Expansion de requêtes par apprentissage automatique dans un assistant pour la recherche d'information

J.C. Bottraud *, — G. Bisson ** — M.F. Bruandet ***

** MRIM-CLIPS, Université Joseph Fourier - 38041 St Martin d'Hères*

*** Apprentissage-Leibniz, 46 Avenue Félix Viallet - 38031 Grenoble Cedex*

{jean-christophe.bottraud, gilles.bisson, marie-france.bruandet}@imag.fr

ABSTRACT: Tools available to search information on the Web have a general approach, ignoring a user's characteristics, and this limits the quality of the results that they may provide. The AIRA system presented here uses document references gathered by the user to build a profile to describe him, and it uses this profile to interpret and filter the results retrieved by Web search engines. In this article, we focus on an algorithm to expand queries, using machine learning techniques, and on the problems caused by the evaluation of systems of this type.

RESUME : Les outils disponibles de recherche d'information sur le Web ont une approche généraliste, ne prenant pas en compte les caractéristiques de l'utilisateur, ce qui limite la qualité des résultats qu'ils sont susceptibles de fournir. Le système AIRA présenté ici utilise les références documentaires rassemblées par l'utilisateur pour construire un profil le représentant, exploité pour interpréter et filtrer les résultats proposés par les moteurs de recherche. Dans cet article nous nous focalisons sur un algorithme d'expansion de requêtes à l'aide de techniques de l'apprentissage machine, et sur les problèmes par l'évaluation de cette famille de systèmes.

KEYWORDS: Information Retrieval, Intelligent Agent, Automatic Clustering, Context, Query Expansion, Evaluation

MOTS-CLES : Recherche d'information, Agent Intelligent, Classification automatique, Contexte, Expansion de requête, Evaluation

1. Introduction

Les services de recherche actuellement disponibles pour la recherche d'information sur le Web, malgré leur succès, restent imparfaits en raison notamment de 3 problèmes : l'ambiguïté intrinsèques de requêtes contenant peu de mots, la non prise en compte du niveau d'expertise de l'utilisateur, et enfin d'éventuels conflits d'intérêt entre ce dernier et les sites. Nous ne reviendrons pas sur ces différents points, déjà évoqués dans Bottraud et al, 2003. L'une des voies d'amélioration proposées par différents auteurs (cf. partie 2) est l'utilisation d'un « assistant » s'exécutant sur la machine de l'utilisateur et qui joue le rôle de médiateur entre celui-ci et le moteur de recherche. De façon complémentaire, cet article propose l'utilisation dans un assistant d'un mécanisme d'expansion de requêtes basé sur des techniques d'apprentissage automatique, exploitant un profil documentaire propre à l'utilisateur. Le système correspond à une extension du système AIRA, présenté dans (Bottraud et al., 2003).

Par ailleurs la réalisation d'un système de ce type exige un assemblage de plusieurs techniques utilisées en apprentissage ou en recherche d'information (indexation, modèle de documents, …), ce qui rend malaisé les comparaisons entre les différentes approches possibles à cause du nombre de variations possibles et de la difficulté d'identifier la cause d'un éventuel écart de performance. A travers la conception de notre système nous proposons une approche systématique permettant d'évaluer les différents composants.

Cet article est organisé comme suit : la première partie présente un ensemble de travaux en rapport avec le système et les objectifs proposés. La partie suivante rappelle les caractéristiques principales de AIRA. Nous détaillons ensuite les mécanismes mis en œuvre pour assister les recherches. Les problèmes posés par l'évaluation de cette classe de système sont ensuite détaillés, avant de décrire l'approche que nous proposons pour y répondre. Enfin, la dernière partie présente les expériences réalisées et les résultats obtenus, de façon à illustrer la démarche d'évaluation proposée et de donner une première évaluation de la stratégie d'assistance à la recherche mise en oeuvre.

2. Travaux existants

Différentes approches ont été proposées pour faciliter, voire automatiser, l'exploration du Web. La plupart des systèmes reposent sur un processus actif *d'adaptation à l'utilisateur* et sont assimilables à des « assistants personnels » de recherche d'information. Dans ce contexte, plusieurs objectifs, non indépendants, peuvent être recensés comme : assister l'utilisateur pour parcourir un ensemble de liens hypertextuels, l'aider à formuler et raffiner une requête dans un moteur de recherche, filtrer et réordonner la liste des documents retournés par ces moteurs.

2.1 Les assistants personnels

L'assistance à la navigation peut être réalisée soit en aidant l'utilisateur à sélectionner les liens pertinents d'une page, soit en lui soumettant des documents en relation avec le document en cours de lecture, via une exploration automatique des liens. Cette approche est illustrée notamment dans Syskill & Webert (Pazzani *et al.*, 1996), Letizia (Lieberman 1995), WAIR (Seo *et al.*, 2000) WEBAce (Boley *et al.*, 1998), WebWatcher (Armstrong *et al.*, 1995) ou les différents modules proposés avec WBI (Maglio *et al.*, 2000). Une autre approche classique consiste à filtrer automatiquement les informations produites par un ensemble de sources sélectionnées par l'utilisateur (magazines électroniques, News de UseNet, …) pour ne présenter que les documents potentiellement intéressants. Alipes (Widyantoro *et al.*, 1999), IDD News Browser (Bloedorn *et al.*, 1996), INFOrmer (Sorensen *et al.*, 1998) et News Dude (Billsus *et al.*, 1999) font partie de cette catégorie de système.

L'amélioration de l'utilisabilité des moteurs de recherche est un autre thème de recherche dans lequel soit on aide l'utilisateur à sélectionner les termes de la requête, soit on réalise de « l'expansion de requêtes » afin de ramener une collection de documents plus large et mieux ciblée. WebMate (Chen *et al.*, 1998), WebNaut (Nick *et al.*, 2001) ou Arachnid (Menczer, 1997) entrent dans cette famille. Une autre approche classique en recherche d'information consiste à réordonner ou classifier les références obtenues via le moteur ; c'est le cas de Grouper (Zamir *et al.* 1999)

Enfin, dans un cadre plus systématique, tel celui d'une veille technologique, d'autres systèmes cherchent à réduire le travail de l'utilisateur via la génération automatique de requêtes, éventuellement combinée avec l'exploration automatique des liens présents dans les documents obtenus. Les systèmes Amalthae (Moukas *et al.*, 1997), InfoSpider (Menczer *et al.*, 2000), InfoFinder (Krulwich *et al.*, 1997), ou WAWA (Shavlik *et al.* 1999, Goecks *et al.* 2000) illustrent ce type d'approche.

