5.2.3 Unité *Web Request Maker (WRM)*

Ce composant reçoit le message Complete Search Set (CSS) de l'IEN et crée une requête qui s'appelle Internet Search Engine Query (ISEQ) et qui doit être envoyée vers le moteur de recherche choisi, Google ou AltaVista. Google accepte environ 32 termes par requête alors que AltaVista en accepte plus de 60. Mentionnons que dans la requête ISEQ qui doit être envoyée à Google, les termes qui sont entrés par l'utilisateur ont un statut *obligatoire,* alors que leurs extensions stockées dans le IPR, ou les nouvelles extensions trouvées dans Wikipedia, ont le statut facultatif. Par exemple, considérons

que l'utilisateur entre le terme "student" dans son interface de recherche et qu'il y a quelques extensions (qui sont déjà positivement évaluées) dans le IPR pour ce terme, comme {"university", "professor"}, ainsi que quelques extensions nouvellement dérivées de Wikipedia comme {"book", "library"}. Dans cet exemple, le message Internet Search Engine Query aura la forme suivante:

student book OR library OR university OR teacher

Ici, l'application d'une autre heuristique est aussi possible: les termes négativement évalués dans le IPR peuvent apparaître dans la requête précédés de l'opérateur *NOT*. Par exemple, si les termes "primary school" et "education" sont des termes dont le poids est négatif dans le IPR de l'utilisateur, la requête ci-dessus aura la forme suivante:

student book OR library OR university OR teacher NOT "primary school" NOT education[33]

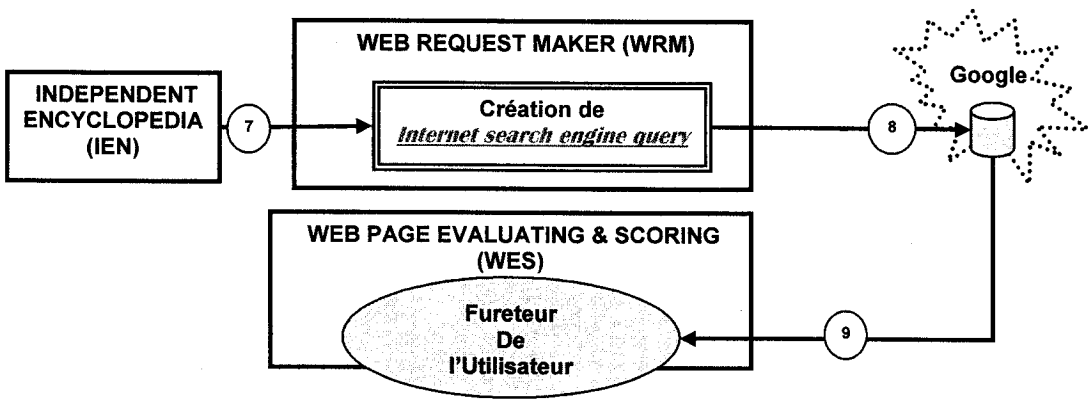

Figure 7 : Unité WEB REQUEST MAKER (WRM)

[33] Opérande NOT peut être illustré par le signe "-" dans la requête

5.2.4 Unité *Web page Evaluating and Scoring(WES)*

Généralement, ce composant est directement en communication avec le moteur de recherche d'Internet. Ce composant obtient directement les pages retournées de Google ou AltaVista (le message Internet Search Engine Answer (SEA)), les montre simultanément à l'utilisateur dans son fureteur Web, et produit un questionnaire simple qui sera placé en haut ou en bas du fureteur Web de l'utilisateur. Ensuite, selon les points accordés par l'utilisateur à la page ouverte, ce composant produit un message Page Score qui sera envoyé à l'IPR pour être enregistré pour les prochaines itérations de recherche. La figure 8 montre la structure de cette unité ainsi que ses sous-composants. Au bas de cette figure, l'interface de cette unité est illustrée.

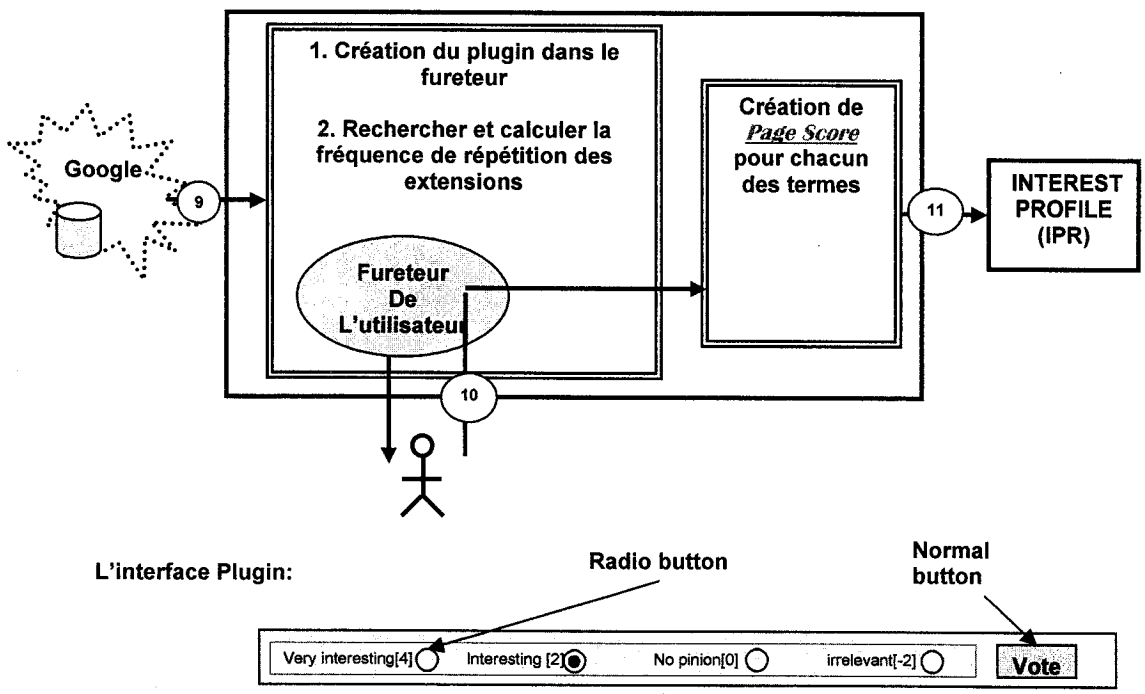

Figure 8 : (en haut) WEB PAGE EVALUATING & SCORING (WES) (En bas) Interface de vote d'utilisateur

Il y a quatre différents niveaux de pertinence que l'utilisateur peut choisir dans cette interface:

- **Très intéressant** (dont la valeur associée est +4): pour les pages qui contiennent tous les aspects du sujet désiré par l'utilisateur.
- **Intéressant** (+2) : Pour les pages qui couvrent partiellement le sujet désiré par l'utilisateur.
- **Aucune opinion** (0) : pour les pages qui sont dans le sujet désiré par l'utilisateur, mais dont le contenu ne répond pas du tout à ses questions. Par exemple, le cas de la requête "agouti", qui retourne une page qui parle de cet animal, mais qui ne répond pas à la question "quel genre d'animal est un agouti?".
- **Non pertinent** (-2) : pour les pages dont le sujet est totalement différent de celui désiré par l'utilisateur. Pour l'exemple ci-dessus, une page avec le titre "agouti" qui ne contient aucun matériel pertinent relié au sujet "animal agouti" et qui parle de musique ou d'un centre d'entretien ou une page publicitaire reçoit une page non pertinente.

Le processus d'évaluation commence quand l'utilisateur ouvre une page HTML dans son fureteur. Le WES recherchera tous les termes du message CSS dans la page et calculera la fréquence d'apparition de chacun d'eux dans la page HTML. Puis il multipliera chaque fréquence par le coefficient correspondant à l'évaluation par

utilisateur, et les enverra (le nom de message est Page Score (PS)) à l'IPR afin d'être enregistré pour les futures recherches.

