Étude et réalisation d'un système intelligent de recherche d'informations : le prototype IOTA

Bruno Defude

THESE

Présentée à

L'INSTITUT NATIONAL POLYTECHNIQUE
DE GRENOBLE
pour obtenir le grade de
docteur de l'Institut National Polytechnique de Grenoble
(arrêté ministériel du 5 juillet 1984)
"Informatique"

par

Bruno DEFUDE

ETUDE ET REALISATION D'UN SYSTEME INTELLIGENT DE RECHERCHE D'INFORMATIONS : *LE PROTOTYPE IOTA.*

Thèse soutenue le 4 juillet 1986 devant la commission d'examen.

J. MOSSIERE	**Président**
E. CHOURAQUI	
H. GALAIRE	
M. ADIBA	**Examinateurs**
Y. CHIARAMELLA	
J.P. LAURENT	

INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE

Président : Daniel BLOCH
Vice-Présidents : B. BAUDELET
H. CHERADAME
R. CARRE
J.M. PIERRARD

Année universitaire 1984-1985

Professeurs des Universités

E.N.S.E.E.G.

BESSON	Jean	LOUCHET	François
BONNETAIN	Lucien	PARIAUD	Jean-Charles
BONNIER	Etienne	RAMEAU	Jean-Jacques
DURAND	Françis	SOHM	Jean-Claude
GUYOT	Pierre	SOUQUET	Jean-Louis

E.N.S.E.R.G.

BARIBAUD	Michel	GENTY	Pierre
BLIMAN	Samuel	GUERIN	Bernard
BUYLE BODIN	Maurice	POUPOT	Christian
CHENEVIER	Pierre	SERMET	Pierre
COHEN	Joseph	ZADWORNY	François
COUMES	André		

E.N.S.I.E.G.

BARRAUD	Alain	JOUBERT	Jean-Claude
BAUDELET	Bernard	JOURDAIN	Geneviève
BLOCH	Daniel	LACOUME	Jean-Louis
BRISSONNEAU	Pierre	LONGEQUEUE	Jean-Pierre
CAVAIGNAC	Jean-François	MASSELOT	Christian
CHARTIER	Germain	MORET	Roger
CHERUY	Arlette	PAUTHENET	René
DURAND	Jean-Louis	PERRET	René
FELICI	Noël	PERRET	Robert
FOULARD	Claude	POLOUJADOFF	Michel
GAUBERT	Claude	SABONNADIERE	Jean-Claude
IVANES	Marcel	SCHLENKER	Claire
JALINIER	Jean-Michel	SCHLENKER	Michel
JAUSSAUD	Pierre		

E.N.S.H.G.

BOIS	Philippe	LESPINARD	Georges
BOUVARD	Maurice	MOREAU	René
LESIEUR	Marcel	PIAU	Jean-Michel

E.N.S.I.M.A.G.

ANCEAU	François	MOSSIERE	Jacques
FONLUPT	Jean	ROBERT	François
LATOMBE	Jean-Claude	SAUCIER	Gabrielle
MAZARE	Guy	VEILLON	Gérard

U.E.R.M.C.P.P.

CHERADAME	Hervé	RENAUD	Maurice
CHIAVERINA	Jean	ROBERT	André
GANDINI	Alessandro	SILVY	Jacques

Professeurs Associés

BLACKWELDER	Ronald	ENSHG
HAYASHI	Hirashi	ENSIEG
PURDY	Gary	ENSEEG

Professeurs à l'Université des Sciences Sociales (Grenoble II)

BOLLIET	Louis
CHATELIN	Françoise

Chercheurs du C.N.R.S.

Directeurs de recherche :

CARRE	René
FRUCHARD	Robert
JORRAND	Philippe
VACHAUD	Georges

Maître de recherche :

ALLIBERT	Michel	JOUD	Jean-Charles
ANSARA	Ibrahim	KAMARINOS	Georges
ARMAND	Michel	KLEITZ	Michel
BINDER	Gilbert	LANDAU	Ioan-Dore
BORNARD	Guy	LASJAUNIAS	Jean-Claude
DAVID	René	MERMET	Jean
DESPORTES	Jacques	MUNIER	Jacques
DRIOLE	Jean	PIAU	Monique
GIGNOUX	Damien	PORTESEIL	Jean-Louis
GIVORD	Dominique	THOLENCE	Jean-Louis
GUELIN	Pierre	VERDILLON	André
HOPFINGER	Emile	SUERY	Michel

Personnalités habilitées à diriger des travaux de recherche
(Décision du conseil scientifique)

E.N.S.E.E.G.

ALLIBERT	Colette	HAMMOU	Abdelkader
BERNARD	Claude	MALMEJAC	Yves (CENG)
BONNET	Roland	MARTIN GARIN	Régina
CAILLET	Marcel	NGUYEN TRUONG	Bernadette
CHATILLON	Catherine	RAVAINE	Denis
CHATILLON	Christian	SAINFORT	(CENG)
COULON	Michel	SARRAZIN	Pierre
DIARD	Jean-Paul	SIMON	Jean-Paul
EUSTATHOPOULOS	Nicolas	TOUZAIN	Philippe
FOSTER	Panayotis	URBAIN	Georges(ODEILLO)
GALERIE	Alain		

E.N.S.E.R.G.

BARIBAUD	Michel	DOLMAZON	Jean-Marc
BOREL	Joseph	HERAULT	Jeanny
CHOVET	Alain	MONLLOR	Christian
CHEHIKIAN	Alain		

E.N.S.I.E.G.

BORNARD	Guy	LEJEUNE	Gérard
DESCHIZEAUX	Pierre	MAZUER	Jean
GLANGEAUD	François	PERARD	Jacques
KOFMAN	Walter	REINISCH	Raymond

E.N.S.H.G.

ALEMANY	Antoine	OBLED	Charles
BOIS	Daniel	ROWE	Alain
DARVE	Félix	VAUCLIN	Michel
MICHEL	Jean-Marie	WACK	Bernard

E.N.S.I.M.A.G.

BERT	Didier	DELLA DORA	Jean
CALMET	Jacques	FONLUPT	Jean
COURTIN	Jacques	SIFAKIS	Joseph
COURTOIS	Bernard		

U.E.R.M.C.P.P.

CHARUEL	Robert

C.E.N.G.

CADET	Jean	NIFENECKER	Hervé
COEURE	Philippe (LETI)	PERROUD	Paul
DELHAYE	Jean-Marc (STT)	PEUZIN	Jean-Claude(LETI)
DUPUY	Michel (LETI)	TAIEB	Maurice
JOUVE	Hubert (LETI)	VINCENDON	Marc
NICOLAU	Yvan (LETI)		

Laboratoires extérieurs

C.N.E.T.

DEMOULIN	Eric	MERCKEL	Gérard
DEVINE	R.A.B.	PAULEAU	Yves
GERBER	Roland		

I.N.S.A. Lyon

GAUBERT	C.

****

ECOLE NATIONALE SUPERIEURE DES MINES DE SAINT-ETIENNE

Directeur	: M. M. MERMET
Directeur des Etudes et de la formation	: M. J. LEVASSEUR
Directeur des Recherches	: M. J. LEVY
Secrétaire Général	: Mle M. CLERGUE

Professeurs de la 1ère Catégorie

COINDE	Alexandre	Gestion
GOUX	Claude	Metallurgie
LEVY	Jacques	Métallurgie
LOWYS	Jean-Pierre	Physique
MATHON	Albert	Gestion
RIEU	Jean	Mécanique-Résistance des matériaux
SOUSTELLE	Michel	Chimie
FORMERY	Philippe	Mathématiques Appliquées

Professeurs de 2ème catégorie

HABIB	Michel	Informatique
PERRIN	Michel	Géologie
VERCHERY	Georges	Matériaux
TOUCHARD	Bernard	Physique industrielle

Directeur de recherche

LESBATS	Pierre	Métallurgie

Maîtres de recherche

BISCONDI	Michel	Métallurgie
DAVOINE	Philippe	Géologie
FOURDEUX	Angeline	Métallurgie
KOBYLANSKI	André	Métallurgie
LALAUZE	René	Chimie
LANCELOT	Francis	Chimie
LE COZE	Jean	Métallurgie
THEVENOT	François	Chimie
TRAN MINH	Canh	Chimie

Personnalités habilitées à diriger des travaux de recherche

DRIVER	Julian	Métallurgie
GUILHOT	Bernard	Chimie
THOMAS	Gérard	Chimie

Professeur à l'UER de Sciences de Saint-Etienne

VERGNAUD	Jean-Maurice	Chimie des Matériaux & chimie industrielle

****

Je tiens à remercier,

Monsieur J.Mossière, Professeur à l'I.N.P.G, pour l'honneur qu'il me fait de bien vouloir accepter de présider le jury de soutenance,

Monsieur E.Chouraqui, Maitre de Recherches au C.N.R.S, qui a bien voulu s'intéresser à ce travail et a accepté d'en être le rapporteur,

Monsieur H.Gallaire, Directeur de l'E.C.R.C, qui a accepté de rapporter sur ce travail. Qu'il trouve ici toute ma reconnaissance et ma gratitude pour les nombreux efforts qu'il a fait pour améliorer le manuscrit. Ces critiques et remarques, toujours intéressantes, ont grandement contribué à la qualité de ce travail,

Monsieur M.Adiba, Professeur à l'U.S.M.T.G, qui fut il y a quelques années un de mes enseignants distingués à la M.I.A.G, d'avoir accepté de participer au jury,

Monsieur Y.Chiaramella, Professeur à l'U.S.M.T.G, qui a dirigé ce travail. Qu'il trouve ici toute ma reconnaissance pour le temps important qu'il m'a consacré, et pour ses efforts patients de compréhension et d'approfondissement de ce travail,

Monsieur J.P.Laurent, Professeur à l'Université de Savoie, qui m'a fait l'honneur de participer à ce jury,

Madame F.Renzetti, documentaliste à l'I.M.A.G, qui a accepté de jouer le rôle d'expert et ce avec beaucoup de dynamisme et d'intérêt. Je profite de l'occasion pour la remercier de tout le travail qui a pu être accompli avec son aide,

Madame M.F.Bruandet, Maitre de Conférences à l'U.S.S.G, qui a grandement participé à ce travail, grâce à de nombreuses discussions et expérimentations sur IOTA. Je la remercie également d'avoir su me supporter pendant quatre longues années,

Je remercie également A.Landelle qui fut à l'origine de ce travail et avec qui j'ai toujours eu plaisir à collaborer, ainsi que D.Kerkouba et P.Palmer pour leur bonne humeur et les nombreuses discussions et échanges fructueux, et enfin tous les membres du groupe "Systèmes Intelligents de Recherche d'Informations" pour l'ambiance et la dynamique qui y règnent,

Je termine cette longue liste de remerciements par celui qui m'initia à l'informatique il y a quelques années, et qui m'a permis de poursuivre dans cette voie. A cette occasion, je suis très heureux d'exprimer toute ma reconnaissance à M.Dang en lui dédiant cette thèse.

TABLE DES MATIERES

CHAPITRE I

INTRODUCTION

1. Le problème de la recherche d'informations

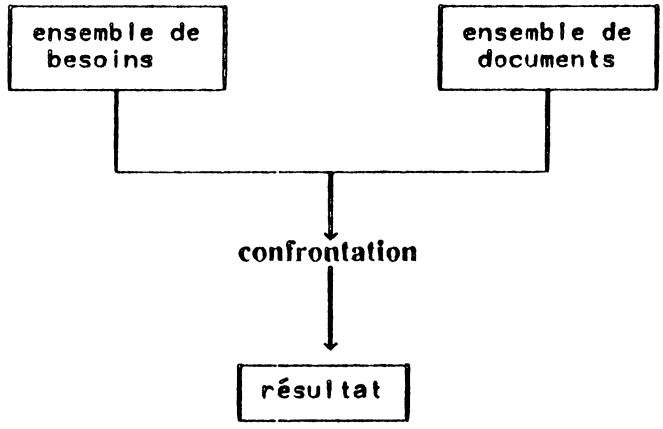
Un Système de Recherche d'Informations (SRI) est composé d'un ensemble de fonctions qui permettent de gérer et de manipuler des documents regroupés dans une base. Cette base constitue un corpus relatif à un thème particulier ou à un ensemble de thèmes donnés. La recherche d'informations (hérité de l'anglais "information retrieval") est la confrontation des besoins d'un usager à ce corpus afin d'en extraire, le ou les documents correspondant à ces besoins.

La notion de document est ici à prendre au sens large, car bien qu'initialement on ait développé cette technique pour des textes écrits, on tente actuellement de généraliser les méthodes utilisées (dans leur plus grande part) à des collections d'informations non structurées ou faiblement structurées (image, son , ...).

La notion de besoins est également large, ces besoins pouvant porter, soit sur la définition externe du document (à savoir pour un texte son auteur, son titre , ...), soit sur sa définition interne (à savoir son contenu sémantique). On considère que les besoins sont exprimés sous la forme d'une suite de critères de recherche.

La confrontation entre les besoins et les documents consiste en la vérification des critères de recherche à l'intérieur de chaque document. Cette vérification peut-être un processus simple (existence d'une chaine de caractères dans le document par exemple), ou bien un processus beaucoup plus compliqué (équivalence sémantique entre le critère et le contenu du document). De plus, ce processus n'est pas nécessairement binaire (document correspondant à la requête ou non) et il faut dans ce cas évaluer la pertinence que l'on attache à ce résultat (c'est à dire la "proximité" estimée entre le document et les besoins). Cette particularité représente une des différences majeures avec l'approche des bases de données classiques où le processus de vérification est toujours binaire (recherche de n-uplets vérifiant certains critères par rapport à un modèle sémantique prédéfini et relativement figé).

Schématiquement, on peut donc représenter le problème de la recherche d'informations comme suit :



L'usager formule ses besoins à travers une requête exprimée dans un certain formalisme qui se situe sur le spectre entre un formalisme libre (langue naturelle) et un formalisme très rigide (langage d'interrogation).

La confrontation consiste en la sélection de l'ensemble des documents correspondant aux besoins exprimés dans la requête. La sélection d'un document susceptible de satisfaire la requête devrait se faire (idéalement) en deux temps :

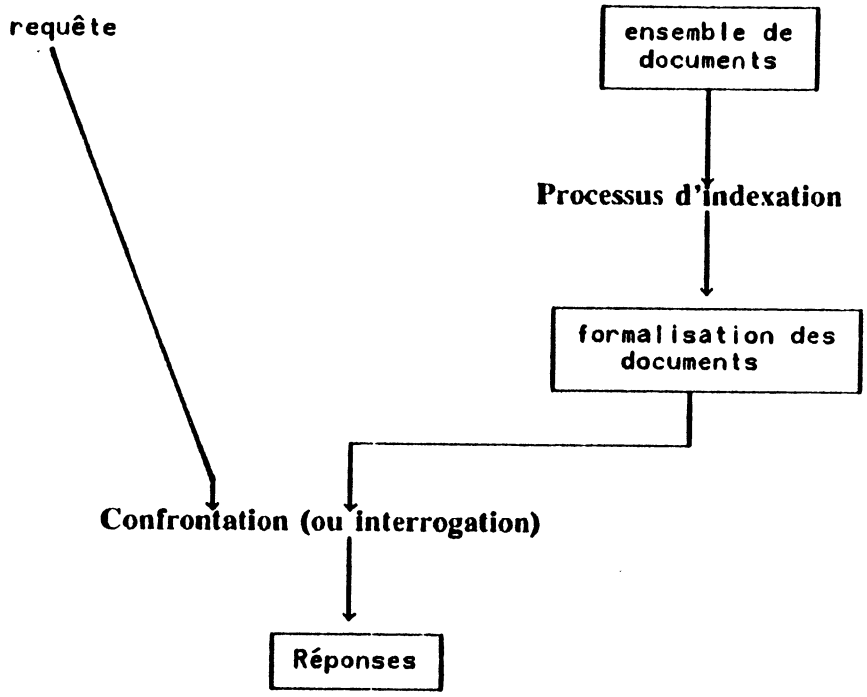
- extraire toute la connaissance exprimée dans le document,
- examiner si ces connaissances correspondent à la requête.

Il est bien évident que ce processus de sélection n'est pas envisageable pour des raisons à la fois d'efficacité (temps et coûts de traitement inacceptables), et à la fois techniques (il est difficile d'extraire toute la connaissance d'un document, et encore plus de la comparer avec une requête quelconque). Pour ces raisons, il est nécessaire d'introduire une étape préalable de traitement des documents (appelée processus d'indexation) qui a donc pour but d'améliorer l'efficacité en diminuant le volume du document et de permettre une comparaison plus aisée en faisant un travail de normalisation et de classification des concepts véhiculés dans l'ensemble des documents.

Ce processus est difficile à définir car on ne sait pas ce qu'est le contenu d'un document. En fonction de la perception que l'on a, ce contenu est variable et il est impossible de l'extraire dans l'absolu. L'indexation consiste en un choix de classes de concepts (substantifs, groupes nominaux, ...) que l'on désigne comme étant les plus porteurs d'information, et qui représentent le niveau de perception des documents. Selon ce modèle, le processus d'indexation normalise et sélectionne des concepts. La forme

indexée est une perception des documents, et par conséquent son contenu sémantique est très restreint par rapport à celui des documents.

Schématiquement, on peut maintenant représenter le SRI comme suit :



De cette analyse, on peut tirer les deux composants principaux des SRI que sont les processus d'indexation et d'interrogation, processus qui interfèrent entre eux fortement.

2. Dualité indexation - interrogation

L'indexation permet de représenter le contenu d'un document, dans une expression équivalente sémantiquement, mais restreinte aux notions les plus caractéristiques du document. Cette restriction permet au processus d'interrogation de ne fournir à l'usager que les documents les plus pertinents relativement à sa requête. L'indexation consiste donc, en l'établissement d'une relation (dite relation d'indexation) entre les documents et l'ensemble des concepts significatifs du domaine considéré.

Le formalisme d'indexation est donc lié au processus d'interprétation d'une requête.

Ce processus d'interprétation peut être décrit comme une correspondance entre la forme indexée des documents, et le contenu sémantique de la requête. Cette correspondance réussit pour un document, si l'expression de la requête est incluse dans l'expression du document ou bien, plus généralement, si elle peut être déduite de l'expression du document.

Cependant, pour que cette correspondance soit possible, il faut que dans l'expression du contenu du document, on ait tenu compte des expressions possibles d'une requête. Cela implique que dans la définition de la relation d'indexation, on tienne compte des propriétés du langage d'interrogation.

De toute façon, l'adéquation réponse - question est problématique et ne peut être qu'estimée.

Cet ensemble de remarques illustre bien la dépendance réciproque indexation - interrogation, ce qui implique que pour la définition de notre système de recherche d'informations, nous avons fait référence à un système d'indexation [KER 84] qui lui même avait été, en partie, conditionné par le système d'interrogation que l'on souhaitait proposer.

3. Les données sémantiques

Les processus d'indexation et d'interrogation ont recours nécessairement à des données sémantiques. Celles ci sont représentées dans ce que l'on appelle classiquement un thésaurus. La richesse de ces données est différente d'un SRI à un autre, mais elles jouent toujours un rôle identique.

3.1. Rôle des données sémantiques

Il faut ici distinguer le rôle classique dans un contexte d'utilisation manuelle, du rôle dans un contexte d'utilisation automatique.

- Rôle classique :

Le thésaurus sert de pivot lors de l'indexation et de l'interrogation :

- il permet la normalisation des concepts lors de l'indexation. Cette normalisation est nécessaire à la fois pour réduire le nombre de concepts sélectionnés, mais aussi pour identifier de façon unique un même concept qui peut apparaitre sous diverses formes.

- il joue également, lors de l'interrogation, le même rôle de normalisation : l'utilisateur doit formuler sa question selon des termes connus du thésaurus.

- Utilisation automatique :

- la normalisation à l'indexation est réalisée par l'accès au thésaurus. Tous les concepts sélectionnés doivent y apparaitre (éventuellement par transformation).

- à l'interrogation, en plus de la normalisation des termes, il permet aussi, grâce à des relations sémantiques, une reformulation de la requête initiale (automatiquement ou non), afin d'obtenir un ensemble de documents plus conforme aux besoins de l'utilisateur.

Cette reformulation peut consister soit en un élargissement (utilisation de relations de généricité), soit en une focalisation ou restriction (utilisation de relations de spécificité), soit en un déplacement (utilisation de relations voir - aussi).

Cette définition de l'utilisation automatique est minimale, et peut être élargie en ne se contentant plus du rôle de normalisateur, mais en en faisant un modèle sémantique qui permet une certaine compréhension des documents en vue de leur indexation, ainsi que des requêtes afin de pouvoir les interpréter. En ce qui nous concerne, nous essayons de mettre en œuvre cette extension (cf V et VI).

3.2. Le thésaurus

Nous présentons la définition d'un thésaurus classiquement employée, de façon à pouvoir répondre aux besoins spécifiés au paragraphe précédent.

Le thésaurus est un ensemble de concepts reliés par des relations sémantiques. Il est représentatif d'un domaine donné, dont il doit être le reflet exhaustif. Les relations sémantiques les plus couramment utilisées sont :

- la généricité (par exemple, arbre est générique de pommier),

- la spécificité (fonction inverse de généricité, par exemple pommier est un spécifique de arbre),
- la synonymie "pure", très rare dans la langue (elle apparait dans les corpus techniques, sous la forme des sigles et de leur définition),
- la synonymie contextuelle, c'est un cas de synonymie dans un contexte sémantique donné,
- le voisin ge sémantique, indique une proximité sémantique entre deux termes (il s'agit d'une généralisation en quelque sorte de la notion de synonymie, où la fonction d'équivalence n'est plus binaire mais floue),
- voir - aussi, désigne les termes sémantiquement connexes d'un terme donné (relation de causalité par exemple).

Ce thésaurus est défini, dans la grande majorité des cas, manuellement par des spécialistes très compétents. Ce travail de construction est très long et il est difficile d'être exhaustif. C'est pourquoi, des outils de construction automatique (ou assistée) sont développés [BRU 85]. Ces outils définissent un thésaurus, non pas sur un domaine analysé ex nihilo, mais à partir d'un ensemble de textes jugés représentatifs du domaine. Ils permettent la sélection des concepts à partir des textes, et effectuent une certaine classification sur ces concepts. La construction des relations sémantiques reste difficile, et la plupart du temps, elle s'effectue manuellement ou de façon assistée.

4. L'évaluation d'un SRI

Le problème de l'évaluation d'un SRI est important, dans la mesure où, comme on vient de le voir, les résultats obtenus ne sont pas certains mais estimés.

L'évaluation d'un SRI n'est pas seulement quantitative (volume de documents que l'on est capable de traiter, et vitesse de traitement), mais elle est également qualitative. En effet, comme nous l'avons déjà évoqué brièvement, le processus de correspondance n'est pas binaire mais continu. Une réponse (un document) est évaluée à travers sa pertinence, qui est censée mesurer la "distance sémantique" entre le document et la requête. Cette notion de pertinence est éminemment subjective et ne peut être qu'estimée à travers des fonctions de calcul fondées sur le modèle de correspondance utilisé.

Un premier critère pour juger de la qualité d'un SRI, consiste donc en sa capacité de fournir des réponses ayant une "bonne" mesure de pertinence.

On peut également essayer de mesurer le résultat global de la requête (ensemble des réponses obtenues), dans ce cas, on utilise deux ratios appelés silence et bruit [SAL 83]. Le silence représente le taux de documents pertinents non retrouvés pour la requête. Le bruit représente le taux de documents non pertinents retrouvés pour la requête.

Un bon système de recherche d'informations est donc un système où les taux de silence et de bruit sont faibles.

Il est donc bien évident que l'on va chercher à diminuer ses deux ratios ; pour ce faire, on peut remédier au silence en effectuant ce que l'on appelle un élargissement de la requête initiale. Cette opération s'effectue lorsque l'on a jugé que le résultat obtenu est insuffisant (c'est l'utilisateur qui juge dans la plupart des cas), et permet d'atteindre un plus grand nombre de documents. Pour cet élargissement, on utilise généralement les relations sémantiques telles que synonymie, généricité , ... par rapport aux concepts initiaux de la requête. Le problème est que cette opération risque d'introduire du bruit (élargissement excessif).

Inversement, si l'on veut remédier au bruit, il faut restreindre la requête (toujours par les environnements sémantiques des concepts initiaux). Cette restriction va donc bien diminuer le nombre de documents non pertinents, mais elle risque également d'augmenter le silence (restriction excessive).

Le problème lié aux taux de bruit et de silence est difficile à résoudre ; si l'on veut augmenter la qualité d'un résultat, il faut utiliser avec prudence l'élargissement et la restriction, sous peine d'obtenir des résultats de mauvaise qualité.

5. Un système intelligent de recherche d'informations

Le travail présenté dans cette thèse concerne la spécification et la réalisation de la fonction d'interrogation d'un système intelligent de recherche d'informations.

De nombreux SRI ont été développés depuis les années 1970, systèmes soit dérivés des modèles de recherche d'informations [SAL 85], soit dérivés des modèles utilisés dans les SGBD (plus récemment). Dans le chapitre II nous présentons les caractéristiques générales de cette génération de systèmes, constituant la très grande part des produits actuellement industrialisés. Nous y soulignons leurs performances, mais aussi leurs

limitations qui nous paraissent principalement liées à la non prise en compte de traitements sémantiques dans la définition des fonctions de correspondance. Nous insistons plus particulièrement dans ce chapitre sur le système SMART qui, bien qu'expérimental, est probablement l'un des systèmes les plus avancés de ce type; à ce titre, il est bien représentatif des limites escomptables pour un SRI faisant abstraction de tout traitement sémantique.

Notre travail consiste ensuite à effectuer une analyse précise de deux approches qui sont, à notre sens, les plus prometteuses pour l'affinement de ces méthodes (approche Base de Données, et approche Intelligence Artificielle); il s'agit là d'entrer dans un débat entamé depuis quelques années, et auquel nous tentons d'apporter notre contribution. Il faut noter en effet que si l'approche Base de Données est contestée (voir ci-dessous), l'approche Intelligence Artificielle l'est tout autant vis à vis des tenants des modèles classiques. On lui reproche non seulement, et de manière classique, les chutes prévisibles des performances qualitatives, mais, plus profondément, de ne pas être réaliste, et même de n'être pas forcément d'un grand apport qualitatif [SAL 83]. Un de nos objectifs essentiels dans ce travail est donc de contrebattre ces assertions; nous le faisons à deux niveaux :

- sur le plan théorique, en montrant que les modèles avancés développés dans les domaines des Bases de Données et de l'Intelligence Artificielle sont effectivement susceptibles d'apporter des améliorations très substancielles aux SRI; cet aspect est développé dans les chapitres III et IV,
- sur le plan pratique, en construisant effectivement un tel système et en l'expérimentant. Les résultats sont présentés dans les chapitres V, VI, VII.

Nous considérons dans ce travail essentiellement les apports qualitatifs aux SRI, qui sont, dans nos propositions, liés aux techniques de l'IA. Les aspects liés aux performances quantitatives sont beaucoup plus liés aux techniques relevant des Bases de Données; nous ne ferons que les aborder dans la mesure où nous n'avons pas, dans le cadre de notre étude, ni le temps ni la possibilité matérielle de les approfondir comme ils auraient mérité de l'être.

Le chapitre III concerne tout d'abord l'approche Base de Données; une critique courante des spécialistes de recherche d'informations vis à vis de cette approche, tient essentiellement à l'insuffisance des modèles actuellement implantés. Si cette critique est, dans une large mesure, fondée, nous montrons que les récents développements dans ce domaine sont assez proches de nos préoccupations (en particulier les extensions des modèles relationnels, entité - association vers ce qui est appelé les SGBD généralisés, et également l'émergence des Bases de Données déductives).

Le chapitre IV traite ensuite de l'approche Intelligence Artificielle. L'extraction, la représentation et le traitement des connaissances constituent le cœur de ce domaine, et il était naturel d'opérer dans un premier temps une analyse précise des outils et méthodes développés dans ce contexte pour évaluer leur apport potentiel et aborder dans un deuxième temps la spécification d'un système intelligent de recherche d'informations (SIRI). Ce travail de spécification est exposé dans la seconde partie du chapitre IV.

Le chapitre V développe en détails l'organisation du système prototype IOTA, dont les objectifs essentiels visent à l'amélioration de la convivialité des interfaces d'interrogation, et celle des résultats qualitatifs au niveau des réponses obtenues. Pour répondre à ces objectifs, nous avons utilisé l'approche intelligence artificielle, et plus particulièrement celle des systèmes experts. Cette approche nous semble intéressante, dans la mesure où elle nous offre des outils d'analyse et de réalisations assez puissants (notamment en ce qui concerne les mécanismes déductifs et d'apprentissage).

Par ailleurs, un tel système doit s'appuyer sur une méthode d'indexation évoluée, nous faisons ici référence à un système d'indexation automatique développé par D.KERKOUBA [KER 84] et qui présente un certain nombre de caractéristiques intéressantes qui seront rappelées en temps utile.

Nous avons choisi une architecture logicielle permettant d'intégrer les nouvelles fonctionnalités voulues. Cette architecture doit pouvoir gérer les différents composants d'un SIRI, qu'ils soient procéduraux (comme l'analyse syntaxique) ou de nature experte (comme l'évaluation d'une réponse). L'architecture choisie est celle d'un système expert étendu, ce qui présente l'avantage d'une gestion intégrée de toutes les activités du SIRI qu'elles soient de nature experte ou non.

Le chapitre VI présente la définition des connaissances expertes du SIRI, qui peut être comprise comme la modélisation du comportement d'un expert humain en recherche d'informations. Il s'agit là d'une première définition, encore expérimentale, destinée surtout à montrer la faisabilité d'une telle modélisation au travers de règles de production.

Enfin, le chapitre VII montre divers résultats expérimentaux, fondés sur l'exploitation d'une base textuelle constituée par un corpus technique (les NEF, Normes d'Exploitation et de Fonctionnement, utilisées pour la conception des autocommutateurs par le CNET).

CHAPITRE II

LES SYSTEMES D'INTERROGATION CLASSIQUES

1. Préambule

Le processus d'interrogation peut se décomposer en trois phases :

- analyse de la requête : consiste à extraire les éléments de la requête nécessaires à la recherche. Ces éléments sont des critères de recherche (concepts ou attributs externes comme titre, ...), et des opérateurs de recherche (si le langage d'interrogation le permet).
- interprétation de la requête : une fois la requête analysée, on recherche tous les documents contenus dans le corpus susceptibles de correspondre à celle ci. La correspondance requête - document est jugée à travers des fonctions de correspondance fondée sur un modèle de recherche associé.
- évaluation du résultat : elle est fondée sur le modèle de recherche employé et peut être dans certains systèmes suivie d'un processus de reformulation ayant pour but un raffinement de la requête pour mieux l'adapter aux besoins de l'utilisateur. Classiquement, cette reformulation est de deux types, élargissement (le résultat a été évalué comme trop précis), focalisation (le résultat a été jugé trop large).

Un système d'interrogation peut être caractérisé par :

- le système d'indexation associé (comme nous l'avons vu précédemment),
- le langage d'interrogation accepté,
- la fonction de correspondance adoptée,
- le processus d'évaluation, et éventuellement de reformulation associé.

Nous allons maintenant, dans un premier temps présenter les différents modèles de correspondance classiques. Nous étudions ensuite le fonctionnement de divers systèmes d'interrogation existants et nous essayons enfin de situer les SRI par rapport aux techniques employées dans les bases de données.

2. Les différents modèles de correspondance

Les différentes fonctions de correspondance utilisées dans les systèmes existants (commercialisés ou à l'état de prototypes) sont fondées sur des modèles théoriques. Aucun de ces modèles n'étant totalement satisfaisant au point de vue des résultats, ou au point de vue opératoire, les fonctions de correspondance utilisées reflètent une combinaison de ces modèles.

Les modèles les plus connus dans les SRI sont [SAL 85] :

- le modèle vectoriel,
- le modèle probabiliste,
- le modèle booléen,
- le modèle linguistique.

2.1. Le modèle vectoriel

C'est le modèle utilisé dans le prototype SMART (cf section 3). Les documents et les requêtes sont représentés sous forme d'un vecteur de concepts, le processus de correspondance consiste à mesurer la proximité entre deux vecteurs. Ce modèle a l'avantage d'être simple conceptuellement, mais a comme hypothèse sous jacente, l'indépendance des concepts pris deux à deux, ce qui ne permet pas de prendre en compte les relations sémantiques entre concepts. Ce modèle est de plus purement statistique et ne fait pas intervenir de sémantique.

2.2. Le modèle probabiliste

L'idée sous jacente à ce modèle, est qu'un document et une requête correspondent si :

- la probabilité P(per | D) que le document D soit pertinent est grande,
- la probabilité P(nonper | D) que le document D soit non pertinent est petite.

En fonction de ces critères, les documents peuvent être classifiés pour une requête donnée, dans l'ordre décroissant de la fonction d'estimation G, définie par :

$$G(D) = (Pr(D \mid per) \cdot Pr(per)) / (Pr(D \mid nonper) \cdot Pr(nonper))$$

où Pr(per) et Pr(nonper) sont les probabilités de pertinence et de non pertinence d'un document D. Pr(D | per) et Pr(D | nonper) sont les probabilités de D d'apparaitre dans les ensembles de documents pertinents et non pertinents respectivement. Cette fonction est malheureusement non opératoire car Pr(D | per) et Pr(D | nonper) sont inconnus.