Ces différentes techniques peuvent être combinées dans la recherche d'une plus grande efficacité. C'est notamment le cas de Alipes (Widyantoro *et al.*, 1999), de InfoSpider (Menczer *et al.*, 2000) ou de Syskill & Webert (Pazzani *et al.*, 1996).

2.2 Modélisation de l'utilisateur

La plupart des systèmes effectuent leur adaptation en construisant un « profil de l'utilisateur » par apprentissage à partir des documents consultés. Ce profil qui s'appuie généralement sur le Vector Space Model (Salton, G., 1971) est constitué de un ou plusieurs vecteurs définis dans un espace de termes ; les coordonnées correspondent aux poids associés aux termes retenus dans le profil. L'utilisation de plusieurs vecteurs correspond à deux préoccupations : pouvoir prendre en compte des centres d'intérêt multiples, et gérer leur évolution dans le temps. Dans ce dernier cas, le système utilise deux groupes de vecteurs évoluant suivant des règles différentes (c'est le cas d'Alipes et de News Dude).

Dans la majorité des cas, l'évolution des profils repose sur des variantes de la méthode de « bouclage de pertinence » (Relevance Feedback, Rocchio J., 1971). L'évaluation des documents retrouvés peut être explicitement fournie par l'utilisateur, ou être déduite de l'observation de l'activité de l'utilisateur, comme dans les systèmes LAW (Edwards P. *et al.*, 1996) ou WAIR (Seo Y.-W. *et al.* 2000). Différents indicateurs (Goecks *et al.*, et 2000 ; Claypool M. *et al.*, 2001) sont utilisables : mouvements et clicks de souris, temps passé sur la page, nombre de liens activés, ... L'ajustement du poids des termes fait de plus en plus appel à des techniques d'apprentissage, comme des réseaux de neurones (Shavlik et al, 1999), des probabilités Bayésiennes (Pohl *et al.*, 1999), des algorithmes à base de règles (Bloedorn E. et al, 1996 ou Krulwich *et al.*, 1997) ou des algorithmes génétiques (Moukas *et al.*, 1997, Menczer F., 1997, Nick *et al.*, 2001).

La construction d'une ontologie personnelle (cf. Huhns *et al.*, 1999) offre une alternative intéressante à ces approches. OBIWAN (Gauch *et al.*, 2003, Chaffee et al, 2000) propose ainsi une approche dans laquelle la personnalisation est fondée sur l'élaboration d'une ontologie personnelle par la sélection dans une ontologie générale de nœuds estimés correspondre aux intérêts de l'utilisateur. Cette sélection est opérée sur la base d'un ensemble de documents organisé par l'utilisateur. La construction d'une ontologie personnelle peut également être réalisée à partir des techniques employées par les systèmes élaborant automatiquement des thésaurus à partir de l'analyse de collections de documents (cf. Baeza-Yates *et al.*, 1999).

2.3 Prise en compte du contexte

Relativement peu de travaux utilisent des *informations contextuelles* non directement liées à la recherche en cours (c'est-à-dire, autres qu'un « feedback » direct ou indirect sur les documents obtenus) pour focaliser la recherche et désambiguïser les requêtes. Remembrance Agent (Bradley *et al.*, 1996) analyse les courriers en cours de création pour proposer des courriers similaires, parmi ceux référencés par l'utilisateur. Watson (Budzik *et al.*, 2000) étudie les documents manipulés par l'utilisateur afin d'engendrer ou étendre des requêtes, mais sans construire de profil explicite. Suitor (Maglio *et al.*, 2003) observe les mouvements de l'utilisateur, notamment la direction de son regard, afin d'identifier le centre d'intérêt courant et adapter le profil. Enfin, WordSieve (Bauer *et al.*, 2001) utilise les documents consultés pour identifier la tâche active, en comparant les termes de ces documents avec des descriptions de contexte de tâche prédéfinies.

2.4 Synthèse

Les « assistants personnels » décrits ci-dessus présentent certains inconvénients concernant l'initialisation, la représentation et l'évolution du profil de l'utilisateur :

– Lorsque la construction du modèle de l'utilisateur est guidée par le *contrôle de pertinence*, l'assistant ne devient opérationnel que lorsque l'utilisateur l'a « entraîné » durant un certain temps avec un risque de démotivation de ce dernier.

– Les caractéristiques retenues pour représenter les centres d'intérêt de l'utilisateur sont souvent peu nombreuses (cent termes environ) et organisées de manière non hiérarchique (partitions) ce qui ne facilite ni leur interprétation ni la prise en compte des différents niveaux de généralités présents dans les connaissances de l'utilisateur.

– Le contexte dans lequel une recherche est effectuée (c'est-à-dire la tâche en cours, comme l'écriture d'un rapport, par exemple) est rarement utilisé pour comprendre la requête et la situer par rapport au profil de l'utilisateur. Dans ce cadre, l'approche préconisée par WordSieve nous semble intéressante.

– Enfin, dans l'évolution du profil, les différentes causes du changement d'intérêt, et les échelles temporelles associées, sont peu ou pas prises en compte : ainsi, une tâche peut typiquement durer de quelques minutes à plusieurs jours, alors que l'évolution des centres d'intérêt peut prendre des semaines voire des mois.

3. Présentation générale du système AIRA

Le fonctionnement du système AIRA (Adaptive Information Research Assistant, Bottraud et al., 2003) repose sur l'hypothèse que l'utilisateur maintient à jour un ensemble de documents ou de références, ou « bibliothèque personnelle », qui est représentative de ses centres d'intérêts. C'est à partir de cet ensemble qu'AIRA extrait et structure les connaissances représentatives de l'utilisateur. Le système (Figure 1) est construit à partir de trois grand blocs : le profil utilisateur, les observateurs d'activités, et les composants intervenants pour gérer une requête.