5.2.5 Unité *Interest PRofile (IPR)*

Cette unité contient la base de données du profil d'intérêt de l'utilisateur et effectue toutes les opérations de recherche, de stockage, de mise à jour, et de suppression de termes. Le détail de sa base de données est montré à la figure 9.

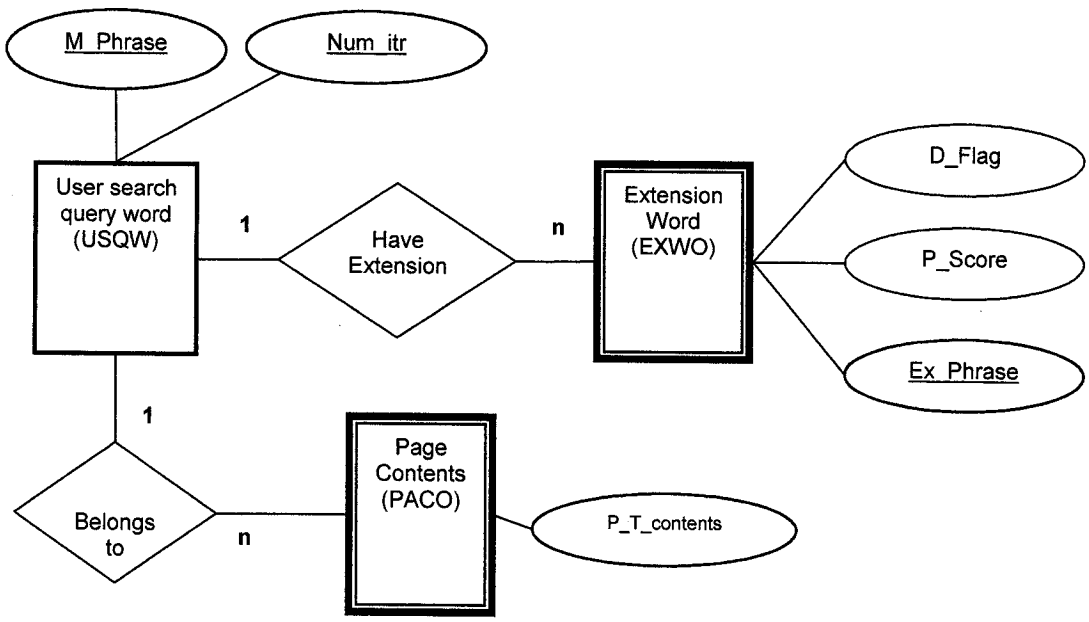

Figure 9 : Le schéma logique de Base de donnée de profile d'utilisateur

Dans cette base de données, il y a trois tables; la table **User Search Query Word** (USQW), la table **EXtension Word** (EXWO), et la table **PAge COntents** (PACO). La première est employée pour stocker les termes initiaux de l'utilisateur (l'attribut M_phrase) et le nombre d'itérations précédentes de recherche faites pour ce terme

(Num_Itr). La deuxième table est employée pour stocker les extensions obtenues de Wikipedia pour la requête initiale (l'attribut Ex_phrase), ainsi que les scores[34] de chacune des extensions par chaque terme (l'attribut P_score), et un attribut qui est employé pour supprimer un terme d'extension si son score devient nul après quelques itérations (l'attribut D_Flag). Cet attribut aide ARIIA à nettoyer le profil d'utilisateur des termes qui ne sont pas importants pour les prochaines expansions de requête. Ces termes sont soit les termes dont le score diminue après quelques itérations de recherche, soit les termes dérivés de Wikipedia qui n'apparaissent pas dans les pages webs évaluées par l'utilisateur. Finalement, la troisième table conserve les sources HTML des pages ouvertes par l'utilisateur pour chaque requête.

Tableau 2 : Format du message Page_score

Le texte de page HTML	**La Terme de message User Data Base Query**	**Vérification D'existence**	**Ensemble des extensions et leurs points polynomial**
<html> … … </html>	Terme 1	Oui/Non	(Extension 1, son score)…(extension n, son score)
	…	…	…
	Terme n	Oui/Non	(Extension 1, son score)…(extension n, son score)

Pour chaque terme de User Data Base Query, l'IPR renverra une variable qui indique si le terme existe ou non. Si le terme existe, le message contient un ensemble de ses extensions et leurs scores. Dans la première version d'ARIIA, nous n'avons pas

[34] Referez au calcule de message Page Score dans le tableau 1

employé les points retournés pour chaque extension en Checked Word and its Extension. Dans le futur, une possibilité est de définir un seuil d'ajustement pour mettre encore plus de termes obligatoires et moins de termes facultatifs dans le message Internet Search Engine Query qui est créée par WRM. En d'autres mots, les termes qui ont des scores relativement élevés peuvent avoir un statut booléen obligatoire dans le message Internet Search Engine Query. La figure 10 illustre la phase de vérification d'existence des termes pour IPR.

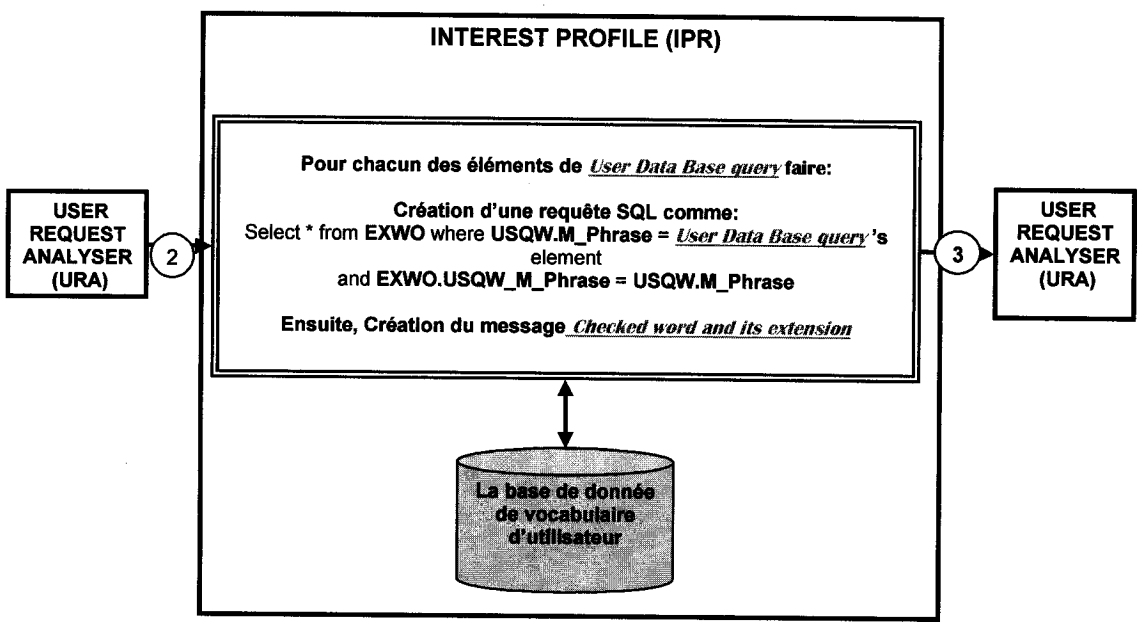

Figure 10 : L'unité Interest PRofile (IPR) : la phase de vérification des extensions

La deuxième tâche de l'IPR est de calculer les scores totaux des extensions et de les enregistrer dans la base de données. La manière la plus simple de faire ceci est de premièrement consulter la base de données afin de trouver chacun des termes contenus

dans le message Page Score. S'ils existent, l'IPR ajoutera leurs points calculés (soit positif soit négatif) à leurs points déjà mémorisés. Et si un terme n'existe pas, l'IPR l'ajoutera à la base de données et ajoutera aussi son score.