De nombreuses fonctions d'estimation de ces deux probabilités existent fondés sur des hypothèses différentes (indexation binaire, indépendance des concepts, ...). Un certain nombre d'extensions existent, concernant une pondération continue de la relation d'indexation, la prise en compte des relations entre termes, ...

Ces modèles restent de toute façon purement probabilistes et ne tiennent pas compte de la sémantique des documents.

2.3. Le modèle booléen

C'est le modèle utilisé dans les systèmes classiques (type fichiers inverses). Ce modèle permet d'inclure deux types de relation entre termes, identifiées par les opérateurs booléens OU et ET :

- la relation OU relie des termes synonymes ou quasi-synonymes,
- la relation ET relie des composants d'une phrase.

Ces deux relations donnent un large pouvoir d'expression au langage d'interrogation (le OU donne un effet d'élargissement, le ET un effet de focalisation).

Une requête booléenne R et un document D étant donnés, la correspondance entre R et D peut être évaluée de la façon suivante :

- chaque terme r_i de la requête est remplacé par la fonction d'évaluation $F(D,r_i)$ qui est calculée lors du processus d'indexation,
- la correspondance est calculée à travers les tables de vérité appropriées des opérations booléennes définissant la requête.

Ces tables de vérité sont fonction des domaines de valeur de $F(D,r_i)$:

- si $F(D,r_i) \in (0,1)$, on retrouve les fonctions booléennes classiques,
- si $F(D,r_i) \in [0,1]$, ce qui correspond à une pondération de la relation d'indexation continue, on applique généralement des opérateurs tirés de la théorie des sous-ensembles flous [KAU 75].

Cette dernière extension permet de mieux prendre en compte la logique inhérente à la requête. En effet, dans le modèle booléen classique, pour une requête ne comprenant que des OU, le fait qu'un document contienne tous les termes ou un seul n'est pas pris en compte. Symétriquement, si une requête ne contient que des opérateurs ET, un document est rejeté quelque soit le nombre de termes non présents (un ou tous). Cette extension permet l'évaluation de tous les documents relativement à la requête, et donc leur classement dans l'ordre décroissant de pertinence.

Le modèle a été également étendu, pour tenir compte à la fois des termes pondérés dans les documents (extension précédente), ainsi que des termes pondérés dans la requête. Cette extension permet à l'utilisateur de raffiner encore davantage sa requête, en hiérarchisant la suite de termes d'interrogation.

2.4. Les modèles linguistiques

Les modèles précédents présentent l'inconvénient d'être basés complètement sur les probabilités ou les statistiques et de ne faire aucun appel à la sémantique induite des documents et des requêtes. De plus, les langages d'interrogation associés à ces modèles, sont peu naturels et difficiles d'accès pour des utilisateurs non spécialistes. C'est pourquoi, l'introduction de modèles linguistiques semble intéressante, car elle permet :

- la définition de langages d'interrogation quasi-naturels,
- une plus grande richesse de représentation des concepts (des morceaux de phrase au lieu des mots clés habituels),
- la possibilité de fournir en résultat, non pas des références de documents, mais la ou les parties de document répondant à la requête (on se rapproche là des systèmes de question - réponses). Cette possibilité implique une analyse fine des requêtes et des documents, tant au niveau syntaxique que sémantique, ainsi que l'utilisation d'une fonction de correspondance permettant la mise en évidence d'équivalence sémantique entre structures syntaxiques.

Ces modèles sont néanmoins difficiles à mettre en œuvre, car ils impliquent l'utilisation :

- d'un analyseur morpho-syntaxique de la langue naturelle,
- d'un modèle sémantique du domaine considéré, permettant la levée d'ambiguités syntaxiques, la compréhension des requêtes et des documents afin de pouvoir les comparer,
- d'une fonction de correspondance entre structures syntaxiques.

Ces objectifs sont envisageables dans le cadre des systèmes experts, où le domaine est, par définition, très limité, mais le sont beaucoup moins dans le cadre des SRI où le domaine est très ouvert et la plupart du temps très large. L'utilisation de modèles linguistiques purs dans les SRI est donc, pour l'instant, utopique et les modèles implantés actuellement sont des modèles mixtes linguistique et statistiques, ou bien restreignent fortement le domaine considéré. La difficulté est de trouver le moyen terme entre les deux modèles et donc de savoir quel niveau de sémantique est obligatoire, et quels renseignements d'ordre sémantique on peut tirer de données statistiques (cf IV-2).

3. Les systèmes basés sur les fichiers inverses

Ce sont les systèmes les plus connus et les plus utilisés sur le plan commercial. Toutes les grosses banques de données existantes utilisent cette technique qui présente l'avantage de supporter des documents très nombreux (quelques millions). Les systèmes les plus connus sont DIALOG (Loockeed), STAIRS (IBM), MEDLARS, ORBIT, MISTRAL (CMB). Nous allons maintenant détailler leurs principales caractéristiques.

3.1. Système d'indexation utilisé

Il s'agit la plupart du temps d'une indexation manuelle, faite presque exclusivement par mots clés, établie par un ensemble d'experts du domaine. Cette indexation est souvent de qualité très moyenne surtout en ce qui concerne la cohérence. En effet, le même concept peut être identifié de deux façons différentes par deux experts différents. Une manière de remédier à ce problème consiste en l'établissement d'un dictionnaire d'indexation recensant l'ensemble des concepts de référence. Malgré cela, il est difficile de garder la cohérence de l'indexation sur des corpus de plusieurs millions de documents traités séparément par un grand nombre d'experts. Certains de ces systèmes (MISTRAL, PASSAT de SIEMENS par exemple) offrent des possibilités d'indexation automatique fondées sur des dictionnaires d'indexation et des calculs fréquentiels.

Un document est donc identifié par un ensemble de concepts. Le stockage de la relation d'indexation est effectué au moyen de fichiers inverses, qui représentent une matrice où les lignes sont les documents (la ligne i désigne le i-eme document de la base), et les colonnes les concepts (la colonne j désigne le j-eme concept du domaine). Un 1 en position (i,j) signifie donc que le concept j indexe le document i, alors qu'un 0 dans cette même position signifie que j n'indexe pas i.

3.2. Le langage d'interrogation

Ces systèmes font appel à des langages d'interrogation figés, spécifiques (ce qui empêche la propagation d'une même requête sur un ensemble de systèmes, et rend donc leur utilisation moins conviviale car il faut se former à une grande variété de langages). Formellement, on peut décrire ces langages d'interrogation de la façon suivante :

REQUETE = CRITERE1 (OPRECH CRITERE2)*

CRITERE = $TERME_i$ / $ATTRIBUT_i$ OPREL CONSTANTE

où $TERME_i$ est un concept (mot clé),

$ATTRIBUT_i$ un attribut externe (titre, auteur, ...),

OPREL un opérateur relationnel classique (égalité, <, >, ...),

CONSTANTE une constante (chaine de caractères, entiers, ...),

et enfin OPRECH un opérateur de recherche.

Ces opérateurs de recherche sont la plupart du temps les opérateurs booléens classiques (ET, OU, SAUF), qui permettent de sélectionner les documents combinant un certain nombre de critères.

La plupart des systèmes ne se limitent pas à ces opérateurs ; on dispose fréquemment des opérateurs d'adjacence et de troncature.

L'opérateur d'adjacence contrôle la proximité dans un même document de deux concepts. Il est donc plus fort que l'opérateur ET, qui se contente de vérifier la présence dans le même document de deux concepts, quelle que soit leur position relative. Les réponses obtenues avec cet opérateur sont donc meilleures que celles obtenues avec le ET car il diminue le bruit (le traitement de cet opérateur est cependant très couteux, car il nécessite une analyse du texte). Cet opérateur constitue une approche des critères syntaxiques, il permet de rechercher des structures syntaxiques plus riches que le mot isolé (du type mot composé).

Exemple : on veut chercher les documents traitant d'informatique de gestion, il est préférable d'établir la requête :

informatique ADJ gestion
plutôt que :
informatique ET gestion
où ADJ représente l'opérateur d'adjacence

L'opérateur de troncature, permet de désigner non pas un seul concept, mais une famille de concepts identifiée par une racine commune. Cet opérateur diminue donc le silence en augmentant le nombre de concepts recherchés, il peut cependant générer un bruit important et il importe de l'utiliser avec précaution. Il permet donc de tenir compte à l'interrogation de problèmes de cohérence dans l'indexation et de l'inexistence d'une organisation des concepts.

Exemple :
inform* représente informatique, information, informationnel, informatisation, informel, ...

3.3. La fonction de correspondance

La fonction utilisée est très simple puisqu'elle consiste en une correspondance exacte : on recherche l'ensemble des documents vérifiant tous les critères de recherche. Ce processus consiste à filtrer les documents sur les attributs externes, puis à appliquer sur les documents sélectionnés les critères de contenu (ceci est effectué par des opérations simples sur les chaines de bits représentant le contenu des documents, qui sont très peu coûteuses en temps).

La correspondance étant exacte, il n'y a pas d'évaluation du résultat, ni par conséquent d'outils de reformulation automatique. La seule aide dans ce cas, est la consultation en ligne du dictionnaire d'indexation qui existe sur certains systèmes.

3.4. Conclusion

Ces systèmes sont précieux, dans la mesure où ils permettent l'accès en ligne à de très grosses masses d'informations, avec des temps de réponse très satisfaisants. Cependant, les performances qualitatives de tels systèmes laissent à désirer, phénomène dû essentiellement à une indexation d'assez mauvaise qualité, ainsi qu'à une fonction de correspondance ne laissant aucune part à la sémantique des documents. Un autre aspect négatif, concerne l'interface d'interrogation constituée d'un langage figé et très peu naturel, qui limite l'accès de ces systèmes aux spécialistes , maitrisant le langage et ayant une bonne connaissance empirique de l'indexation effectuée.

4. Le système expérimental SMART

SMART est un système universitaire développé par G.SALTON [SAL 71] de façon à améliorer la qualité des SRI classiques basés sur les fichiers inverses. Ce système quoique assez ancien au niveau des idées, est toujours d'actualité en ce qui concerne la richesse des innovations employées. Les principaux apports de SMART sont :

- un système d'indexation automatique,
- une classification des documents permettant une optimisation de la recherche,

- une fonction de correspondance basée sur la similarité document - requête, les réponses étant classées dans l'ordre décroissant de similarité,
- un ensemble de procédures de reformulation automatique d'une requête.

4.1. Le système d'indexation

Chaque document i est identifié par un vecteur de termes :

$DOC_i = (terme_i^1, terme_i^2, \ldots, terme_i^t)$

où $terme_i^j$ représente le poids (importance) du concept j dans le document i. Les $terme_i^j$ sont supérieurs ou égaux à 0, (un poids 0 indiquant que le concept n'intervient pas dans le document). Un terme est une identification de concept (mot, groupe nominal, ...).

Une collection de documents est donc représentée par une matrice où les lignes représentent les documents et les colonnes les concepts.

Cette matrice est construite de façon automatique à partir des documents source, à l'aide de méthodes linguistiques et statistiques.

4.2. Le langage d'interrogation

Une requête j est identifiée également par un vecteur $(qterme_j^1, qterme_j^2, \ldots, qterme_j^t)$ où $qterme_j^i$ représente le poids du terme i dans la requête j. Ce vecteur est construit automatiquement à partir de la requête, avec les mêmes outils que ceux utilisés pour les documents.

4.3. La fonction de correspondance

La fonction de correspondance adoptée n'est pas exacte. Au lieu de choisir une méthode donnant la correspondance document - requête, si tous les poids non nuls du document et de la requête coincident, il a été choisi une mesure de similarité estimant la proximité entre un vecteur de document et un vecteur de requête. Une mesure souvent utilisée pour SMART est :

$$\cos(doc_i, req_j) = \sum_k (terme_i^k * qterme_j^k) / \sqrt{\sum_k (terme_i^k)^2 * \sum_k (qterme_i^k)^2}$$

Cette mesure peut s'interpréter comme le cosinus de l'angle formé par un vecteur de document et un vecteur de requête, dans un espace à t dimensions.

Cette mesure offre l'avantage d'être maximale si tous les termes coincident avec un poids maximum, et d'être minimale si aucun des poids non nuls ne correspondent.

4.4. La classification des documents

Le processus de correspondance de chaque document pour une requête donnée, peut être très couteux si le nombre de documents est élevé. C'est pourquoi il a été introduit une technique de classification automatique des documents afin d'optimiser le nombre de correspondances à effectuer.

Les documents sont classsifiés suivant un critère portant sur les thèmes véhiculés. Tous les documents relatifs (proches) d'un thème particulier sont regroupés en un même ensemble. Ce découpage n'est pas une partition, les ensembles pouvant se recouper.

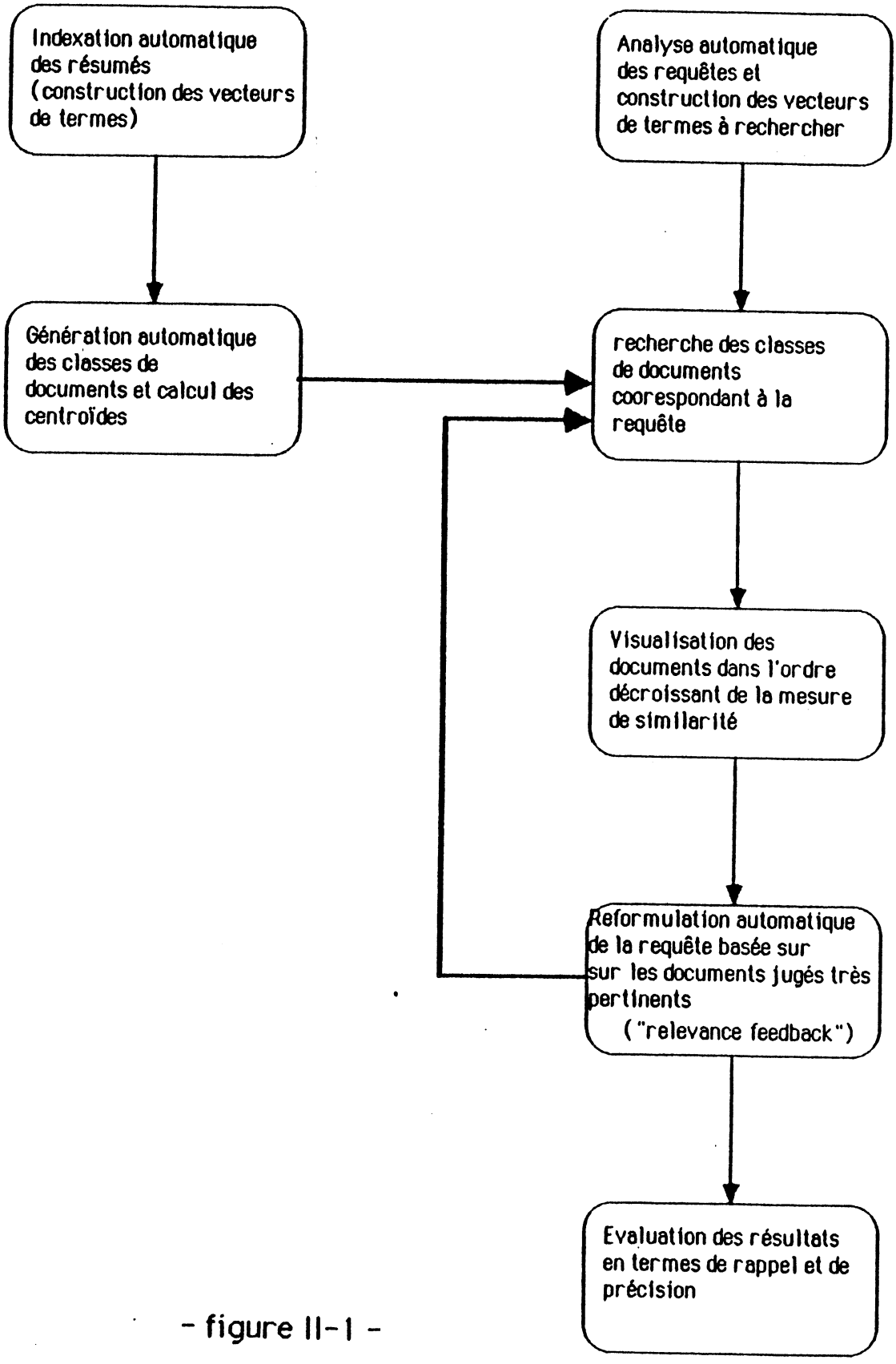
Chaque ensemble est ensuite identifié par un élément fictif appelé centroïde qui est calculé comme le barycentre de l'ensemble (centroïde). Une fois cette classification effectuée, le processus de recherche est le suivant :

- correspondance du vecteur de la requête avec tous les centroïdes, afin de déterminer quels sont les ensembles voisins de la requête,
- correspondance du vecteur de la requête avec tous les éléments des ensembles sélectionnés, ce qui donne la réponse finale.

Cette technique de classification est évidemment généralisée à plusieurs niveaux, ce qui donne une hiérarchie d'ensembles.

4.5. La reformulation automatique

SMART offre un certain nombre de procédures de reformulation automatique des requêtes. Cette reformulation de la requête initiale est basée sur deux opérations complémentaires :



- figure II-1 -

Le fonctionnement de SMART

- les termes jugés pertinents par l'utilisateur dans les documents trouvés initialement, sont rajoutés à ceux de la requête initiale, ou voient leur poids augmenté dans la nouvelle requête.
- les termes jugés non pertinents sont enlevés de la requête initiale, ou bien leur poids est diminué.

Une fois que l'utilisateur a estimé la pertinence des termes, SMART reformule automatiquement la requête initiale et relance le processus d'interrogation.

Le fonctionnement général du système est résumé dans la figure II-1.

4.6. Conclusion sur SMART

SMART introduit un nombre important d'améliorations par rapport aux systèmes classiques, qui sont essentiellement l'indexation automatique, la fonction de similarité et les procédures de reformulation automatique. Il offre de plus une interface beaucoup plus conviviale puisqu'il admet le langage naturel.

Par contre, il ne permet pas l'usage dans la requête d'opérateurs de composition ce qui oblige à faire plusieurs requêtes pour réaliser l'équivalent de l'opérateur OU par exemple. La fonction de correspondance fait néanmoins peu appel à la sémantique, et les procédures de reformulation automatique font fortement intervenir l'utilisateur.

5. Conclusion

Parmi les systèmes actuellement réalisés, même à un niveau aussi poussé que SMART, aucun n'est réellement satisfaisant sur le plan qualitatif. Le principal problème réside dans la non prise en compte de la sémantique des documents, ainsi que dans l'utilisation de langages d'interrogation peu conviviaux.

Tous sont fondés sur des fonctions de correspondance question - document définies à partir de critères purement statistiques ou empiriques. La représentation du contenu sémantique des documents (le langage d'indexation) est très pauvre, de même que les langages d'interrogation qui en découlent; en conséquence, l'usager doit recourir à des artifices complexes (expressions booléennes, opérateurs d'adjacence), et à de multiples opérations successives (reformulations manuelles) pour obtenir des réponses satisfaisantes.

Pour améliorer ces systèmes, l'utilisation de modèles linguistiques et sémantiques semble souhaitable mais est difficile à mettre en œuvre par les méthodes classiques. Le recours à des outils nouveaux parait nécessaire pour produire des résultats intéressants. Nous développons dans les deux chapitres suivants les apports potentiels liés aux Bases de Données tout d'abord, puis à l'Intelligence Artificielle.

CHAPITRE III

SRI ET SGBD

1. Introduction

Les progrès réalisés dans le domaine des Systèmes de Gestion de Bases de Données (SGBD) depuis une dizaine d'années sont très importants tant sur le plan théorique que pratique. De nombreux systèmes commerciaux ont vu le jour (SYSTEM R, INGRES, ...) intégrant un certain nombre de fonctionnalités nouvelles. Leurs principaux apports sont :

- la mise en œuvre du modèle relationnel, qui présente les avantages d'être fondé sur des bases théoriques solides (le concept mathématique de relation), de fournir des langages d'interrogation à fort pouvoir d'expression, et de permettre une mise à jour dynamique des données [DEL 82].
- l'utilisation des systèmes de bases de données dans un grand nombre d'applications où la manipulation des informations est non triviale (bureautique, conception assistée par ordinateur, génie logiciel, ...). A ces nouvelles applications correspondent des besoins qui impliquent l'enrichissement des modèles et langages existants.

Les évolutions actuelles des SGBD, se font principalement sur deux axes :

- prise en compte d'objets de nature diverse, soit faiblement structurés (texte, image, ...), soit fortement structurés (objets classiques des SGBD). Ceci nécessite un modèle de données plus riche permettant la description et la manipulation de façon unifiée de tels objets. Ces nouveaux SGBD sont appelés SGBD généralisés.
- prise en compte de la sémantique des données (du moins en partie), afin de ne pas se contenter de retrouver des données stockées, mais de permettre la production de nouvelles données à partir de celles déjà stockées, et de règles de déduction associées. Ceci tend à faire se rapprocher les SGBD des systèmes de l'intelligence artificielle, en les dotant d'outils déductifs (SGBD déductifs).

Après avoir brièvement rappelé les fondements de ces travaux, nous analyserons dans la section 4 de ce chapitre les apports possibles de ces domaines aux SRI.

2. Les SGBD généralisés

Ils sont fondés sur des modèles de données aptes à saisir toute la structure et toute la sémantique des différents types de données que l'on veut traiter [ADI 85].

Le modèle relationnel est insuffisant pour ces nouveaux objectifs (il ne permet pas notamment la prise en compte de la sémantique). Il semble par ailleurs qu'un modèle généralisé universel soit difficile à mettre en évidence ; c'est pourquoi les modèles dits généralisés sont en fait définis dans le cadre de domaines d'application (CAO, ...).

On rencontre deux grands courants :

- modèles basés sur des extensions du modèle entité - association (TIGRE [VEL 84], BIG [CRA 83]). Ces extensions concernent généralement l'introduction de mécanismes pour définir et manipuler des objets généralisés : champs de longueur arbitraire, objets complexes, nouveaux types de données, ...
- modèles orientés objets, offrant un cadre dynamique pour la déclaration et la construction d'objets au sens large [BEN 85].

A ces modèles sont associés de nouveaux langages de manipulation, en général du niveau de SQL, munis d'interfaces évoluées (graphiques la plupart du temps) [ZLO 77], [BEG 85], [LEM 85].

3. Les SGBD déductifs

3.1. Présentation générale

"Une base de données déductive est une base de données dans laquelle de nouveaux faits peuvent être dérivés de faits qui ont été explicitement introduits" Gallaire, Minker, Nicolas [GAL 84]

Les BD déductives sont définies en rapport avec la logique du premier ordre. Alors que les BD conventionnelles sont considérées, par la logique, comme une interprétation d'une théorie de la logique du premier ordre, les BD déductives sont considérées comme une théorie de la logique du premier ordre.

Une BD déductive peut être représentée, sous forme clausale, de la façon suivante :

- un ensemble d'axiomes représentant les faits, ces axiomes sont des clauses où la partie gauche est vide et la partie droite ne contient qu'un seul prédicat.

  Exemple : --> GRANDPERE(jean,julie)

- un ensemble de règles de déduction de nouveaux faits, représenté par des clauses dont la partie gauche est non vide.

  Exemple : PERE(x,z),PERE(z,y) --> GRANDPERE(x,y)

- un ensemble de règles d'intégrité, décrites dans le même formalisme que les règles de déduction.

Suivant les restrictions apportées sur la forme clausale (partie droite réduite à un seul prédicat ou non), on peut classifier les BD déductives en BDD définies (un seul prédicat), et BDD indéfinies (plus d'un prédicat).

Pour des raisons opératoires, on se restreint aux BDD définies.

3.2. Définition opératoire des BDD définies

Une BDD définie est constituée :

- d'un ensemble d'axiomes A, lui même composé d'un ensemble d'axiomes A1 (les faits élémentaires de la base de données) et d'un ensemble A2 (les règles de déduction),
- d'un ensemble de contraintes d'intégrité CI,
- une meta-règle "negation as failure" (négation si échec), un fait P est faux si on n'a pas réussi à inférer P (non P est déduit si et seulement si on ne peut déduire P).

Les règles de déduction et les contraintes d'intégrité correspondent toutes deux à une connaissance générale du monde modélisé par la base de données. Il est donc difficile de choisir sous quelle forme représenter une telle connaissance (déduction ou intégrité). Un

certain nombre de critères existent dans la littérature, fondés essentiellement sur des aspects opératoires (par exemple, une règle de déduction ne doit pas manipuler de fonctions car on risque sinon d'obtenir des réponses indéfinies ou non explicites).

3.3. Utilisation des règles de déduction

Les régles de déduction peuvent être utilisées de deux façons différentes :

- elles sont activées lors de l'interprétation d'une requête, pour trouver les faits déductibles (implicites). Ce mode est appelé mode de dérivation.
- elles sont activées lors de l'introduction de nouveaux faits, afin de rendre explicites (stocker) les faits déductibles. L'interprétation d'une requête est alors identique à ce qui se fait dans les BD conventionnelles. Ce mode est appelé mode de génération.

Les avantages respectifs de l'un ou l'autre mode sont discutés dans [NIC 82].

4. SRI et SGBD

4.1. Les apports des SGBD classiques

Il semble intéressant de déterminer quels peuvent être les apports des SGBD dans les SRI. Les SRI manipulant un volume de données important, il est assez naturel de vouloir utiliser les SGBD comme outil de stockage :

- les textes peuvent être stockés dans une base de données textuelles (SGBD généralisée permettant la gestion de documents),
- la relation d'indexation ainsi que les attributs externes des documents peuvent être décrits en termes d'une base de données conventionnelle.

Exemple :

Relation ATTRIBUT(DOC,AUTEUR,TITRE,ANNEE, ...)
où DOC est l'ensemble des références des documents et AUTEUR, TITRE, ANNEE des attributs externes de documents.

Relation INDEXATION(DOC,CONCEPT,POIDS)
où CONCEPT décrit l'ensemble des concepts du domaine considéré, et POIDS donne la mesure de pertinence associée au couple concept - document.

INDEXATION décrit une relation d'indexation continue.

A l'aide de cette description, on peut aisément écrire une fonction d'interrogation correspondant au modèle booléen simple, en termes des opérateurs de l'algèbre relationnelle.

Exemple de requête :

donner les titres des documents traitant d'informatique et de gestion parus depuis 1982.

Dans le langage SQL (SYSTEM R), on l'exprime de la façon suivante :

```
SELECT TITRE
FROM ATTRIBUT
WHERE ANNEE >= 1982 AND
    DOC IN
        SELECT DOC
        FROM INDEXATION
        GROUP BY DOC
        HAVING SET CONCEPT CONTAINS (informatique,gestion)
```

Un SRI de type modèle booléen simple peut donc être facilement décrit en termes du modèle relationnel [MCL 85]. Le schéma peut également être enrichi de façon à y intégrer la description d'un thésaurus, et à pouvoir ensuite l'interroger pour connaitre les concepts utilisés dans le domaine.

De plus, en utilisant un SGBD généralisé, on peut inclure les textes dans le schéma et les manipuler. Les opérateurs sur ce type texte sont cependant assez pauvres dans les SGBDG ; ce sont la plupart du temps des opérateurs sur des chaines de caractères, ce qui permet néanmoins l'ajout des opérateurs de troncature et d'adjacence. Les textes peuvent être représentés sous forme hiérarchique (structure logique du document), ce qui est important dans des applications type documentation technique.

Au niveau de l'interface, les langages associés au modèle relationnel (malgré leur puissance d'expression et leur indépendance vis à vis du stockage physique), sont interdits aux usagers non informaticiens. Le recours à d'autres types d'interfaces, notamment graphi-

ques est largement souhaitable [BEG 85], [LEM 85].

L'apport des SGBD aux SRI est cependant tempéré par la difficulté d'étendre de tels systèmes pour mettre en œuvre des modèles de recherche plus élaborés, comme les modèles probabilistes ou vectoriel. Cette difficulté provient de la différence fondamentale entre SGBD et SRI, les SGBD fournissant un modèle de recherche exact sur des requêtes précises, et les SRI des modèles de recherche approchée sur des requêtes imprécises.

Cependant il est souhaitable de permettre une vision unifiée de la recherche d'informations à travers une base de données, en offrant des opérateurs spécifiques à chaque type d'information stockée. Il parait donc intéressant, dans le cadre des SGBD généralisés, de développer des opérateurs de recherche sur le type texte plus élaborés que de simples manipulations de chaine de caractères. On peut signaler dans ce sens, le travail de Deogun et Raghavan [DEO 85] visant à un modèle de recherche d'informations unifié, qui intégre le modèle de recherche exact pour les attributs classiques, et un modèle de recherche probabiliste sur les attributs de type texte.

4.2. Les apports des BD déductives

L'approche des SRI à travers les BD déductives n'a pas encore été, à notre connaissance, envisagée. Il semble pourtant qu'elles offrent certaines possibilités intéressantes en ce qui concerne l'utilisation d'un modèle de recherche linguistique.

On peut définir les fonctionnalités d'un SRI, en tant qu'approche déductive dans le cadre d'une théorie du premier ordre. Ceci nécessite de modéliser aussi bien les données (relation d'indexation, thésaurus, requêtes), que le processus d'interprétation d'une requête.

4.2.1. Les données

Les données vont être formalisées comme un ensemble d'axiomes.

a) la relation d'indexation est représentée par un prédicat INDEXE à trois arguments : référence de document, concept, poids du concept dans le document.

Exemple : INDEXE(d,informatique,0.8)

b) le thésaurus est représenté par un ensemble de prédicats décrivant les différentes relations sémantiques :

- SYNONYME prédicat à deux arguments : concept, concept
Exemple : SYNONYME(ordinateur,calculateur)
- GENERIQUE prédicat à deux arguments : concept, concept
Exemple : GENERIQUE(avion,DC10)
- VOISIN prédicat à trois arguments : concept, concept, poids
Exemple : VOISIN(bateau,navire,0.9)

Les relations sémantiques peuvent être décrites statiquement (comme dans un modèle relationnel classique), ou bien générées en partie dynamiquement par des règles de dérivation définissant les propriétés des relations. Il faudrait donc définir les règles de dérivation suivantes :

La relation SYNONYME est symétrique et transitive (on n'utilise pas la réflexivité) :
SYNONYME(a,b) --> SYNONYME(b,a)
SYNONYME(a,b) --> SYNONYMES(a,b)
SYNOMYMES(a,c),SYNONYME(c,b) --> SYNONYMES(a,b)
où SYNONYMES exprime la fermeture transitive de la relation SYNONYME.
De même pour la relation de voisinage sémantique :
VOISIN(c1,c2,p1) --> VOISINS(c2,c1,p1)
VOISINS(c1,c3,p2),VOISIN(c3,c2,p3),EGAL(p1,f(p2,p3))
--> VOISINS(c1,c2,p1)
où EGAL est un prédicat marquant l'égalité sur les réels, et
VOISINS exprime la fermeture transitive de la relation VOISIN.
La relation GENERIQUE est seulement transitive :
GENERIQUE(a,b) --> GENERIQUES(a,b)
GENERIQUES(a,c),GENERIQUE(c,b) --> GENERIQUES(a,b)
où GENERIQUES exprime la fermeture transitive de la relation GENERIQUE.
On peut également définir la relation SPECIFIQUE comme symétrique de la relation GENERIQUE :
GENERIQUE(b,a) --> SPECIFIQUE(a,b)
SPECIFIQUE(a,b),SPECIFIQUE(b,c) --> SPECIFIQUE(a,c)

Cette définition des relations sémantiques a l'avantage d'être moins couteuse en place, mais présente des difficultés opératoires importantes liées à la définition récurrente des règles (la synonymie produit des réponses non finies) et à l'utilisation de fonctions (on ne

peut garantir des réponses finies et explicites).

- une requête est définie comme une formule du calcul des prédicats : QUI-INDEXE(c,d) où c est une variable instanciée par un concept et d une variable libre décrivant l'ensemble des documents. On peut facilement étendre cette définition à un modèle booléen, en représentant l'opérateur de recherche OU par un OU logique, l'opérateur de recherche ET par un ET logique et le SAUF par un ET NON logique.

- un document est représenté par un ensemble d'axiomes décrivant les différents attributs que l'on veut stocker.
  Par exemple, AUTEUR prédicat à deux arguments : AUTEUR(document,personne)
  TITRE prédicat à deux arguments : TITRE(document,chaine de caractères)

  On peut de même représenter les relations entre documents (voisinage sémantique et citations bibliographiques par exemple) :
  VOISINDOC prédicat à trois arguments : VOISINDOC(document,document,similarité)
  CITE prédicat à deux arguments : CITE(document,document)

4.2.2. La fonction de correspondance

Elle va être définie à partir de règles de déduction. Par exemple :

- (1) CONTIENT(d,c,p),GRAND(p) --> QUI-INDEXE(c,d)
  (2) INDEXE(d,c,p) --> CONTIENT(d,c,p)
  Ces règles permettent de retrouver tous les documents indexés par un concept donné, avec une pertinence estimée suffisante (définie par le prédicat GRAND).