Le **profil utilisateur** est construit à partir de la bibliothèque de l'utilisateur et est organisé grâce à plusieurs composants ayant chacun un rôle particulier : l'*observateur* de la bibliothèque chargé d'en fournir une représentation manipulable par le système et d'en surveiller les évolutions, le *profil conceptuel*, qui organise les documents de la bibliothèque en une hiérarchie stricte[1] de classes pouvant être donnée « a priori » ou construite automatiquement par un mécanisme de classification incrémentale, et le *profil opérationnel*, dérivé du profil conceptuel, associant à chaque classe une description intentionnelle sous forme de liste de termes, directement exploitable lors de la recherche.

Les **observateurs d'activités** sont chargés de collecter de façon autonome les sous-produits textuels des activités de l'utilisateur, par interaction avec les composants logiciels qu'il utilise (contenu du presse-papier, pages WWW visualisées, documents édités, …). Ces éléments textuels sont enregistrés dans un

[1] La possibilité pour un document d'appartenir à plusieurs classes dépend de l'algorithme utilisé. Cobweb, utilisé ici, ne le permet pas.

journal chronologique des actions correspondant à la mémoire « à court terme » du système.

Enfin, les **composants utilisés pour la recherche** correspondent au *gestionnaire de contexte*, chargé d'exploiter le journal des actions et le profil opérationnel pour identifier le contexte de travail actuel, et au *gestionnaire de requête*, qui utilise le contexte pour interpréter la requête, la soumettre aux moteurs de recherche sélectionnés et enfin afficher les résultats collectés, après qu'ils aient été analysés et reclassés.

Le fonctionnement du système est ainsi basé sur trois tâches : l'observation continue de l'activité de l'utilisateur afin de garder « à jour » le journal des actions ; la gestion de l'évolution du profil, déclenchée par une modification de la bibliothèque personnelle ; la gestion de la recherche d'information proprement dite, déclenchée à l'initiative de l'utilisateur lorsqu'il soumet une requête.

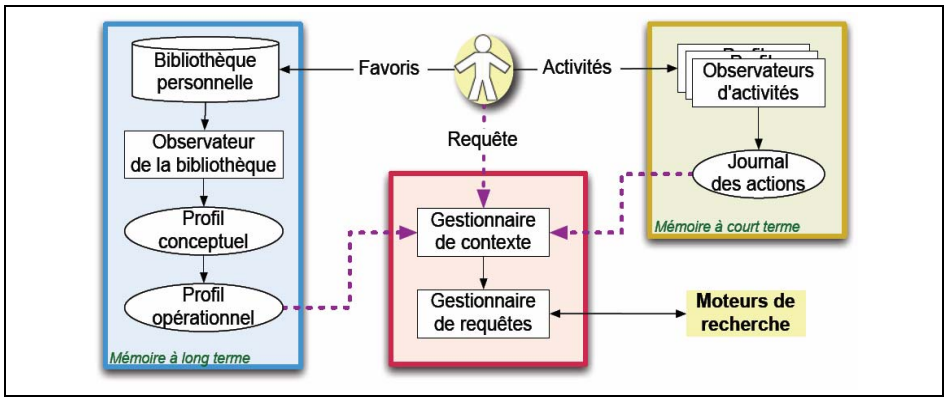

Figure 1. *Principaux composants du système AIRA*

4. Hypothèses

Le processus de gestion des requêtes par l'assistant AIRA repose sur plusieurs hypothèses portant à la fois sur le comportement de l'utilisateur et les caractéristiques du système documentaire (cf. ***Figure 2***):

– Hypothèse **1 – *Requête décentrée:*** Les termes sélectionnés par l'utilisateur ne sont pas nécessairement les plus efficaces. Il peut exister des synonymes ou des termes au sens lié permettant de mieux décrire les documents recherchés. On suppose que ces termes existent dans le contexte, parce qu'ils sont apparus en liaison avec les termes de la requête ou ceux issus des actions l'utilisateur.

– Hypothèse **2 – *Suffisance :*** La combinaison de la requête et du contexte fournissent une base raisonnable pour l'évaluation de la pertinence d'un document

– Hypothèse **3 – *Utilisateur persévérant :*** Il est possible de mieux explorer l'espace documentaire accessible en utilisant des requêtes générées à partir des

termes contenus dans la requête initiale et de ceux contenus dans le contexte, c'est-à-dire que les actions de l'utilisateur et les références enregistrées dans sa bibliothèque conservent une cohérence globale, en rapport avec les sujets d'intérêt de l'utilisateur : ce dernier persévère dans les mêmes thèmes.

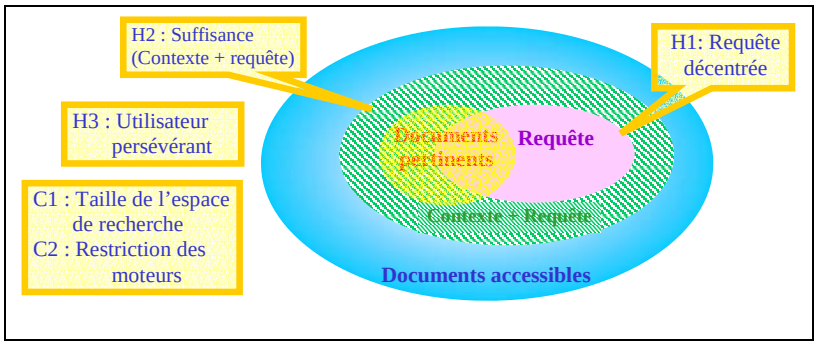

Figure 2. *Sous-espaces de l'espace documentaire*

Parallèlement à ces hypothèses, il est nécessaire de prendre en compte les contraintes imposées par l'environnement dans lequel opèrent les agents chargés de la recherche. Ces contraintes sont essentiellement de deux types, selon qu'elles sont liées à la nature même du World Wide Web, ou aux moteurs de recherche : les contraintes associées au World Wide Web sont liées à son ampleur et au temps nécessaire à l'accès à un grand nombre de documents ; les contraintes imposées par les moteurs de recherche sont essentiellement dues aux limites imposées sur le nombre de termes pouvant figurer dans une requête (~une quinzaine), aux capacités d'expression des langages de requêtes, et enfin au risque de définir des requêtes trop spécialisées ne fournissant pas de résultat.