Ainsi, nous pouvons nous assurer que lorsque l'utilisateur perd son intérêt sur un domaine d'un sujet recherché, le score des termes qui y sont associés sera diminué par ses évaluations négatives, ou par le fait que leur score n'augmente pas et sera éventuellement surpassé par d'autres termes.

Parfois le score d'un terme demeure nul après plusieurs recherches sur le même sujet. Dans ce cas, l'IPR sait que ce terme ne peut plus calibrer la requête de recherche. L'IPR mettra donc automatiquement le D_flag de ce terme à "oui" afin qu'il soit éliminé de la base de données à l'avenir. Finalement le IPR enregistre le contenu des pages dans le tableau PACO. La figure ci-dessous montre la phase d'enregistrement de termes de l'IPR.

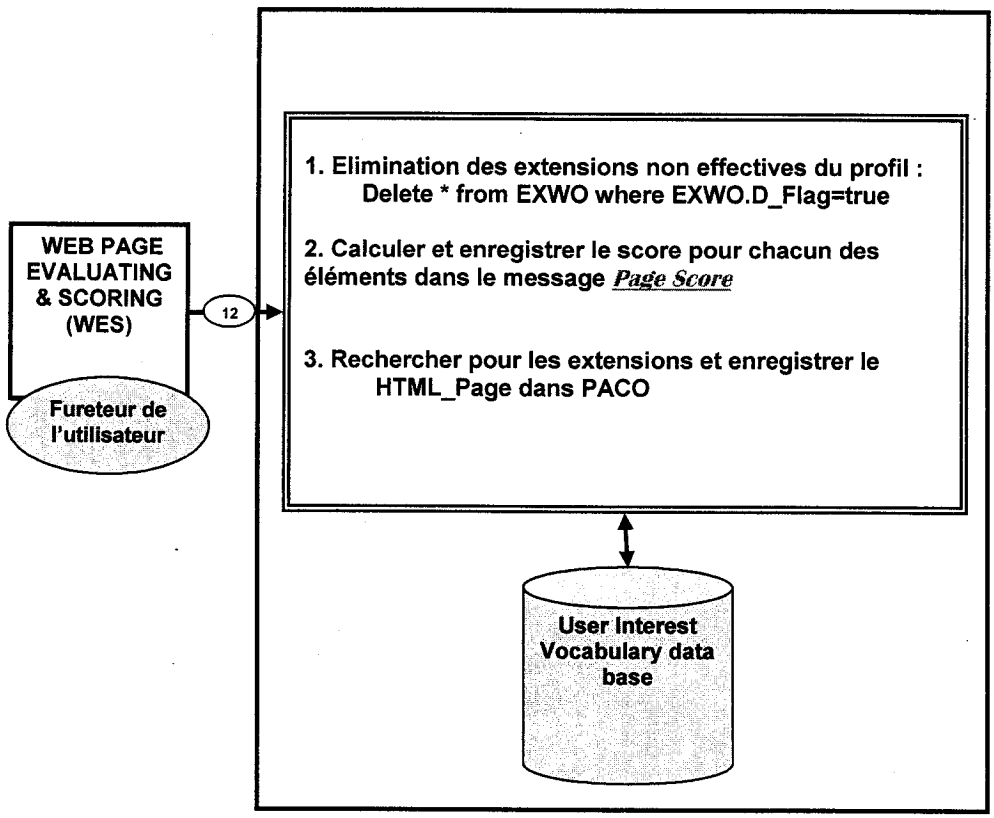

Figure 11 : L'unité Interest Profile (IPR) : la phase d'enregistrement des termes

5.3 Implémentation

Notre prototype d'ARIIA a été conçu de manière à ce que son fonctionnement puisse être contrôlé par quelques paramètres réglables. Ces paramètres sont :

- **Choix du moteur de recherche:** c'est un paramètre qui donne la possibilité de choisir soit Google soit AltaVista comme moteur de recherche.
- **Inclusion de termes booléens négatifs dans la requête de recherche:** En choisissant ce paramètre, les termes qui sont négativement évalués par l'utilisateur seront précédés de l'opérateur NOT dans la requête envoyée au

moteur de recherche (le message Internet Search Engine Query). Nous croyons qu'il vaut mieux éviter ce dispositif, puisque le nombre maximum de mots-clés dans la requête de Google est déjà trop limité. En ajoutant des termes négatifs, on laisse moins de place aux termes positivement évalués. Pour nos tests ce paramètre est totalement inactif.

- **Création d'une requête contenant obligatoirement les termes initiaux:** ce dispositif force le WRM à créer le message Internet Search Engine Query d'une façon à contenir les termes du message User Search Query sous forme booléenne obligatoire. En d'autres mots, l'activation de ce dispositif force une recherche focalisée sur les termes fournis par l'utilisateur plutôt que leurs extensions. Ce paramètre est actif dans nos tests.
- **Élimination des termes d'extension contenant des caractères spéciaux:** l'activation de ce dispositif force l'unité IEN à éliminer toutes les extensions de Wikipedia qui sont des *nombres*, des *dates*, qui contiennent *une combinaison de chiffres et des lettres de l'alphabet*. Ce dispositif est activé dans nos tests.
- **Choix du ratio entre les extensions enregistrées dans le profil et les extensions trouvables dans le Wikipedia:** Après plusieurs itérations de recherche sur un sujet particulier, employer ce dispositif semble être nécessaire puisque le nombre d'extensions qui sont pondérées positivement dans le IPR sera tellement élevé qu'il empêchera d'ARIIA d'ajouter de nouveaux termes d'extension provenant de Wikipedia à la requête de recherche qui doit être soumise à Google. Donc il doit y avoir un équilibre entre le nombre de termes

dans l'extension provenant de l'unité IPR et les termes trouvés dans Wikipedia. Cet équilibre nous assure un dynamisme qui donne une chance aux nouvelles extensions trouvées dans Wikipedia. Pour le premier test d'ARIIA, ce rapport est ajusté à 16 termes du IPR parmi les 32 termes qui sont soumis à Google. Cela signifie que dans une limite de 32 termes de requête de recherche sur Google, il y a environ 16 termes qui constituent une extension de la requête originale dont les termes sont ceux qui ont les meilleurs scores lors des itérations précédentes. D'autre part, selon la longueur initiale de la requête (le message User_Search_Query), qui contient au moins un terme, le nombre de termes provenant de Wikipedia dans l'extensions varie entre 1 et 15.

CHAPITRE 6 : Stratégies de validation et résultats obtenus

Pour évaluer ARIIA nous avons utilisé les critères suivants:

1. **Jugement d'utilisateurs** sur deux ensembles de résultats : l*es résultats retournés par ARIIA pour une requête de recherche* et *les résultats retournés par Google ou AltaVista pour la même requête*. Ce test a la forme d'un sondage qui nous indique le niveau du satisfaction de l'utilisateur quant aux résultats obtenus avec ARIIA.
2. **Performance de ARIIA** pour chacun des différents types de recherche (navigationnelle, exploratoire, informationnelle). Ce critère vise à connaître les points forts et les faiblesses de ARIIA pour chacun des types de recherche.
3. **Vitesse moyenne de convergence de résultats de recherche d'ARIIA** avec les résultats désirés pour chaque session et requête de recherche prédéfinis. Il s'agit du nombre d'itérations nécessaires à ARIIA pour arriver à une page Web désirée dans ses 10 premiers résultats retournés.
4. **Niveau d'adaptabilité d'ARIIA** par rapport aux variations de l'intérêt de l'utilisateur sur un sujet de recherche. Il s'agit de tester la détection de décalage, de perte, ou d'apparition de nouvel intérêt sur un sujet qui est toujours lancé par la même requête.