- (3) CONTIENT(d,c1,p),SYNONYME(c,c1) --> CONTIENT(d,c,p)
  Permet de retrouver également l'ensemble des documents indexés par les synonymes d'un concept donné.

- (4) CONTIENT(d,c1,p1),VOISIN(c,c1,p2),EGAL(p,f(p1,p2))
  --> CONTIENT(d,c,p)
  Permet de retrouver les documents indexés par les concepts voisins d'un concept donné, en pondérant la représentativité du concept en fonction d'une mesure de proximité entre les deux concepts.

Avec ces quatre règles de déduction, on peut donc traiter les requêtes portant sur une conjonction ou une disjonction de concepts. Si on veut traiter la négation (pour représenter le SAUF), il faut introduire une meta règle supplémentaire ("negation as failure"), qui indique que si un fait n'est pas stocké ou déductible, alors sa négation est vraie.

Par une approche purement déductive, on peut définir un SRI intégrant un modèle sémantique du domaine. On se rapproche ainsi de la définition d'un modèle linguistique. Ce SRI se composerait, par exemple :

- d'un ensemble d'axiomes A qui décrit la relation d'indexation et le thésaurus,
- des quatre règles de déduction (1) (2) (3) (4) qui réalisent la fonction de correspondance,
- d'une meta règle "negation as failure".

On peut également étendre ce modèle, en tenant compte des voisinages entre documents et des citations.

5. Conclusion

De nombreux efforts sont faits actuellement pour intégrer les SRI et les SGBD, notamment dans le cadre de SGBD généralisés. Des modèles de données et des langages associés ont déjà été définis, et permettent de manipuler des documents. Cependant les opérateurs de manipulation demeurent simples (recherche de chaines de caractères) et donc peu satisfaisants de notre point de vue. Cette faiblesse provient de la non prise en compte de la sémantique des textes, tant au niveau de la description que de la manipulation (il semble que le modèle relationnel ne soit pas satisfaisant pour fournir une fonction de correspondance autre qu'une fonction booléenne classique).

Il semble qu'il faille plutôt s'orienter vers un partage des tâches entre un SRI et un SGBD, le SRI définissant une fonction de correspondance évoluée, et le SGBD un stockage efficace et un accès rapide aux documents. Nous verrons au chapitre IV qu'un tel SRI peut être décrit dans une approche intelligence artificielle.

A plus long terme, l'intégration des SRI et des SGBD peut s'envisager par le développement des BD déductives qui intégrent les approches intelligence artificielle et base de données (les différentes tâches seront effectuées par une BD déductive).

CHAPITRE IV

SRI ET INTELLIGENCE ARTIFICIELLE

1. Introduction

Depuis quelques années, les systèmes d'Intelligence Artificielle (IA) font leur apparition dans le monde informatique. D'abord limitée aux problèmes tirés de la théorie des jeux ou aux problèmes de traitement de la langue naturelle, on assiste actuellement à l'entrée de l'intelligence artificielle dans tous les domaines, notamment par le biais des systèmes experts.

Il est donc naturel d'essayer de formuler une analyse des potentialités de l'IA dans le domaine des systèmes de recherche d'informations. Une présentation générale de ce problème a été déjà faite par SMITH [SMI 80] qui propose une sélection des grandes techniques d'IA intéressant la recherche d'informations. SPARCK JONES [SPA 78] traite également des outils linguistiques utilisés dans les systèmes de questions - réponses, et leur application aux SRI.

Il ressort de ces différentes réflexions que l'apport potentiel de l'IA aux SRI est très important (un certain nombre de SRI utilisent d'ailleurs certaines techniques héritées de l'IA), les apports principaux étant surtout :

a) *l'utilisation d'outils linguistiques*, à la fois pour le traitement de requêtes en langue naturelle, et pour la sélection de concepts dans les textes (indexation automatique). Cela permet une amélioration de la convivialité ainsi qu'une augmentation de la qualité des performances par la prise en compte de concepts plus riches sémantiquement que de simples mots-clés (structure syntaxique comme les groupes nominaux par exemple), ainsi que d'opérateurs de recherche plus puissants que les opérateurs booléens classiques (opérateurs de causalité, de localité, ...).

b) *l'utilisation de modèles de représentation des connaissances de l'IA*, permettant la modélisation des données et des traitements nécessaires à la recherche d'informations. Ces modèles sont des formalismes permettant la description des connaissances d'un système; les possibilités de traitement sont fortement dépendantes du formalisme choisi. Dans le cadre des SRI, les connaissances à modéliser sont nombreuses (documents, requêtes, thésaurus) et le choix d'un formalisme difficile.

c) *la résolution de problèmes* :

Elle consiste dans le choix de connaissances à appliquer pour atteindre un but déterminé. Elle peut être vue comme la recherche d'un chemin dans un graphe où les nœuds représentent des états et les arcs des connaissances modifiant des états. Le nœud initial représente l'état initial du système, et le nœud final un état dans lequel le but à résoudre est vérifié.

La résolution de problèmes peut fournir un cadre de modélisation à la recherche d'informations. De ce point de vue, la recherche d'informations est considérée comme l'identification, en réponse à chaque requête, des documents à retrouver. La résolution de problèmes consiste donc au développement d'une stratégie de recherche et l'utilisation de mécanismes d'inférences (comme l'équivalence sémantique de deux concepts par exemple).

d) *la prise en compte des problèmes d'apprentissage* :

En effet, il est difficile d'envisager un SRI "intelligent" (c'est à dire possédant un minimum de connaissances d'ordre sémantique) sans mécanisme d'apprentissage. Les SRI ont par définition un domaine d'application ouvert et il est donc impossible de définir a priori toute la connaissance nécessaire.

L'apprentissage en IA est considéré comme le moyen d'augmenter les performances d'un système dans le temps [MIC 83]. Les techniques requises concernent donc la possibilité d'évaluer les performances afin de juger de l'amélioration produite, ainsi que les moyens de stocker et d'utiliser les résultats d'expérimentations antérieures.

Un des cas les plus simples d'apprentissage, est le mécanisme d'essais - erreurs, où le système enregistre chaque action effectuée et son résultat, afin de, lorsqu'on se retrouve dans la même situation, réeffectuer les actions jugées bonnes et d'éviter celles jugées mauvaises.

L'apprentissage dans les SRI doit pouvoir être envisagé sous deux aspects :

- apprentissage à court terme : il s'agit de l'adaptation des réponses pendant le traitement d'une requête afin de mieux répondre aux besoins de l'utilisateur. C'est le cas de la reformulation automatique des requêtes par exemple, où on substitue aux termes jugés peu satisfaisants des termes susceptibles d'améliorer les réponses,

- apprentissage à long terme : il s'agit de modifications durables du système concernant la mise à jour de connaissances (introduction ou modification de mots ou concepts nouveaux, par exemple) à partir des interrogations produisant des réponses jugées satisfaisantes.

L'apprentissage pose néanmoins le problème de la confiance à accorder au processus apprenant. Il peut être dangereux de mettre à jour des données à partir de déductions provenant d'une ou de peu d'interactions avec l'utilisateur. Il est donc souhaitable de faire la mise à jour dans une base séparée de la base principale, qui ne sera consultée qu'en cas d'échec sur la base principale. Lorsque ces données sont validées, soit par un expert humain, soit par un grand nombre de validations concourantes, la base principale peut être mise à jour.

e) *l'utilisation des techniques de reconnaissances des formes* ("pattern recognition") comme modèle de la recherche d'informations :

La reconnaissance de formes est l'identification d'un certain objet ayant un ensemble particulier de caractéristiques, comme membre d'une certaine classe d'objets. Trois grandes classes de techniques sont utilisées :

- le tri, les critères d'appartenance à une classe sont donnés explicitement,
- la correspondance de prototypes ("prototype matching"), chaque classe est identifiée par un ensemble d'éléments (prototypes),
- classification ("clustering"), les classes sont créées sur la base de similarité et dissimilarité mutuelle entre objets.

Ces différentes approches sont toutes les trois fondées sur l'existence d'une mesure de similarité, déterminée à partir des caractéristiques utilisées pour décrire chaque objet.

Dans ce cadre, on peut aisément décrire le problème de la recherche d'informations comme un problème de reconnaissance de formes. Les différentes classes étant des classes de documents (notamment la classe des documents pertinents à une réponse, et celle des documents non pertinents), la forme à reconnaitre étant une requête. Le problème est donc de savoir quelles classes de documents correspondent à la requête. Cette approche (notamment en utilisant la technique de "clustering") a d'ailleurs été utilisée avec succès dans de nombreux SRI [CRO 80], [SAL 83].

On peut rajouter à ces cinq catégories d'outils d'IA, une autre qui semble particulièrement intéressante aujourd'hui, qui est la modélisation de l'usager. En effet, il semble de plus en plus évident que l'amélioration de la qualité d'un quelconque système interactif (à la fois sur les aspects performances et convivialité) ne peut se faire sans prise en compte d'un modèle de l'utilisateur permettant de mieux connaitre ses besoins. Ce problème se pose

avec force dans les systèmes d'Enseignement Assistée par Ordinateur (EAO) et la plupart des techniques développées l'ont été dans ce cadre.
En ce qui concerne les SRI, l'apport d'un modèle d'utilisateur concerne surtout le choix des réponses en fonction du niveau estimé de l'utilisateur.

Dans la suite de ce chapitre, nous allons détailler l'ensemble des points que nous venons d'introduire à l'exception des problèmes de reconnaissance des formes qui sont déjà bien traité dans la littérature concernant les SRI [SAL 83] [RIJ 79], et de l'apprentissage. Nous faisons ensuite la présentation succinte de SRI utilisant certaines des techniques d'IA présentées et nous terminons ce chapitre en définissant ce que nous entendons par "système intelligent de recherche d'informations" (SIRI).

2. Modèles linguistiques et SRI

Les applications des modèles linguistiques sont bien évidemment importantes dans les SRI. Elles peuvent s'envisager à deux niveaux :

- tout d'abord, le traitement de requêtes d'interrogation en langue naturelle : l'utilisation de la langue naturelle rend l'interface plus souple et offre plus de possibilités à l'utilisateur pour exprimer ses besoins (il n'a pas à s'adapter à un langage figé qui ne convient pas forcément à la formulation de ses besoins),
- le problème de la représentation des documents (le processus d'indexation) : La plupart des documents traités dans les SRI sont des textes écrits en langue naturelle, dont il est intéressant d'en faire l'analyse linguistique afin de déterminer leur contenu.

On peut noter cependant que ces deux applications nécessitent des modèles linguistiques assez différents :

(1) pour le traitement des requêtes, les mécanismes mis en jeu sont des mécanismes de compréhension permettant la mise en évidence des éléments nécessaires à la recherche ultérieure dans la base de documents, ce sont :

- les attributs sur lesquels porte la recherche : le contenu du ou des documents recherchés, ou bien des attributs externes comme titre, auteur, ...

- les valeurs associées aux attributs : les concepts à rechercher pour l'attribut de contenu, un titre pour l'attribut titre, ...
- la logique de la requête, c'est à dire la logique liant les divers attributs, ainsi que la logique existant à l'intérieur d'un attribut donné.

Ceci implique l'utilisation de mécanismes d'analyse sophistiqués car l'analyse de la requête doit être complète et assez fine. De nombreux problèmes linguistiques doivent être traités (dans la mesure du possible), comme par exemple :

- le problème des référents, référence pronominale par exemple (détermination de l'antécédent d'un pronom relatif),
- la détermination de l'épithète des adjectifs,
- la prise en compte des verbes,
- la mise en évidence des opérateurs logiques explicites (conjonctions de coordination, ...) ou implicites (souvent marqués par des signes de ponctuation), ainsi que de leur portée.

Ces problèmes nécessitent une analyse morphologique et syntaxique complète qui est malheureusement assez souvent insuffisante (on peut produire plusieurs analyses syntaxiquement correctes d'une même requête), il faut alors ajouter une composante sémantique permettant la levée des ambiguïtés. La difficulté dans les SRI est que le domaine couvert est très large et qu'il est donc impossible d'avoir un modèle sémantique complet. On peut cependant limiter ce modèle à la sémantique nécessaire aux besoins spécifiques de l'interface (on n'intègre que les verbes exprimant l'interrogation par exemple), et utiliser des méthodes heuristiques dans les cas non encore résolus.

Les outils employés sont donc assez puissants (ATN [WOO 70], grammaires à trous [MCC 82], grammaires logiques [PER 80]).

(2) pour la représentation des documents :
Le but ici n'est pas de parvenir à une compréhension détaillée du document, mais à une compréhension partielle consistant à en extraire les concepts représentatifs, susceptibles de devenir des termes d'indexation. L'utilisation des données sémantiques est très réduite dans la mesure où le domaine, beaucoup trop large, n'est absolument pas modélisable.

La plupart des outils déjà développés font donc appel à des analyses morpho-syntaxiques dites de surface, car ne visant qu'à une analyse partielle des phrases, limitée à la reconnaissance des syntagmes jugés intéressants. Les ambiguïtés éventuelles sont résolues par des méthodes heuristiques (on essaie de tirer un maximum d'informations sémantiques de la syntaxe). Une erreur dans l'analyse (une ambiguïté mal résolue) n'a pas ici une

importance aussi grande que dans le cadre de traitement des requêtes d'interrogation. En effet, les documents sont beaucoup plus longs qu'une requête, un concept important se répète a priori assez souvent, ce qui fait que l'on peut supposer que statistiquement une erreur intervient peu ou pas dans le résultat final de l'indexation.

D'après de nombreuses études [DEW 81], [DEB 82], [FLU 77], il semble, en français en tout cas, que les éléments intéressants soient les syntagmes nominaux car ils sont le plus porteurs de sens. Ce sont donc les syntagmes nominaux que l'on cherche à reconnaitre dans les documents, les syntagmes verbaux étant ignorés.

Les outils employés sont donc essentiellement morphologiques et syntaxiques, et l'effort porte surtout sur la définition de mécanismes heuristiques pour la levée d'ambiguïtés (avec dans certains cas des possibilités d'apprentissage) [FLU 77], [DEB 82], [PAL 85]. Ces systèmes sont pour l'instant expérimentaux ou en cours de développement, mais les résultats déjà produits sont fort intéressants sur le plan qualitatif.

3. Représentation des connaissances et SRI

Les SRI gèrent de grosses masses de données souvent assez peu structurées, et fréquemment mises à jour. Le modèle de représentation choisi doit donc être relativement souple et permettre une mise à jour aisée. Sur le premier point au moins, il semble que les modèles tirés de l'IA sont satisfaisants par rapport aux modèles de données classiques des bases de données.

On peut distinguer quatre grands types de représentation des connaissances en IA :

- les modèles tirés de la logique du premier ordre :
  Assez en vogue actuellement à travers l'utilisation du langage PROLOG [COL 83], ils présentent l'avantage de bénéficier d'un cadre théorique riche et de mécanismes d'inférences bien définis (principe de résolution entre autre). Les problèmes de mise à jour, par contre, sont plus difficile à traiter si l'on veut préserver la monotonicité (pour chaque connaissance ajoutée, il faut vérifier qu'elle ne soit pas en contradiction avec une connaissance existante ou pouvant être dérivée de connaissances existantes).

- les modèles de type règles de production :
  Ce sont les plus utilisés dans le cadre de la représentation de connaissances pour les systèmes experts. Ils présentent l'avantage d'être simples conceptuellement (ils modélisent une connaissance de type prémisses - conclusion), d'être puissants (un grand nombre de connaissances peuvent être représentées sous cette forme), de

disposer de mécanismes d'inférences (parcours de graphes d'états, par exemple).

- les modèles type réseaux sémantiques [LOP 79] :
  Ils sont utilisés le plus souvent pour traiter les problèmes relatifs à la sémantique des langues naturelles. Ils présentent l'avantage d'être puissants (on peut pratiquement tout représenter), mais sont par contre difficiles à gérer (temps de traitement et place mémoire occupée), et les mécanismes d'inférences ne sont pas toujours faciles à mettre en évidence et à réaliser.

- les modèles type "objets" :
  Ils se développent beaucoup actuellement. On retrouve sous cette dénomination les frames [KUI 75], les scripts [SCH 77], les objets et classes des langages SIMULA ou SMALLTALK. L'idée est d'établir une représentation à deux niveaux, un premier niveau (la classe) permet de décrire toutes les caractéristiques (attributs, propriétés) d'un objet, alors que le second niveau règle les relations entre classes (le plus souvent par une structure hiérarchique). Le mécanisme d'inférences utilisé est l'héritage de propriétés à travers la hiérarchie de classes. Ces modèles se trouvent implantés dans des langages de programmation comme SMALLTALK, LOOPS [STE 82], FLAVORS, ...

En plus de ces grandes catégories, on trouve souvent des modèles "hybrides" qui sont des mélanges des catégories précédentes (logique plus objets [ALB 84], [GAL 85] ou réseaux sémantiques et règles de production [DUD 77]).

Les connaissances à représenter dans le cadre des SRI sont fort nombreuses. On peut citer :

- les concepts (termes d'indexation ou concepts du domaine),
- les documents,
- les requêtes,
- les connaissances sur la recherche d'informations (stratégies de recherche, heuristiques).

Le problème est de choisir un modèle de représentation permettant l'intégration harmonieuse de ces connaissances. Nous allons examiner tour à tour chaque catégorie de modèle dans le cadre des SRI.

3.1. Modèles type "règles de production" et SRI

Les règles de production sont très utilisées dans les systèmes experts pour représenter des connaissances provenant d'un expert humain [DAV 77]. Leur intérêt provient de leur simplicité conceptuelle, de l'indépendance des connaissances entre elles, ce qui implique une mise à jour aisée de la base (si on ne prend pas en compte le problème de cohérence de la base vis à vis du mécanisme d'inférences utilisé).

Formellement, ces règles se décrivent sous la forme :

SI <condition> ALORS <conclusion>

Des extensions à ce formalisme sont souvent utilisées, comme par exemple :

- l'évaluation de la vraisemblance de la conclusion (ou connaissance incertaine). Elle permet de prendre en compte le caractère non certain d'une déduction sous forme d'une fonction de pondération indiquant la confiance à accorder à la conclusion :

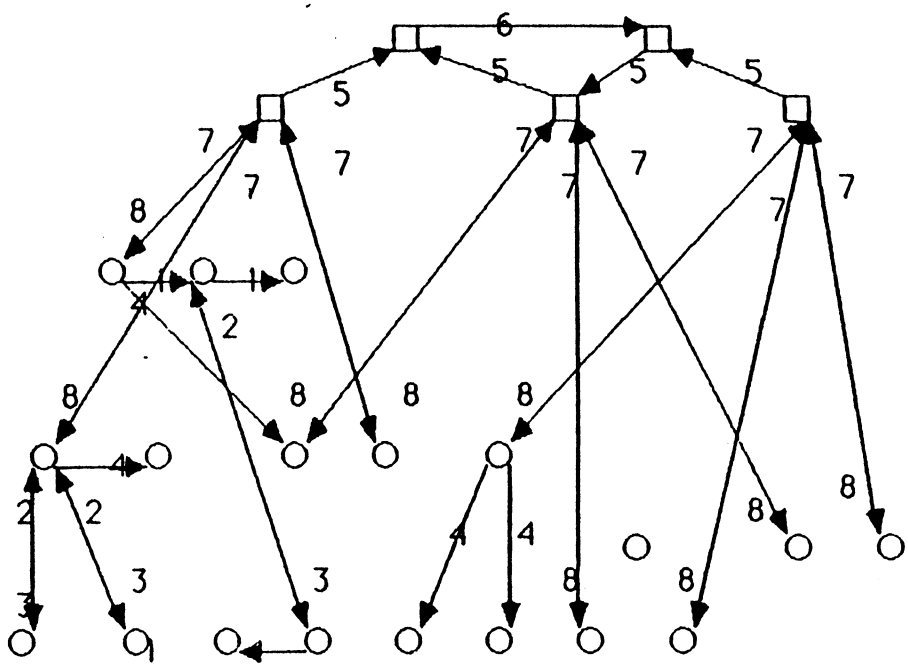
  SI <condition> ALORS <conclusion> AVEC CERTITUDE p

- l'utilisation de conditions supplémentaires infirmant ou confirmant la conclusion initiale :

  SI <condition1> ALORS <conclusion1> AVEC CERTITUDE p1
  ET SI <condition2> ALORS <conclusion1> AVEC CERTITUDE p2
  si p2 > p1, <condition2> renforce la conclusion
  sinon, il infirme la conclusion

Dans le cadre des SRI, ce formalisme s'applique difficilement aux relations statiques entre concepts et documents, mais peut par contre définir les connaissances nécessaires à la recherche d'informations (stratégies de recherche, heuristiques, ...).

Exemple :
SI le nombre de réponses > 50 ALORS reformulation nécessaire



□ représente un noeud de type document

○ représente un noeud de type concept

→n représente un arc étiqueté par n

- figure IV-1 -

Un exemple de réseau sémantique

3.2. Modèles logiques et SRI

Comme nous l'avons déjà vu dans le cadre des bases de données déductives (cf III-4), la logique du premier ordre offre un bon moyen de modélisation du problème de la recherche d'informations.

3.3. Réseaux sémantiques et SRI

Les réseaux sémantiques sont un moyen assez naturel de représenter les connaissances d'un SRI. En effet, on peut décrire les données comme un graphe orienté avec des arcs étiquetés où :

- un nœud est soit un document, soit un concept,
- un arc est étiqueté de plusieures manières :
  1- représente une relation synonyme
  2- " " générique
  3- " " spécifique
  4- " " de voisinage sémantique entre concepts

  Ces quatre types d'arcs joignent deux nœuds de type concept.
  5- représente une relation de citation bibliographique
  6- " " de voisinage sémantique entre documents

  Ces deux types d'arcs joignent des nœuds de type document.
  7- représente la relation d'indexation entre un nœud document et un nœud concept
  8- représente la relation d'indexation entre un nœud concept et un nœud document

  Exemple : voir figure IV-1

Le graphe obtenu peut être très gros (plusieurs dizaines de milliers de nœuds et d'arcs) et par conséquent difficile à gérer. Pour faciliter le traitement d'un tel graphe, il est souhaitable que l'ensemble des sous-graphes obtenus à partir du graphe général, en ne considérant qu'un seul type d'arc, soit sans cycles.

La recherche des documents indexés par un concept peut être effectuée de la façon suivante :

- accès au nœud du graphe correspondant au concept à rechercher,
- pour ce nœud et pour tous ses voisins par les arcs de type 1,
  accès aux documents par les arcs de type 8,

On obtient ainsi l'ensemble des documents indexés par un concept et l'ensemble de ses synonymes. On peut généraliser à des disjonctions, des conjonctions, des négations de concepts (modèle booléen classique). On peut également prendre en compte les autres relations pour la recherche (citations, voisinages, ...).

Cette expression de la recherche d'informations en termes de parcours de graphe nécessite que l'on puisse accéder directement à n'importe quel nœud du graphe. Pour cela, il faut définir une structure d'index sur les nœuds permettant l'accès direct. On trouve un exemple d'implantation d'un SRI à l'aide d'un réseau sémantique dans [CRO 83].

3.4. Modèles type "objets" et SRI

Les représentations en objets sont issues d'une conjonction d'idées très diverses, les schémas, les frames, les scripts, les prototypes, les objets, les formes, les acteurs, les types abstraits (pour une bibliographie assez complète sur tous ces éléments voir [BON 84]).

Le terme d'objet recouvre toutes les notions comprises dans les différentes appellations évoquées. On peut définir un objet de façon générale, comme un ensemble d'aspects ou d'attributs (qui caractérisent l'objet). Ces aspects peuvent contenir des facettes standards décrivant chaque aspect.

Par exemple, on peut définir les facettes VALEURS-POSSIBLES, DEFAUT qui donnent le domaine de valeurs de l'aspect, et sa valeur par défaut.

La notion d'objet est donc analogue à celle de canevas définissant un ensemble d'éléments ayant les mêmes propriétés. A chaque objet est donc attaché un ensemble d'éléments correspondant au modèle de l'objet, qui sont appelés instances de l'objet.

Un objet peut être représenté dans une hiérarchie et avoir des objets plus ou moins généraux que lui (par exemple, l'objet document est plus général que l'objet chapitre, mais est moins général que l'objet corpus). De plus, un objet hérite des propriétés des objets plus généraux que lui (l'héritage peut être éventuellement multiple, si on admet que la hiérarchie des objets n'est pas un arbre). Ce mécanisme d'héritage de propriété est le mécanisme d'inférences associé aux modèles objets.

L'utilisation des objets est faite de manière diverse suivant les applications considérées. Dans le cadre des langages orientés objets, elle se fait par passation de messages. Dans ces langages, le vocabulaire est différent, un objet est une classe, et une instance d'objet un objet. L'expédition d'un message à un objet provoque un changement d'état de cet objet, ainsi que l'expédition d'un message en retour. Cela implique que l'on détermine dans la définition de la classe, la liste des messages pouvant accepter les objets ainsi que la façon de les traiter. Ce mode d'emploi concernant les messages (le protocole de communication) est généralement appelé une méthode.

Dans beaucoup d'autres cadres (comme les frames par exemple), le mécanisme de base est le filtrage qui permet de déterminer à quel objet rattacher un ensemble d'observations (cas de reconnaissance des formes).

L'intérêt des modèles type objets dans les SRI parait peu évident, si ce n'est dans une optique reconnaissance des formes ("pattern recognition") dont nous avons parlé dans l'introduction de ce chapitre. Dans ce cas, les documents sont les objets et une requête les observations, et il faut déterminer quels sont les objets correspondants à la requête. Ceci se formalise bien à l'aide de frames hiérarchisés suivant une classification sémantique des documents (obtenue par des méthodes de "clustering" par exemple).

En dehors de cette approche, les modèles objets ne semblent pas trop satisfaisants pour représenter le thésaurus ou les documents. En effet, il n'y a pas à proprement parler de classes d'objets (les seules que l'on peut mettre en évidence sont les concepts et les documents) avec des propriétés communes intéressantes. La hiérarchie de classe est par conséquent fort réduite et le mécanisme d'héritage n'intervient pas.

4. Résolution de problèmes et SRI

La résolution de problèmes consiste à chercher le moyen de parvenir à un résultat donné, en utilisant un certain nombre de connaissances. Classiquement on opère une distinction entre connaissances algorithmiques et heuristiques (connaissances empiriques permettant le plus souvent d'effectuer un choix dans l'application des connaissances algorithmiques; ce peut être des fonctions de valuation par exemple).

Le processus de résolution consiste à chercher une séquence de connaissances permettant de passer du problème à sa solution. Deux types de cheminement sont possibles :

- on applique sur le problème un certain nombre de connaissances (dans l'ordre prémisses - conclusion) jusqu'à ce qu'on arrive dans un état où il est résolu : résolution par chaînage avant,

- on connait déjà l'ensemble potentiel des résultats du problème, et on cherche alors, en partant de chaque résultat, à remonter au problème initial par les connaissances appliquées dans l'ordre conclusion - prémisses : résolution par chaînage arrière.

Dans le cadre des SRI, le problème à résoudre est une requête, et le but est l'ensemble des documents correspondant à la requête. Les connaissances utilisées sont de plusieurs types :

- décomposition d'un problème en un ensemble de sous-problèmes,
  Exemple : "traiter une requête revient à traiter chacun de ses concepts séparément puis à composer le résultat"

- purement algorithmique,
  Exemple : "accès aux dictionnaires"

- heuristique,
  Exemple : "choix d'une stratégie de recherche"

Ces connaissances sont représentées dans un formalisme propre à la résolution de problèmes, et fonctionnent ensuite en chainage avant ou arrière pour résoudre une requête.

5. Modèles de l'utilisateur et SRI

La modélisation de l'utilisateur et le problème de l'apprentissage sont fortement connectés. En effet, le modèle de l'utilisateur va permettre de mieux apprécier ses besoins et par conséquent d'améliorer le processus d'apprentissage.

La modélisation de l'utilisateur n'est intéressante que si elle peut s'apprendre, ou tout au moins si, à partir d'un nombre restreint d'informations sur l'utilisateur, on est capable de déduire un modèle beaucoup plus complet [RIC 79].

Ces modèles peuvent être utilisés dans les SRI à la fois au niveau de l'interrogation et au niveau de l'indexation.

5.1. Interrogation et utilisateurs

Pour l'interrogation, on essaie de construire un modèle général de l'utilisateur à partir des interactions qui sont effectuées, c'est à dire principalement la requête. Les différentes déductions que l'on peut en tirer sont :

- tout d'abord, la connaissance du niveau de l'utilisateur dans le domaine questionné permet de connaitre plus finement ses besoins en informations. On peut supposer que si l'utilisateur est spécialiste, il attend des références précises et de haut niveau en réponse, alors que ce n'est pas le cas pour un utilisateur débutant.

- le niveau de l'utilisateur permet également d'évaluer la confiance que l'on peut accorder à des notions apprises à partir de ses interactions. On accorde une plus grande confiance à un spécialiste qu'à un débutant.

- le niveau permet également un choix de stratégie de recherche. Pour un spécialiste on choisira plutôt une stratégie favorisant la précision par rapport au rappel.

- le niveau de l'utilisateur dans la manipulation du système permet également de choisir le niveau d'interactions entre l'usager et le système. Un débutant a besoin de nombreux messages d'aide et si possible rédigés de façon claire. De même l'affichage des réponses peut dépendre du niveau.

5.2. Indexation et utilisateurs

A partir de modèles généraux de classes d'utilisateurs, on essaie d'adapter l'indexation à ces différentes classes. La forme indexée constituant une certaine perception du contenu d'un document (cf I), il est intéressant de moduler cette perception en fonction de l'utilisateur du SRI.

On peut, en effet, supposer que plus un utilisateur est spécialiste du domaine traité, plus la perception qu'il a des documents est aigüe et précise. Par ailleurs, une différence entre la perception choisie pour l'indexation et celle de l'utilisateur, peut être génératrice de bruit et/ou de silence à l'interrogation.

L'idéal serait de réindexer l'ensemble des documents pour chaque nouvel utilisateur, de façon à ce qu'il y ait toujours adéquation. Cette solution est malheureusement inopératoire, car trop coûteuse. Plusieurs autres approches peuvent être poursuivies :

- choix d'une perception moyenne (barycentre de celle des utilisateurs affectés des cœfficients de fréquence d'interrogation), ce qui implique de connaître a priori les types d'utilisateurs du SRI. Cette solution présente l'avantage d'être facile à implanter.

- choix d'une perception maximale (celle d'un spécialiste) pour l'indexation. La correspondance à l'interrogation entre la forme indexée et la requête est fonction de la différence entre le niveau estimé de l'utilisateur et un spécialiste. Par exemple, si l'utilisateur est spécialiste, la différence est nulle et la correspondance réussit si les concepts de la requête sont très proches sémantiquement de ceux du document ; par contre, pour un débutant, la différence est grande et on se contente d'une proximité faible via des génériques.

- choix d'une indexation par niveaux, chaque niveau correspondant à une perception d'une classe d'utilisateur (l'ensemble des classes est prédéfini). A l'interrogation, en fonction du niveau estimé de l'utilisateur, on confronte sa requête au niveau d'indexation correspondant. Cette hiérarchie peut être utilisée pour la reformulation où sans modifier la requête, on peut la confronter de nouveau, non pas au niveau d'indexation initial, mais à ses voisins dans la hiérarchie.

Parmi ces diverses approches, il semble que l'indexation par niveau soit la plus intéressante. Elle permet de disposer d'une représentation des documents riche, qui peut être également utilisée pour la reformulation. Le coût d'une telle indexation est malheureusement prohibitif et cette solution n'est donc pas opérationnelle.

6. Systèmes existants

Un certain nombre de systèmes proches des systèmes de recherche d'informations mettant en œuvre des techniques d'intelligence artificielle sont développés depuis quelques années. La plupart mettent l'accent sur l'aspect interface langue naturelle, mais peu d'entre eux réunissent les fonctionnalités complètes d'un SRI.

6.1. Les systèmes de question - réponses

Les systèmes de question - réponses sont des systèmes de l'intelligence artificielle. Ils ont été développé surtout dans les années 1960 [COO 64], (on trouve également une présentation générale dans [SAL 83]). Leur but est de permettre l'accès à des faits stockés dans une base de connaissances.

Généralement, les connaissances enregistrées sont de deux types :

- un grand nombre de faits relatifs à un domaine particulier,
- des règles générales permettant l'utilisation des faits dans le contexte de l'interrogation.

Les questions et les réponses sont le plus souvent exprimées en langue naturelle. Il n'y a pas de restrictions sur les questions, cependant la plupart des prototypes réalisés traitent celles de type oui - non (c'est à dire qui admettent pour réponse oui ou non). Dans ce cas, les questions sont évidemment très précises et assez riches sémantiquement ce qui complique leur analyse.

Le processus de traitement d'une question est le suivant :

- extraction des éléments intéressants pour la recherche,
- comparaison de ces éléments avec les faits stockés et composition de la réponse à partir des faits jugés pertinents et des règles générales.

La difficulté inhérente à ce processus est grande, car l'analyse de la langue naturelle dans des domaines relativement larges n'est pas résolue et il est également difficile de mettre en évidence des règles d'utilisation des faits suffisamment générales.