Pour assister l'utilisateur, deux stratégies peuvent donc être définies :

– La première, relativement simple, utilise le contexte pour filtrer et réordonner les résultats retournés par le ou les moteurs de recherche auxquels la requête a été soumise. Cette stratégie a l'inconvénient de ne pas explorer l'espace en dehors du sous-espace initialement définie par la requête, mais elle permet de recentrer les résultats sur l'intersection de ce sous-ensemble avec celui des documents pertinents. C'est celle initialement décrite dans (Bottraud et al. 2003)

– La seconde, plus élaborée, utilise un mécanisme « d'expansion de requêtes » exploitant les informations disponibles dans le contexte. En générant des requêtes dérivées de la requête initiale, l'espace exploré est plus important ce qui accroît les chances d'obtenir des documents pertinents.

5. Expansion de requête et apprentissage

5.1 Introduction

De nombreux travaux ont portés sur l'expansion de requête. Ils peuvent être répartis en deux groupes principaux, suivant que les transformations de requêtes exploitent l'évaluation par l'utilisateur des résultats obtenus par la requête initiale (« relevance feedback »), ou qu'elles utilisent directement les résultats obtenus.

L'approche proposée par Rocchio (Rocchio, 1971, Buckley et al., 1995) dans le cadre du modèle d'espace vectoriel de documents est emblématique du premier groupe. Différentes approches ont été également proposées sur des bases probabilistes ou de théorie de l'information (cf. notamment Carpineto et al., 2001).

L'autre groupe utilise une analyse des documents obtenus pour identifier des associations entre termes en étudiant les cooccurrences, les termes les plus proches des termes de la requête étant utilisés pour l'étendre. De façon similaire, une analyse des documents de tout un corpus peut également fournir des associations entre termes. La construction automatique d'un thésaurus peut éventuellement être utilisée pour définir des concepts, les lier par des relations et les associer à des termes. Ces deux types d'approche peuvent être combinés (Xu et al., 1996, Baeza-Yates, 1999).

En ce qui concerne les assistants personnels, comme nous l'avons mentionnés dans la partie 2.1, plusieurs systèmes mettent en œuvre des mécanismes d'expansion faisant appel à des approches d'apprentissage automatique (WebMate - Chen et al., 1998, WebNaut - Nick et al., 2001, Arachnid - Menczer, 1997 , …). L'assistant que nous proposons ici repose également sur un paradigme d'apprentissage.

5.2 Algorithme d'apprentissage pour l'expansion

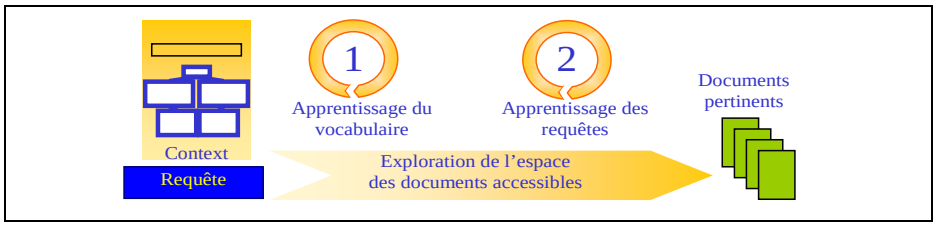

Figure 3. *Vue d'ensemble du processus de requête"QueryExpansion"*

Dans AIRA, l'expansion de requêtes repose sur un double mécanisme itératif classique d'apprentissage de type « beam search », explorant en parallèle une liste d'hypothèses candidates. A chaque itération, les hypothèses retenues sont spécialisées. L'utilisation qui en est faite ici met potentiellement en jeu deux phases. Dans la première phase, le système apprend un vocabulaire en identifiant les termes les plus « efficaces » pour la recherche des documents (c'est à dire les termes

permettant d'obtenir les documents ayant la meilleure évaluation de pertinence). Dans la deuxième phase, il recherche la combinaison optimale des termes ainsi sélectionnés, ce qui correspond à la génération effective des requêtes.

A l'issue de ces deux phases, les documents perçus comme les plus pertinents sont présentés à l'utilisateur (classés par pertinence décroissante).

Les premiers tests réalisés nous ont rapidement montrés que la réalisation de deux cycles d'apprentissage exigeait la génération et le traitement d'un nombre de requêtes incompatible avec la présentation de réponses à l'utilisateur dans un délai raisonnable. La phase d'apprentissage du vocabulaire a donc été abandonnée, le vocabulaire utilisé dans la génération des requêtes est donc directement déduit du contexte, de même que les poids associés aux termes.

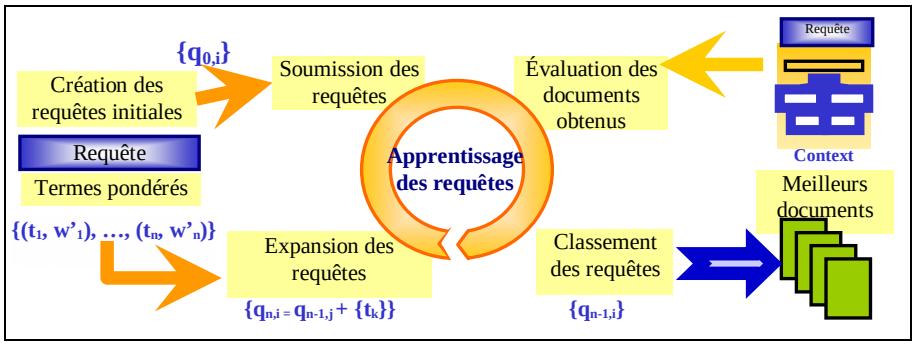

Figure 4 *Agent "QueryExpansion" - Apprentissage de la requête*

```
soit s un service de recherche
soir D l'ensemble des documents retrouvés, avec leur évaluation de pertinence
D ← Ø
soit q_init la requête initiale de l'utilisateur
soit Q_i l'ensemble des requêtes, et des poids associés, générées à l'itération i
Q_1 ← {(q_init, 1)}
soit V_q,t = {(t_i, w_i)} le vocabulaire disponible
soumettre q_init à s et évaluer la pertinence des documents doc_init,k obtenus.
D ← D U{( doc_init,k, evaluation(doc_init,k))}
pour i = 1 à max_iter faire {
        sélectionner n requêtes q_j par tirage pondéré dans Q_i
        Q_i+1 ← Ø
        pour j = 1 à n faire  {
                sélectionner L nouveaux termes par tirage pondéré dans V_q,t
                générer une nouvelle requête q_j' en ajoutant à q_j ces L termes
                soumettre q_j' à s et évaluer la pertinence des documents doc_i,j,k obtenus.
                D ← D U{(doc_i,j,k, evaluation(doc_i,j,k))}
                score w_i de q_j'= Max(evaluation(doc_i,j,k))
                Q_i+1 ← Q_i+1 U{( q_j', w_i)}
        }
}
Retourner les m meilleurs documents de D
```

Figure 5. *Agent "QueryExpansion" - Algorithme pour l'apprentissage des requêtes*

La phase de génération des requêtes se déroule donc de la façon schématisée Figure 4. Elle met en jeu un cycle contenant plusieurs étapes qui consiste à spécialiser progressivement la requête initiale jusqu'à une limite pré-établie ou jusqu'à ce que les requêtes n'obtiennent plus de document. L'algorithme correspondant est présenté Figure 5.