Pour respecter les délais de projet, nous avons dû limiter notre effort de validation et nous nous sommes donc limité aux deux premiers critères pour la recherche

informationnelle et exploratoire mais pas pour la recherche navigationnelle. Ce dernier type de recherche est moins pertinent pour un système comme ARIIA. En plus, vu la nature itérative de la recherche web par ARIIA et le temps qui est nécessaire pour examiner les résultats de chaque itération, nous avons divisé l'évaluation en deux niveaux: une évaluation réalisée par trois utilisateurs différents pour la première itération de recherche, et un test *supplémentaire* qui est la continuation du test précédent pour les deuxième et troisième itérations de recherche par ARIIA. Les résultats de ce second test sont basés sur un seul utilisateur web avec un ensemble de requêtes prédéfinies. La raison pour laquelle nous avons choisi un seul utilisateur pour tester la deuxième et la troisième itération est que le premier test s'avère très long et il aurait été difficile de poursuivre l'expérience avec trois personnes.

6.1 Choix des exemples de test

Pour notre évaluation, nous avons choisi un ensemble de cinq thèmes des questions de la conférence TREC2004 (*VOORHEES E.M. 2004*) qui fournit une base solide et reconnue de comparaison pour des tâches de recherche d'information. Chacun de ces titres généraux contient de 4 à 5 sous-questions (23 sous-questions au total). Toutes ces sous-questions sont employées pour notre évaluation de recherche informationnelle. Le tableau suivant illustre tous les exemples choisis. Chacun des thèmes généraux contient un sous-thème "Other" qui est utilisé pour notre recherche exploratoire. La sous-question "Other" est associée aux pages web retournées par ARIIA qui contiennent des

informations autres que celles concernées par les sous-questions de chaque thème. Par exemple, pour le thème "Agouti", si l'utilisateur trouve des informations autre que les informations qui répondent aux 3 sous questions détaillées " In what countries are they found?"," What kind of animal is an agouti?", et " What is their average life span?", le score donné pour la sous-question "Other" sera positif. L'utilisateur donne le score en fonction de son évaluation subjective de la quantité d'information supplémentaire dans la page.

Tableau 3 : L'ensemble des exemples de TREC2004 choisis pour le test

Thème de la question	Sous question détaillée
agouti	In what countries are they found?
	What kind of animal is an agouti?
	What is their average life span?
	Other
Black Panthers	Who have been members of the organization?
	When was it founded?
	Where was it founded?
	Who founded the Black Panthers organization?
	Other
Cassini space probe	How much did it cost to build?
	When was Cassini launched?
	What is its destination?
	What planets will it pass?
	Other
Hale Bopp comet	How often does it approach the earth?
	In what countries was the comet visible on its last return?
	When was the comet discovered?
	Other
Horus	Horus is the god of what?
	Who was his mother?
	What country is he associated with?
	Who was his father?
	Other

Dans notre expérimentation, toute les recherches ont été effectuées par une requête ne contenant que le thème de la question. Par exemple pour trouver les réponses aux sous-questions " In what countries are they found?"," What kind of animal is an agouti?", et " What is their average life span?" pour le thème "Agouti", nous n'avons utilisé que le terme "Agouti" dans la requête. En effet les requêtes des conférences TREC peuvent être utilisées pour des systèmes qui traitent la langue naturelle, ce qui n'est pas notre cas. En utilisant cette stratégie, nous avons lancé la requête au moteur de recherche Google et à ARIIA.

Les sous-questions et les 10 premiers résultats retournés par Google et par la première itération de ARIIA ont été donnés à chacun des trois évaluateurs. Chaque utilisateur a dû lire chacune des pages soigneusement et déterminer si la réponse à chacune des sous-questions s'y trouvait. Comme il y avait 23 sous-questions au total, pour la première itération il y avait environ 230 évaluations faites pour les pages retournées par Google, et le même nombre pour ARIIA. Moins de 7 pourcent des évaluations ont nécessité des corrections. Par exemple, pour le titre "the black panthers", certains des utilisateurs ont positivement évalué les pages retournées par ARIIA qui parlent de "the New Black Panthers" qui correspondent à un sujet semblable mais quand même différent de ce qui est désiré. Leurs réponses ont été ajustées par l'expérimentateur et nous leur avons demandé une confirmation finale sur la justesse de la correction effectuée. Deux indicateurs ont été utilisés pour comparer les résultats:

- **Moyenne des Scores (*MS*)**: Cet indicateur compare la moyenne cumulative pour la somme de valeurs de score pour chacune des sous-questions. Il y a aussi

un facteur d'efficacité qui est calculé a partir d'un ratio entre la moyenne cumulative des scores pour le thème de question et la valeur maximum possible de moyenne cumulative pour chaque thème. Pour mieux comprendre ce calcul nous donnons un exemple. Par exemple, on considère que les résultats obtenus (soit par Google soit par ARIIA) sont illustrés dans le tableau 4 et sont les scores assignés par l'utilisateur à chacune des 10 premières pages d'une requête de recherche pour une des sous-questions:

Tableau 4 : Les votes d'utilisateur sur les 10 premières pages retournées d'ARIIA

Le score donné à la Page 1	Le score donné à la Page 2	Le score donné à la Page 3	Le score donné à la Page 4	Le score donné à la Page 5	Le score donné à la Page 6	Le score donné à la Page 7	Le score donné à la Page 8	Le score donné à la Page 9	Le score donné à la Page 10
4	2	0	4	4	2	2	-2	2	0

Comme déjà mentionné, les valeurs de votes sont les nombres entiers pairs qui varient entre -2 et +4. Pour éliminer les points négatifs, d'abord nous avons normalisé les points en leur ajoutant (+2). Alors la valeur de ***MS*** se calcule comme :

$$\underline{\boldsymbol{MS}} = \frac{\sum_{i=1}^{10}(V(P_i)+2)}{10} = (6+4+2+6+6+4+4+0+4+2) / 10 = 3{,}8$$

Et ***Efficacité de MS*** (***EMS***) pour cet exemple sera calculé comme:

$$\underline{\boldsymbol{EMS}} = \frac{MS}{6} = (6+4+2+6+6+4+4+0+4+2) / (60) = 63{,}33\%$$

L'importance de cette formule est qu'elle nous une échelle limité entre les deux valeurs fixes, 0% et 100%, pour comparaison des résultats.

- **Moyenne des Scores Exponentiels (*SE*) :**

Le deuxième indicateur de notre comparaison est un indicateur pour lequel nous avons pondéré l'importance de l'ordre des pages de résultats à l'aide d'un cœfficient exponentiel à base de 2. On accorde ainsi un poids deux fois plus important à la page 1 qu'à la page 2, et la page 2 a un poids deux fois plus grand que la page 3, et ainsi de suite. Puis nous avons calculé le *SE* de ces résultats pour chaque sous question détaillée. Les cœfficients utilisés sont énumérés dans le tableau 5:

Tableau 5 : les coefficients exponentiels de pondération des pages de résultats

Le coeffici ent de page 1	Le coeffici ent de page 2	Le coeffici ent de page 3	Le coeffici ent de page 4	Le coeffici ent de page 5	Le coeffic ient de page 6	Le coeffici ent de page 7	Le coeffici ent de page 8	Le coeffici ent de page 9	Le coeffici ent de page 10
512	256	128	64	32	16	8	4	2	1

Donc pour l'exemple du tableau 4 nous avons:

$$\underline{\textbf{SE}} = \frac{\sum_{i=1}^{10} 2^{(10-i)} * (V(P_i) + 2)}{10} = (6*512 + 4*256 + 2*128 + 6*64 + 6*32 + 4*16 + 4*8 + 0*4 + 4*2 + 2*1)/10 = 50{,}46$$

Et à la même façon pour ***l'Efficacité de SE*** ***(ESE)*** nous avons:

$$\underline{\textbf{ESE}} = \frac{\sum_{i=1}^{10} 2^{(10-i)} * (V(P_i) + 2)}{\sum_{i=1}^{10} 2^{(10-i)} * 6} = (6*512 + 4*256 + 2*128 + 6*64 + 6*32 + 4*16 + 4*8 + 0*4 + 4*2 + 2*1)/ (6*512 + 6*256 + 6*128 + 6*64 + 6*32 + 6*16 + 6*8 + 6*4 + 6*2 + 6*1) = 82\%$$

Dans les prochaines sections nous verrons les résultats obtenus pour les deux indicateurs ci-dessus.