Ces problèmes font que ces systèmes sont restés à l'état de prototypes sur des domaines d'application très restreints.

L'approche suivie pour la composition des réponses est la plupart du temps déductive et fondée sur la logique du premier ordre (principe de résolution), ce qui fait que l'on peut rapprocher les systèmes de question - réponses des SGBD déductives.

Il est intéressant de faire un parallèle entre les SQR et les SRI, car ils sont relativement proches sur un certain nombre de points.

- l'analyse préalable de la requête pour en tirer les éléments intéressants est du même type que ce que l'on fait en recherche d'informations, dans le cadre des modèles linguistiques.

- le travail de comparaison des éléments aux faits est semblable à la correspondance terme d'interrogation - concept, à la différence essentielle près que celle ci doit être exacte et non pas estimée.

Le domaine d'application est très différent, très restreint dans le cas des SQR, et ouvert dans le cas des SRI. Les objectifs poursuivis par un système intelligent de recherche d'informations (SIRI) sont néanmoins proches de ceux des SQR, en rajoutant les contraintes liées au domaine ouvert (pas de modèle sémantique détaillé, questions imprécises).

6.2. IRUS (Information Retrieval using the RUS parsing system)

IRUS [BAT 83] est un système permettant l'accès en langue naturelle (l'anglais) à des bases de données hétérogènes. Le principal objectif est la transportabilité qui est envisagée sur trois axes :

- la possibilité de changement de domaine,

- la possibilité de changement de base de données à l'intérieur d'un même domaine,

- la possibilité de changement de système de bases de données.

L'effort a été porté sur la définition d'une interface utilisateur transformant la requête en une représentation interne qui est ensuite interprétée en fonction des différentes bases visées (langage pivot). Cette représentation interne (appelée MRL "Meaning Representation Language") est obtenue après une série de traitements linguistiques :

- tout d'abord un analyseur morpho-syntaxique, basé sur les techniques des ATN (réseaux de transition décorés [WOO 70]),

- un interprète sémantique incrémental, qui permet la levée de certaines ambiguités syntaxiques ainsi que la mise en évidence de quantificateurs,

- un certain nombre de modules spécialisés traitant de problèmes linguistiques comme ellipses, anaphores, ...

La représentation ainsi obtenue peut être décrite comme une extension du calcul des prédicats, dans laquelle les domaines de quantifications des variables sont rendus explicites.

Exemple :
La chaine d'entrée
"Show the books charged out by people in department 45 in january."
est transformée dans l'expression MRL suivante :

```
(for all x / book :
    (for some y / person :
        (for the z / dept :
            (= (dept# z) 45) ;
            (dept-member y z));
        (for some w / day :
            (in-month w january) ;
            (borrow x y w))) ;
    (print x))
```

La forme MRL d'une requête étant construite, elle est projetée sur le schéma de la base de données visée, et les expressions correspondantes du langage de manipulation sont générées.

IRUS présente donc une interface langue naturelle puissante, avec un mécanisme de transformation du MRL vers les schémas de bases de données. Le principal problème réside dans la construction manuelle des modèles de connaissance du domaine ainsi que des règles de transformation, ce qui limite l'intérêt du système. La résolution des phénomènes linguistiques comme ellipses et anaphores reste également faible.

6.3. Un système expert pour la consultation de données bibliographiques

Ce système [POL 83] a été développé (en MICRO-PROLOG) sur un sous ensemble de sujets médicaux concernant le cancer tirés de MEDLINE. Il s'agit d'un système de recherche d'informations muni d'une interface utilisateur par menus qui permet la sélection de termes de recherche.

Le modèle de connaissances utilisé est constitué par un ensemble d'entêtes de sujets médicaux représenté sous forme de frames organisés hiérarchiquement. Un total de 41 frames a été construit sur un maximum de sept niveaux.

Le problème de la recherche d'informations est traité par un ensemble de règles de production permettant :

- la sélection des frames (pour le choix des termes de recherche),
- le traitement des termes de recherche sélectionnés,
- la génération d'expressions de recherche en termes du langage d'interrogation de MEDLINE,
- la gestion de l'interface avec MEDLINE.

Exemple :

```
REGLE NO 1 :
IF "START" ON CONTROL-BOARD
THEN ERASE "START" FROM CONTROL-BOARD
    CLEAN SITE-BOARD
    CLEAN TYPE-BOARD
    CLEAN THERAPY-BOARD
    DISPLAY FRAME 2
    CONTEXT FRAME 2
```

L'enchainement des règles se fait à travers une structure de données commune type "blackboard" (tableau noir), qui décrit l'état du système.

Ce codage en termes de règles de production permet une assez grande souplesse notamment en ce qui concerne la transformation des termes sélectionnés en expressions de recherche.

Une évaluation du système a été faite, en comparant les résultats obtenus pour un certain nombre de requêtes avec ceux fournis par un expert indexeur du domaine. Sur cet échantillon, les résultats sont à peu près équivalents. Il reste cependant que la grande précision des réponses provient surtout du niveau de détail des frames, ce qui empêche la généralisation de la technique à un domaine plus large.

6.4. RUBRIC (RUle Based Retrieval of Information by Computers)

RUBRIC [TON 85] est un système de recherche d'informations fondé sur une approche règles de production.

La base d'indexation est constituée d'un fichier inverse des mots apparaissant dans les documents. A chaque mot est associé un certain nombre d'informations contextuelles (comme les documents où il apparait et à quelle position), ce qui permet de traiter des opérateurs de recherche de type positionnel (existence de deux mots dans une même phrase par exemple).

L'utilisateur formule sa requête sous forme d'un ensemble de règles qui définit un filtre de recherche sur les documents. Ceux ci sont classés suivant une mesure de pertinence estimée. Cette mesure est calculée en interprétant les règles comme une hiérarchie de concepts à rechercher et de sous concepts. Par conséquent, en désignant un concept, l'utilisateur met en œuvre automatiquement une recherche dirigée par le but dans l'arbre défini par tous les sous concepts utilisés dans la définition du concept cherché. Les sous concepts de plus bas niveau sont eux mêmes définis en termes d'expressions sur des chaines de caractères permettant la recherche par mots clés, par position, ...

L'évaluation de la pertinence globale d'un document se fait en fonction des chaines qu'il contient, ainsi qu'à partir des mesures de pertinence attachées aux règles, propagées dans le graphe de recherche par des fonctions héritées des logiques multi-valuées [LUK 30].

La syntaxe choisie pour les règles est une extension des règles de production permettant de rajouter un antécédent auxiliaire renforçant ou diminuant la valeur de la conclusion.

Exemple :
IF "the story contains the literal string 'bomb' "
THEN "it is about an explosive device with degree 0.6"
BUT IF "it also mention a boxing match"
THEN "reduce the strength of the conclusion to 0.3"

Un prototype a été évalué sur un certain nombre de documents (une trentaine) et quelques requêtes, et semble produire des réponses de bonne qualité.

Il parait cependant, que la qualité du résultat est assez fortement liée à la qualité de formulation de la requête en termes de règles. Il faut donc associer à ce système, un mécanisme d'aide à la formulation permettant une expression plus conviviale et plus simple de la requête, notamment pour les utilisateurs non spécialistes du domaine qui vont avoir des difficultés à donner des définitions précises des concepts en sous concepts. La définition des cœfficients attachés aux règles est également très difficile à faire, surtout pour un utilisateur ne connaissant pas leur utilisation dans le modèle de recherche.

L'idée de décrire la pertinence d'un document en termes de règles de production est intéressante, mais il semble qu'elle peut être associée avec la prise en compte de la typologie de l'usager. En effet, ces règles pourrait être fournies non pas par l'utilisateur, mais intégrées au système pour chaque classe d'utilisateur mise en évidence. Cela permettrait de simplifier la requête qui serait restreinte à une expression de concepts, sans perdre la possibilité d'évaluer chaque document produit.

RUBRIC est donc un SRI classique qui offre la possibilité à l'utilisateur de définir des critères sémantiques d'évaluation des résultats. Cette approche offre l'intérêt de ne pas nécessiter de modèle sémantique a priori, mais le recours à un spécialiste du système pour formuler ces critères semble obligatoire.

6.5. Un intermédiaire comme système expert

Ce système actuellement développé parallèlement au notre [CRO 85], a pour objectif la modélisation du comportement d'un expert humain en recherche d'informations. Cet expert contrôle toute l'interaction utilisateur - système, en utilisant un certain nombre de connaissances concernant différents types d'utilisateurs, des méthodes de formulation de requêtes, différentes stratégies de recherche, les façons de visualiser l'information.

De façon plus détaillée, les capacités du système sont les suivantes :

(1) construction et maintenance des modèles d'utilisateurs. Ces modèles représentent les caractéristiques d'utilisateurs individuels, qui sont indépendants des requêtes. Cela inclut des connaissances sur le domaine, fournies par l'utilisateur lors de la formulation de sa requête, ainsi que l'information dérivée des descriptions des types d'utilisateurs.,

(2) construction d'un modèle de l'information demandée. Ce modèle de requête est construit durant la formulation de celle-ci, et peut être modifié par des interactions ultérieures,

(3) sélectionner des stratégies de recherche en fonction des modèles de requête et d'utilisateur,

(4) permettre le balayage de la base de données durant le processus de formulation d'une requête,

(5) présenter l'information à l'utilisateur et obtenir un retour ("feedback"). Le type d'information affichée et la façon de l'afficher, dépend du modèle de la requête, de l'utilisateur, et des capacités du moniteur,

(6) fournir des explications du fonctionnement.

La réalisation d'un tel système implique l'intégration d'outils d'intelligence artificielle et de recherche d'informations. Les techniques d'IA sont utilisées pour représenter les connaissances expertes sur la formulation de requêtes (règles de production), ainsi que comme stratégies de recherche contrôlant les actions du système. Les techniques d'indexation automatique (par extraction de mots clés) et des stratégies de recherche (modèle binaire, modèle binaire étendu, classification, modèles de dépendance) sont utilisées pour leur efficacité et leur indépendance relativement au domaine traité.

Les connaissances sur le domaine font partie de celles de l'expert. Elles sont fournies, soit par l'utilisateur, soit sous forme d'un thésaurus pré-établi. La représentation des documents est produite par les techniques d'indexation augmentée d'informations de type bibliographiques comme les auteurs, les citations, ...

Ce système n'est pas pour l'instant implanté, mais l'architecture logicielle prévue est celle d'un ensemble de modules spécialisés, coopérant au travers d'une structure de données commune.

Ce travail est l'un des plus ambitieux présenté jusqu'à ce jour, puisqu'il présente une vision générale de la modélisation d'un intermédiaire à la recherche d'informations (il ne se contente pas de développer une interface en langue quasi naturelle). Sur bien des points, il est similaire au notre (modèles de l'utilisateur, de la requête, architecture logicielle). Il offre en plus l'utilisation potentielle de plusieurs modèles de recherche, mais il ne prend pas en compte l'évaluation des réponses et la reformulation automatique des requêtes.

7. Conclusion : les fonctionnalités d'un SIRI

Les techniques d'IA constituent un champ d'investigation extrêmement riche pour les SRI. Nous avons montré que ces techniques permettent de résoudre (au niveau théorique du moins) les problèmes actuellement rencontrés dans les SRI classiques, c'est à dire manque de convivialité et faibles performances sur le plan qualitatif. Nous pensons donc que l'intelligence artificielle est, pour l'instant, la seule voie réellement productive dans le développement de systèmes évolués de recherche d'informations.

Si nous avons montré l'intérêt théorique de l'intelligence artificielle, il nous reste à démontrer que cette approche est réaliste sur un plan pratique. Pour cela, nous avons défini et réalisé un SIRI sous la forme d'un système expert (SE) en recherche d'informations modélisant l'activité d'un documentaliste. Cette approche a déjà été utilisée ([DAV 80], [YIP 81]), mais les systèmes réalisés étaient allés moins loin dans leur modélisation de l'activité (ils ne traitent pas ou peu les aspects évaluation et reformulation notamment).

Nous faisons tout d'abord une présentation fonctionnelle de notre système, puis nous abordons les principaux problèmes d'architecture posés par l'intégration d'un système expert et d'un SRI.

7.1. Un système expert en recherche d'informations

Le but est donc de modéliser le comportement d'un expert en recherche d'informations (la personne qui réalise l'interrogation des bases documentaires) de façon à le simuler. Ce travail consiste à déterminer la séquence d'actions entreprises par l'expert, ainsi que, pour chacune d'entre elles, l'ensemble des connaissances mises en œuvre. Si la première partie ne pose pas énormément de problèmes, il est par contre assez difficile de mettre en évidence l'ensemble des connaissances requises (les experts n'en ont pas toujours une perception très nette).

Nous avons effectué ce travail à partir des études existantes et d'interviews réalisées avec un expert (la documentaliste de notre institut). On peut donc décomposer la tâche de l'expert de la façon suivante :

(1) détermination des concepts à rechercher :
Elle se fait à partir d'un dialogue avec l'utilisateur, qui exprime ses besoins à la fois en donnant les thèmes visés par la recherche, mais également en donnant le type de réponse visée (cela peut être le type du document, livre, article, ... ou bien le niveau de

connaissances véhiculées par le document, présentation générale, article "pointu", ...).

(2) construction du plan de recherche :

A partir du dialogue précédent, l'expert construit un plan de recherche indiquant l'ordonnancement des critères de sélection et les façons de combiner ces critères. Cette phase n'est pas complétement indépendante des SRI dont dispose l'expert pour sa recherche, car ce plan doit pouvoir être traduit en expressions du langage d'interrogation de ces SRI. La plupart du temps, ce plan se réduit à un ensemble de termes à rechercher connectés par des opérateurs logiques.

(3) choix de la base à interroger :

A partir du plan de recherche et des connaissances qu'il a sur les bases documentaires dont il dispose, l'expert doit choisir une ou plusieurs bases sur lesquelles il va effectuer ses recherches. Les connaissances sur les bases concernent surtout le contenu du fonds documentaire et son adéquation à la recherche proposée; mais elles peuvent être d'ordre plus technique (on sait que les documents ont été bien indexés, ou bien le langage d'interrogation dispose d'opérateurs plus puissants, ...).

(4) transformation du plan de recherche en expressions du langage d'interrogation cible :

Une fois la base choisie, l'expert transforme le plan de recherche en expressions du langage d'interrogation. Le problème est double :

* tout d'abord transformer les termes du plan en termes d'indexation pour la base. L'expert dispose pour cela de ses connaissances empiriques (les interrogations déjà effectuées sur cette base), et le cas échéant d'un thésaurus (sur support informatique ou papier) lui indiquant les équivalences entre termes. Il est à noter qu'un terme du plan de recherche peut se transformer en une expression du langage d'interrogation (et pas simplement en un seul terme d'indexation).

* transformation des opérateurs de recherche du plan en opérateurs de recherche pour la base. C'est une opération assez délicate qui nécessite de bien maitriser la sémantique des opérateurs de la base visée.

(5) exécution de la requête d'interrogation,

(6) évaluation des réponses :

C'est la tâche la plus difficile à modéliser. On peut distinguer deux phases dans cette évaluation :

* évaluation globale de la réponse :
Il s'agit de déterminer si l'ensemble des réponses obtenues est conforme à l'attente de l'utilisateur. Les critères de choix sont ici surtout fonction du nombre et de la qualité des réponses fournies par rapport aux besoins de l'utilisateur formulés lors de la détermination des concepts à rechercher. En effet, si l'utilisateur veut se faire une bibliographie générale sur un domaine particulier, il attend un assez grand nombre de documents en réponse; par conséquent si on ne lui fournit que quelques documents il n'est pas satisfait. De façon symétrique, si l'utilisateur cherche un article "pointu", il ne veut pas cinquante références en réponse. De même si le système associe une mesure de pertinence aux résultats fournis, l'utilisateur peut vouloir qu'au moins une réponse soit de bonne qualité.

* évaluation d'une réponse :
L'utilisateur peut vouloir estimer chaque réponse, par exemple pour savoir laquelle est la plus satisfaisante relativement à son problème. Pour cela, il faut essayer d'estimer la pertinence du document relativement à la requête. Si le système dispose d'une mesure, c'est un critère important, sinon on peut évaluer la réponse en regardant la liste de ses termes d'indexation : si tous ces termes sont proches des termes cherchés alors la réponse est jugée bonne.

(7) reformulation du plan de recherche :
En fonction de l'évaluation qui est faite des réponses, l'expert va reformuler son plan de recherche initial pour essayer d'améliorer les réponses obtenues. Pour cela, il dispose de diverses connaissances, les termes d'indexation associés aux documents qu'il a jugés très pertinents, et les connaissances empiriques qu'il possède sur ces termes lui permettant de déterminer les termes trop utilisés du domaine (ils génèrent un nombre de réponses important) et les termes peu utilisés du domaine (ils génèrent peu ou pas de réponses).

L'expert en recherche d'informations a donc un ensemble de tâches assez diverses à accomplir, et utilise pour cela des connaissances nombreuses, de nature différente (connaissances purement liées à la recherche d'informations, liées au domaine, liées aux SRI utilisés).

Dans un SRI classique, la seule fonction assurée est l'exécution d'une requête d'interrogation (phase 6), toutes les autres fonctions présentées ci-dessus sont donc prises en charge par le système expert d'interrogation.

7.2. Architecture d'un système intelligent de recherche d'informations

Le système que nous voulons réaliser intègre les fonctionnalités d'un SRI avec celles d'un système d'IA. Cette intégration de deux techniques différentes nous pose un problème d'architecture logicielle pour notre SIRI [DEF 84]. En effet, les architectures de chacune des deux composantes sont assez différentes. Le SRI, d'une part, défini classiquement de façon procédurale comme un ensemble de modules utilisant des données pour en produire de nouvelles, et le système d'IA, d'autre part, défini de façon déclarative comme un ensemble de connaissances et un mécanisme inférentiel indépendant, permettant la manipulation de ces connaissances afin de résoudre un problème donné. Pour mieux situer ce problème, nous rappelons tout d'abord les fondements des systèmes experts.

7.2.1. L'architecture d'un système expert

Les SE ont été développés à la fin des années 1970 pour traiter de problèmes nouveaux en informatique [BAR 77], [LAU 81], [HAY 83], [SIC 83]. Ces problèmes laissaient une large part à l'activité humaine et mettaient en jeu de grandes quantités d'informations et de connaissances, difficiles à maitriser informatiquement. Les domaines d'applications visés étaient surtout les problèmes de diagnostic, activités typiquement réalisées par des experts humains. Les SE sont donc une nouvelle approche de l'informatique permettant de traiter des problèmes d'une telle complexité. Cette approche consiste en un changement d'architecture des systèmes informatiques : à l'équation "algorithme + structure de données = programme" on substitue l'équation "connaissances + mécanisme de raisonnement + contrôle = intelligence".

Cette architecture en trois composantes est héritée de celles des systèmes d'intelligence artificielle. Les systèmes experts sont une spécialisation de ces systèmes modélisant l'expertise des spécialistes d'un domaine, c'est à dire des faits (connaissances), des règles (raisonnement), des heuristiques (contrôle).

L'avantage de cette architecture est qu'elle permet de séparer des problèmes jusqu'alors traités globalement au sein des algorithmes (le concepteur peut donc se consacrer à chacun des trois aspects séparément).

Nous allons détailler un peu plus ces trois composantes :

- les connaissances :

Elles sont représentées dans un formalisme différent de celui utilisé classiquement. Les modèles de représentation doivent être généraux, sémantisés (les connaissances manipulées

sont plus riches que de simples faits). Ils varient suivant les connaissances à représenter. On distingue cependant trois grandes approches, les modèles hérités de la logique, les modèles à règles de production, les modèles à approche par les données (modèles objets, frames, réseaux sémantiques, ...). Ces approches sont complémentaires, certains modèles représentent bien les faits (modèles orientés données), d'autres les règles (modèles logiques, à règles de production).

- les mécanismes de raisonnement :

Ils constituent le cœur des SE, puisqu'ils donnent les capacités inférentielles aux systèmes sur lesquels ils sont implantés. Ces mécanismes sont plus ou moins généraux, liés aux modèles de représentation et traduisent des principes de raisonnement (par l'absurde, par hypothèse et test, par exhaustivité, par analogie, ...). On peut distinguer, en simplifiant, trois grandes classes de modèles de raisonnement :

* les modèles de nature logique fondés sur les logiques classiques, ou plus rarement sur des logiques non classiques (modales, temporelles, ...). Ils offrent l'avantage de présenter un cadre théorique bien établi pour le mécanisme d'inférence,

* les modèles type règles de production qui s'apparentent aux modèles logiques, sans posséder le cadre théorique lié à l'inférence,

* les modèles type objets où la représentation des connaissances (la structure des objets) contraint les inférences possibles (par exemple, le mécanisme d'héritage des propriétés dans les langages orientés objets).

- le contrôle des raisonnements :

Il a pour but de réduire l'espace de recherche des solutions. Il s'agit de méthodes heuristiques, dépendantes du problème traité mais qui visent à l'efficacité des SE.

Par exemple, dans le cadre d'un système à règles de production, diverses heuristiques de choix d'une règle active existent :
- la première règle rencontrée,
- la règle la plus récente,
- la règle la plus contrainte.

Les mécanismes de raisonnement et de contrôle sont souvent fusionnés dans ce qu'on appelle un moteur d'inférences [LAU 84]. On peut résumer le fonctionnement d'un SE de la façon suivante :

(1) sélection de l'ensemble des connaissances actives par rapport à l'état courant du système,
(2) choix des connaissances à utiliser (contrôle)
(3) application des connaissances choisies suivant le mécanisme de raisonnement utilisé,
(4) si la solution au problème traité est atteinte, on s'arrête
sinon aller à (1)

7.2.2. Intégration d'un SRI et d'un Système d'Intelligence Artificielle

L'intégration de ces deux approches peut être envisagée de trois manières distinctes :

- l'intégration du système d'IA dans le SRI :
Elle est difficile à envisager, car elle implique de figer les mécanismes d'IA de façon à pouvoir les représenter de façon procédurale. Dans ce cas, les avantages liés à la dynamicité des techniques disparaissent, et par conséquent le système obtenu ne peut répondre aux objectifs que nous nous sommes fixés.

- l'intégration du SRI dans le système d'IA :
Elle implique la définition du SRI en termes d'un système d'IA. Nous avons vu au chapitre précédent que cette redéfinition est possible théoriquement, mais qu'elle risque de se faire au détriment des performances du système (à cause de la masse très importante de connaissances à gérer, qui de plus ne nécessitent pas toujours l'utilisation de mécanismes inférentiels).

- la coopération du système d'IA et du SRI :
Elle doit permettre de conserver les avantages respectifs des deux architectures (aspect performances des SRI, et aspect dynamique des systèmes d'IA), tout en fournissant de nouvelles fonctionnalités à travers la coopération des deux techniques. Le découpage fonctionnel entre les deux systèmes doit être défini précisément, de même que le protocole de coopération.

Nous allons maintenant étudier plus en détails les deux dernières approches.

7.2.2.1. Intégration du SRI dans le SE

Nous considérons ici un SRI classique, composé d'une base d'indexation (ensemble de couples formés d'un terme d'indexation et d'une référence vers un document), d'un thésaurus (ensemble de termes et de relations sémantiques classiques les reliant) consultable en ligne, et permettant l'interrogation du corpus via des requêtes d'interrogation booléennes.

Le but est de décrire ce SRI en termes de SE, de façon à obtenir un SE en recherche d'informations (en y rajoutant les fonctions expertes évoquées au début de ce chapitre). Pour cela, il faut trouver la répartition entre les faits, les règles et les heuristiques. On considère que :

- les éléments de la base d'indexation sont des faits; la relation d'indexation est une donnée statique qui ne permet pas de dérivation d'informations nouvelles, et on représente donc naturellement un élément de la relation d'indexation par un fait,

- les éléments du thésaurus sont de la même manière des faits,

- pour les relations du thésaurus deux approches sont possibles. On peut tout d'abord considérer l'ensemble des relations sémantiques entre termes (y compris celles obtenues par l'utilisation des propriétés des relations, comme la transitivité par exemple) que l'on représente alors par un ensemble de faits. L'autre solution consiste à différencier à l'intérieur de chaque relation sémantique, les éléments élémentaires, et les propriétés génératives qui permettent la recomposition de toute la relation. Dans ce cas, les éléments élémentaires sont représentés par des faits alors que les propriétés des relations (symétrie, transitivité, ...) sont représentées par des règles. La première approche est donc coûteuse en place occupée mais est rapide en temps d'exécution, alors que la seconde approche offre les avantages et désavantages inverses,

- les fonctions du SRI et les fonctions expertes sont des règles et des heuristiques; les fonctions classiques décrites habituellement sous forme de procédures peuvent être recodées par un ensemble de règles de production enchainées par un mécanisme déductif déterministe (réalise l'équivalent de l'exécution séquentielle). Les fonctions expertes modélisant une activité humaine, sont composées d'un ensemble de règles représentant les connaissances à utiliser, ainsi que d'heuristiques permettant la résolution de choix conflictuels entre différentes connaissances.

- Choix d'une représentation :
  Les faits peuvent être aisément représentés par un ensemble de formules du calcul des prédicats du premier ordre, ou bien par un réseau sémantique. Par contre le formalisme des règles de production n'est pas adapté.
  Les règles peuvent également être représentées par des formules du calcul des prédicats du premier ordre, ou par des algorithmes d'exploration du réseau sémantique. Le formalisme des règles de production est là très adapté. Si on veut une représentation uniforme des faits et des règles, le choix se réduit donc au calcul des prédicats du premier ordre et aux réseaux sémantiques. L'avantage de l'approche prédicative est qu'elle permet une meilleure souplesse au niveau de la définition des règles (il n'est pas toujours facile de décrire les fonctionnalités désirées en termes du réseau sémantique) mais, par contre, le réseau sémantique permet de traiter rapidement des problèmes comme la détermination des synonymes d'un terme (il suffit d'examiner les voisins du terme dans le réseau), avantage il est vrai qui est payé par la place occupée.

- Conséquences de l'intégration :
  On va avoir une base de faits extrêmement importante à gérer (plusieurs dizaines de milliers de faits) et donc difficile à manipuler. Il est nécessaire de mettre en œuvre des mécanismes permettant de subdiviser la base de faits de façon à n'utiliser, à un instant donné, que les éléments nécessaires à la tâche à accomplir : par exemple, lorsque l'on veut reformuler une requête, il suffit de considérer les éléments du thésaurus voisins de ceux de la requête.

  La cohérence de la base est très difficile à assurer, étant donné sa taille; on ne peut donc garantir que le mécanisme de raisonnement produise un résultat correct. De plus, la base risque d'être soumise à de nombreuses mises à jour, car elle est modifiée à chaque ajout ou retrait de document.

  Le thésaurus peut être utilisé comme un modèle de connaissances, dans la mesure où, faisant partie de la base de faits, des mécanismes déductifs peuvent lui être appliqué. Cela permet une consultation plus riche pour l'utilisateur, et des possibilités de vérification de la cohérence du thésaurus un peu plus étendues (fermeture transitive des différentes relations par exemple).

  Le SE obtenu est complétement lié au domaine considéré et au SRI utilisé, ce qui ne permet donc pas sa réutilisation.

7.2.2.2. Coopération d'un SE et d'un SRI

La coopération d'un SRI avec un SE pose deux problèmes essentiels :
- quelle est la répartition des tâches entre les deux composants ?
- qui gère la coopération ?

La réponse à la première question est claire : on peut reprendre la répartition faite précédemment entre fonctions du SRI et fonctions expertes. En ce qui concerne la coopération, trois cas peuvent être envisagés :
a) le SRI gère la coopération :
C'est lui qui lance les appels au SE et vérifie l'exécution correcte de ces appels, ce qui implique que l'on doit rajouter un niveau au SRI permettant de déterminer quand on doit faire appel au SE et comment contrôler son fonctionnement.

b) le SE gère la coopération :
C'est lui qui lance l'exécution des primitives du SRI et les vérifie. Cette tâche de coopération est décrite en termes de règles incluses dans la base de connaissances.

c) la gestion est faite par un contrôleur :
On définit un troisième composant (un contrôleur) dont l'activité va être de déterminer la séquence d'actions à exécuter, et qui ensuite lance les appels aux deux autres composants en fonction de la nature de l'action à entreprendre.

Les deux dernières solutions semblent les plus adaptées, elles ne nécessitent pas de modifier le SRI, elles offrent une indépendance du processus de coopération vis à vis du SRI; et enfin la définition du mécanisme de coopération est plus souple et modulaire (un ensemble de règles à changer ou un module spécialisé à réécrire).

Le SE en recherche d'informations se compose donc d'un ensemble de faits, règles et heuristiques permettant de réaliser les fonctions expertes (et la coopération avec le SRI si on choisit la solution 2 de coopération).

- choix d'une représentation :
  Il est bon de donner ici, en français, quelques exemples des connaissances que l'on a à représenter :
  "dans le cadre de l'évaluation globale du résultat, on peut dire que si l'usager est spécialiste, que le nombre de réponses est petit, et que leur qualité est jugée bonne alors on considère que le résultat est satisfaisant."

Ce type de connaissances peut être représenté par des formules du calcul des prédicats du premier ordre ou bien par des règles de production :

Si on veut représenter la règle concernant l'évaluation globale, on obtient :
SPECIALISTE(usager), PETIT(résultat), APPARTIENT(r,résultat),
BONNEMESURE(r) implique EVALUATION(résultat,correct)
ou bien (en termes de règles de production) :
SI usager = spécialiste ET nombre(résultat) < 10 ET
  POUR TOUT r de résultat mesure(r) > 0.7
ALORS
  evaluation(résultat) <--- correct

- Conséquences de la coopération :
La base de connaissances est relativement petite (quelques dizaines de règles) et est donc facile à manipuler et à gérer.

Le SE obtenu est relativement indépendant du SRI; si on veut le réutiliser pour d'autres SRI ayant le même domaine traité, il suffit de redéfinir la partie concernant la coopération SE - SRI, ainsi que les connaissances éventuelles sur le SRI utilisé (qualité de l'indexation, connaissance du fonds documentaire). Par contre, si on change de domaine, la partie de la base qui est liée au domaine est à redéfinir (il n'y a que la partie concernant l'expertise générale de l'expert qui reste valide).

Les ajouts ou retraits de documents ne remettent pas en cause le SE.

La manipulation du thésaurus en tant que modèle de connaissances est plus difficile à mettre en œuvre (il faut développer des outils spécifiques).

7.3. Conclusion sur l'architecture

Un certain nombre de possibilités d'architecture sont possibles pour un SIRI, chacune présentant ses caractéristiques propres. Nous avons choisi la dernière solution proposée pour notre système prototype IOTA, et nous en expliquons les raisons au chapitre V.

CHAPITRE V

LE PROTOTYPE IOTA

Nous présentons successivement dans ce chapitre, l'architecture choisie pour IOTA ainsi que ses fonctionnalités. Le détail de la réalisation des fonctions de IOTA est également donné, alors que les connaisances expertes seront définies au chapitre suivant.

1. L'architecture choisie

Comme nous venons de le voir, suivant le choix d'architecture effectué, on obtient des SE en recherche d'informations assez différents. En ce qui nous concerne, nous avons choisi la solution de la coopération car elle présente un certain nombre d'avantages :
- tout d'abord, elle offre une modularité importante, ce qui pendant la phase de développement d'un prototype est fort intéressant,

- le SE est beaucoup plus général,

- et surtout, on peut effectuer une analyse beaucoup plus fine de la tâche de l'expert en recherche d'informations en mettant bien en évidence (grâce à l'expérimentation de différents domaines et de différents SRI) les connaissances générales de la recherche d'informations, les connaissances liées au domaine, et les connaissances liées au SRI.

La gestion de la coopération est faite par le SE, ce qui offre l'avantage de ne pas avoir à écrire un contrôleur, et de pouvoir décrire le processus de coopération sous forme d'un ensemble de règles modulaire et donc modifiable.

1.1. L'architecture du système intelligent de recherche d'informations

Nous allons détailler l'architecture logicielle du SE en recherche d'informations que nous avons développé. Pour cela, nous présentons tout d'abord le détail des fonctionnalités du système, puis nous décrivons l'ensemble des données et connaissances utilisées, et nous terminons cette présentation par l'exposé du fonctionnement du SE construit.

1.1.1. Les fonctionnalités du système

Les fonctionnalités du système sont celles décrivant le comportement d'un expert en recherche d'informations. Il nous nous faut donc modéliser ce comportement de façon à pouvoir le réaliser informatiquement. A partir de cette modélisation, on peut alors donner le schéma de traitement d'une requête ainsi que le contrôle qui est fait de ce traitement par le mécanisme de coopération.