Tout au long de cette phase, les références de tous les documents accédés et les évaluations de pertinence associées sont conservées. Une fois la phase terminée, les meilleurs documents obtenus sont présentés à l'utilisateur, classés par évaluation de pertinence décroissante.

5.3 Exemple d'itérations

Prenons comme exemple la requête « object oriented design », pour laquelle le système a établi le contexte suivant, sous forme d'un vecteur de termes pondérés :

{(0.346, java), (0.189, class), (0.163, xml), (0.151, uml), (0.143, use), (0.140, thread), (0.132, program), (0.119, model),(0.112, application), (0.110, rmi), (0.108, method), (0.107, servlet), (0.103, new), (0.103, system)}

Si à chaque itération 3 nouvelles requêtes sont générées par addition d'un terme aux requêtes existantes, les requêtes suivantes pourraient être obtenues:

Itération	Requêtes générées	Meilleur score obtenu	Commentaires
initiale	object oriented design	0.4	
1	object oriented design java object oriented design class object oriented design uml	0.5 0.2 0.35	Expansion de la requête initiale
2	object oriented design java xml object oriented design java class object oriented design uml java	0.48 0.4 0.55	Tirage pondéré : la requête la plus « performante » est reprise deux fois
3	object oriented design java xml object oriented design uml java class object oriented design uml java use	0.6 0.62 0.43	Tirage pondéré mais aléatoire

Tableau 1. *Exemple de cycle de génération de requêtes*

6. Méthodologie d'expérimentation

L'évaluation de ce type de système est rendue difficile par l'aspect dynamique de l'environnement du système et le caractère fortement adaptatif de ce dernier. Il est de plus extrêmement difficile de définir une métrique d'évaluation absolue, utilisable quelque soit le contexte. La comparaison entre des configurations différentes est également complexe parce qu'il est difficile de les mettre dans des

conditions identiques [2] . Pour effectuer cette comparaison, plusieurs agents, représentant les différentes configurations à évaluer (profil/gestionnaire de contexte/gestionnaire de requêtes), sont donc utilisés simultanément par l'assistant pour traiter les requêtes de l'utilisateur. Chaque agent dispose de toutes les informations disponibles et les combine comme il le souhaite. Un agent « par défaut» est également activé, n'utilisant aucune information particulière : il se contente de transférer la requête à un moteur, sans modification. Il sert de référence en permettant d'évaluer l'intérêt du système par rapport aux systèmes disponibles actuellement sur le World Wide Web.

Le fonctionnement du système peut alors être schématisé de la façon représentée Figure 6 : l'utilisateur saisit une requête dans la page correspondante de l'interface de AIRA. Cette requête est transmise au composant « Agence », qui la transmet à l'agent concerné, soit ici un agent gérant plusieurs agents de recherche. L'agent accuse réception de la demande (étapes 5 à 7) et effectue la recherche (étapes 8 à 9). Il effectue les tâches administratives liées à l'enregistrement de la requête, et la transmet à chacun des agents de recherche. Ces derniers traitent la requête an fonction de leurs attributs, et renvoient les résultats obtenus à l'agent gestionnaire. Celui-ci fusionne les résultats, puis les enregistre (accompagnés des informations permettant de savoir quel agent a fourni quelle référence) avant d'envoyer la liste des références consolidées à l'utilisateur (étape non présente sur la figure).

La consolidation des résultats pose le problème du classement relatif des résultats fournis par chacun des agents. Il n'est en effet pas garanti que les scores fournis soient comparables entre eux, dans la mesure où rien ne peut être affirmé a priori sur la façon dont un agent estime la pertinence des documents.

La seule information effectivement disponible avec une interprétation raisonnablement constante au niveau de l'utilisateur est l'ordre de classement des documents. Nous avons donc décidé d'ordonner les documents fournis par chaque agent en utilisant le rang comme indicateur de pertinence, c'est-à-dire en attribuant comme pertinence à un document la valeur $p = (n - 1 - r) / (n - 1)$, où n est le nombre de documents considérés et r [3] le rang du document considéré dans le classement de ces n documents. Le document classé premier a donc une pertinence évaluée à 1, et le dernier une pertinence évaluée à 0. L'évaluation de pertinence attribuée à chaque document lors de la consolidation correspond alors à la moyenne des rangs attribués par les agents ayant fournis ce document.

Les résultats fournis par les différents agents sont présentés ainsi à l'utilisateur afin que celui-ci puisse donner son estimation sur l'intérêt de chaque document vis à

---

[2] Il serait possible d'utiliser une collection de documents issus de la « filtering track » de TREC, en initialisant la bibliothèque personnelle et le contexte avec une partie des documents pertinents puis en comparant AIRA à une simple indexation. Cette approche a l'inconvénient de ne pas refléter les conditions réelles d'utilisation.

[3] r varie de 0 à n-1. L'évaluation de pertinence du premier document est donc 1, celle du dernier 0.

vis de la requête initiale (Figure 7). Cette présentation peut être rendue « anonyme » (les agents qui ont retrouvés le document ne sont pas indiqués).

On enregistre pour chaque requête la liste des documents obtenus et, pour chaque document, l'estimation de pertinence établie par chaque agent et celle fournie par l'utilisateur. La comparaison des notes fournies par les agents avec les évaluations de l'utilisateur permet de classer les différentes configurations en fonction de leur capacité à prévoir la pertinence d'un document pour l'utilisateur.