6.2 Les résultats de la première itération

Les résultats de la première itération ont été obtenus pour les recherches exploratoire et informationnelle.

6.2.1. Recherche exploratoire

Pour notre recherche exploratoire nous avons considéré les résultats obtenus pour toutes les sous-questions, incluant la sous-question "Other". Selon le tableau 6 nous pouvons conclure les résultats suivants:

- L'indicateur de ***EMS*** de ARIIA (**74,23%**) est environ 20% mieux que celui de Google (**55,20%**).
- L'indicateur de ***ESE*** de ARIIA (**69,23%**) est environ 10% mieux que celui de Google (**59%**).

Tableau 6 : Les résultats obtenus pour la recherche exploratoire

	Moyenne de *MS* par thème pour les trois évaluateurs			**Moyenne de *SE* (ordonnés) par thèmes pour les trois évaluateurs**	
Thèmes de question	***Google***	***ARIIA***		***Google***	***ARIIA***
hale bopp	1,36	1,06		1295,91	1310,33
black panthers	1,11	1,2		707,53	1739,00
cassini	1,31	1,85		2550,86	2213,33
agouti	0,89	2,0		375,93	2843,06
horus	1,89	6,16		2923,66	2882,08
EMS	**55,20%**	**74,23%**			
ESE				**59,00%**	**69,23%**

Nous notons que pour le thème " Cassini " les pages retournées par ARIIA ne sont pas forcément les pages non pertinentes. Par contre il y a des pages qui sont parmi les meilleures pages (par exemple les pages http://saturn.jpl.nasa.gov/home/index.cfm et http://www.space.com/cassini/) qui contiennent toutes les réponses, mais elles sont évaluées comme les pages moins pertinentes par les évaluateurs car les réponses se trouvent dans leurs hyperliens mais pas dans leurs contenus de texte. Ceci est probablement un effet de l'algorithme PageRank et du fait que ces pages ont plusieurs liens entrants.

6.2.2. Analyse des scores de recherche exploratoire pure (les sous questions "Other")

Ce type de recherche est un sous-ensemble de la recherche exploratoire qui se concentre seulement sur les informations supplémentaires des pages web. Pour cette section nous avons considéré les résultats obtenus pour toutes les sous-questions "Other". Les deux indicateurs ont été calculés pour la recherche exploratoire comme le montre le tableau suivant. Puisque l'évaluation de la sous-question "Other" était basée sur la justification de l'utilisateur nous avons regroupé les résultats de tableau 7 selon les votes des utilisateurs.

Tableau 7 : Les résultats obtenus pour la recherche exploratoire (les sous-questions "Other")

	Moyennes des scores pour chacune des sous-questions "Other"		Moyennes des scores exponentielles pour chacune des sous-questions "Other"	
	Google	**ARIIA**	**Google**	**ARIIA**
Utilisateur 1	1,2	1,8	1462	3102,4
Utilisateur 2	2,4	2,28	2918	3334,4
Utilisateur 3	1,6	2,02	2143	2596,2
EMS	**62,22%**	**67,22%**		
ESE			**68,85%**	**80,26%**

Les résultat de tableau 7 montre que l'indicateur ***EMS*** d'ARIIA est environ 67,22 – 62,22 = 5,00% mieux que celui de Google. De l'autre côté, l'indicateur de ***ESE*** d'ARIIA est environ 80,26 – 68,85 = 11,41% mieux que celui de Google.

6.2.3. Recherche informationnelle (les sous questions sans "Other")

Ce type de recherche est un sous-ensemble de recherche exploratoire qui se concentre seulement sur les informations précises des pages web. Pour cette section nous avons considéré les résultats obtenus pour toutes les sous-questions sauf "Other". Les deux indicateurs ont été calculés pour la recherche informationnelle et sont rapportés dans le tableau suivant:

Tableau 8 : Les résultats obtenus pour la recherche informationnelle

	Moyennes des scores pour chacun des sous questions sauf "Other"		Moyennes des scores exponentielles pour chacun des sous questions sauf "Other"	
	Google	**ARIIA**	**Google**	**ARIIA**
Utilisateur 1	1,2	1,52	1490	2405,2
Utilisateur 2	1,3	1,47	1159	1789,4
Utilisateur 3	1,3	1,59	1426,33	1753,9
Efficiency of MS	**54,44%**	**58,77%**		
Efficiency of SE			**55,46%**	**65,63%**

Les résultat de tableau 8 montre que l'indicateur ***EMS*** d'ARIIA est environ 58,77 – 54,44 = 4,33% mieux que celui de Google. D'un autre côté, l'indicateur de ***ESE*** d'ARIIA est environ 80,26 – 68,85 = 10,17% mieux que celui de Google.

6.3 Les résultats pour les deuxième et troisième itérations

Les deuxième et troisième itérations de recherche sont faites par un seul utilisateur. Environ 690 évaluations ont été faites (230 évaluations pour chacune des itérations).

6.3.1. Recherche exploratoire (toutes les sous-questions contenant "Other")

Le tableau 9 illustre la comparaison de tous les résultats des première, deuxième et la troisième itérations de recherche de l'utilisateur. Dans ce tableau, les résultats obtenus par ARIIA sont montrés par rapport à ceux de Google. Généralement, la performance d'ARIIA pour les deux indicateurs ***EMS*** et ***ESE*** est plus haute que celle de Google. La différence d'indicateur ***EMS*** d'ARIIA a une tendance à la hausse en fonction du nombre d'itérations de recherche. Au contraire, pour l'indicateur ***ESE*** cette tendance est à la baisse mais néanmoins toujours mieux que le résultat de Google (la deuxième ligne du tableau 9). En général on observe que l'indicateur ***SE*** d'ARIIA est supérieur à de celui de Google :

Tableau 9 : Les résultats des itérations de recherche obtenus pour la recherche exploratoire

	Google	ARIIA: première itération	ARIIA: deuxième itération	ARIIA: troisième itération
EMS	55,32%	61,27%	63,58%	64,67%
ESE	60,14%	71,90%	71,61%	68,33%

En bref, l'indicateur ***EMS*** d'ARIIA est de 6% à environ 9% mieux de celui de Google et croît avec le nombre d'itérations de recherche. D'autre part, l'indicateur ***ESE*** est 8,19% mieux que celui de Google pour le pire des cas (la troisième itération) mais l'écart tend à diminuer avec le nombre d'itérations.

Si nous comparons les indicateurs ***SE*** des thèmes généraux des questions du tableau 3 nous trouvons deux comportements différents comme illustré à la figure 12. Cette figure montre la variation de ***SE*** entre chacune des itérations d'ARIIA avec Google.