1.1.1.1. Modélisation du comportement de l'expert humain

A partir de l'analyse du comportement exposée précédemment (cf IV-7-1), on peut déduire une modélisation des tâches effectuées en termes de fonctions d'un processus informatique [DEF 85]. Il faut déterminer ici dans quelle mesure ces tâches peuvent être matérialisées informatiquement, et dans ce cas en déterminer un découpage en actions élémentaires. Si on reprend la séquence de tâches de l'expert, on obtient :

(1) détermination des concepts à rechercher et construction du plan de recherche :
Si l'on veut modéliser le plus fidèlement possible cette tâche, il faut la matérialiser par un système de dialogue avec l'utilisateur simulant une interview. Ce système doit être capable d'analyser finement les besoins exprimés par l'utilisateur pour déterminer d'éventuels problèmes (incohérences, manque de précision, ...). Les mécanismes informatiques nécessaires sont donc très puissants puisqu'il faut réaliser l'analyse, la compréhension et la génération de phrases en langue naturelle. Pour cette raison, nous avons choisi de réduire ce dialogue à une seule requête émise par l'utilisateur sans retour immédiat de la part du système. Un autre aspect important de cette tâche est l'évaluation (très subjective) du niveau de l'utilisateur dans le domaine questionné, à partir de l'interview réalisée. Cette activité est très difficile à modéliser (surtout si l'on part de l'hypothèse que la requête est la seule connaissance que l'on a sur l'utilisateur).
Nous avons donc choisi de modéliser cette tâche par deux actions (appliquées sur la requête) permettant la mise en évidence des concepts recherchés (construction d'une

équation primaire de recherche) et l'évaluation de la typologie de l'utilisateur.

En ce qui concerne la construction de l'équation de recherche primaire, de nombreux problèmes d'ordre linguistique se posent. Tout d'abord, si on se place dans un cadre général, les besoins exprimés par l'utilisateur peuvent porter soit sur des attributs externes (titre, auteur, ...), soit sur le contenu. Le traitement effectué sur la requête doit donc être capable de faire la distinction entre ces attributs. La détermination des liens logiques entre les deux attributs, ou à l'intérieur d'un attribut est également difficile, à cause notamment des problèmes de portée des opérateurs logiques et de leur mise en évidence (les opérateurs logiques ne sont pas toujours marqués explicitement dans la requête). Pour les attributs externes on doit être capable de distinguer le désignateur de l'attribut (son type) de sa valeur souhaitée. Pour l'attribut de contenu, il faut déterminer les différents concepts recherchés, ce qui pose un certain nombre de problèmes linguistiques (détermination de l'épithète d'un adjectif, de l'antécédent d'un pronom relatif, ...).

(2) choix de la base :
Pour être matérialisée, cette tâche nécessite l'utilisation de modèles de connaissances généraux décrivant le contenu des différentes bases ainsi qu'un certain nombre d'informations les concernant (qualité de l'indexation, puissance du langage d'interrogation, ...). Le problème est alors de chercher la base dont le contenu est le plus proche de celui de la requête (problème de voisinage sémantique).

(3) Transformation du plan de recherche en expressions du langage d'interrogation cible :
Il s'agit ici d'effectuer le passage entre les concepts exprimés par l'usager dans l'équation primaire de recherche, et les concepts connus du système; ce processus de correspondance transforme l'équation primaire de recherche en équation finale de recherche; aux concepts reconnus on a substitué l'ensemble des concepts connus correspondants. Le problème réside donc dans la définition de la fonction de correspondance, qui à tout concept reconnu formé d'un ensemble de mots connus du système, doit associer une expression logique de concepts connus du système.

(4) Exécution de la requête d'interrogation :
L'équation finale de recherche est ensuite interprétée en substituant à chaque concept son ensemble de références associées dans la relation d'indexation, et en appliquant ensuite les opérateurs de recherche composant l'équation. On obtient alors l'ensemble de références satisfaisant la requête. Le problème principal réside dans la détermination de la pertinence d'une réponse relativement à la requête. Cette pertinence doit être fonction du poids de la

relation d'indexation entre chaque concept et la réponse, ainsi que de la logique introduite par les opérateurs.

(5) Evaluation des références :
Le but est d'évaluer l'ensemble des réponses obtenues de façon à déterminer si cet ensemble est correct (dans ce cas le traitement s'arrête avec l'affichage des réponses à l'utilisateur), ou s'il est incorrect (dans ce cas la requête doit être reformulée). Cette évaluation va être assurée par un ensemble de règles expertes. Le problème ici est de déterminer l'ensemble des connaissances nécessaires à l'évaluation (connaissances la plupart du temps très empiriques) et au diagnostic (que faut il changer pour que les réponses soient jugées bonnes).

(6) Reformulation :
Il s'agit ici de reformuler automatiquement la requête, de façon à obtenir un ensemble de réponses plus conforme à l'attente de l'utilisateur. Les problèmes qui se posent ici sont le choix du concept à reformuler, le choix de la relation sémantique à utiliser et enfin le niveau de reformulation à effectuer.

Parallèlement à cette modélisation de l'activité du documentaliste, nous avons ajoutés des possibilités d'apprentissage, visant à améliorer la convivialité du système (inférence de mots inconnus dans une requête), et les performances (enregistrement des inférences réalisées pendant le processus de correspondance par exemple).

1.1.1.2. Le processus de coopération

C'est lui qui réalise l'enchainement séquentiel des tâches que nous venons de décrire. Il est important de noter que cet enchainement représente le traitement maximum d'une requête et qu'en fonction des requêtes traitées, il peut être différent (des tâches peuvent être supprimées). En effet, on peut s'apercevoir pendant l'exécution d'une tâche qu'il est inintéressant de passer à la suivante car elle ne peut rien apporter.
Par exemple, si on s'aperçoit qu'un des concepts de la requête est très générique vis à vis du domaine, et que par conséquent il indexe beaucoup de documents, on peut tout de suite reformuler la requête car le nombre de réponses sera trop grand.

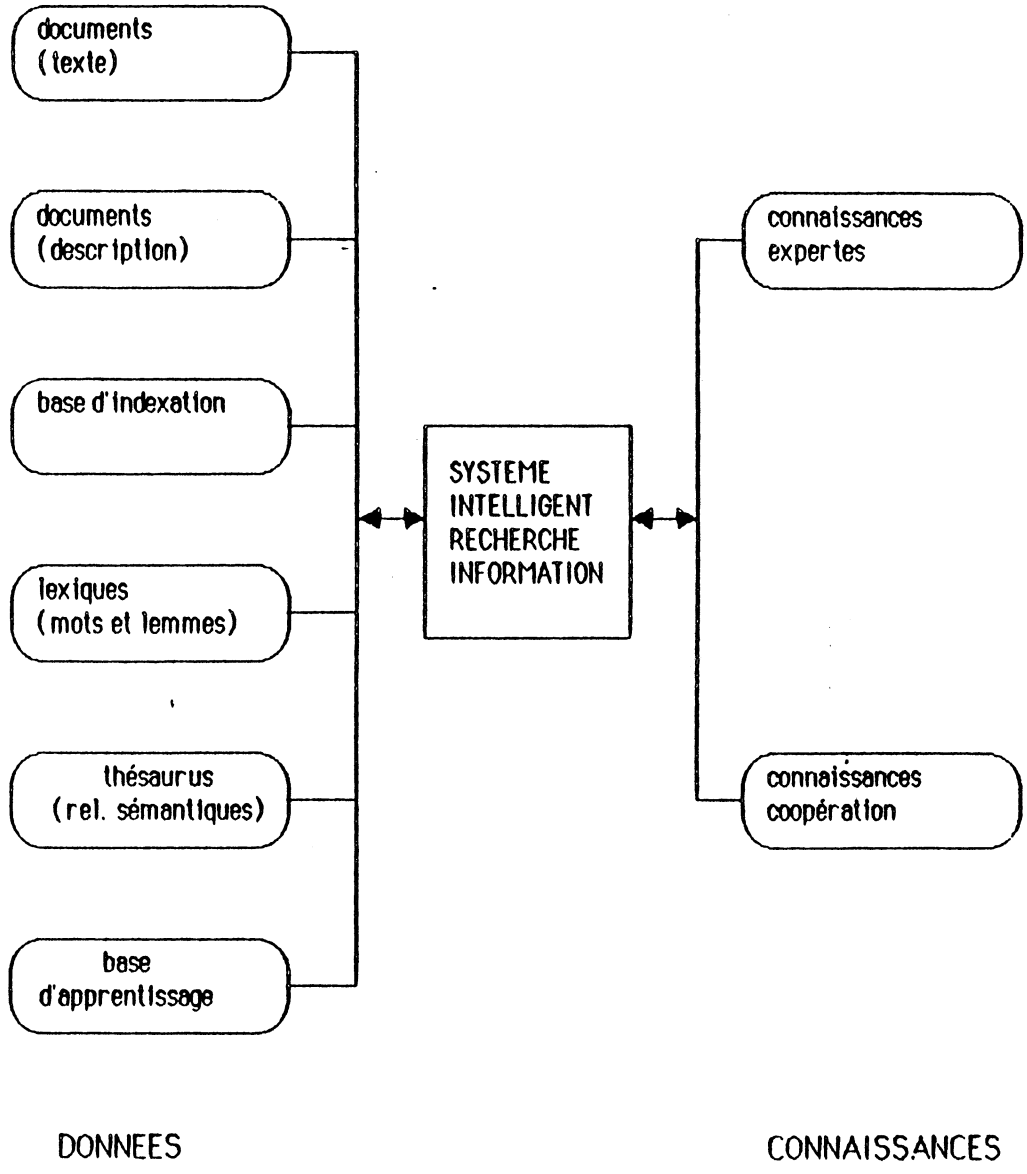
Le processus de coopération doit donc pouvoir plannifier l'ensemble des tâches à accomplir en fonction d'une requête, de manière à optimiser le processus.

1.1.2. Données et connaissances utilisées

Le système dispose d'un grand nombre de données et connaissances à gérer (voir figure V-1). Les données (les plus nombreuses) sont utilisées par le SRI alors que les connaissances sont utilisées par le SE.

Les différentes données utilisées sont :

- les textes intégraux,
- la base d'indexation, il s'agit de quadruplets formés d'un terme d'indexation, d'une référence vers un élément de la structure logique d'un document, de la mesure de représentativité du terme d'indexation dans le document, de la mesure de représentativité du document par rapport au terme d'indexation [KER 84]. Le processus d'indexation automatique utilisé présente les caractéristiques suivantes :
  - afin d'augmenter la richesse de représentation de la forme indexée, les concepts extraits des documents ne sont pas simplement des mots isolés (mots-clés), mais sont représentés sous forme de syntagmes nominaux, ce qui permet une plus grande précision de représentation d'un concept, et par conséquent permet d'augmenter la précision des réponses fournies par le système d'interrogation.
  - ce système d'indexation s'appliquant à des corpus techniques (application de type documentation technique), où le souci de précision d'une réponse est primordial, l'indexation effectuée prend en compte la structure logique du document (découpage en chapitres, sous-chapitres, ...). La relation d'indexation porte donc sur l'ensemble des concepts et sur l'ensemble des éléments de la structure logique.
  - les éléments de la relation d'indexation (termes d'indexation) sont normalisés par référence à un thésaurus [BRU 85], ce qui permet un gain au niveau du stockage de la relation et une certaine factorisation sémantique des concepts tirés des textes. Cette factorisation, permet à l'interrogation de diminuer le bruit.
  - la relation d'indexation n'est pas binaire, mais continue. La relation entre un concept et un élément de la structure logique est pondérée (par une mesure comprise dans l'intervalle [0,1], qui évalue la représentativité mutuelle concept-élément), ce qui évite une sélection un peu "brutale" des concepts représentatifs d'un élement de la structure. Cette non sélection diminue le silence à l'interrogation.
- la description de la structure logique des documents (découpage en chapitre, ...),
- les lexiques, qui vont servir à l'analyse de la requête,
- le thésaurus, ensemble de relations sémantiques entre les termes d'indexation, nécessaire au processus de reformulation,
- la base d'apprentissage, base dans laquelle sont stockées les données ayant été "apprises" pendant le traitement des requêtes (éléments des lexiques ou de la base d'indexation).


documents
(texte)
documents
(description)
base d'indexation
lexiques
(mots et lemmes)
thésaurus
(rel. sémantiques)
base
d'apprentissage
SYSTEME
INTELLIGENT
RECHERCHE
INFORMATION
connaissances
expertes
connaissances
coopération
DONNEES
CONNAISSANCES

- figure V-1 -

Les différentes connaissances sont :
- l'expertise générale d'un expert en recherche d'informations,
- les connaissances sur le domaine (relations sémantiques, niveau de généricité ou de spécificité des termes d'indexation) sont considérées comme des données et stockées dans le thésaurus. Nous les considérons, pour l'instant, comme données car l'utilisation que nous en faisons ne nécessite pas de mécanismes inférentiels (nous nous contentons d'y faire des accès),
- les connaissances sur le SRI (qualité de l'indexation, ...) ne sont pas prise en compte pour le moment,
- les connaissances sur la coopération sont stockées en partie dans la base de connaissances (tout ce qui concerne la planification), alors que les points d'entrée dans le SRI sont stockées dans une base de primitives.

1.1.3. Le fonctionnement du système expert

Le SE a pour but la réalisation des tâches expertes de recherche d'informations ainsi que du processus de coopération. Il doit incorporer de façon harmonieuse des connaissances expertes "classiques" et des appels à des procédures. Nous avons donc choisi un noyau de SE (voir figure V-2) permettant cette intégration, inspiré du modèle des systèmes experts procéduraux [GEO 83].

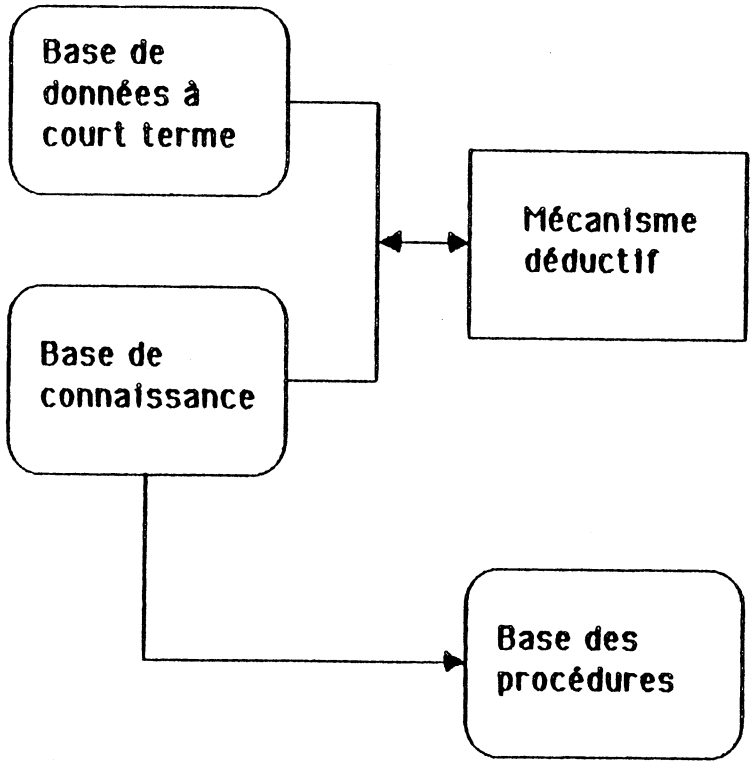
1.1.3.1. La base de connaissances

Nous avons choisi de représenter l'ensemble des connaissances sous la forme de règles de production dans lesquelles on permet l'appel de procédures en partie droite (le problème d'intégration est donc réglé). Le formalisme utilisé est le suivant :

SI <cond> ALORS <action>
où <cond> exprime une condition portant sur les variables d'état du système.

<action> exprime l'action à entreprendre lorsque la partie gauche est vérifiée.
Cette action est une combinaison d'éléments pouvant être :
- des modifications de variables d'états,
- l'ajout de buts à accomplir,
- l'exécution de procédures.

Exemple :



- figure V-2 -

Le fonctionnement du système expert

SI tâche = evaluation-globale ET usager = spécialiste ET nbréponses > 20
ALORS évaluation := pas bonne, tâche := reformulation

Cette règle exprime que si le nombre de réponses est grand, l'usager spécialiste alors la réponse courante n'est pas satisfaisante et la nouvelle tâche à accomplir est la reformulation.

1.1.3.2. La base de données à court terme

Elle décrit l'état courant du système expert. C'est à travers cette structure que tous les éléments communiquent. Elle contient :
- l'ensemble des variables d'états du système, à savoir la requête initiale, l'équation de recherche en l'état courant, l'évaluation de la typologie de l'utilisateur, la tâche courante, ...

- l'ensemble des solutions, chaque solution étant représentée par un triplet formé d'une référence à une entité de structure, de la représentativité estimée de la requête par rapport à l'entité, de la représentativité estimée de l'entité par rapport à la requête,

- la pile des buts à accomplir, c'est elle qui gère les tâches expertes.

1.1.3.3. L'ensemble des procédures

Les noms des procédures utilisables en partie droite des règles sont stockées dans une base de procédures. L'exécution d'une procédure peut produire deux types de résultats :
- une modification de l'ensemble des solutions,
- une modification des variables d'états.

Cette définition implique que l'on fasse un découpage en procédures qui soit atomique vis à vis du processus de coopération : l'exécution du corps d'une procédure doit toujours être indépendante du système expert.

1.1.3.4. Le mécanisme déductif

On a un mécanisme de déduction de type chainage avant, qui correspond au parcours "profondeur d'abord" de l'arbre de résolution. Il fonctionne suivant le schéma classique suivant :

(1) détection de l'ensemble des règles candidates,
(2) choix d'une règle,
(3) application de la partie droite de la règle.

(1) détection de l'ensemble des règles candidates :
Elle se fait par filtrage sur la partie gauche de l'ensemble des règles. Si cet ensemble est vide, deux cas peuvent se produire :
- on s'est trompé dans les choix précédents du SE, il faut donc effectuer un retour arrière à la dernière alternative non explorée. Le problème ici est que ce retour n'est pas toujours possible. En effet si la dernière alternative se trouve dans le traitement d'une tâche déjà achevée, on ne peut revenir en arrière (le choix est irréversible), on est alors en échec. Par contre durant le traitement d'une tâche, le retour arrière est possible.

- on est en échec : on ne peut pas résoudre la requête posée,

(2) choix d'une règle :
C'est la partie la plus importante et la plus délicate (pour les performances) du système déductif. Pour faire un choix dans l'ensemble des règles candidates deux solutions sont à envisager :
- un choix par meta-connaissance : s'il existe une meta-règle résolvant le conflit on l'utilise, sinon on utilise une heuristique par défaut (règle la plus contrainte par exemple).
- un choix par heuristique.

(3) application de la règle choisie :
L'application d'une règle consiste, soit à ajouter un nouveau but, soit à appeler une fonction, soit à mettre à jour les variables d'états.

Le système est initialisé avec un premier ensemble de variables d'états et la pile de buts à vide. Le processus d'arrête sur un cas d'échec.

1.2. Conclusion sur l'architecture

Parmi les différentes possibilités analysées, l'architecture choisie nous permet d'intégrer aisément et de façon modulaire les différentes fonctionnalités désirées du SIRI. La distinction opérée entre fonctions classiques et fonctions expertes, nous permet de garder les avantages respectifs des approches procédurale (efficacité) et déclarative (puissance d'expression, modularité). L'architecture d'un système expert permet d'intégrer de façon harmonieuse les deux types de fonctions en les gérant de manière uniforme au moyen de règles de production réalisant la coopération. Le mécanisme déductif est aussi bien utilisé par les différentes tâches expertes que pour la coopération. Cette approche permet la définition et la réalisation séparées des différentes tâches propres au système d'IA ou au SRI, ce qui autorise une mise au point rapide et aisée.

2. Réalisations et stratégies

Le traitement d'une requête est un processus complexe qui nécessite l'utilisation de nombreuses fonctions. Ces fonctions peuvent être soit classiques (nous les détaillons dans la suite du chapitre), soit expertes c'est à dire réalisant la modélisation d'une activité typiquement humaine (nous les détaillons dans le chapitre suivant).

Le processus défini est dynamique et s'applique différemment d'une requête et d'un utilisateur à un autre. Pour cela, des choix et des stratégies d'actions sont utilisées et reflètent notre vision de la recherche d'informations.

Nous allons tout d'abord présenter les différents modules du système, puis nous donnons et justifions les différentes stratégies mises en œuvre.

2.1. Algorithme général de traitement d'une requête

Le processus de traitement d'une requête peut être vu comme l'exécution séquentielle de différents modules (voir figure V-3). Les étapes sont les suivantes :

(1) Analyse de la requête :
A partir de la requête en langue naturelle, on construit l'équation primaire de recherche qui est une expression booléenne des concepts de la requête.

(1') Traitement des mots inconnus :

Si lors de l'étape précédente on rencontre un certain nombre de mots inconnus, on va chercher à inférer leur catégorie grammaticale et à proposer un certain nombre de possibilités de mots de substitution à l'utilisateur. Celui ci peut alors soit choisir un de ces termes et le traitement continue à l'étape (2), soit choisir de reformuler entièrement sa requête.

(2) Processus de correspondance :

On applique sur chaque concept de l'équation primaire de recherche une fonction de correspondance, qui lui associe une expression booléenne de termes d'indexation. On obtient alors l'équation finale de recherche qui est interprétable.

(3) Evaluation du niveau de dégradation de la requête :

On cherche à déterminer dans quelle mesure les processus de correspondance ou de reformulation ont dégradé l'équation primaire de recherche (cette mesure est utilisée par les processus d'évaluation et de reformulation). Cette évaluation s'effectue donc à chaque modification de l'équation finale de recherche.

(4) Evaluation de la typologie de l'utilisateur :

Le niveau estimé de l'utilisateur dans le domaine et sur le système est évalué à partir des équations primaire et finale de recherche (cette mesure est utilisée par les processus d'évaluation et de reformulation). Cette évaluation est faite une seule fois, à partir de la première équation finale de recherche.

(5) Interprétation de l'équation finale de recherche :

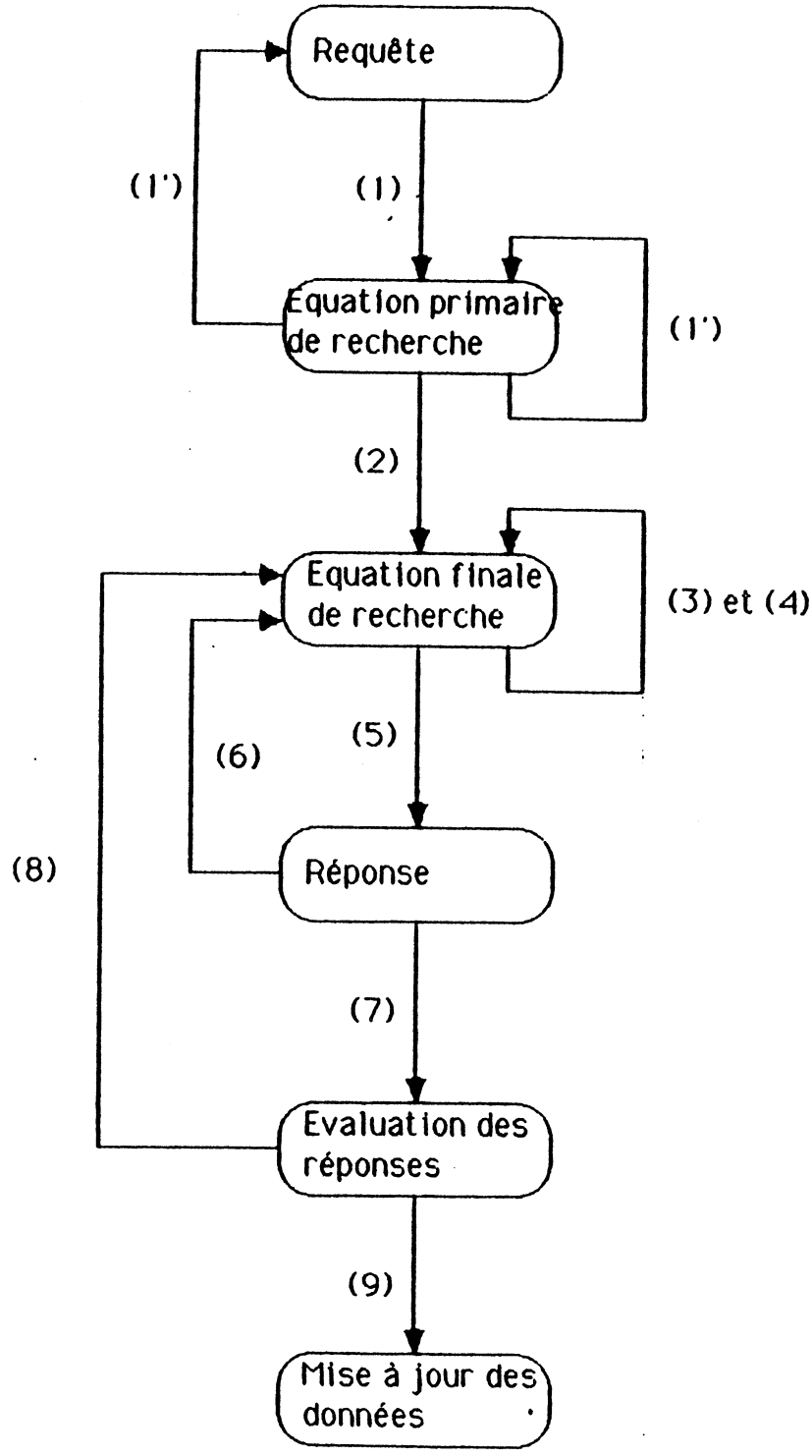
L'équation finale de recherche est interprétée : on détermine l'ensemble des références qui lui correspond (on utilise la relation d'indexation associée à chaque terme d'indexation ainsi que la sémantique des opérateurs logiques).

(7) Evaluation du résultat :

Le résultat obtenu par l'étape (5) est évalué, c'est à dire on cherche à déterminer s'il est conforme à l'attente de l'utilisateur (c'est typiquement une tâche experte). Si le résultat est jugé non satisfaisant, un diagnostic est émis indiquant dans quel sens il faut reformuler l'équation finale de recherche afin d'obtenir un résultat plus conforme.

(8) Reformulation :

A partir du diagnostic émis à l'étape (7), une nouvelle formulation de l'équation finale de recherche est déterminée. Le choix de la nouvelle formulation est heuristique, par conséquent soumis à validation ultérieure (voir étape (6)).



- figure V-3 -

Le fonctionnement de IOTA

(6) Validation de la reformulation :
Cette étape a pour but la validation de la reformulation effectuée. Pour cela elle compare le résultat de l'interprétation après reformulation avec le résultat précédent. Si la variation va dans le sens souhaité, la reformulation est validée et on reprend le traitement à l'étape (7), sinon l'équation précédente est restaurée et on réapplique une nouvelle reformulation (étape (8)).

(9) Apprentissage des inférences réalisées :
Les différentes inférences réalisées (sur les mots ou concepts inconnus) peuvent être apprises à la fin du traitement, si elles ont été jugées profitables. Cet apprentissage correspond à l'enregistrement de ces inférences, qui peuvent ensuite servir à mettre à jour les données (après validation par l'administrateur du système).

2.2. Les stratégies mises en œuvre

Nous avons défini un certain nombre de stratégies visant à améliorer la recherche d'informations effectuée par notre système. Le but de tout le traitement est de trouver, en fonction d'une requête et d'un utilisateur donnés, les inférences qu'il faut effectuer sur celle ci pour produire un résultat le plus satisfaisant possible.
Pour cela diverses stratégies sont envisageables, concernant le moment où faire les inférences (au niveau de la fonction de correspondance ou bien du processus de reformulation), et concernant le mécanisme de contrôle de celles ci :

- quand faire les inférences ?
Nous avons choisi de placer les inférences au niveau du processus de reformulation et non pas au niveau de la fonction de correspondance (la seule inférence qui y est faite est l'utilisation de synonymes). Nous pensons qu'il est plus facile de faire les "bonnes" inférences pendant la reformulation car nous disposons à ce niveau d'informations plus riches qu'au cours de la correspondance. En effet, nous disposons alors des résultats de l'interprétation de l'équation finale de recherche et de leur évaluation, ce qui nous permet d'orienter le sens des inférences à réaliser selon un objectif précis.

- le mécanisme de contrôle des inférences :
Plutôt que de produire l'ensemble des inférences possibles sur une requête et de choisir ensuite la formulation produisant le meilleur résultat, nous avons choisi une stratégie de recherche de la première formulation satisfaisante. Le processus inférentiel n'étant pas déterministe (il n'y a pas une façon unique de reformuler une équation de recherche), la stratégie mise en œuvre est un mécanisme heuristique d'essais et retours arrières éventuels

successifs, jusqu'à la production d'une formulation satisfaisante ou l'épuisement de toutes les formulations (dans ce cas il n'existe pas de formulation satisfaisante).
Cette solution permet de s'assurer que les inférences réalisées amènent réellement un résultat de meilleure qualité que la formulation initiale.
Dans cette optique de dérivations successives à partir d'une formulation donnée, il est important de savoir si les critères de satisfaction vis à vis des résultats sont stables quelque soient les inférences réalisées, ou bien s'ils varient. Dans la mesure où les inférences font s'éloigner la formulation courante de la formulation intiale de la requête, nous considérons que ces critères doivent varier; plus l'éloignement par rapport à la formulation intiale est grand, plus les critères doivent être faciles à atteindre (ce choix a pour but de limiter la dégradation des formulations au détriment peut être de la qualité des résultats).

Nous avons défini également une stratégie d'utilisation des connaissances apprises. Les différentes bases (lexiques, termes d'indexation) ne sont pas mises à jour directement après l'apprentissage (il est dangereux de prévoir une stratégie d'apprentissage purement automatique); les connaissances apprises sont stockées dans une base d'apprentissage et ne sont utilisées pour mettre à jour les bases qu'après validation par l'administrateur du système.

Dans la base d'apprentissage, on stocke chaque mot inconnu (c'est à dire une forme et non pas un lemme puisque qu'aucune analyse morphologique n'a été effectuée sur ce mot), ainsi que le lemme jugé voisin par l'utilisateur. A un concept inconnu, on associe l'expression booléenne générée par le processus de correspondance ainsi que les références associées.
Si un mot inconnu est ajouté dans les lexiques, on enregistre donc ce mot dans le lexique des formes avec comme lemme associé celui stocké dans la base d'apprentissage.
Si un concept inconnu est ajouté comme terme d'indexation, on l'enregistre avec comme relation d'indexation celle de l'expression booléenne jugée équivalente (le résultat du processus de correspondance ayant été appliqué avec succès à ce concept inconnu).
Par exemple, supposons que la base d'apprentissage ait le contenu suivant :

"distribuées" (mot inconnu) ---> "réparti" (adjectif) (lemme associé et catégorie grammaticale)

Après validation par l'administrateur, on enregistre "distribuées" dans le dictionnaire de formes et on lui associe un lien sur "réparti" dans le dictionnaire de lemmes.

"structure détaillée de réseaux locaux" (concept inconnu) ---> ET("structure détaillée","réseaux locaux") (expression booléenne correspondante)

Après validation, le terme d'indexation "structure détaillée de réseaux locaux" est ajouté

dans la base d'indexation et on lui associe le sous-ensemble de la relation d'indexation correspondant à l'interprétation de l'expression booléenne donnée comme équivalente.

La base d'apprentissage est néanmoins utilisée pendant le traitement d'une requête pour éviter de refaire des actions déja entreprises dans la même situation. Elle est donc utilisée en cas de mot inconnu (on évite de refaire un traitement qui peut être très long, et on propose d'abord à l'utilisateur ce que l'on a appris précédemment), ainsi que dans le cas du concept inconnu.

3. L'implantation de IOTA

Le prototype IOTA [CHI 86a], [CHI 86b] est codé complètement en LELISP (dialecte LISP de l'INRIA [CHA 85]) sur un VAX 11/780 UNIX 4.2 BSD. Toutes les fonctions du système sont écrites en LELISP, ainsi que toutes les connaissances expertes et toutes les données (sauf le texte des documents qui est stocké dans des fichiers externes). Le volume occupé par IOTA est par conséquent très important ce qui est un obstacle sérieux à son transport sur d'autres machines (comme des micro-ordinateurs par exemple), mais par contre l'accès aux données est très rapide (gestion en mémoire centrale). L'ensemble des fonctions de IOTA représente environ 3000 lignes de LELISP réparties de la façon suivante :

- analyseur morpho-syntaxique : 200 lignes
- fonction de correspondance : 360 lignes
- interprétation de l'équation finale de recherche : 360 lignes
- processus de reformulation : 500 lignes
- traitement des mots inconnus et apprentissage : 250 lignes
- moteur d'inférences : 300 lignes
- fonctions diverses (gestion du terminal, ...) : 1000 lignes

Il est bien évident que le prototype réalisé est expérimental et vise essentiellement à valider qualitativement un certain nombre d'idées et d'hypothèses quant à la recherche d'informations. De nombreuses améliorations visant à l'efficacité sont nécessaires (notamment la compilation du code source). Nous avons cependant tenté de profiter au mieux des possibilités offertes par l'outil LELISP disponible.

Nous décrivons tout d'abord les données et connaissances nécessaires à IOTA, ainsi que leur implantation en LELISP, puis nous décrivons les différentes fonctions composant le prototype.