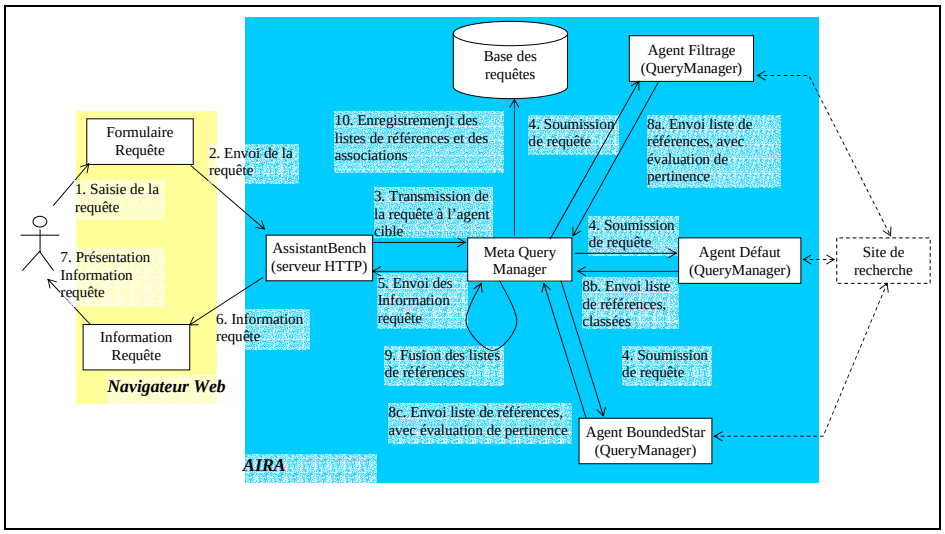

Figure 6. *Soumission d'une requête*

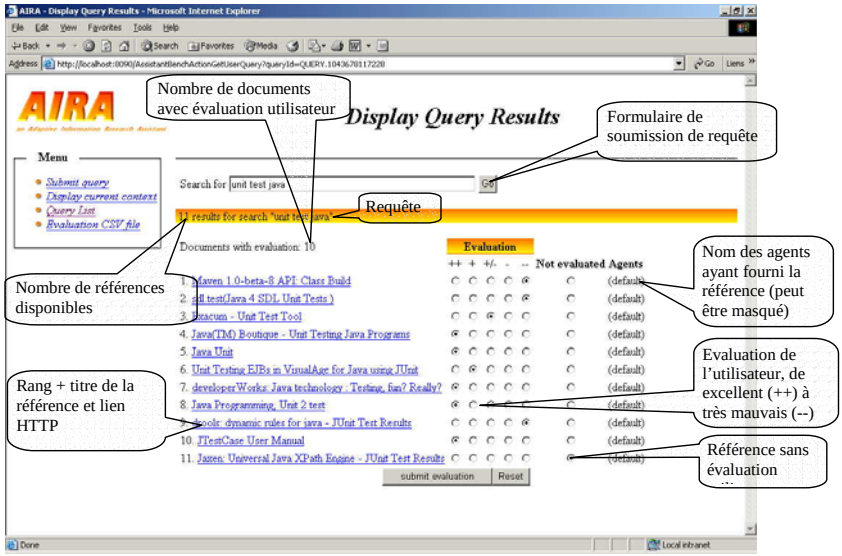

Figure 7. *Interface pour l'évaluation des documents obtenus*

Bien qu'il soit possible d'imaginer de nombreuses façon de comparer les agents, nous avons choisi un critère relativement simple, à l'interprétation aisée, la Qualité des documents: $Q(A_i,R_j)$. Directement lié à la valeur perçue par l'utilisateur des documents obtenus, ce critère correspond à la moyenne des carrés des évaluations de pertinence fournies par l'utilisateur pour les *N* documents d_k sélectionnés par l'agent A_i pour la requête R_j (plus la valeur est haute, meilleur est l'agent):

$$Q(A_i, R_j) = \frac{1}{N} \sum_{k=1 à N} User\_evaluation\ (d_k)^2$$

7. Discussion des résultats

Cette expérimentation a été effectuée avec trois agents : un agent « par défaut » qui fournit le résultat brut ramené par Google, un agent de filtrage « Filtering Weighted» (présenté dans Bottraud et al., 2003), et enfin l'agent « QueryExpansion », qui utilise le mécanisme d'expansion de requête détaillé au paragraphe 5, en association à un filtrage des résultats. Ces deux derniers agents utilisent un contexte construit avec une méthode de sélection avec pondération des nœuds dans le profil. Les paramètres suivants ont été utilisés pour ajuster le fonctionnement de ces agents :

– Agent « Filtering Weighted » : 50 documents ont été accédés, pour n'en conserver que 10 (et donc en rejeter 40).

– Agent « QueryExpansion » : l'étape d'apprentissage des requêtes a été paramétrée de façon à exécuter 2 itérations, en générant 10 requêtes à chaque itération. Les requêtes sont obtenues à partir de la requête initiale, par addition de termes sélectionnés dans le vocabulaire. Les 10 meilleurs documents sont conservés.

– Finalement, l'agent « MetaQueryManager » assemble les 10 meilleurs documents fournis par chacun des agents, agent « Défaut » inclus.

Dans cette expérience, un profil a été construit en identifiant un ensemble de thèmes, puis en identifiant plusieurs références pour chaque thème de façon à obtenir une liste de Favoris (« bookmarks») contenant 130 pointeurs. Un ensemble de requêtes a été ensuite défini et traité. Les résultats obtenus ont été évalués par 3 personnes ayant une expertise variable dans les thèmes correspondant aux requêtes. La généralité des différentes requêtes proposées, définie par le nombre de documents proposés par Google, est inégale, allant de 7630 à 9 750 000 documents.

Environ 600 documents ont été obtenus en réponse aux requêtes (les paramètres initiaux limitaient les résultats à environ 30 documents par requête). Il est important de noter que l'intersection entre les documents qui ont servi à construire le profil et ceux retrouvés par le moteur de recherche est inférieure à 5 %.