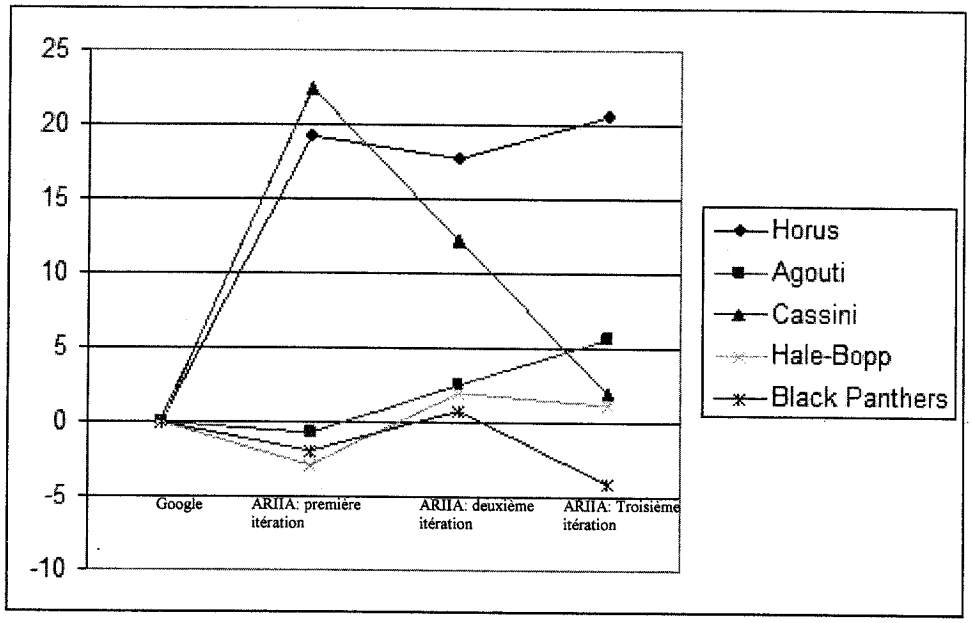

Figure 12 : Différence entre les valeur d'indicateur SE pour les thèmes généraux des questions de tableau 3

On distingue facilement deux situations différentes. La première concerne les thèmes *Cassini* et *Horus* qui ont leur maximum d'écart avec les résultats de Google à leur première itération. La deuxième situation se présente avec les thèmes "*Agouti*","*Hale-Bopp*", et "*Black panthers*", qui ont une légère diminution de l'écart ***SE*** à leur première itération et un résultat plus élevé pour leur deuxième itération de recherche[35].

Nous avons analysé les données pour tenter de comprendre les différences de résultats pour les thèmes des questions. Une différence remarquable de la taille des pages HTML est observable pour les deux titres *Cassini* et *Horus*, ce qui signifie qu'en général, le contenu visible des pages HTML retournées à l'utilisateur est plus élevé par

[35] Comme nous avons mentionné dans la section 6.2.1, la tendance décroissante de ***SE*** de thème "*Cassini*" ne signifie pas que les pages retournés par ARIIA ne sont pas pertinents.

rapport à *Agouti* et *Hale-Bopp*. Nous pouvons donc déduire informellement que la *taille de la page HTML* peut être l'une des raisons de la différence de comportement observée. Le tableau 10 illustre quelques paramètres statistiques calculés selon les informations de la base de donnée de profil d'utilisateur qui a évalué l'ARIIA pour les trois itérations. Ce tableau montre une autre explication de la différence observée.

Tableau 10 : Statistiques sur les nombres des extensions de chacun des titres des questions

Les thèmes généraux	Nombre d'extensions évaluées (les extensions qui ont le score positif ou négatif)	Nombre des extensions qui ont un score positif	Nombre des extensions qui ont un score négatif	Somme des scores	Ecart type des scores	Moyenne des scores	Valeur minimale des scores	Valeur maximale des scores
AGOUTI	28	26	2	3434	332,69	122,64	-16	1672
BLACK PANTHERS	183	133	50	21284	635,33	116,30	-644	8188
CASSINI PROBE	145	145	0	43595	644,33	300,65	2	5068
HALE-BOPP	82	82	0	9942	366,43	121,24	8	2540
HORUS	177	173	4	43782	827,86	247,35	-50	8284

Dans ce tableau, la première colonne montre le thème général des questions. La deuxième colonne montre le nombre d'extensions trouvées pour chaque thème. La troisième colonne montre le nombre d'extensions positivement évaluées par l'utilisateur. La quatrième colonne montre les extensions négativement évaluées. Les autres colonnes montrent la somme, l'écart type, la moyenne, le minimum, et le maximum des scores attribués aux extensions de chacun des thèmes. On voit que le nombre d'extensions obtenues de Wikipedia pour les trois thèmes *Cassini*, *Horus*, et *black panthers* est beaucoup plus élevé (environ deux fois plus) que pour les deux

autres thèmes (*Agouti* et *Hale-Bopp*). D'autre part, il reste à expliquer la différence entre le comportement de *black panters* et les deux autres thèmes *Cassini* et *Horus*. Pour expliquer cette différence, on peut facilement observer un autre facteur relatif en comparant le nombre d'extensions qui sont positivement (ou négativement) évaluées par l'utilisateur.

Malgré qu'il ait un grand nombre d'extensions positivement évaluées, le thème *black panthers* a aussi un grand nombre d'extensions qui sont négativement évaluées. Le ratio entre ses extensions favorables et défavorable est environ 50/133 = 0,37, ce qui est de beaucoup supérieur au ratio obtenu pour *Cassini*(0,0) et le *Horus*(0,02), ainsi que Agouti(0,07) et Hale Bopp(0,0).

En observant la moyenne des scores, et la somme des scores des extensions obtenues de Wikipedia pour chacun des titres *Agouti*, *Hale-Bopp*, et *Black Panthers*, on observe qu'elles se distinguent nettement des valeurs obtenues par les extensions de *Cassini* et de *Horus*. Ceci signifie que ces deux catégories ont des extensions plus favorables et plus positivement évaluées par rapport aux autres.

La figure 13 montre la variation de l'indicateur *MS* pour les thèmes. Ce diagramme affiche une relativement grande variabilité selon les thèmes en fonction des itérations de recherche par ARIIA.

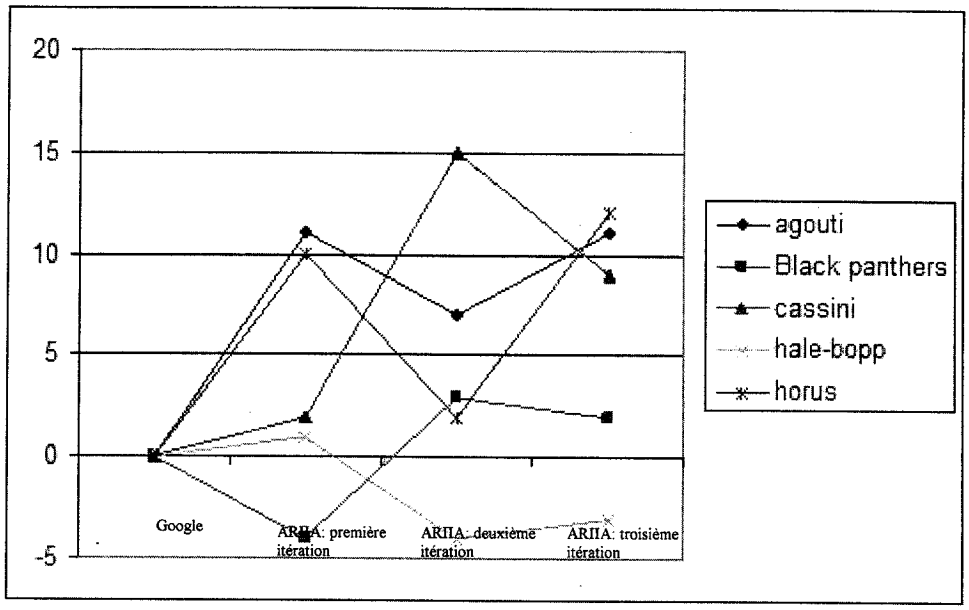

Figure 13 : Différence entre les valeur d'indicateur MS pour les thèmes généraux des questions de tableau 3

6.3.2. Recherche exploratoire pure (les sous-questions "Other")

Le tableau 11 illustre les résultats de la deuxième et la troisième itération de recherche exploratoire faite par ARIIA et Google.