3.1. Les données et connaissances utilisées

3.1.1. Les données

Les différentes données manipulées par IOTA sont les textes, les dictionnaires (utilisés par l'interface utilisateur), la base de connaissance et la base d'apprentissage. Le modèle de données utilisé est simple :

- le corpus est considéré comme un ensemble de textes,

- chaque texte est stocké selon sa structure logique (c'est à dire une hiérarchie de chapitres, ...). Nous appelons entité de structure les éléments correspondants à n'importe quel sous arbre de la structure logique,

- à chaque entité de structure est associé un titre (s'il existe), un ensemble de termes d'indexation (qui représentent le contenu de l'entité), et le fragment de texte correspondant à l'entité,

- le vocabulaire utilisé par l'interface utilisateur est stocké dans deux dictionnaires, l'un contenant l'ensemble des formes pouvant être utilisées dans les requêtes (mots avec leurs variantes grammaticales), et l'autre contenant l'ensemble des mots lemmatisés (on associe à un ensemble de formes un lemme qui est la représentation canonique de ces formes). Le dictionnaire de lemmes permet la factorisation de toutes les variantes grammaticales d'un mot donné ainsi que d'éventuels synonymes.

Exemple :

"bateau" peut être choisi comme lemme représentant "bateau", "bateaux", "navire", "navires", ...

- la base de connaissances est composée des relations sémantiques sur les termes d'indexation. Ces relations sont décrites de façon hiérarchique (généricité, spécificité) ou non (voisinage),

- la base d'apprentissage est un ensemble de formes et concepts inconnus. A chaque forme inconnue est associé un lemme connu (résultat de l'inférence du mot inconnu), et à chaque concept inconnu est associée une expression booléenne de termes d'indexation (résultat de la fonction de correspondance).

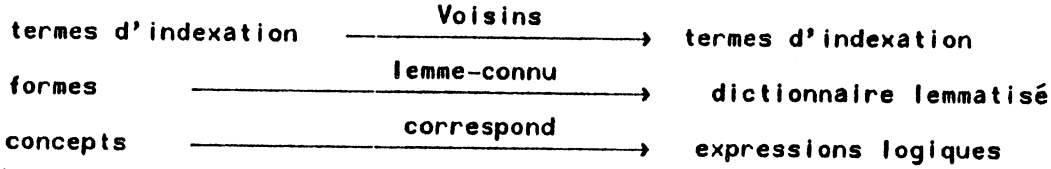
Nous donnons ci dessous la description précise du système d'informations constitué par les différentes données utilisées par IOTA. Nous donnons cette description selon le modèle Z0 [ABR 74] qui permet de décrire des ensembles d'objets et les associations les liant. Ce modèle peut être vu comme un graphe orienté où les ensembles sont des sommets et les arcs étiquetés et orientés sont les fonctions de dépendance exprimant l'association. Les étiquettes définissent le nom des fonctions de dépendance (un nom commençant par une lettre majuscule dénote une association multivaluée alors qu'une lettre minuscule dénote une association monovaluée).

Ensembles d'entités :

dictionnaire de forme = {formes}
dictionnaire lemmatisé = {lemmes}
base d'indexation = {termes d'indexation}
pondération relation d'indexation = {(référence textuelle, p∈[0,1], q∈[0,1])}
références = {références textuelles}
fragment de textes = {textes pleins}
titre = {titres}
forme = {mots inconnus}
concept = {concepts inconnus}
expressions logiques = {expression booléenne de termes d'indexation}

Associations :

dictionnaire de forme —lemme→ dictionnaire lemmatisé
dictionnaire lemmatisé —Formes→ dictionnaire de forme
dictionnaire lemmatisé —Termes→ base d'indexation
dictionnaire lemmatisé —Contient→ base d'indexation
base d'indexation —Réfère-à→ pondération relation d'inde
références —réfère-à→ fragments de texte
références —Fils→ références
références —père→ références
références —titre-de→ titres
termes d'indexation —générique→ termes d'indexation
termes d'indexation —Spécifiques→ termes d'indexation



Tous les ensembles et les fonctions sont décrits en LISP (hormis les textes pleins qui sont sur fichiers externes). Les éléments de chaque ensemble sont des atomes LISP et les associations sont matérialisées par des listes de propriétés associées à chaque atome. Cette représentation est très efficace en temps d'accès, et LISP garantit une bonne optimisation de l'espace mémoire.

3.1.2. Les connaissances

IOTA manipule un certain nombre de connaissances expertes décrites sous forme de règles de production. On distingue deux grands types de connaissances :
- les connaissances sur la coopération :
Ce sont les règles réalisant l'enchaînement des différents modules de IOTA.

- les connaissances expertes :
Ce sont les règles décrivant le comportement de l'expert en recherche d'informations (cf VI). Ces règles sont découpées en classes correspondant aux différentes tâches à réaliser (évaluation, reformulation, apprentissage). Cette partition permet d'optimiser le processus de filtrage des règles actives, car elle diminue le nombre de règles candidates.

Le nombre de règles dont nous disposons est, pour l'instant, peu élevé (une quarantaine) mais est susceptible d'évoluer rapidement en fonction des expérimentations futures.

Nous avons déjà dit (cf 1.1.3.1) que le formalisme choisi était celui des règles de production; la matérialisation en LISP de ces règles est assez simple. Nous avons choisi de représenter une règle par un atome LISP ayant pour nom le nom de la règle. On associe à cet atome deux listes de propriétés : une décrivant la partie condition, et l'autre la partie action.
La partie condition est une expression logique LISP, c'est à dire une s-expression s'évaluant soit à t (vrai), soit à nil (faux). Les opérateurs utilisés dans cette partie sont donc les opérateurs logiques (AND, OR, NOT), les opérateurs relationnels (<, >, =, ...), et des fonctions utilisateur à valeur dans (t,nil).
Pour savoir si la partie condition d'une règle est vérifiée, il suffit donc d'évaluer la liste de propriété CONDITION de l'atome correspondant (si elle est évaluée à t, la règle est filtrée,

sinon elle est rejetée).

La partie action est également une s-expression LISP, composée d'une liste de fonctions à exécuter.

Si on considère l'exemple de règle :
SI <usager spécialiste> ET <référence pertinente>
ALORS <ajouter 1 au nombre de références jugées bonnes>
Cette règle s'écrit alors (nous supposons que son nom est r1) :
(putprop 'r1 '((and (> typologie 0.66) (> (rep reference) 0.8))) 'condition)
(putprop 'r1 '((+ 1 nbreponsebonnes)) 'action)
Dans cette définition, nous avons instancié les meta-symboles tels que <usager spécialiste> par des actions sur des variables d'états manipulées par IOTA (voir au chapitre 6 la définition complète de ces évaluations).

Ce type de représentation est à la fois simple et efficace (c'est un choix classique de représentation de règles de production en LISP). Les règles doivent être directement décrites dans le formalisme LISP, nous n'avons pas défini d'outils de traduction d'une règle de production vers la représentation choisie.

3.2. Les différents modules réalisés

Nous décrivons ici de manière détaillée les principaux modules utilisés par IOTA (hormis les connaissances expertes qui sont décrites dans le chapitre suivant).

3.2.1. L'analyseur de requête

L'analyse de la requête a pour but le passage d'une forme syntaxiquement riche et potentiellement ambiguë (une requête en langue naturelle), à une forme syntaxiquement simple et non ambiguë (l'équation primaire de recherche), qui est une expression de recherche sur la base.

Nous définissons tout d'abord le langage d'interrogation accepté, puis la syntaxe de l'équation primaire de recherche, et enfin le processus de transformation.

3.2.1.1. Le langage d'interrogation

Le langage d'interrogation est le français, mais son interprétation est restreinte à la reconnaissance d'une expression de recherche sur des groupes nominaux.

Les mots employés doivent être connus du système, c'est à dire être présents dans les lexiques (un processus d'inférences sur les mots inconnus est utilisé, mais il ne fonctionne convenablement que si l'environnement du mot inconnu dans la requête est connu, cf V.3.2.2).
Nous distinguons les opérateurs de recherche des autres mots, et nous voyons chaque requête comme une succession de fragments de phrase séparés par des opérateurs de recherche.
Dans l'état actuel du prototype, les opérateurs de recherche sont restreints aux opérateurs booléens classiques (ET, OU, SAUF), mais leur extension à des opérateurs plus riches sémantiquement (opérateur de causalité par exemple) est à l'étude.

Au niveau lexical, la requête est donc composée de mots appartenant à deux catégories, les opérateurs de recherche et les autres.

OPERATEUR ::= ET / OU / SAUF
MOT ::= suite de caractères non blancs

3.2.1.2. L'équation primaire de recherche

L'équation primaire de recherche est la première étape vers la "compréhension" par le système de la requête. On peut la considérer comme une expression booléenne des concepts extraits de la requête (elle est encore très proche de la requête).
Sa syntaxe est définie de la façon suivante :

EPR ::= CONCEPT [OP EPR]*
OP ::= ET / OU / SAUF
CONCEPT ::= GROUPENOMINAL (voir processus de transformation)

3.2.1.3. Le processus de transformation

Le but de ce processus est de reconnaitre dans la requête l'ensemble des concepts présents et les opérateurs logiques qui les lient. Dans l'absolu cet objectif est très ambitieux car de nombreux problèmes linguistiques sont à traiter (c'est un problème de compréhension d'une phrase) nécessitant des outils syntaxiques puissants ainsi qu'une analyse sémantique permettant la levée d'un certain nombre d'ambiguités.

Dans le cadre d'un SRI de telles approches sont difficilement envisageables étant donnée l'étendue du domaine couvert. Nous avons donc choisi une approche heuristique [BOG 82], [GUI 83] qui vise à une compréhension superficielle de la phrase à l'aide d'un minimum de sémantique et d'une analyse morpho-syntaxique de surface, et non pas à la compréhension totale de la requête [SCH 80], [EIS 85].
Bien entendu, par cette approche la compréhension risque de produire des résultats erronés dans un certain nombre de cas, et ce sera à l'utilisateur de rectifier les erreurs d'interprétation.
L'analyse consiste à chercher dans la requête les différents concepts qui s'y trouvent ainsi que les opérateurs logiques les liant. Nous avons pour cela défini syntaxiquement un concept comme un sous ensemble du syntagme nominal (de nombreuses études réalisées montrent que les éléments les plus porteurs d'informations sont les syntagmes nominaux), et nous ne considérons que les opérateurs logiques qui apparaissent explicitement dans la requête.
Les hypothèses utilisées sont donc suffisament restrictives pour nous conduire à un analyseur simple, ne produisant pas de formes ambiguës.
Le processus de transformation de la requête en une équation primaire de recherche consiste donc en une analyse morpho-syntaxique suivie de transformations :

- fragmentation de la phrase en mots et détermination de leur catégorie grammaticale (cette phase est réalisée à partir d'un lexique des formes connues auquel est associé un lexique des lemmes donnant la représentation canonique de la forme et sa catégorie grammaticale),

- fragmentation de la phrase en segments autour des opérateurs logiques reconnus et vérification de l'arité de ces opérateurs,

- analyse syntaxique de chaque segment de phrase afin d'en extraire tous les éléments vérifiant la syntaxe des groupes nominaux (si plusieurs groupes nominaux sont extraits d'un même segment, ils sont supposés connectés par un ET). Cette dernière phase de traitement comporte donc éventuellement des transformations.

La syntaxe des groupes nominaux est la suivante :

GROUPENOMINAL ::= GROUPESIMPLE [PREP GROUPESIMPLE]*
GROUPESIMPLE ::= NOMINAL [ADJECTIF]* / [ADJECTIF]* NOMINAL
NOMINAL ::= NOM / NOM NOM
ADJECTIF ::= adjectifs
NOM ::= noms
PREP ::= prépositions

La technique d'analyse utilisée est celle des automates d'états finis. L'automate est activé à chaque début potentiel de groupe nominal (à savoir un nom ou un adjectif). Dès qu'une des trois règles (GROUPENOMINAL, GROUPESIMPLE, NOMINAL) est vérifiée, on a reconnu un groupe (on cherche à reconnaitre le groupe le plus long, donc on cherche à vérifier la règle GROUPENOMINAL tant qu'il n'y a pas d'échec).
La règle GROUPENOMINAL permet de reconnaitre des groupes très long; ceux-ci correspondront rarement à des termes d'indexation qui sont eux constitués au plus de deux groupes simples. Nous avons donc introduit une première règle de transformation des groupes reconnus, destinée à décomposer les groupes trop longs en une suite de groupes nominaux comportant au plus deux groupes simples connectés par l'opérateur logique ET. Cependant, pour éviter de perdre trop d'informations, cette décomposition s'accompagne d'un recouvrement partiel des groupes :

GS_1 PREP GS_2 ... PREP GS_n -->
GS_1 PREP GS_2 ET GS_2 PREP GS_3 ... ET GS_{n-1} PREP GS_n
où GS_i est une instance de GROUPESIMPLE

Par exemple, si on a reconnu le GROUPENOMINAL "support de documentation d'un autocommutateur", il est transformé en "support de documentation" ET "documentation d'un autocommutateur".

Dans l'équation primaire de recherche, les groupes reconnus subissent ensuite une seconde transformation permettant de simplifier les étapes de traitement ultérieurs : chaque groupe simple est mis sous la forme noms suivis des adjectifs éventuels.

ADJ* NOMINAL --> NOMINAL ADJ*

Par exemple le groupe simple "jolie petite maison" est transformé en "maison jolie petite".

Exemple d'analyse complète de requête :
Considérons la requête
"je veux tout ce qui concerne les grandes bases de données distribuées
et les réseaux locaux"
l'équation primaire de recherche produite est alors :
ET("base de données distribuées grandes","réseaux locaux")

Il est certain que la technique mise en œuvre est approximative dans un nombre assez important de cas; la solution proposée ici ne constitue qu'une première étape destinée à faciliter la mise au point du système. Un autre analyseur beaucoup plus puissant a été réalisé l'année dernière dans notre groupe par J.NIE [NIE 85]. Il permet la reconnaissance dans la requête des attributs externes (auteur, type de documents, ...) et de contenu (les thèmes de recherche), ainsi que la résolution heuristique d'un certain nombre de problèmes linguistiques (antécédent d'un pronom relatif, nominalisation des verbes significatifs, ...) permettant un niveau beaucoup plus fin de compréhension de la requête. Une autre particularité consiste en une meilleure interprétation des opérateurs de recherche à partir des constructions syntaxiques (interprétation des conjonctions et de la ponctuation notamment).
Ce prototype, réalisé en PROLOG doit être complété et réécrit en LELISP pour être intégré dans une prochaine version du système comme générateur de l'équation primaire de recherche.

3.2.2. Inférences de mots inconnus

Les SRI gérant par définition des domaines ouverts, il n'est pas possible de définir des lexiques contenant tous les mots de ces domaines. Un certain nombre de mots n'appartenant pas aux lexiques peuvent donc être rencontrés dans une requête. Dans ce cas, plutôt que de demander à l'utilisateur de définir le ou les mots inconnus, on cherche à inférer le plus de choses possibles concernant ces mots. Etant donné le peu d'informations dont on dispose (modèle sémantique peu riche, et environnement du mot dans la requête), les inférences réalisées sont forcément limitées. On peut cependant essayer d'inférer la catégorie grammaticale du mot inconnu, et, dans certains cas, proposer à l'utilisateur un ensemble de termes de substitution possibles. L'utilisateur peut alors soit choisir une des propositions (et le traitement de la requête peut continuer), soit reformuler entièrement sa requête.

Tout le processus d'inférences est basé sur le contexte du mot inconnu (c'est à dire les mots qui lui sont syntaxiquement liés dans la requête). Si ce contexte est vide, on ne peut donc pas faire d'inférences.
En ce qui concerne l'inférence de la catégorie grammaticale du mot, on utilise le fait qu'un certain nombre de catégories sont soit parfaitement connues (mots-outil : cas des prépositions, des conjonctions par exemple), soit peu intéressantes pour notre analyse de la requête (verbes, adverbes).
Deux catégories restent donc possibles, les substantifs et les adjectifs. Le choix entre les deux catégories va s'effectuer par référence au contexte et aux termes d'indexation (on aurait également pu utiliser les règles d'accord en genre et en nombre qui peuvent résoudre un certain nombre d'ambiguïtés, mais nous ne les avons pas encore intégrées dans notre prototype).

La proposition d'un ensemble de mots de substitution s'effectue après l'analyse syntaxique de la requête. On effectue une analyse syntaxique par catégorie grammaticale proposée. Pour chaque analyse, deux cas peuvent se présenter :

- le mot inconnu n'apparait pas dans les résultats de l'analyse ou bien il apparait comme un mot isolé : on ne peut faire aucune inférence (le contexte reconnu est vide),
- le mot inconnu apparait dans un ou plusieurs groupes reconnus, on choisit alors le groupe reconnu le plus long (plus le contexte est large, plus l'inférence réalisée est précise), et on confronte ce groupe avec l'ensemble des termes d'indexation (on utilise la même fonction de correspondance que celle décrite en V.2.2.3, à la substitution du mot inconnu par un mot de même catégorie grammaticale près). On propose à l'utilisateur tous les mots se substituant au mot inconnu (si il y en a).

Exemple :
Soit la requête :
"Donnez moi les livres parlant de bases de données réparties et de réseaux locaux"
Supposons que le mot "réparties" soit inconnu. On lui affecte donc les catégories possibles substantif et adjectif.
(a) Groupes reconnus avec la catégorie substantif :
"livres", "bases de données réparties (sub)", "réseaux locaux"
on cherche ensuite tous les termes d'indexation correspondant à "base de données (sub)"
(b) Groupes reconnus avec la catégorie adjectif :
"livres", "bases de données réparties (adj)", "réseaux locaux"
on cherche ensuite tous les termes d'indexation correspondant à

"base de données (adj)"
Il est probable que si la base parle d'informatique, on va trouver des solutions pour la catégorie adjectif (par exemple "relationnelles", "distribuées", "déductives", ...) et non pas pour la catégorie substantif.

On propose donc à l'utilisateur au plus deux ensembles de mots correspondant chacun à une des catégories possibles.

Si l'utilisateur valide une des propositions, le mot inconnu peut être appris à la fin du traitement de la requête (cf VI.2.3).

3.2.3. La fonction de correspondance

A partir de l'équation primaire de recherche, le problème est d'obtenir une équation interprétable. Pour cela, il faut que tous les groupes reconnus composant l'équation soient des termes d'indexation. Comme il est peu probable que cette condition soit vérifiée d'emblée, nous devons définir un processus de correspondance entre un groupe de la requête et l'ensemble des termes d'indexation.

Dans l'absolu ce processus de correspondance doit fournir "l'équivalent sémantique" du groupe en fonction des termes d'indexation, mais cela nécessiterait une base de connaissances beaucoup plus complète.
Pour notre part, nous essayons de réaliser "l'équivalence sémantique" à l'aide de règles fondées sur une correspondance syntaxique.
L'algorithme de correspondance utilisé est le suivant (un exemple complet est décrit figure V-4 et sa notation algorithmique est donnée figure V-5) et s'applique sur chaque groupe de l'équation primaire de recherche :

(1) phase d'inclusion initiale :
Nous construisons l'ensemble IND des termes d'indexation incluant totalement le groupe de la requête. L'inclusion est basée sur l'identité des mots composants le groupe et le terme d'indexation. Trois cas se présentent :
(a) IND est vide :
Aucun terme d'indexation ne contient l'ensemble des mots du groupe : le groupe reconnu est inconnu du système. Nous sommes donc amenés à transformer le groupe pour le décrire en fonction des termes d'indexation (voir phase (2)).

(b) IND contient le groupe :

Le groupe est un terme d'indexation, la correspondance est donc immédiate.

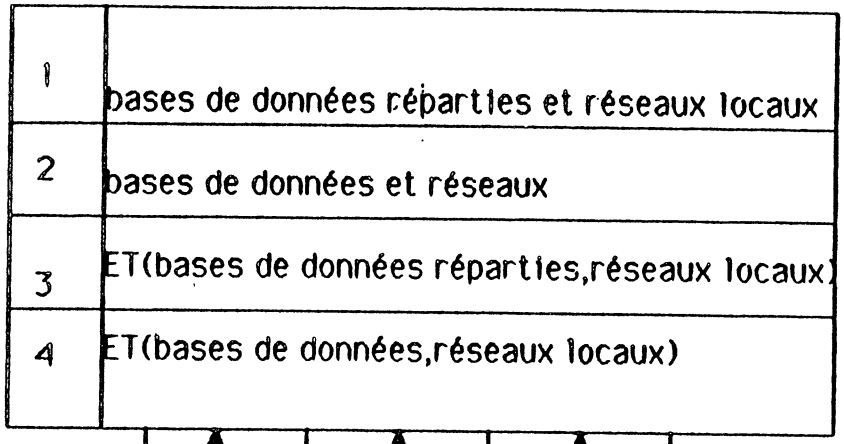
(c) IND ne contient pas le groupe :
Un ou plusieurs termes d'indexation recouvrent le groupe. Nous allons chercher dans IND l'ensemble INC1 des termes d'indexation très voisins du groupe, c'est à dire incluant syntaxiquement celui ci. L'inclusion syntaxique est définie de la façon suivante :
un groupe G1 est inclus syntaxiquement dans un groupe G2, si et seulement si :
- le squelette de G1 (groupe privé de ses adjectifs) est inclus (au sens des listes) dans le squelette de G2,
- la liste d'adjectifs associée à chaque substantif de G1 est incluse (au sens des ensembles) dans celles du même substantif de G2.
Par exemple, le groupe "structure d'une base de données distribuées" inclut syntaxiquement les groupes "base de données distribuées", "structure d'une base de données", "base de données", ... mais n'inclut pas "structure détaillée d'une base de données", "base d'une structure", ...

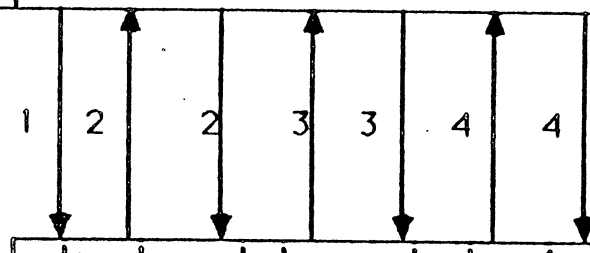
Si INC1 est vide, nous passons à la phase (2), sinon le premier des plus petits éléments de INC1 est choisi comme terme d'indexation le plus proche du groupe initial (nous n'avons pas de critères objectifs de choix).

(2) deuxième phase d'inclusion :
La première phase de transformation du groupe consiste à lui enlever ses adjectifs (c'est la transformation qui dégrade le moins le groupe). Pour cela, on construit l'ensemble J des termes d'indexation contenant l'ensemble des mots du squelette du groupe. Là, deux cas sont à considérer :
- J est vide :
La dégradation appliquée n'est pas suffisante, il va falloir transformer plus profondément le groupe (voir phase (3)).

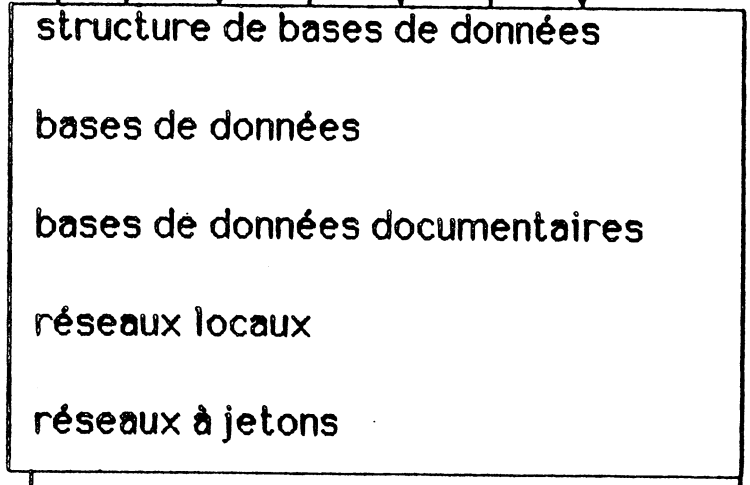
- J n'est pas vide :
Nous cherchons dans J l'ensemble INC2 des termes d'indexation incluant syntaxiquement le squelette du groupe (il faut noter qu'on ne tient pas du tout compte des adjectifs composant le groupe).
Si cet ensemble est vide nous passons à la phase (3), sinon le premier des plus petits éléments de INC2 est choisi comme terme d'indexation le plus proche du groupe initial.

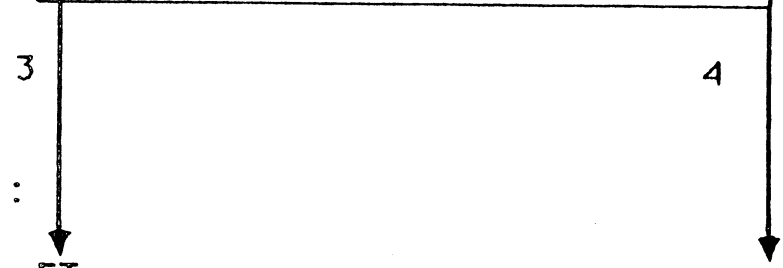
Concepts :



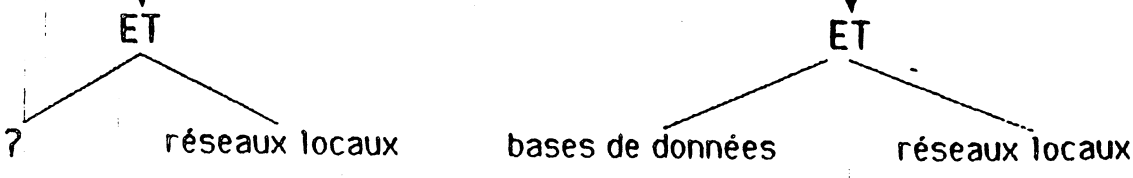


Termes d'indexation





Equation correspondante :



- figure V-4 -

Un exemple de correspondance

```
Fonction correspondance (groupe) --> expression booléenne
|
|  IND <-- termes d'indexation incluant groupe
|  INC1 <-- termes d'indexation incluant syntaxiquement groupe
|
|  choix
|  |
|  |  IND ≠ vide et groupe ∈ IND : --> groupe
|  |
|  |  IND ≠ vide et groupe ∉ IND et INC1 ≠ vide :
|  |              --> premier des plus petits éléments de INC1
|  |
|  |  (IND ≠ vide et groupe ∉ IND et INC1 = vide) ou IND = vide :
|  |              /* il faut transformer groupe */
|  |
|  |              J <-- termes d'indexation incluant squelette(groupe)
|  |              INC2 <-- termes d'indexation incluant syntaxiquement
|  |                   squelette(groupe)
|  |
|  |              choix
|  |              |
|  |              |  J ≠ vide et INC2 ≠ vide :
|  |              |    /* le retrait des adjectifs est une transformation
|  |              |    suffisante */
|  |              |    --> premier des plus petits éléments de INC2
|  |              |
|  |              |  J = vide ou INC = vide :
|  |              |    /* il faut faire une décomposition syntaxique */
|  |              |    --> ET(pourtout élément de décomposition(groupe) :
|  |              |          correspondance(élément))
|  |              |
|  |              fin choix
|  fin choix
fin fonction correspondance
```

- figure V-5 -

L'algorithme de correspondance

(3) phase de décomposition syntaxique :

Pour assurer la correspondance entre le groupe et les termes d'indexation, nous sommes conduits à opérer une décomposition syntaxique de ce groupe. En effet, si la correspondance n'est pas vérifiée, cela provient du fait que le groupe est trop long. Il faut donc le décomposer afin d'augmenter les possibilités de correspondance.

Nous utilisons pour cela, un ensemble de règles de cassure syntaxique indiquant, pour chaque modèle syntaxique de squelette, les décompositions syntaxiquement correctes :

GS_1 PREP GS_2 --> ET(GS_1,GS_2)

NOM_1 NOM_2 --> NOM_1

Les adjectifs qui n'ont pas été pris en compte dans la décomposition sont ensuite rajoutés aux sous-groupes générés, et les nouveaux groupes ainsi constitués sont remis en entrée du processus de correspondance (phase (1)).

Par exemple, le groupe "structure détaillée d'une machine séquentielle" est transformé par le processus de décomposition syntaxique en :

"structure détaillée" et "machine séquentielle"

La perte de la liaison prépositionnelle est compensée (en partie seulement) par l'ajout d'un opérateur ET sur les sous-groupes générés (dans notre exemple on obtient donc ET("structure détaillée","machine séquentielle")).

Le processus de décomposition continue récursivement jusqu'à ce que la correspondance réussisse pour chaque sous-groupe généré (on est certain d'y parvenir puisqu'on décompose au plus jusqu'au niveau d'un substantif isolé, qui correspond forcément à un terme d'indexation, le vocabulaire connu étant celui des termes d'indexation).

Le processus de correspondance fournit donc comme résultat soit le terme d'indexation le plus proche du groupe initial, soit la conjonction de termes d'indexation correspondant le plus à celui-ci.

3.2.4. Le processus d'interprétation

Ce processus comprend deux étapes :

(1) accès à la liste des références correspondant à chaque groupe de l'équation finale de recherche :

Il s'agit de substituer à chaque groupe (ou terme d'indexation) sa relation d'indexation. On

obtient alors une expression booléenne de listes de références (l'arbre d'interprétation).

(2) composition de la réponse et évaluation de la pertinence des références :
Elles se font en fonction de la sémantique associée aux opérateurs booléens. Le résultat final est une liste de références (éventuellement vide) satisfaisant à l'équation finale de recherche, avec leurs mesures de pertinence associées (représentativités requête - référence et référence - requête).

Nous décrivons tout d'abord les algorithmes liés aux opérateurs logiques et nous donnons ensuite la définition des fonctions de pondération qui leur sont associées.

3.2.4.1. Les algorithmes de composition du résultat

Il est important de souligner que, les références désignant des entités de la structure logique des documents, l'interprétation des opérateurs logiques doit prendre en compte la relation d'inclusion qui peut exister entre deux entités.
L'algorithme général va être un parcours postfixé de l'arbre d'interprétation. Durant le traitement, on associe à chaque nœud non terminal le résultat correspondant au sous arbre dont il est racine, sous la forme d'un couple nombre de références, représentativité moyenne références - requête et requête - références. Cette décoration est ensuite utilisée durant le processus de reformulation (cf VI.2.2).

a) Conjonction de références : ET
Soient L1 et L2 des listes de références. ET(L1,L2) correspond à la liste des sous-arbres qui appartiennent à la fois à L1 et L2. Ces sous-arbres sont déterminés par l'algorithme suivant (a_1 est un élément de L1, a_2 est un élément de L2) :

- $a_1 = a_2$: a_1 appartient au résultat,
- a_1 est un sous-arbre de a_2 : a_1 appartient au résultat,
- a_1 et a_2 sont des arbres disjoints : ni a_1 ni a_2 appartiennent au résultat.

b) Disjonction de références : OU
OU(L1,L2) correspond à la liste des sous-arbres appartenant soit à L1, soit à L2, soit aux deux. L'algorithme de construction du OU est le suivant (a_1 et a_2 ont la même signification que précédemment) :

- $a_1 = a_2$: a_1 appartient au résultat,
- a_1 est un sous-arbre de a_2 : a_2 appartient au résultat,
- a_1 et a_2 sont des arbres disjoints : a_1 et a_2 appartiennent au résultat.

c) Négation d'une référence : SAUF

SAUF(L1,L2) correspond à la sous liste de L1 dont les éléments n'incluent aucun élément de L2; l'algorithme de construction du SAUF est le suivant :

- $a_1 = a_2$: a_1 n'appartient pas au résultat,
- a_1 est un sous-arbre de a_2 : a_1 n'appartient pas au résultat,
- a_2 est un sous-arbre de a_1 : a_2 est enlevé de a_1, le résultat est l'ensemble des sous-arbres de plus haut niveau de a_1 qui ne contiennent pas a_2 ("élagage" de a_1),
- a_1 et a_2 sont disjoints : a_1 appartient au résultat.

3.2.4.2. Les fonctions de pondération

Elles ont pour but l'évaluation de la pertinence des références résultat et sont donc, comme les mesures de représentativité terme - document et document - terme (rep^1 et rep^2 respectivement, cf VI.3.3), définies sur [0,1].
Le principe de l'évaluation est de considérer, étant données deux références R1 et R2 avec leurs mesures associées (rep_1^1,rep_1^2) et (rep_2^1,rep_2^2), la fonction Fop qui fournit comme résultat la mesure de pertinence associée à op(R1,R2).
Cette fonction dépend bien entendu de l'opérateur "op".