Le tableau ci-dessous (cf. Tableau 2) présente les résultats obtenus. Ceux-ci sont encourageants, montrant une amélioration de la qualité perçue pour les agents

utilisant le profil pouvant être de plus de 20 % par rapport à la note de l'agent témoin (« Defaut »). Les meilleurs résultats sont obtenus avec un agent utilisant une stratégie de génération de requêtes permettant d'ajuster la précision de la requête fournie par l'utilisateur. Un examen plus détaillé des résultats montre cependant qu'il existe une variabilité notable dans la qualité observée : les meilleurs résultats obtenus concerne environ deux tiers des requêtes, mais certaines requêtes montrent un renversement des scores. Par ailleurs, l'amélioration des résultats par l'agent « QueryExpansion » est d'autant meilleure que les scores de l'agent « Defaut » sont faibles. En outre, il ne semble pas y avoir de relation entre les scores obtenus par ce dernier (qui reflète l'ambiguïté de la requête initiale) et ceux de l'agent « QueryExpansion ».

Identifiant de l'agent	Qualité observée moyenne
Defaut	0,312516
FilteringWeighted	0,350034
QueryExpansion	0,380849

Tableau 2. *Moyenne des qualités observées*

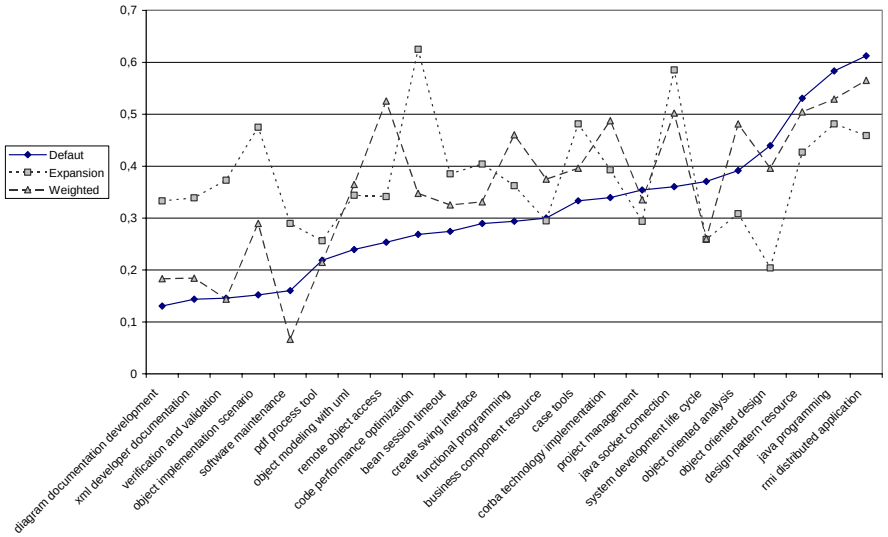

Figure 8. *Comparaison de la qualité des agents de recherche. Note : les requêtes ont été triées selon la qualité évaluée pour l'agent « Deéfaut » de manière à mieux faire ressortir les résultats des différents agents (la régularité de la courbe de l'agent Defaut est donc un artefact du mode de construction)*

Il serait intéressant de connaître la cause des variations observées. Elles pourraient venir a priori soit d'un effet lié au degré de généralité de la requête par rapport au corpus documentaire exploré, soit d'un effet dû à la relation de la requête aux thèmes présents dans le profil. Dans le premier cas, un indicateur de cette généralité pourrait être le nombre de documents associés par Google (ou un autre service de recherche du Web) à cette requête. Dans le second cas, la relation pourrait

être évaluée à partir du nombre de nœuds du profil associés à la requête, ou par la profondeur maximum atteinte dans le profil. Chacun des facteurs pris isolément ne permet cependant pas d'expliquer les variations constatées.

8. Conclusion

L'architecture logicielle mise en œuvre dans le système AIRA repose sur une architecture générale mettant l'accent sur la création et l'assemblage dynamique de composants (les agents), et une construction de ces composants également très ouverte, isolant les algorithmes essentiels de façon à permettre des substitutions aisées. Par ailleurs cette architecture contient l'infrastructure nécessaire à une évaluation comparative de différentes approches, en se focalisant sur les parties du logiciel directement concernées, et en faisant abstraction des autres composants nécessaires à la constitution d'un système opérationnel.

Les résultats présentés ici sur les aspects d'expansion des requêtes à l'aide d'un profil documentaire hiérarchique montrent l'intérêt de cette approche logicielle, en permettant de positionner un nouvel algorithme par rapport à une référence.

L'algorithme d'expansion proposé offre l'avantage de ne pas exiger de l'utilisateur de travail préliminaire (de type entraînement ou relevance feedback). Il améliore globalement les résultats d'un système de filtrage, ou d'un système de recherche standard. Actuellement le plus gros problème se pose lorsque la requête n'est pas ambiguë (le score de l'agent « Defaut » est bon), car on observe une détérioration des scores, ce qui pourrait s'expliquer par des problèmes au niveau des techniques de filtrage (identiques dans les deux agents). Plusieurs approches d'amélioration sont possibles soit au niveau de l'indexation (aujourd'hui faite exclusivement sur des termes isolés), soit au niveau des stratégies de classification de document et de sélection des termes.

9. Références

Armstrong R., Freitag D., Joachims T., Mitchell T., WebWatcher: A Learning Apprentice for the World Wide Web, *AAAI Spring Symposium on Information Gathering from Heterogeneous, Distributed Environments, Stanford*, 1995.

Baeza-Yates R., Ribeiro-Neto B., Modern Information Retrieval, *Addison Wesley*, , 1999

Bauer T., Leake D. B., Real Time User Context Modeling for Information Retrieval Agents, *CIKM '01, Atlanta, Georgia USA, November 5-10,* pp.568-570, 2001

Billsus, D. and Pazzani, M. (1999). A Hybrid User Model for News Story Classification, *Proceedings of the Seventh International Conference on User Modeling (UM '99),* 1999.