Tableau 11 : Les résultats de recherche exploratoire

	Google	ARIIA; première itération	ARIIA; deuxième itération	ARIIA; troisième itération
EMS	60,33%	67,00%	68,00%	70,67%
ESE	68,34%	75,73%	84,16%	80,36%

La résultat obtenu par ARIIA pour l'efficacité de ***MS*** est de 6,57% (première itération) à 10,67%(troisième itération) mieux que celui de Google. Pour l'efficacité de ***SE*** l'augmentation du résultat d'ARIIA par rapport à Google varie entre 7,39% (première itération) et 15,82% (deuxième itération).

6.3.3. Recherche informationnelle (les sous-questions sans "Other")

Le tableau 11 illustre les résultats des deuxième et troisième itérations de recherche informationnelle fait par ARIIA et Google.

Tableau 12 : Les résultats de recherche exploratoire

	Google	ARIIA; première itération	ARIIA; deuxième itération	ARIIA; troisième itération
EMS	53,74%	57,1%	62,14%	62,9%
ESE	57,75%	62,19%	70,34%	64,94%

Les résultats obtenus par ARIIA pour l'efficacité de ***EMS*** est d'environ 3,2% (première itération) à plus de 8,4% (deuxième itération) mieux que ceux de Google. Pour l'efficacité de ***ESE*** le résultat d'ARIIA varie entre 4,44% (première itération) et 12,59% (deuxième itération).

CHAPITRE 7 : Conclusion et travaux futurs

Le travail visait à présenter une méthode adaptative et dynamique de recherche de l'utilisateur sur Internet dans le cadre d'une architecture itérative appelée ARIIA (Adaptive Reinforcement Iteration-based Internet Agent). ARIIA est un agent de recherche qui est employé comme un complément à des moteurs de recherche comme Google ou AltaVista afin d'améliorer les deux types de recherches : l'informationnelle et l'exploratoire. L'architecture proposée a une certaine originalité dans deux domaines de recherches : les architectures d'agent de côté d'utilisateur de recherche sur le Web, et l'application de Wikipedia pour l'expansion de requête de recherche pour un contexte général. La première nouveauté d'ARIIA est qu'il emploie le corpus évolutionnaire et dynamique de Wikipedia comme ressource d'ontologie pour l'expansion de requête. Cet avantage devient plus évident quand l'aspect multilingue de Wikipedia est considéré. Une autre nouveauté d'ARIIA est sa structure simple et efficace qui exécute une expansion itérative en ligne de la requête originale de l'utilisateur. Enfin, une autre nouveauté d'ARIIA est que son architecture est conçue d'une manière qui adapte les résultats de recherche aux changements de l'intérêt de l'utilisateur particulier durant une longue période de recherche sur un sujet.

Les résultats de test d'ARIIA sont toujours mieux par rapport aux résultats de Google. Dans les pires des cas, ARIIA améliore les résultats de Google de 4% à 6%, selon le type de recherche réalisé. Pour certains types de recherche cette amélioration atteinte a plus de 20% par rapport des résultats de Google. Dans la majorité des cas,

cette amélioration se stabilise à environ 10% et parfois nous avons une augmentation jusqu'à 20% par rapport à Google.

Puisque l'architecture proposée est relativement nouvelle dans ce domaine, elle peut être examinée et validée par plusieurs stratégies différentes. D'un côté, il est préférable que quelques autres validations et tests soient réalisés. Dans notre projet, nous avons réalisé un test de base sur la pertinence des recherches avec ARIIA pour les requêtes simples contenants un seul mot-clef. Il serait intéressant de valider ARIIA avec des requêtes comportant plusieurs mots clefs dès la première itération. D'autre part, une autre stratégie de test peut être de mesurer sa capacité d'adaptativité de recherche selon les changements de l'intérêt de l'utilisateur. Comme nous avons déjà mentionné, l'architecture itérative de l'ARIIA couvre naturellement ce dispositif, mais nous n'avons toujours aucune idée de sa rapidité et sa performance puisque nous ne les avons pas mesurées spécifiquement. En observant les résultats des itérations de recherche, il y a une baisse de ***ESE*** en troisième itération de recherche d'ARIIA par rapport de sa deuxième itération. Comme travail future sur l'analyse de performance d'ARIIA, nous avons besoin de rassembler plus informations détaillées sur chacune des itérations de recherche d'ARIIA. Cette étape nous permettra de mieux interpréter ce comportement de recherche par ARIIA selon les attributs de profil de l'utilisateur et les documents web.

Par ailleurs, l'architecture d'ARIIA est largement extensible. Une avenue de recherche intéressante serait d'explorer le feedback implicite[36] de l'utilisateur sur son niveau d'intérêt pour chacune des pages web retournées. Actuellement, ARIIA nécessite un feedback explicite de l'utilisateur. Ceci signifie que pour chaque page web ouverte, l'utilisateur doit effectuer une opération supplémentaire qui est l'évaluation de cette page. Il peut donc peut se sentir agacé avec cette opération. Il est préférable qu'à l'avenir ARIIA détecte l'intérêt de l'utilisateur sans besoin d'une évaluation explicite.

Finalement, certaines lacunes évidentes pourraient être abordées. Il faudrait notamment éliminer des extensions non valables, « bruyantes », qui surviennent fréquemment avec les itérations. Par exemple, les termes "date" et "time" sont parmi les extensions très fréquentes et bruyantes. Elles surviennent parce que leur présence est très probable dans chacune des pages HTML retournées. Dans ce travail, nous n'avons pas éliminé ces extensions bruyantes et notre résultat de test contient tous les termes de ce type qui sont saisis dans le profil de l'utilisateur. Une autre étape pour améliorer l'exécution d'ARIIA peut être l'utilisation d'une stratégie qui filtre certaines de ces données bruyantes et permet à ARIIA de mieux sélectionner les meilleurs termes dans son expansion de requête.

[36] Pour les informations complémentaires sur le feedback implicite référez aux articles de référence.

BIBLIOGRAPHIE

ANTONIOU G., HARMELEN F. 2004. *A Semantic Web Primer*: 2nd ed. MIT press

ARONSON A. R., T. C. RINDFLESCH. 1997. "Query Expansion Using the UMLS Metathesaurus". *National Library of Medicine, Bethesda, MD 20894* [En ligne]: http://skr.nlm.nih.gov/papers/references/query_expansion.97.pdf. (Page consulté le 6 juin 2006).

BHARAT K. 2005. "SearchPad: Explicit Capture of Search Context to Support Web Search". *Compaq Systems Research Center, Conference on the Web: the Next Generation.* [En ligne]: http://www9.org/w9cdrom/173/173.html. (Page consulté le 6 juin 2006).

BRIN S., PAGE L. 1998. "The Anatomy of a Large-Scale Hypertextual Web Search Engine". *Proceedings of WWW7 / Computer networks conference.* [En ligne]: http://www-db.stanford.edu/~backrub/google.html. (Page consulté le 6 juin 2006).

CLARKE G., CORMACK G., KEMKES G., LASZLO M., LYNAM T., TERRA L., TILKER P. 2002. "Statistical Selection of Exact Answers(Multi text experiments for TREC 2002)". *School of computer science, University of Waterloo, Draft for TREC Conference 2002 Notebook.*

COLLINS K., JAMIE CALLAN Th. "Query Expansion Using Random Walk Models". *Germany : Proceedings of ACM CIKM'05 Conference.* October / November. 2005.

DIMITROVA M., KUSHMERICK N., RADEVA P., VILLANUEVA J. "User Assessments in Visual Web Genre Classifier", *Proceedings of Int'l. Conference on Visualization, Imaging, and Image Processing,* 2003 [En ligne]: www.smi.ucd.ie/nick/research/download/dimitrova-viip2003.pdf. (Page consulté le 6 juin 2006).