(a) Fonction de pondération associée au OU : Fou

Les fonctions choisies doivent avoir les mêmes propriétés que l'opérateur booléen auquel elles sont rattachées. Pour le OU, les propriétés à respecter sont la commutativité et l'associativité.
La fonction doit également refléter la sémantique de l'opérateur; on peut considérer le OU comme un opérateur partitionnant la requête en deux sous-requêtes, le résultat étant l'union des résultats de chaque sous-requête. Pour les références produites par l'une des deux sous-requêtes, la mesure résultante ne fait intervenir que la mesure résultat de la sous-requête, par contre pour les références produites par les deux sous-requêtes la mesure résultante est la plus grande de celles des deux sous-requêtes. Nous avons choisi de ne pas donner un effet de renforcement au OU lorsque les deux opérandes sont présents dans la même entité. Il nous semble que si l'utilisateur voulait que l'on attache une pertinence plus grande à une entité vérifiant cette propriété, il aurait formulé sa requête avec un ET. S'il ne

l'a pas fait, c'est qu'il veut atteindre un nombre plus grand de références en spécifiant des synonymes d'un même concept. De plus, un renforcement implique qu'une requête ne portant que sur un des deux concepts fournit une pertinence toujours inférieure, ce qui ne parait pas très normal.

$$Fou(R1,R2) = MAX [rep_1^i, rep_2^i]$$
avec i = 1,2

b) Fonction de pondération associée au ET : Fet

Les propriétés à respecter sont la commutativité et l'associativité. La sémantique du ET est plus difficile à mettre en évidence. Le ET fait jouer un rôle symétrique aux deux opérandes, la fonction choisie doit donc toujours donner un poids identique à chacun des opérandes. Nous avons utilisé pour cela une mesure donnant un effet de moyenne sur les opérandes :

$$Fet(R1,R2) = (rep_1^i * rep_2^i) / (1 - rep_1^i - rep_2^i + 2 * rep_1^i * rep_2^i)$$
avec i = 1,2

Il faut noter que cette fonction n'est pas idempotente (c'est à dire $Fet(p,p) \neq p$). Cela signifie que $Fet(R1,R1)$ est différent de rep_1^i. Si cela est génant au point de vue théorique, il est peu probable que ce phénomène se produise au niveau d'une requête (une requête bien formulée ne fait pas intervenir deux fois le même opérande pour un opérateur donné).

c) Fonction de pondération associée au SAUF : Fsauf

Le SAUF est bien entendu non commutatif. La sémantique du SAUF implique que le deuxième opérande soit absent de la référence résultat. La fonction d'évaluation choisie ne tient donc pas compte du deuxième opérande :

$$Fsauf(R1,R2) = rep_1^i$$
avec i = 1,2

3.2.5. Reformulation automatique

3.2.5.1. Introduction

Le processus de reformulation automatique a pour but de modifier l'équation finale de recherche afin de fournir à l'utilisateur un ensemble de références plus conforme à ses besoins.
Ce processus utilise le diagnostic fourni par le processus d'évaluation, qui peut être (cf VI-2-2-4) :
- le nombre de références est trop faible,
- le nombre de références est trop grand,
- la qualité des références est trop faible.

Pour résoudre ces problèmes, l'équation finale de recherche peut être modifiée de deux façons :
- on modifie les concepts de cette équation (c'est à dire on substitue à un concept, un ou plusieurs concepts tirés de ses relations sémantiques).

- on modifie la logique de cette équation (c'est à dire on enlève des opérateurs logiques). Cette modification consiste à supprimer un opérande d'un OU (on considère que cela ne change pas fondamentalement l'équation).

Une fois le type de modification choisi, il faut déterminer les modalités de la modification proprement dite, qui ne sont pas toujours parfaitement évidentes a priori (notamment si on utilise des relations sémantiques, quel niveau de généricité applique-t'on sur le concept à reformuler par exemple). La modification dégradant par définition l'équation finale de recherche, il faut mettre à jour le niveau de dégradation de la requête.
Une fois ces diverses opérations effectuées, on peut reprendre la séquence de traitement d'une requête avec la nouvelle équation finale de recherche (avant le processus d'interprétation). Avant de reprendre l'étape d'évaluation, il faut s'assurer que la reformulation choisie a bien amélioré le résultat dans le sens souhaité (validation de la reformulation).

On peut résumer le processus de reformulation de la façon suivante :

(1) à partir du diagnostic et de l'équation finale de recherche
choix de la modification à apporter
(2) choix du niveau de modification
(3) réalisation de la modification
(4) mise à jour du niveau de dégradation
(5) retour au processus d'interprétation

(6) validation de la reformulation

3.2.5.2. Le processus de reformulation

Nous allons maintenant détailler tour à tour chacun des cinq points mis en évidence.

(1) Choix de la modification à effectuer :

Elle est fonction du problème à résoudre et de l'équation finale de recherche à traiter. Il nous faut tout d'abord déterminer quelles sont les modifications possibles pour un problème donné :

- trop de références :

Pour diminuer le nombre de références résultat, on peut soit modifier la logique de l'équation, soit reformuler un concept par ses spécifiques (par hypothèse, plus un élément est spécifique plus le nombre de références qu'il indexe est faible). Les différents cas à envisager sont les suivants :

+ pour un concept isolé, on lui substitue la disjonction de ses spécifiques,
+ pour un OU, on supprime l'opérande apportant le plus de références, (cela revient à supprimer un OU)
+ pour un ET, on substitue à l'opérande apportant le plus de références la disjonction de ses spécifiques,
+ pour un SAUF, on substitue au premier opérande la disjonction de ses spécifiques, ou bien au deuxième la conjonction de ses génériques.

- trop peu de références :

Pour augmenter le nombre de références, on joue également sur la logique de l'équation ou sur les concepts (utilisation de la relation de généricité). Les différents cas sont les suivants :

+ pour un concept isolé, on lui substitue la conjonction de ses génériques,
+ pour un OU ou un ET, on substitue à l'opérande apportant le moins de références, la conjonction de ses génériques,
+ pour un SAUF, on substitue au premier opérande la conjonction de ses génériques, ou au deuxième la disjonction de ses spécifiques.

- mauvaise qualité des références :

On joue également sur la logique de l'équation et la reformulation des concepts (utilisation de la relation de voisinage sémantique). Les différents cas sont les suivants :

+ pour un concept isolé, on lui substitue la disjonction de ses voisins,
+ pour un OU, on supprime l'opérande apportant les références de plus mauvaise qualité,
+ pour un ET, on substitue à l'opérande apportant les références de plus mauvaise qualité la disjonction de ses voisins,
+ pour un SAUF, on substitue au premier opérande la disjonction de ses voisins (le deuxième opérande d'un SAUF n'intervenant pas dans l'évaluation de la pertinence résultat, il ne sert à rien de le modifier).

Il est noter que pour chaque problème à résoudre, les modifications possibles ne sont pas toutes équivalentes relativement à la dégradation de l'équation; la stratégie de choix d'une modification doit donc garantir le niveau de dégradation minimum. Pour cela, on hiérarchise l'ensemble des modifications possibles dans l'ordre de dégradation croissant :

- suppression d'un opérande dans un OU,
- substitution d'un opérande dans un OU,

On considère que la modification associée à un OU est la moins dégradante, car elle ne modifie en rien l'opérande non modifié.

- substitution d'un opérande dans un ET ou un SAUF,

La modification associée à un ET et un SAUF est par contre plus dégradante, car elle affecte la sémantique des deux opérandes.

A partir de l'ensemble des modifications possibles et de l'équation à reformuler, il faut donc choisir la modification adéquate. Ce choix s'effectue tout d'abord à partir de la hiérarchisation des modifications (on cherche à faire les modifications les moins dégradantes possibles), ensuite pour un type de modification donné, il faut choisir l'opérande sur lequel opérer la modification. Pour cela on utilise la décoration de l'arbre d'interprétation de l'équation finale de recherche (chaque nœud est décoré par un couple formé du nombre de références produit par le sous arbre dont il est racine, et de la qualité moyenne de ces références). Deux cas sont à traiter :

- suppression d'un opérande d'un OU :

On cherche l'opérande d'un OU de plus bas niveau dans l'arbre d'interprétation (cela modifie le moins possible la logique de l'équation) et fournissant le plus de références (ou les plus mauvaises suivant le problème à résoudre). Cet opérande peut être un concept isolé ou un arbre lui même.

- substitution d'un concept d'un OU, ET, SAUF (le problème est identique) :

L'élément choisi est donc forcément une feuille. On cherche donc l'opérande d'un OU de plus bas niveau et fournissant les moins nombreuses références, et l'opérande d'un ET ou

d'un SAUF de plus bas niveau et fournissant (suivant le problème à traiter) les plus nombreuses, les moins nombreuses ou les plus mauvaises références.

Il faut noter cependant que l'algorithme de choix d'une modification est fortement heuristique et par conséquent ne peut garantir l'amélioration du résultat dans tous les cas. Par exemple, dans le cas de la suppression d'un opérande d'un OU, l'algorithme n'effectue pas toujours le meilleur choix :
Soit l'expression booléenne OU(A,B), et L(A) et L(B) les relations d'indexation associées à, respectivement, A et B. L'algorithme de choix de l'opérande à supprimer produit les effets suivants :
- L(A) et L(B) disjoints :
Supposons que A (ou B symétriquement) soit l'opérande ayant la plus mauvaise qualité moyenne. En enlevant A (ou B) on enlève donc bien les plus mauvaises références (en moyenne), et les autres références ne sont pas altérées (les ensembles sont disjoints).
De même si A (ou B) est l'opérande fournissant le plus grand nombre de références, sa suppression réduit bien le nombre de références total, sans effet sur les autres références.

- L(A) inclus dans L(B) :
Supposons que A soit l'opérande ayant la plus mauvaise qualité moyenne de références. En enlevant A, on n'améliore pas la qualité des références puisque toutes les références données par A le sont également par B, et la fonction de pondération associée au OU est le MAX. Le fait d'enlever A ne sert à rien pour les références de l'intersection de L(A) et L(B) pour lesquelles A fournit la plus faible mesure de pertinence, et au contraire diminue la qualité de celles pour lesquelles A fournit la plus forte mesure de pertinence.
Si B est l'opérande ayant la plus mauvaise qualité moyenne des références, sa suppression n'améliore pas la qualité des références mais les supprime.
Si B est l'opérande fournissant le plus de références, sa suppression va bien diminuer le nombre de références résultat (il n'y a même plus de références satisfaisant l'expression).

- L(A) et L(B) ont une intersection non vide et différente de L(A) ou L(B) :
Dans ce cas, l'algorithme de choix produit des effets oscillant entre ceux du premier cas (L(A) et L(B) disjoints) et du deuxième (L(A) est inclus dans L(B)).

De même, la substitution de concepts ne garantit pas un changement dans le sens souhaité.

Un autre problème important est l'effet de bord qui peut être produit par les modifications. En effet, même si une modification améliore le résultat dans le sens souhaité, elle peut le détériorer sur une autre dimension.

Par exemple, la suppression d'un opérande d'un OU peut diminuer le nombre de références en résultat (ce que l'on souhaite), mais également détériorer fortement la qualité des références (l'opérateur de combinaison des représentativités associé au OU est l'opérateur MAX).

(2) Choix du niveau de reformulation :

Si la modification choisie est la reformulation d'un concept par des concepts tirés de ses relations sémantiques, il est nécessaire de connaitre le niveau de reformulation à appliquer, c'est à dire l'éloignement par rapport au concept initial qu'on autorise.
Cet éloignement est une fonction du niveau de connaissances de l'utilisateur dans le domaine. Plus l'utilisateur est spécialiste, plus la reformulation va être faible (quitte à faire un grand nombre de reformulations pour arriver à un résultat conforme).
Nous avons donc fixé pour chaque catégorie d'utilisateur un seuil de reformulation associé au graphe des relations sémantiques :

- pour un utilisateur spécialiste :
  on prend le générique et les spécifiques à un niveau d'écart,
  on prend l'élément le plus immédiat dans la chaine de voisinage (cf V.2.1.1).
- pour un utilisateur moyen :
  on prend les génériques et les spécifiques à un niveau d'écart,
  on prend les voisins immédiats (les deux premiers éléments dans la chaine de voisinage).
- pour un utilisateur débutant :
  on prend les génériques et les spécifiques à deux niveaux d'écart,
  on prend les voisins immédiats.

(3) Réalisation de la modification :

Elle consiste seulement, soit à supprimer un opérande d'un OU (ce n'est pas forcément un concept isolé, ce peut être un arbre), soit à substituer un concept par un arbre.

(4) Mise à jour du niveau de dégradation :

La mesure de niveau de dégradation (DEG) est mise à jour comme défini au VI.3.2.

(5) Retour au processus d'interprétation :

La nouvelle équation finale de recherche est ensuite interprétée (le processus est celui décrit au V.2.2.4).

(6) Validation d'une reformulation :

Elle a pour but de s'assurer que la reformulation va bien dans le sens de l'amélioration de la réponse souhaitée et qu'elle ne produit pas d'effets de bord importants. Pour cela, on va

comparer la réponse fournie par la reformulation à la réponse précédente.
Dans l'optique de l'évaluation globale d'une réponse, les éléments importants du résultat sont le nombre et la qualité des références obtenues (les diagnostics du processus d'évaluation sont fonction de ces éléments cf VI.2.1.4). On associe donc à chaque nouvel ensemble de références produit par le processus d'interprétation, un couple V formé du nombre de références obtenues (n) et de la qualité moyenne des références (q).
Pour évaluer l'amélioration du résultat, trois cas sont à envisager (un par diagnostic possible) :

- on veut plus de références :

Soit V_i le couple associé à l'équation de recherche courante, et V_{i-1} celui associé à l'équation de recherche précédente. L'équation courante est "meilleure" que l'équation précédente, si et seulement si :
$n_i > n_{i-1}$ et q_i voisin ou supérieur à q_{i-1}

- on veut moins de références :

Dans ce cas, l'équation de recherche courante est "meilleure" que la précédente, si et seulement si :
$n_i < n_{i-1}$ et q_i voisin ou supérieur à q_{i-1}

- on veut augmenter la qualité des références :

L'équation de recherche courante est meilleure que la précédente, si et seulement si :
n_i voisin n_{i-1} et $q_i > q_{i-1}$

Si l'équation courante est jugée meilleure, la reformulation est validée et le traitement de la requête se poursuit. Dans le cas contraire on procéde alors à un retour arrière à l'équation finale de recherche précédente.

3.2.6. Le moteur d'inférences

C'est le cœur de IOTA, c'est à travers lui que coopérent les différents modules du système. C'est donc la partie la plus importante et la plus délicate à réaliser. Plutôt que d'utiliser un moteur d'inférences existant, nous avons choisi de développer notre propre outil. Ce choix est dicté uniquement par des critères techniques; le moteur devait être écrit dans un langage accessible depuis LELISP, et sa taille devait être peu importante puisqu'il doit cohabiter avec IOTA. Nous n'avons pas trouvé de moteurs correspondant à ces critères et c'est pourquoi nous avons été amené à développer notre propre outil.

Les caractéristiques un peu particulières de notre application (nombre de règles de production peu élevé, mécanisme de retour arrière peu utilisé, pas d'instance de règles à gérer) font que le moteur réalisé est très simple (nous n'avons pas à nous préoccuper des problèmes délicats d'optimisation du filtrage des règles, de mécanisme de retour arrière performant et de gestion efficace des instances de règles).

Le moteur réalisé permet donc la manipulation des règles de production décrites précédemment, à travers deux structures de données, la base de données à court terme et la pile de buts.
La pile de buts est utilisée par les modules experts pour permettre la réalisation de fonctions comme l'évaluation d'une référence ou l'évaluation du résultat, qui mettent en œuvre des connaissances expertes. Les modules classiques (interprétation, accès aux textes, ...) ne nécessitant pas de connaissances expertes n'utilisent pas cette pile de buts (les variables d'états suffisent à assurer l'enchainement séquentiel des modules).

Un mécanisme simple de retour arrière est utilisé. Il faut noter que le retour arrière n'est possible qu'à l'intérieur d'une tâche donnée, car la réalisation d'une tâche a un caractère irréversible. Dans notre application, le retour arrière consiste donc à relancer une tâche en échec en effectuant de nouveaux choix (cela n'est possible que pour les tâches expertes comme évaluation et reformulation, les autres étant parfaitement déterministes). Si toutes les alternatives d'une tâche ont été épuisées, la tâche est en échec. Nous considérons cependant que cet échec est partiel, c'est à dire qu'il n'y a pas de solutions parfaites pour cette tâche mais que la solution courante est néanmoins la meilleure possible. Le traitement peut donc continuer même dans ce cas (une règle d'échec est associée à chaque tâche et définit alors la suite du traitement).

Nous allons maintenant décrire la base de données à court terme et la pile de buts, puis nous donnons les algorithmes du moteur d'inférences.

3.2.6.1. La base de données à court terme

Elle décrit l'état courant du système. Elle contient les variables suivantes :

- tâche : donne la tâche courante du système,
- typdom, typsys, typologie : donnent la typologie de l'utilisateur dans le domaine, sur le système, et globalement,
- degradation : donne le niveau de dégradation de l'équation finale de recherche,
- requête, eqprimaire, eqfinale : donnent respectivement la requête initiale de l'utilisateur, l'équation primaire de recherche correspondante, l'équation finale de recherche courante,

- modifeqfinale : donne la liste des substitutions effectuées sur l'équation finale de recherche (permet la restauration des équations précédentes le cas échéant),
- arbreinterp : donne l'arbre d'interprétation de l'équation finale de recherche,
- référencesrésultat : donne l'ensemble résultat courant,
- référencecourante : donne la référence courante (utilisée pendant l'évaluation locale d'une référence),
- nbréponses : donne le nombre d'éléments de référencesrésultat,
- nbréponsebonnes : donne le nombre de références estimées bonnes,
- évaluation : donne l'évaluation globale du résultat courant,
- diagnostic : donne le diagnostic sur le résultat courant.

3.2.6.2. la pile de buts

Cette pile n'est utilisée que par les tâches expertes. Elle représente la hiérarchie de buts à accomplir pour résoudre une tâche experte. Le but de sommet de pile est le but courant à résoudre.

3.2.6.3. Les algorithmes du moteur d'inférences

Nous donnons une description des algorithmes utilisés.

```
action moteur-d'inférences
    initialisation-des-variables-d'états
    répéter
        détection-des-règles-actives
        choix
            ensemble candidat = vide : retour-arrière
            ensemble candidat ≠ vide : choix-d'une-règle
                                       application
        fchoix
    jusqu'à erreur
fin action moteur-d'inférences
```

```
action détection-des-règles-actives
    choix
    | pilebut = vide : /* on filtre sur la base de données à court terme */
    | pilebut ≠ vide : /* on cherche à résoudre le premier but de la pile */
    fchoix
fin action détection-des-règles-actives

action choix-d'une-règle
    /* on prend la règle active la plus contrainte, c'est à dire dont la
       partie condition est la plus longue */
fin action choix-d'une-règle

action application
    /* exécution séquentielle des s-expressions composant la partie action
       de la règle choisie, mise à jour des historiques (pour un retour
       arrière éventuel) et mise à jour de la pile de buts le cas échéant */
fin action application

action retour-arrière
    choix
    | pilebut = vide : erreur
    |       /* échec dans une tâche non experte ou fin du traitement*/
    | pilebut a un élément :
    |       /* il y a un échec dans le premier but d'une tâche experte,
    |          on ne peut donc pas la résoudre, il faut reprendre avec le
    |          traitement prévu en cas d'échec partiel */
    | pilebut a plus d'un élément :
    |       /* cas d'échec dans le déroulement d'une tâche experte mais
    |          toutes les alternatives n'ont pas été explorées. On refait
    |          donc la tâche en restaurant l'état initial et en enlevant
    |          de la pile de buts les buts associés à la tâche */
    fchoix
fin action retour-arrière
```

4. Conclusion

Le système IOTA constitue une première approche d'un système intelligent de recherche d'informations. Il permet de dériver, à partir d'une requête donnée, une formulation (ou une reformulation) "équivalente" à la requête initiale mais fournissant un ensemble de références plus satisfaisant. A notre connaissance, il s'agit d'un des tous premiers systèmes prototypes de cette nature, pouvant se prêter à des expérimentations réalistes. L'architecture choisie est originale, et bien adaptée à des développements ultérieurs (intégration de modules spécialisés plus efficaces, enrichissement des connaissances, ...). On peut notamment citer deux extensions intéressantes :

- bien que IOTA soit destiné à l'origine à des applications type bureautique sur des textes techniques structurés, il est possible de le réutiliser comme système d'interrogation intelligent de serveurs de bases de données. Dans ce cas, il assure toutes les tâches du SIRI hormis l'exécution des requêtes d'interrogation. Dans l'état actuel de développement, seules les requêtes portant sur le contenu pourraient être traitées (il n'y a pas de reconnaissance des attributs externes lors de l'analyse des requêtes).

Aux fonctionnalités déjà présentées, il faut adjoindre la transformation de l'équation finale de recherche en expressions de recherche du serveur, ainsi que la gestion des communications.

Différentes données (base d'indexation, thésaurus) peuvent être récupérées des serveurs et transformées dans une forme exploitable, et les lexiques en être tirés. Les connaissances sur la recherche d'informations demeurent inchangées. Les fonctions de transformation et de communication sont décrites sous forme de modules classiques. Le module de transformation est assez simple à réaliser pour générer des expressions booléennes, puisqu'il consiste en une simple réécriture de l'équation finale de recherche, avec quelques modifications d'ordre syntaxique.

Le seul module qui doit réellement changer, est la fonction de correspondance qui doit être modifiée de façon à tenir compte des spécificités des langages d'interrogation visés (c'est à dire l'opérateur d'adjacence et les jokers). Ces opérateurs doivent être introduits dans les règles de transformation des concepts reconnus.

Par exemple :
le groupe "informatique de gestion" peut être transformé en :
"informatique ET gestion", ou bien de façon plus précise en :
"informatique ADJACENT gestion", ou encore en :
"informatique* ADJACENT gestion*"

De cette façon, on peut pallier au manque de richesse des termes d'indexation. L'utilisation

de IOTA comme interface intelligente pourrait nous permettre de bien tester et mettre au point l'ensemble des connaissances relatives à la recherche d'informations ainsi que les stratégies utilisées (pour l'évaluation et la reformulation notamment).

- il est possible de définir une fonction de choix d'une base. Cette fonction doit être fondée sur deux critères :

  * le contenu de la base : caractérisé par le domaine des documents, ainsi que leur type (articles spécialisés, livres généraux, thèses, ...),

  * la qualité de la base, perçue à travers la puissance du langage d'interrogation (existence d'opérateurs plus ou moins sophistiqués) et de la qualité de l'indexation (surtout au niveau de la cohérence des termes d'indexation utilisés).

Toutes ces caractéristiques ne sont pas faciles à évaluer (notamment celles concernant la qualité de la base), et doivent donc être fournies par un expert.

En ce qui concerne le choix d'une base en fonction du contenu, une stratégie à deux niveaux peut être employée. Tout d'abord il faut disposer d'un thésaurus de haut niveau décrivant les concept principaux décrits par chaque base. Chaque requête est alors confrontée à ce thésaurus qui permet de sélectionner un ensemble de bases candidates (des bases ayant des contenus voisins ne peuvent être distinguées). Le choix parmi les bases candidates est alors fait en appliquant le processus de construction de l'équation finale de recherche sur chacune d'elle. La base choisie sera celle où la dégradation résultante est la plus faible et la typologie de l'utilisateur sur le domaine la plus élevée.

Le critère de qualité va intervenir surtout comme un premier filtrage sur l'ensemble des bases (on utilisera une base estimée de mauvaise qualité que lorsqu'elle sera la seule à correspondre au contenu de la requête). A ce niveau, il serait intéressant de définir des primitives d'apprentissage qui permettent d'estimer la qualité de la base en référence aux requêtes déjà traitées. IL serait encore plus intéressant de pouvoir définir alors des stratégies d'utilisation de ces bases qui permettraient de pallier aux insuffisances relevées.

Globalement, le prototype souffre encore de quelques limitations :

- notre notion de formulation équivalente reste limitée dans la mesure où nous utilisons des mécanismes inférentiels peu puissants. Cette restriction provient essentiellement de la base de connaissances qui contient uniquement des relations sémantiques entre les termes d'indexation, et n'est donc pas un véritable modèle sémantique du contenu des documents. Nous n'avons donc pas de moyens de produire des formulations sémantiquement équivalentes à une formulation donnée, ce qui est très restrictif. Par conséquent, les inférences réalisées sont très liées à la structure des groupes manipulés, puisqu'elles consistent soit en des modifications de structures syntaxiques (processus de correspon-

dance), soit en des substitutions de groupes nominaux (processus de reformulation).

- il reste un travail important de validation de la base de connaissances, permettant d'affiner la définition des relations sémantiques qu'il serait souhaitable d'inclure, ainsi que les connaissances expertes existantes.

- le langage d'interrogation accepté reste relativement fruste, tant au niveau du vocabulaire puisque nous ne disposons pas de dictionnaires importants, ni d'outils morphologiques permettant de n'utiliser qu'un dictionnaire de lemmes (toutes les formes possibles d'un lemme pouvant être alors dérivées), qu'au niveau de la compréhension effectuée.

Ces remarques soulignent que les différents modules ou ensembles de données manipulés par IOTA ont actuellement un degré de sophistication variable dans le prototype. Certains d'entre eux (notamment le module d'interprétation fine de la requête, et le thésaurus) font l'objet de développements parallèles dans le groupe, en vue de leur intégration ultérieure dans IOTA. Notre travail, relativement à ces points, a donc consisté à en donner des versions simplifiées, permettant essentiellement de valider les solutions d'architecture et de stratégies choisies. Des exemples illustrant le fonctionnement de IOTA sont donnés dans le chapitre VII.

CHAPITRE VI

DEFINITION DES CONNAISSANCES EXPERTES

1. Introduction

Nous avons situé, dans les chapitres précédents, les différents modules mettant en œuvre des connaissances expertes (cf V-2-1), et nous avons déjà assez largement précisé leur nature (cf V-1-1-3-1). La tâche de définition proprement dite de ce type de connaissances est, de manière classique, un problème délicat; dans notre étude, cette difficulté se double d'un inconvénient supplémentaire lié au fait que nous cherchons principalement à modéliser le *comportement* d'un opérateur humain (le documentaliste). Ce comportement se traduit par la détermination permanente de plans d'actions en fonction de l'appréciation d'une situation donnée (par exemple, la détermination de la typologie d'un usager, de la qualité d'une réponse, pour inférer une stratégie de reformulation). Les connaissances expertes reposent donc sur la définition de plusieurs variables permettant au système d'inférer une situation.

Dans un premier temps, nous présentons (cf 2) les différents niveaux de connaissances utilisées dans le prototype; nous y précisons chaque fois la nature et le rôle des variables impliquées. Ces connaissances ont été mises en évidence et spécifiées grâce à un travail de collaboration avec une documentaliste très au fait de l'utilisation des systèmes de recherche d'informations. Le nombre assez restreint de règles présentées s'explique par l'état encore expérimental du prototype; comme dans beaucoup d'autres travaux de même nature, elle seront affinées et complétées au travers d'une utilisation intensive par l'expert. Il faut également remarquer qu'une définition plus réaliste de ces connaissances dépend de l'expérimentation sur des corpus importants et variés, susceptibles de mieux refléter la diversité des situations réelles d'interrogation.

Le rôle des variables s'exprime, dans la première partie, par des notations de type <usager spécialiste>; il nous faut naturellement préciser les modalités d'évaluation de ces propriétés. Ce travail, souvent empirique du fait même de la nature de la plupart de ces

critères, est présenté dans la deuxième partie du chapitre (cf 3). Les valeurs prises par ces variables dans un contexte donné sont fonction d'un certain nombre de paramètres (par exemple, la généralité des termes utilisés, l'effort qu'a à faire le système pour comprendre la requête, pour situer la typologie de l'utilisateur). La base de connaissances utilise actuellement six paramètres de ce type pour affecter l'ensemble des variables. On remarquera qu'ils correspondent tous à des données numériques, et que les variables sont affectées en fonction de combinaisons de valeurs de paramètres, par rapport à des seuils prédéfinis. L'ajustement de ces seuils doit également être affiné par une expérimentation plus poussée du système.

2. Les domaines d'expertise

2.1. La typologie de l'utilisateur

2.1.1. Introduction

Le problème de la modélisation de l'usager est un des problèmes fondamentaux de l'intelligence artificielle, car il constitue un passage obligatoire pour la définition et la réalisation d'interfaces homme - machine réellement conviviales et performantes. Malgré de nombreux travaux réalisés dans ce domaine [RIC 83], [RIC 85], la modélisation de l'usager dans un contexte général reste très partielle, et les techniques employées pour l'instant sont assez dépendantes du domaine d'application.

Pour notre part, notre définition de la typologie de l'utilisateur est différente de la définition classique. Nous ne voulons pas évaluer une typologie dans l'absolu, mais relativement au domaine de connaissance couvert par les documents, et à l'utilisation du système (ce sont les informations qui nous intéressent dans les tâches expertes).

Deux approches sont possibles pour la détermination de la typologie :

- On dispose d'une connaissance sur les différents types d'utilisateurs de la base, et d'une caractérisation de chaque type. Le problème consiste à déterminer, à partir d'une requête donnée, la catégorie d'utilisateurs correspondante. Cette détermination s'effectue la plupart du temps à partir d'une mesure de distance entre la caractérisation d'un type et la caractérisation de la requête (on choisit le type le plus proche de la requête). Cette

approche nécessite la connaissance préalable des usagers potentiels de la base, ainsi que la définition de caractérisations de type très discriminantes. La définition d'une distance entre une requête et un type d'utilisateur est également difficile.

- On détermine la typologie sans connaissances sur les utilisateurs autres que la requête. L'objectif est de comparer les concepts utilisés dans la requête par rapport aux concepts connus du système. Il est nécessaire de définir des critères de comparaison (spécificité des concepts par exemple) fondés sur des hypothèses concernant l'expression de requêtes par un type d'utilisateur donné (par exemple, on suppose qu'un utilisateur spécialiste emploie beaucoup de concepts spécifiques dans sa requête).

Nous avons choisi pour notre part la deuxième approche, surtout parce que nous ne disposons d'aucune donnée expérimentale sur l'utilisation d'une base telle que la notre. Il n'est cependant pas impossible que nous évoluions vers une approche mixte (si nous pouvons expérimenter notre système en vraie grandeur, avec un nombre significatif d'utilisateurs).

2.1.2. Notre modèle d'évaluation

Partant de la seule donnée brute d'une requête, il nous a paru possible de caractériser la typologie de l'utilisateur à travers deux critères :

a) Son expertise dans le domaine couvert par le corpus :
Le domaine sémantique est principalement reflété pat le thésaurus. Un usager faisant référence à des concepts très spécifiques, très discriminants, sera jugé spécialiste. Inversement, d'une requête impliquant essentiellement des concepts génériques, peu discriminants, on déduira qu'il s'agit d'un débutant. Naturellement, les notions de référence (spécificité, généricité) sont définies par rapport au thésaurus; il s'agit donc bien d'une estimation relative au domaine couvert. Si l'on considérait par exemple, un corpus extrêmement spécialisé, d'une discipline donnée, un usager jugé par ailleurs spécialiste pourrait éventuellement être considéré comme débutant pour peu que ses requêtes ne correspondent pas à la finesse des connaissances enregistrées pour ce corpus.

b) Son expertise dans la connaissance du système :
Ce niveau d'expertise est évalué relativement à l'effort que doit faire le système pour "comprendre" la requête et dériver l'équation finale de recherche. Sera donc jugé spécialiste (à ce niveau) un utilisateur connaissant les concepts significatifs du domaine couvert, et qui les formulera de manière très proche ou identique à leur définition dans le

thésaurus : le système effectue alors un minimum de travail pour retrouver les termes d'indexation correspondants (on dira aussi qu'il apprend peu de choses à l'usager). Inversement, une requête profondément dégradée lors de la dérivation de l'équation finale de recherche sera considérée comme le fait d'un usager débutant, connaissant mal les concepts significatifs du domaine et/ou les exprimant mal.

L'affectation d'un usager dans une catégorie de la typologie (spécialiste, moyen, débutant) dépend concrétement de la considération de ces deux critères, eux mêmes étant définis à partir de l'évaluation de trois paramètres évalués sur la requête R (se reporter à la section 3.1 ci-après pour une présentation détaillée de ces paramètres ainsi que de leur mode d'évaluation) :

- la spécificité moyenne des concepts utilisés, notée M1(R)
- la particularité moyenne des concepts utilisé, notée M2(R)
- les connaissances apportées à l'usager dans le processus de dérivation de l'équation finale, notée M3(R).

Tous ces paramètres correspondent à des fonctions réelles $\in [0,1]$, ayant comme argument l'équation de recherche. Nous montrons ci-après leur utilisation dans la détermination de la typologie.