Bloedorn, E., Mani, I. and MacMillan, T.R., Machine Learning of User Profiles: Representational Issues, *Proceedings of the Thirteenth National Conference on Artificial Intelligence (AAAI-96),, AAAI/MIT Press,* pp. 433-438, 1996

Boley D., Gini M., Gross R., Han E.-H., Hastings K., Karypis G., Kumar V., Mobasher B., Moore J., *Document Categorization and Query Generation on the World Wide Web Using WebACE, AI Review,* 1998

Bottraud J.C., Bisson G., Bruandet M.F., *Apprentissage des profils pour un agent de recherche d'information, Actes de CAP 2003 (Conférence d'APprentissage), Laval, 1-4 juillet 2003, PUG,* pp31-46, 2003

Bradley J. R., Starner T., Remembrance Agent A continuously running automated information retrieval system, *Procs of the 1st International Conference on The Practical Application Of Intelligent Agents ad Multi Agent Technology (PAAM '96),* pp 487-495, 1996

Buckley C., Saltion G., Allan J., Singhal A., Automatic Query Expansion Using SMART: TREC 3, *in D.K. Harmond Eds, NIST Special Publication 500-225: the First Text Retrieval Conference (TREC 3),* pp 69-80, 1995

Budzik J., Hammond K. J., *User Interaction with Everyday Application as Context for Just-in-time Information Access, IUI 2000, New Orleans LA,* pp.44-51, 2000

Carpineto C., De Mori R., Romano G., Bigi B., An Information-Theoretic Approach to Automatic Query Expansion, *ACM Transactions on Information Systems, Vol. 19, No. 1,* 2001.

Chaffee J., Glauch S., Personal ontologies for web navigation, *Proceedings of the 9th international conference on Information and knowledge management, McLean, Virginia, United States,* 2000

Chen L., Sycara K., WebMate: A Personal Agent for Browsing and Searching, In *Procs of the 2nd International Conference on Autonomous Agents and Multi Agent Systems,* 1998

Claypool M., Phong Le, Wased M., Brown D., Implicit Interest Indicators, *IUI'01, January 14-17, 2001, Santa-Fe, New Mexico, USA,* pp. 33-40, 2001

Edwards P., Bayer D., Green C. L., Payne T. R., Experience with Learning Agents which Manage Internet-Based Information, *AAAI Stanford Spring Symposium on Machine Learning in Information Access,* 1996

Gauch S., Chaffe J., Pretschner A., Ontology-Based User Profiles for Search and Browsing, in J. User Modeling and User-Adapted Interaction: The Journal of Personalization Research , *Special Issue on User Modeling for Web and Hypermedia Information Retrieval,* 2003

Goecks J. , Shavlik J., Learning User's Interests by Unobtrusively Observing their Normal Behavior, *Proceedings of the 2000 International Conference on Intelligent User Interfaces, , ACM,* pp129-132, 2000

Huhns, M. N., Stephens, L. M., Personal Ontologies, *Internet Computing, Vol. 3, No. 5,* pp. 85-87, Sept./Oct. 1999

Krulwich B., Burkey C., The InfoFinder Agent: Learning User Interests through Heuristic Phrase Extraction, *IEEE Expert,* pp 22-27, September/October 1997

Lieberman H., Letizia : An Agent That Assists Web browsing, *Proceedings of the 1995 International Joint Conference on Artificial Intelligence, Montreal Canada,* August 1995

Maglio, P. P., Campbell, C. S., Attentive agents, Communications of the ACM, Volume 46, N° 3, pp. 47-51, March 2003

Maglio, P. P., Barrett, R., Intermediaries personalize information streams, *Communications of the ACM, 43,* N° 8, 2000

Menczer F., ARACHNID: Adaptive Retrieval Agents Choosing Heuristic Neighborhoods for Information Discovery, *Machine Learning: Procs of the 14th International Conference,* 1997

Menczer F., Belew R. K., Adaptive Retrieval Agents: Internalizing Local Context and Scaling up to the Web, *Machine Learning 39 (2-3), Kluwer Academic Publishers*, pp. 203-242, 2000

Moukas A., User Modeling in a MultiAgent Evolving System, Proceedings, *workshop on Machine Learning for User Modeling, 6th Int. Conf. on User Modeling* , 1997

Moukas A., Zacharia G., Evolving a multi-agent information filtering solution in Amalthaea, Proc. of the 1st International Conference on Autonomous Agents, *ACM.*, pp.394-403, 1997

Nick Z. Z., Themis P., Web Search Using a Genetic Algorithm, *IEEE Internet Computing, vol. 5, n° 2*, pp. 18-26, March-April 2001

Pohl W., Nick A., Machine Learning and Knowledge-Based User Modeling in the LaboUr approach, *User Modeling: Proceedings of the 7th International Conference, UM99 , Edited by Judy Kay, Springer Wien New York*, pp. 179-188, 1999

Pazzani M., Muramatsu J., Billsus D., Syskill and Webert: Identifying Interesting Web Sites, *AAAI-96*, pp54-61, 1996

Rocchio J., Relevance feedback information retrieval. *In G. Salton, ed., The Smart Retrieval System-Experiments in Automatic Document Processing, Prentice-Hall*, pp 313-323, 1971

Salton G., The SMART Retrieval System – Experiments in Automatic Document Processing, *Prentice Hall, Inc., Englewood Cliffs, NJ*, 1971

Salton G., Buckley C., Term-Weighting Approaches in Automatic Text Retrieval, 1988, *in Readings in Information Retrieval, , Sparch Jones K., Willet P. eds, Morgan Kaufmann*, 1997

Seo Y.-W., Zhang B.-T., A Reinforcement Learning Agent for Personalized Information Filtering, *Proceedings of the 2000 International Conference on Intelligent User Interfaces, New-Orleans, USA, Jan 9-12, 2000, ACM*, pp248-251, 2000

Shavlik J, Calcari S., Eliassi-Rad T., Solock J., An Instructable, Adaptive Interface for Discovering and Monitoring Information on the World-Wide Web, *Procs of the 1999 International Conference on Intelligent User Interfaces, ACM*, pp257-260, 1999

Sorensen H., O'Riordan A., O'Riordan C., Profiling with the INFOrmer Text Filtering Agent, *Journal of Universal Computer Science, vol. 3, n° 8*, October 1998

Widyantoro D. H., Yin J., Seif El Nasr M., Yang L., Zacchi A, Yen J., Alipes : a Swift Messenger in Cyberspace, *Procs of the Spring Symposium on Intelligents Agents in Cyberspace,* , 1999

XU J., CROFT W. B. 1996. Query expansion using local and global document analysis. *In Procs of the 19th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, H.-P. Frei, D. Harman, P. Schaüble, and R. Wilkinson, Chairs. ACM Press,* pp4–11, 1996

Zamir O., Etzioni O., Grouper : A dynamic clustering interface to Web Search Results, *Computer Networks, 31(11-16)*, pp 1361-1374, 1999