DING L., FININ T., JOSHI A., PENG Y., COST R. S., SACHS J., PAN R., REDDIVARI P., DOSHV. I. "Swoogle: a Semantic Web search and Metadata Engine", *Proceedings of the Thirteenth ACM Conference on Information and Knowledge Management.* February 2004.

DING L., ZHOU L., FININ T. "Trust Based Knowledge Outsourcing for Semantic Web Agents", *Proceedings of IEEE/WIC International Conference on Web Intelligence (WI'03) , Halifax Canada.* October 2003.

DING Z., PENG Y., PAN R.. "A Bayesian Approach to Uncertainty Modeling in OWL Ontology", *Proceedings of the International Conference on Advances in Intelligent Systems - Theory and Applications , Baltimore MD, US.* November 2004.

FENSEL D. "The semantic Web and its Languages", December 2000, [En ligne]: www.computer.org/intelligent/ex2000/pdf/x6067.pdf. (Page consulté le 6 juin 2006).

FOX S., KARNAWAT K., MYDLAND M., DUMAIS S. WHITE T. "Evaluating Implicit Measures to Improve Web Search", *ACM Journal of Transactions on Information Systems, Vol. 23, No. 2.* April 2005. Pages 147–168.

FRIENDLY M. "Accent principles for effective graphical display" [En ligne]: http://www.math.yorku.ca/SCS/Gallery/.(Page consulté le 6 juin 2006).

GAGNON M., KARDOOST M., SUN CH.. "Towards a semantic web site for an academic department", *Proceedings first annual scientific conference of the LORNET Research Network,* 15 October 2004.

GUHA R., McCOOL R. "TAP: A Semantic Web Test-bed", *Journal of Web Semantics, Volume 1, Issue 1,* December 2003.

GUHA R., McCOOL R. "TAP: Towards the semantic web", [En ligne]: http://tap.stanford.edu/www2002.ppt. (Page consulté le 6 juin 2006).

GUHA R.V., McCOOL R. "TAP: A System for integrating Web Services in to a Global Knowledge Base", [En ligne]: http://tap.stanford.edu/sw002.html. (Page consulté le 6 juin 2006).

HIJIKATA Y. "Implicit User Profiling for On Demand Relevance Feedback", *Proceedings of ACM IUI'04 Conference, Madeira, Funchal, Portugal,* P. 198 – 205. January 2004.

"History of internet and WWW"[En ligne]: http://www.netvalley.com/intvalstat.html. (Page consulté le 6 juin 2006).

"HTML Relative URLs" [En ligne]: http://www.webreference.com/html/tutorial2/3.html. (Page consulté le 6 juin 2006).

IMAI H., COLLIER N., TSUJII J. "A Combined Query Expansion Approach for Information Retrieval" , *Presented at the Genome Informatics Workshop 1999, , Garden Hall, Yebisu Garden Place, Tokyo, Japan, Published as: i, T. (eds.) "Genome Informatics 1999", Universal Academy Press, Tokyo.* December 14-15, 1999.

IWAYAMA M. "Relevance Feedback with a Small Number of Relevance Judgments: Incremental Relevance Feedback vs. Document Clustering", *Proceedings of ACM SIGIR 2000 Conference 7/00, Athens, Greece.* P. 10 – 16. July 2000.

JOACHIMS T. "Optimizing Search engines using Clickthrough Data", In Proceedings of the Eighth ACM SIGKDD International Conference on Knowledge Discovery and Data Mining, 1-58113-567-X/02./0007. 2002.

JOACHIMS Th. GRANKA L., PAN B., GAY G., "Accurately Interpreting the clickthrough Data as Implicit Feedback", *Proceedings of ACM SIGIR'05 Conference, Salvador, Brazil.* P. 154 – 161. August 2005.

KartOO technologies, The KartOO intranet users technical reference [En ligne]: http://www.kartoo.net/e/eng/doc/intranet2.pdf. (Page consulté le 6 juin 2006).

KartOO technologies, The KartOO sitebox commercial reference [En ligne]: http://www.kartoo.net/e/eng/doc/kartOOsitebox2.pdf. (Page consulté le 6 juin 2006).

KELLY D., NELKIN N. "Modeling Characteristics of User's Problematic Situation with Information Search and Use Behaviors", *Proceedings of ACM/IEEE Joint conference on Digital Libraries (JCDL'02), Portland OR. US.* 2002.

KELLY D., TEEVAN J. "Implicit feedback for inferring user preference: a bibliography", *Proceedings of ACM SIGIR Conference, Forum Volume 37, Issue 2.* P. 18 – 28. Fall 2003.

LE GRAND B. "Extraction d'information et visualisation de systèmes complexes sémantiquement structurés ", [En ligne]: http://www-rp.lip6.fr/~blegrand/index.html. (Page consulté le 6 juin 2006).

LITA VIAD L., HUNT W. A., NYBERG E. "Resource Analysis for Question Answering", *Proceedings of the 42nd Annual Meeting of the Association for Computational Linguistics (ACL 2004), Barcelona, Spain.* July 2004.

McKENZIE B., COCKBURN A. "An Empirical Analysis of Web Page Revisitation", *IEEE 0-7695-0981-9/01, Proceedings of 34th Annual Hawaii Intl' Conf. on System Sciences (HICSS-34)- Track 5, Maui* . January 2001.

MISHNE G., DE RIJKE M., JIJKOUN V. "Using a Reference Corpus as a User Model for Focused Information Retrieval", *Journal of Digital Information Management, Volume 3 Number 1.* Mar. 2005.

MITCHEL T. M. 1997. *Machine Learning*: 4th ed. MIT press & The McGraw-Hill co-publisher Inc.

MOLDOVAN D., RUS V. "Explaining answers with extended WordNet", *Proceedings of Proceedings of the Annual Meeting of the Association for Computational Linguistics (ACL), Toulouse, France.* 2001

MOSTAFA J., MUKHOPADHYAY S., LAM W., PALAKAL M. "A Multilevel Approach to Intelligent Information Filtering: Model, System, and Evaluation", *ACM Volume 15, Issue 4* P. 368 – 399. October 1997.

NANAS N., UREN V., DOMINGUE J., DE ROECK A. "Nootropia: a User Profiling Model based on a Self-Organising Term Network", *Proceedings of ICARIS, Catania, Italy.* Feb.2004.

NELSON D. L., McEVOY C. L., SCHREIBER T.A. 1998. "The University of South

Florida word association, rhyme, and word fragment norms" [En ligne]: http://www.usf.edu/FreeAssociation/.(Page consulté le 6 juin 2006).

"Search Engine Watch" [En ligne]: http://www.searchenginewatch.com. (Page consulté le 6 juin 2006).

SHEN X., TAN B., ZHAI CH. "Context-Sensitive Information Retrieval Using Implicit Feedback". *Proceedings of SIGIR'2005, Salvador, Brazil,* P. 43 - 50. 2005.

SHEN X., TAN B., ZHAI CH. "UCAIR: "Capturing and Exploiting Context for Personalized Search", *In Proceedings of SIGIR'2005 IRiX Workshop, Salvador, Brazil* 2005.

"Six degrees of Wikipedia" [En ligne]: http://kohl.wikimedia.org/~kate/cgi-bin/six_degrees?from=Bush_43&to=president. (Page consulté le 10 janvier 2006).

VOORHEES E.M. "Overview of the TREC 2004 Question Answering Track", *National Institute of Standards and Technology, MD 20899*, Gaithersburg.

WHITE R., RUTHVEN I., JOSE J. "A Study of Factors Affecting the Utility of Implicit Relevance Feedback", *ACM SIGIR'05 Conference, Salvador, Brazil*, P. 35 – 42. August 2005.