2.1.3. Règles de détermination de la typologie

a) L'expertise de l'usager dans le domaine est mesurée uniquement à travers le paramètre M1; la catégorie est déduite de la requête R dans la variable TYPDOM en fonction des valeurs suivantes de M1(R) :

M1(R)	TYPDOM
<= 0,3	débutant (utilise surtout des concepts génériques)
> 0,3 et <= 0,7	moyen (situation intermédiaire)
> 0,7	spécialiste (utilise surtout des termes spécifiques)

b) L'expertise de l'usager relativement au système est mesurée par les paramètres M2 et M3; la catégorie de l'usager est déduite de R dans la variable TYPSYS en fonction des valeurs suivantes de M2(R) et M3(R) :

M2(R)	M3(R)	TYPSYS
<= 0,3	<= 0,3	débutant
<= 0,3	>= 0,7	impossible M3 <= M2
>= 0,7	<= 0,3	spécialiste
>= 0,7	>= 0,7	débutant

On peut synthétiser ces résultats en évaluant la mesure M2(R) - M3(R), qui évalue les connaissances préalables de l'utilisateur sur le système :

M2(R) - M3(R)	TYPSYS
<= 0,3	débutant
> 0.3 et <= 0.7	moyen
> 0.7	spécialiste

c) Evaluation globale de la typologie :
On réutilise les deux variables précédentes TYPDOM et TYPSYS pour déduire la valeur définitive de la typologie de l'utilisateur (variable TYPOLOGIE) selon la table de vérité suivante :

TYPDOM	TYPSYS	TYPOLOGIE
débutant	débutant	débutant
débutant	moyen	débutant
débutant	spécialiste	moyen
moyen	débutant	moyen
moyen	moyen	moyen
moyen	spécialiste	moyen
spécialiste	débutant	moyen
spécialiste	moyen	spécialiste
spécialiste	spécialiste	spécialiste

Soit, en termes d'expressions logiques :
TYPOLOGIE = débutant ⇔ TYPDOM = débutant & TYPSYS ≠ spécialiste
TYPOLOGIE = spécialiste ⇔ TYPDOM = spécialiste & TYPSYS ≠ débutant
La TYPOLOGIE "moyen" se déduit par négation des deux précédentes :
TYPOLOGIE = moyen ⇔ (TYPDOM ≠ débutant OU TYPSYS = spécialiste) & (TYPDOM ≠ spécialiste OU TYPSYS = débutant)

2.2. L'évaluation des références

2.2.1. Objectifs

Le but de l'évaluation est de déterminer si l'ensemble des références fournies par le processus d'interprétation de l'équation finale de recherche est conforme aux besoins de l'utilisateur. Si ce n'est pas le cas, le processus d'évaluation doit également établir un diagnostic indiquant pourquoi l'ensemble des références est jugé non conforme, et les changements à effectuer pour qu'il puisse devenir conforme.

Cette tâche d'évaluation est typiquement une tâche experte faisant intervenir des connaissances empiriques. Nous avons choisi de la modéliser par un ensemble de règles de production faisant intervenir un certain nombre de variables.

La première variable mise en évidence est bien entendu le niveau de l'utilisateur dans le domaine. Un utilisateur spécialiste a des besoins d'informations différents d'un débutant (il veut des références très spécialisées, pas d'ordre général), en conséquence, la qualité estimée d'une réponse n'est pas absolue, mais jugée relativement à la typologie du demandeur.

Une autre variable importante est l'existence, dans l'ensemble des références, d'une ou plusieurs références jugées très pertinentes relativement à la requête. Il est sûr que si aucune référence n'est jugée très pertinente, il est difficile de trouver l'ensemble résultat conforme. La détermination de la qualité d'une référence va être réalisée à travers un certain nombre de règles expertes évaluant une référence.

Le nombre total de références est également intéressant (un spécialiste ne veut généralement pas un nombre élevé de références).

A ces variables directement liées à la tâche d'évaluation, nous devons ajouter deux éléments prenant en compte le traitement effectué sur la requête. Le premier élément est lié à l'ensemble des références résultat. Dans la mesure où le processus d'indexation automatique utilisé n'a pas une stratégie de sélection des concepts, mais une stratégie de pondération, la taille moyenne de la relation d'indexation est assez grande et certains poids associés peuvent être faibles. Par conséquent, le nombre de références produit par une requête peut être grand avec des mesures de pertinence associées assez faibles. Il peut donc être nécessaire de filtrer l'ensemble des références de façon à n'obtenir que des références réellement intéressantes. La définition de cette propriété est bien entendu subjective, et nous avons utilisé là encore des règles expertes pour réaliser ce filtrage.

Le deuxième élément est lié au cycle de traitement d'une requête. Tout au long de ce cycle, une requête peut être modifiée. Il est certain que plus la requête initiale est modifiée, plus il est difficile d'évaluer l'ensemble résultat relativement aux besoins de l'utilisateur; c'est pourquoi les critères d'évaluation doivent être atténués en fonction du degré de modification de la requête. Nous utilisons donc le niveau de dégradation de la requête dans notre processus d'évaluation.

La tâche d'évaluation s'effectue donc en trois étapes :

(1) filtrage de l'ensemble des références
(2) évaluation de chaque référence (est ce une "bonne" référence ?)
(3) évaluation globale du résultat et établissement du diagnostic le cas échéant

Nous allons maintenant détailler ces trois étapes.

2.2.2. Le filtrage des références

Il est basé sur deux variables, la typologie de l'utilisateur et le degré de modification de la requête. En effet, plus l'utilisateur est spécialiste plus le filtrage doit être sévère. Par contre, si la requête est très dégradée une référence même peu pertinente est intéressante.

A partir de ces deux variables, nous avons défini des règles fixant des seuils de qualité des références, la qualité étant définie à partir des poids de la relation d'indexation (représentativité du concept dans le document, et du document relativement au concept).

SI <utilisateur spécialiste> ET <dégradation faible>
ALORS SEUIL <-- (0.7 0.7)
SI <utilisateur spécialiste> ET <dégradation moyenne ou forte>
ALORS SEUIL <-- (0.6 0.6)
SI <utilisateur moyen> ET <dégradation faible>
ALORS SEUIL <-- (0.5 0.5)

2.2.3. Evaluation d'une référence

Le but est ici de déterminer si une référence est très pertinente relativement à la requête. Différents paramètres interviennent dans cette évaluation. Tout d'abord le niveau de connaissance de l'utilisateur dans le domaine (s'il est spécialiste une bonne référence est une référence très pertinente), la pertinence des références est également un élément important (représentativité concept - document et document - concept), et enfin le niveau de dégradation de la requête (si la dégradation est importante, les autres paramètres sont affaiblis).
Il faut noter deux cas d'évaluation absolue, indépendants de toute autre considération, une référence ayant une pertinence maximale (représentativité requête - référence $\simeq$ 1 et représentativité référence - requête $\simeq$ 1) qui est considérée comme bonne, et une référence ayant une pertinence quasi nulle qui est considérée comme mauvaise.

Sur ces paramètres, nous avons défini une vingtaine de règles d'évaluation permettant de décider si une référence est bonne ou non. Ces règles fixent des seuils de représentativité pour chaque type d'utilisateur, en dessous desquels une référence n'est pas estimée pertinente. Ces seuils sont, bien évidemment, d'autant plus élevés que l'utilisateur est spécialiste. Le niveau de dégradation de la requête permet de pondérer ces seuils (plus la requête est dégradée, plus les seuils sont abaissés). Voici quelques exemples de règles :

SI REP(document,requête) > 0.7 ET REP(requête,document) > 0.7 ET
  <usager spécialiste>
ALORS
  référence bonne

SI REP(document,requête) < 0.3 ET REP(requête,document) < 0.3 ET
  <usager moyen>
ALORS
  référence mauvaise

SI REP(document,requête) < 0.3 ET REP(requête,document) ≥ 0.3 ET
  <usager débutant> ET <dégradation forte>
ALORS
  référence bonne

SI REP(document,requête) < 0.3 ET REP(requête,document) ≥ 0.3 ET
  <usager débutant> ET <dégradation faible>
ALORS
  référence mauvaise

2.2.4. Evaluation globale de la réponse

Comme nous l'avons vu les différents paramètres utilisés sont le nombre de références, le nombre de références jugées bonnes (fourni par l'étape précédente), le niveau de connaissance dans le domaine de l'utilisateur et enfin le niveau de dégradation.

Nous avons un certain nombre de règles permettant d'évaluer l'ensemble des références, et diagnostiquant si besoin est les modifications à effectuer. Trois diagnostics sont possibles :
- on constate que le nombre de références est trop grand, c'est le cas notamment pour les

utilisateurs spécialistes;

- on constate que le nombre de références est trop petit, c'est le cas notamment pour les utilisateurs débutants;

- on constate que la qualité des références est trop faible, c'est le cas notamment pour les utilisateurs spécialistes; ce diagnostic est prioritaire si l'un des deux précédents s'applique déjà sur la réponse (la qualité de la réponse est plus importante que la quantité de références).
Ces trois diagnostics sont ensuite utilisés pour orienter le processus de reformulation de la requête ayant produit ce résultat.

Quinze règles d'évaluation et de diagnostic ont été définies. Nous pouvons les résumer pour chaque type d'utilisateur :
- utilisateur spécialiste :
Pour que le résultat soit correct, il faut qu'il y ait un nombre de références faible (sinon on diagnostique un nombre de références trop élevé) et des références évaluées comme bonnes (sinon on diagnostique une mauvaise qualité de la réponse). Si le niveau de dégradation est fort, on accepte un nombre de références plus élevé.

- utilisateur moyen :
Pour que le résultat soit correct, il faut qu'il y ait un nombre de références moyen (sinon on diagnostique trop ou pas assez de références) et des références jugées bonnes (sinon on diagnostique une mauvaise qualité des références). Si le niveau de dégradation est fort, le paramètre sur le nombre de références n'est plus utilisé (on considère que c'est le processus de reformulation qui a apporté ce grand nombre de références).

- utilisateur débutant :
Pour que le résultat soit correct, il faut qu'il y ait un nombre de références élevé (sinon on diagnostique trop peu de références) et des références jugées bonnes (sinon on diagnostique une mauvaise qualité des références). Si le niveau de dégradation est fort, on diagnostique une bonne réponse même avec peu de références.

Voici quelques exemples de règles d'évaluation :

SI <usager spécialiste> ET <nombre de réponses faible> ET
<il y a des références bonnes>
ALORS
résultat correct

SI <usager spécialiste> ET <nombre de réponses grand> ET
<dégradation faible>
ALORS
trop de réponses

SI <usager débutant> ET <nombre de réponses faible>
ALORS
pas assez de réponses

SI <il n'y a pas de référence bonnes> ET <dégradation faible>
ALORS
mauvaise qualité des réponses

2.3. L'apprentissage

Comme nous l'avons vu, dans le cours de traitement d'une requête, le système peut être amené à faire un certain nombre d'inférences lui permettant de continuer le traitement (mot ou concept inconnu). Il est intéressant de pouvoir enregistrer ces inférences si elles ont permis de résoudre de façon satisfaisante la requête. Nous avons donc défini un certain nombre de règles réalisant ces deux formes d'apprentissage, en tenant compte du résultat obtenu (si la requête n'est pas satisfaite, on considère que les inférences réalisées ne sont pas bonnes) et de la typologie de l'utilisateur (plus il est spécialiste, plus on accorde de crédit aux inférences réalisées; les critères de satisfaction d'une requête étant, dans le cas d'un usager spécialiste, très stricts).

SI <usager est spécialiste> ET <évaluation bonne> ET
<dégradation faible>
ALORS <faire l'apprentissage>

3. Evaluation des paramètres

3.1. Typologie de l'utilisateur : les paramètres M1, M2, M3

Nous décrivons ici les fonctions d'évaluation associées aux trois paramètres M1. M2, M3 utilisés dans la section 2.1.3 de ce chapitre :

(1) Evaluation de la spécificité des concepts :
La mesure de spécificité du concept par rapport au domaine est obtenue à partir de la base de connaissances où sont enregistrées les relations sémantiques entre les termes d'indexation. Chaque concept appartient à une hiérarchie de génériques - spécifiques, ou bien est isolé.
On considère que les concepts isolés sont très spécifiques. Pour les autres concepts, l'évaluation se fait à partir du niveau dans la hiérarchie; les concepts de plus haut niveau sont les plus génériques et ceux de plus bas niveau les plus spécifiques. La mesure de spécificité est la suivante :

si c appartient à une hiérarchie :
$m(c)$ = niveau de c / nombre de niveaux dans la hiérarchie
sinon
$m(c) = 1$
avec $0 \leq m(c) \leq 1$

$m(c) = 0$ signifie que "c est un concept très générique"
$m(c) = 1$ signifie que "c est un concept très spécifique"

Le calcul de la spécificité moyenne d'une requête M1(R) s'effectue à partir de l'équation finale de recherche (EFR), pour laquelle on calcule la mesure de spécificité de chaque concept. La valeur résultante est la moyenne arithmétique des mesures :

$M1(R) = M1(EFR) =$ Moyenne $m(c)$
avec $0 \leq M1(EFR) \leq 1$

(2) Evaluation de la particularité des concepts :
Un concept particulier est un concept qui est peu présent dans le corpus et exprime donc une notion "pointue" relativement à celui-ci, alors qu'un concept général est présent dans tout le corpus et exprime une notion générale.

La mesure de généralité ou de particularité d'un concept est estimée par la discriminance du concept dans la base. La discriminance d'un concept est calculée à partir du nombre de documents indexés par ce concept :

si c est un terme d'indexation :

DISC(c) = 1 - (NBDOCINDEXE(c) / NBDOCTOTAL)

sinon

DISC(c) = 0

On a donc $0 \leq DISC(c) < 1$

où c est un concept, NBDOCINDEXE(c) représente la taille de la relation d'indexation associée à c, et NBDOCTOTAL le nombre de documents présents dans le corpus.

$DISC(c) \simeq 1$ s'interprète comme "c est un concept particulier (très discriminant)"

$DISC(c) \simeq 0$ s'interprète comme "c est un concept général (très peu discriminant) ou inconnu"

Le calcul de la particularité moyenne d'une requête M2(R) s'effectue sur l'équation finale de recherche, à partir de la mesure DISC(c) de chaque concept.

M2(R) = M2(EFR) = Moyenne DISC(c)

avec $0 \leq M2(EFR) < 1$

(3) Connaissances apportées à l'usager :

On veut mesurer la connaissance que possède l'utilisateur sur le système : rappelons que ce niveau d'expertise exprime la connaissance que l'usager peut avoir des concepts significatifs du domaine, tels qu'ils sont enregistrés dans le système. Nous avons choisi d'exprimer cette notion à travers la quantité d'information éventuellement apportée à l'usager par le processus de correspondance. Cette quantité est définie comme la différence de pouvoir discriminant entre les termes de l'équation primaire EPR et de l'équation finale EFR de recherche.

M3(R) = M2(EFR) - M2(EPR)

Il est facile de voir que si les concepts de l'équation primaire sont des termes d'indexation, on aura $M2(EFR) \simeq M2(EPR)$, et par conséquent $M3(R) \simeq 0$: l'usager s'étant directement exprimé à partir de termes connus du système, le processus de correspondance ne lui a rien apporté. Inversement, si une grande partie des termes de EPR sont inconnus du système, leur pouvoir discriminant est nul et on aura $M2(EPR) \simeq 0$. Le processus de correspondance va alors tenter de dériver des termes d'indexation de ces concepts inconnus; s'il y parvient, on peut dire que $M2(EFR) - M2(EPR) \simeq M2(EFR)$ mesure

l'information discriminante apportée à l'usager par le système.

Remarques :
$0 \leq M3(EFR) < 1$, $M2(EFR) \geq M2(EPR)$, $M3(R) = 1 \Rightarrow M2(EFR) = 1$ & $M2(EPR) = 0$
En effet, $M2(EFR) < M2(EPR)$ impliquerait l'existence d'un concept c appartenant à EPR, tel que DISC(c) > DISC(correspondance de c dans l'EFR). Or deux cas peuvent se présenter, soit c est un terme d'indexation et il est son propre correspondant dans l'EFR (DISC(c) = DISC(c)), soit c n'est pas un terme d'indexation et DISC(c) = 0 et DISC(correspondance de c dans EFR) ≥ 0.

3.2. Le niveau de dégradation de la requête

Cette quantité mesure le niveau de dégradation sémantique de l'équation de recherche courante par rapport à l'équation primaire de recherche. A chaque modification de l'équation de recherche (processus de correspondance, reformulation), cette quantité est mise à jour. Nous considérons donc qu'une requête est peu dégradée lorsque sa compréhension et son traitement nécessitent peu d'ajustements dans la formulation de ses concepts ou dans sa logique.

Nous avons utilisé une mesure définie sur [0,n] pour évaluer la dégradation (0 signifie qu'il n'y a pas de dégradation, et plus n est grand, plus la dégradation est grande : l'équation courante n'a plus rien de commun avec l'équation primaire de recherche initiale).

On peut distinguer entre les dégradations occasionnées par le processus de correspondance et celles produites par le processus de reformulation.
- dégradations lors du processus de correspondance :
Le processus de correspondance est exclusivement fondé sur des outils syntaxiques, les dégradations consistent donc en une modification de la structure syntaxique du concept initial.

- dégradations lors du processus de reformulation :
Les dégradations liées au processus de reformulation peuvent être soit la suppression d'une partie de l'équation (cas de suppression d'un opérande d'un OU), soit la modification d'un concept (substitution d'un concept au concept initial), soit l'ajout d'une expression booléenne (substitution d'un ensemble de concepts au concept initial).

Il n'est pas possible d'évaluer sémantiquement la dégradation résultante de telles modifications. Dans le premier cas la dégradation est essentiellement une décomposition syntaxique et dans le deuxième l'évaluation sémantique nécessiterait l'utilisation de fonctions de voisinage sémantique dont nous ne disposons pas. La seule façon d'évaluer la dégradation est donc la comparaison de structure entre les équations de recherche (nous supposons que plus la structure est modifiée, plus la dégradation sémantique est grande).
Nous utilisons pour cela une fonction de comparaison ayant comme critère de comparaison le rapport entre le nombre de nœuds modifiés (c'est à dire soit mis à jour, soit ajoutés, soit supprimés) dans l'équation courante par rapport à l'équation initiale et la taille de l'équation primaire de recherche :

$$DEG(EFR_i,EPR) = \text{nombre de nœuds changés dans } EFR_i \text{ / taille EPR}$$

Cette dégradation est calculée après chaque pas de reformulation en référence à l'équation primaire de recherche. Cette mesure tient bien compte de tous les changements effectués et est très simple à calculer. Par contre, le fait qu'elle ne fasse aucune référence à la sémantique rend son interprétation plus difficile.

Nous utilisons une partition sur cette mesure :

- $DEG(EFR_i,EPR) < 1$: la dégradation est faible,
- $1 \leq DEG(EFR_i,EPR) < 2$: la dégradation est moyenne,
- $DEG(EFR_i,EPR) \geq 2$: la dégradation est forte.

3.3. Les mesures de représentativités

Nous utilisons les mesures tirées de la relation d'indexation qui ont été définies par [KER 84], [KER 85]. Ces mesures ont pour but l'évaluation de la représentativité mutuelle entre un terme d'indexation (t) et une référence (r).
La représentativité de t par rapport à r (notée REP(t,r)) mesure l'importance de t dans le contenu sémantique de r, par rapport aux autres termes d'indexation. La fonction d'évaluation utilisée est la suivante :

REP(t,r) = Occurrence(t,r) / taille de r
où Occurrence(t,r) représente la fréquence d'occurrence du terme t dans le document r.

La représentativité de r par rapport à t (notée REP(r,t)) définit inversement l'importance de r dans l'expression de t, par rapport aux autres références du corpus. La fonction d'évaluation utilisée est la suivante :

REP(r,t) = Occurrence(t,r) / Occurrence(t,corpus)

L'interprétation aux valeurs limites de ces deux mesures est la suivante :

REP (t,r)	REP (r,t)	Conclusion
1	1	r meilleure réponse pour t
1	0	r bonne réponse pour t qui est un terme général
0	1	r meilleure réponse pour t qui est un terme particulier
0	0	r non pertinente relativement à r

4. Conclusion

Nous avons défini l'ensemble des connaissances relatives aux tâches d'évaluation et de reformulation. Il est bien évident que cette définition n'est que provisoire et que des expérimentations en vraie grandeur nous permettront d'affiner ou de modifier ces définitions. Cette remarque vaut surtout pour les seuils (typologie de l'utilisateur, niveau de reformulation, ...) qui ont été définis de manière totalement empirique. Le choix d'une implantation des connaissances expertes en règles de production va nous permettre, le cas échéant, de les modifier aisément.

CHAPITRE VII

EXPERIMENTATIONS

1. L'évaluation d'un SRI

L'évaluation d'un SRI n'est pas une chose très aisée à entreprendre et ceci pour diverses raisons :
- de nombreux composants sont utilisés,
- on peut utiliser de nombreux critères,
- il y a une grande dépendance vis à vis des données.

Il faut donc fixer un cadre à notre évaluation. Tout d'abord, nous voulons évaluer le composant système d'interrogation en faisant abstraction le plus possible du système d'indexation (ce qui est difficile vu la dualité indexation - interrogation cf I).

Les critères d'évaluation d'un système d'interrogation sont classiquement le rappel et la précision, qui représentent respectivement le pourcentage de documents pertinents retrouvés par rapport au nombre total de documents pertinents, et le pourcentage de documents pertinents retrouvés par rapport au nombre total de documents retrouvés.

Ceci implique de connaitre pour chaque requête le nombre de documents pertinents relativement au corpus utilisé (il s'agit de la grande difficulté de l'évaluation d'un SRI).

Nous avons donc construit sur le corpus traité, un certain nombre de requêtes que nous avons résolues manuellement. Nous présentons tout d'abord le corpus traité, puis nous faisons une évaluation de IOTA sous deux aspects :
- évaluation globale à travers les deux ratios de rappel et précision par rapport à la résolution manuelle des requêtes (nous avons fait intervenir la typologie de l'utilisateur dans ces résolutions, puisque le processus d'interprétation fait largement appel à celle ci),
- évaluation plus fine de IOTA à travers l'observation des incidences des choix effectués en ce qui concerne l'utilisation de la typologie de l'utilisateur et le mécanisme de reformulation.

Nous terminons ce chapitre avec quelques réflexions concernant les connaissances expertes utilisées, et enfin nous donnons des exemples de sessions complètes sous IOTA.

2. Le corpus traité

L'expérimentation en cours a porté sur un corpus de documents de type technique très structurés. Il s'agit des NEF (Normes d'Exploitation et de Fonctionnement des autocommutateurs) du CNET.
Ce texte est organisé en 15 chapitres principaux fortement hiérarchisés en sections, sous-sections,... Il est composé d'environ 30000 mots appartenant à un dictionnaire de 4415 formes différentes. Le dictionnaire lemmatisé correspondant contient 2643 lemmes. Nous présentons ici des résultats d'interrogation portant sur un des chapitres les plus significatifs (pour des raisons de capacité nous n'avons pas pu charger des données plus importantes). Le chapitre choisi a une taille de 3000 mots. La base d'indexation est composée de 469 termes, alors que le dictionnaire de formes a 431 éléments et le dictionnaire de lemmes 429 (ils représentent le vocabulaire des termes d'indexation).
Ce document est hiérarchisé en 118 entités textuelles réparties sur 6 niveaux au maximum.

L'ensemble du chapitre est relatif à la documentation des autocommutateurs, et est une longue suite de normes et conseils relatifs aux différents documents à constituer.

En ce qui concerne le thésaurus mis en place, nous avons défini les relations sémantiques autour de quelques concepts seulement, ce qui explique que les requêtes expérimentées concernent toutes les mêmes notions. Les données du thésaurus sont complétées par les notions de spécificité et de généricité syntaxiques : nous considérons que le fait d'enlever des qualifications à un concept (suppression des adjectifs, cassure de liaisons prépositionnelles) nous fournit un générique de ce concept, alors que le fait de rajouter des qualifications à un concept (adjectifs, complément de nom) nous fournit des spécifiques de celui ci.

3. Evaluation globale de IOTA

Nous avons choisi un certain nombre de requêtes que nous avons traitées sur les deux dernières versions de IOTA, la première (appelée IOTA 2 dans la suite) qui est une version préliminaire au prototype décrit dans ce travail, et la deuxième (appelée IOTA 3 dans la suite) qui constitue la réalisation actuelle de IOTA.

Les principales différences entre IOTA 2 et IOTA 3 sont les suivantes :
- la fonction de correspondance de IOTA 2 est plus large que celle de IOTA 3, en effet si un concept n'est pas reconnu comme terme d'indexation, on lui fait correspondre non pas le plus petit des termes d'indexation l'approchant (cf V.3.2.3), mais l'ensemble des termes d'indexation l'approchant,
- il n'y a pas de prise en compte de la typologie de l'utilisateur dans IOTA 2,
- il n'y a pas de mécanismes d'évaluation et de reformulation, les références résultats sont simplement classées par ordre décroissant de pertinence estimée.
La comparaison entre IOTA 2 et IOTA 3 va nous permettre d'estimer globalement le performances de IOTA 3 par rapport à un système plus classique.

Nous donnons dans le tableau 1 les différentes requêtes choisies, les résultats obtenus avec IOTA 2 et IOTA 3, et le résultat théorique (évalué manuellement sur le texte). Les résultats expérimentaux sont composés d'un triplet référence, représentativité requête - document, représentativité document - requête. La référence est désignée par la donnée du chemin dans la structure logique du document (par exemple u010201 désigne le paragraphe 1 du sous-chapitre 2 du chapitre 1).

Nous avons ensuite mesuré les taux de rappel et de précision sur chaque requête et globalement (pour déterminer quels sont les documents pertinents parmi les résultats expérimentaux, nous avons tenu compte des critères d'inclusion des références, par exemple si u010201 est théoriquement pertinent nous considérons que u01020101 l'est également).

Nous résumons dans le tableau ci dessous les différentes mesures :

REQUETES	IOTA2		IOTA3	
	Rappel	Précision	Rappel	Précision
Numéro 1	2/3	2/2	2/3	2/2
Numéro 2	1/3	1/1	1/3	1/1
Numéro 3	2/3	2/4	1/3	1/1
Numéro 4	ind	0/3	ind	0/1
Numéro 5	2/4	3/5	2/4	2/4
Numéro 6	1/2	1/1	1/2	1/1
Numéro 7	1/1	1/1	1/1	1/1
Numéro 8	2/3	1/3	3/3	3/4
Numéro 9	1/2	1/2	1/2	1/3
Numéro 10	1/3	1/2	1/3	1/2
Numéro 11	1/3	1/2	3/3	2/3
Numéro 12	4/4	4/8	4/4	4/4
Numéro 13	0/1	0/1	1/1	1/1
Moyenne	54.16%	53.60%	68%	77%

REQUETES	IOTA 2	IOTA 3	RESULTAT THEORIQUE
Numéro 1 : diffusion de documentation technique de référence	u010401/1/1 u01040301/1/1	u010401/1/.33 u01040301/1/.33	u010401 u01040301 u010404
Numéro 2 : document nécessaire à l'exécution de marché	u01010301/.1/1	u01010301/.1/1	u010104 u01010301 u01030201
Numéro 3 : type de support	u01020501/.14/.25 u01010405/.11/.25 u01020301/.06/.25 u01030202/.00/.25	u01030203/.02/1	u01020301 u01020501 u01030201
Numéro 4 : type de support sauf papier	u01020501/.14/.25 u01010405/.11/.25 u01030202/.00/.25	u01030203/.02/1	aucune référence
Numéro 5 : type de support ou papier ou microfiche	u01020301/.06/1 u01020501/.14/.25 u01010405/.11/.25 u01040403/.12/.16 u01030202/.00/.25	u01020301/.06/1 u01020501/.14/.25 u01010405/.11/.25 u01030202/.00/.25	u01020301 u01020501 u01030203 u01020401
Numéro 6 : format et papier	u01020301/.66/.18	u01020301/.66/.18	u01020401 u01020301
Numéro 7 : format de support papier	u01020401/.10/1	u01020401/.10/1	u01020401
Numéro 8 : bordereau de documentation	u01030203/1/.02 u01040501/1/.02 u01040201/1/.02 u01040303/1/.02 u01040304/1/.02 u010104 /1/.02 u01030202/1/.02	u01010404/1/1 u01030203/1/1 u01040302/1/1 u01010502/1/.03 u01010503/1/.03	u010104 u010105 u01030203

Tableau 1

REQUETES	IOTA 2	IOTA 3	RESULTAT THEORIQUE
Numéro 9 : papier	u01040403/.12/.16 u01020301/.06/.16	u01020401/.2/1 u01040302/.05/.33 u01040501/.02/.33	u01020401 u01020301
Numéro 10 : norme de documentation	u01020702/1/.03 u01030202/1/.03	u01020702/1/.03 u01030202/1/.03	u010201 u0202 u01020702
Numéro 11 : norme	u01020702/.12/.33 u01030202/.00/.33	u0102/1/.5 u0202/.2/1 u01010105/.1/.5	u02 u010201 u01020702
Numéro 12 : matériel ou logiciel	u01040203/1/.04 u01010404/.11/.16 u01010202/.14/.08 u01010501/.11/.08 u01030202/.01/.13 u01030201/.06/.08 u01040304/.06/.04 u01020202/.05/.04	u01030202/.00/1 u01020501/.14/.5 u01010501/.11/.5 u01030201/.06/.5	u01030201 u01010501 u01030202 u01020501
Numéro 13 : établissement de documentation technique de référence	aucune référence	u01040103/.02/.16	u010401

Tableau 1 (suite)

Si nous observons donc globalement les mesures, on constate que IOTA 3 est supérieur à IOTA 2 sur les deux dimensions rappel et précision. L'accroissement est assez sensible (+ 14% pour le rappel et surtout + 23.4% pour la précision). On s'aperçoit donc que c'est surtout la précision qui est augmentée, ce qui n'était pas forcément évident, dans la mesure où classiquement on considère que la reformulation permet surtout d'augmenter le rappel.

On peut d'ores et déjà considérer après ces premières expérimentations que l'optique prise permet d'augmenter assez sensiblement les performances qualitativ es d'un SRI.

4. Evaluation des composants

Nous allons essayer d'évaluer l'incidence de la typologie et de la reformulation sur le comportement de IOTA 3.

4.1. L'incidence de la typologie

Pour évaluer cette incidence, nous avons choisi deux requêtes donnant la même équation finale de recherche initiale avec une typologie différente. Les deux requêtes choisies sont :
"bordereau de documentation"
"bordereau et documentation"
Le déroulement de leur résolution est résumé sur le tableau 2.1. La première requête est estimée moyenne car on a été amené à casser le groupement "bordereau de documentation" en "bordereau ET documentation", alors que pour la seconde requête, nous n'avons pas eu de dégradation à effectuer (estimée spécialiste).
La résolution initiale des deux requêtes est identique ainsi que le diagnostic de l'évaluation (il ya trop de références résultats). Par contre le processus de reformulation est différent puisque l'amplitude de la reformulation est différente (spécificité à un niveau pour les spécialistes et à deux niveaux pour les moyens).
Le résultat final est donc différent quant au nombre de réponses (une seule très pertinente pour le spécialiste, et cinq également très pertinentes pour le moyen). La stratégie de reformulation mise en place est donc bien cohérente avec notre modèle d'évaluation des références.

REQUETES	TYPOLOGIE	RESULTAT	DIAGNOST
bordereau de documentation **(formulation initiale)**	M	u01030203 1 .02 u01040501 1 .02 u01040201 1 .02 u01040303 1 .02 u01040304 1 .02 u010104 1 .02 u01030202 1 .02	trop
bordereau joint ou exemplaire de bordereau ou partie de bordereau ou état de bordereau et documentation **(reformulation)**	M	u01010404 1 1 u01030203 1 1 u01040302 1 1 u01010502 1 .03 u01010503 1 .03	bon
bordereau et documentation **(formulation initiale)**	S	u01030203 1 .02 u01040501 1 .02 u01040201 1 .02 u01040303 1 .02 u01040304 1 .02 u010104 1 .02 u01030202 1 .02	trop
bordereau joint et documentation **(reformulation)**	S	u01040302 1 1	bon

Tableau 2.1

Au niveau qualitatif, on peut noter que la référence résultat de "bordereau et documentation" n'est pas dans le résultat théorique du concept de "bordereau de documentation" (voir tableau 1). Ceci est principalement du à la dégradation résultant de la perte de la liaison prépositionnelle. La requête "bordereau de documentation" fournit par contre un ensemble de références beaucoup plus pertinent (rappel = 1 et précision = 0.75). Il est normal que le rappel soit supérieur puisque la reformulation a été plus large, par contre il n'était pas évident que la précision soit bonne après une reformulation d'une telle ampleur.

4.2. Evaluation de la reformulation

Nous observons les effets de la reformulation dans les 3 cas de diagnostics possibles :
(1) Faible qualité des références :
Nous avons choisi la requête "norme" (voir tableau 2.2). Après la première résolution, le résultat obtenu présente une qualité assez faible, et nous reformulons donc par la relation de voisinage associée. La nouvelle requête ("normalisation OU qualité de documentation") obtient par contre des références avec de bonnes mesures de pertinence. De plus relativement au concept initial, les taux de rappel et de précision sont bons (respectivement 1 et 0.66).

(2) Faible nombre de références :
Nous avons choisi la requête "établissement de la documentation technique de référence" (voir tableau 2.2). La première résolution ne nous fournit aucun résultat, la reformulation se fait donc par les génériques. Nous obtenons alors une référence estimée de faible qualité. IOTA essaye alors d'améliorer cette qualité mais ne peut y parvenir. Nous sommes donc là dans le cas d'échec sur la reformulation, qui est résolu en choisissant le meilleur résultat obtenu (au sens de la qualité).
La référence obtenue est de plus partiellement pertinente (voir tableau 1), puisqu'elle est une référence fille de la référence pertinente.

(3) Grand nombre de références :
Nous avons choisi la requête "matériel ou logiciel" (voir tableau 2.3). La première résolution nous fournit un grand nombre de références (8), bien trop important pour un utilisateur spécialiste. La reformulation s'effectue d'abord en supprimant l'opérande du OU apportant le plus grand nombre de références (ici "matériel"). Cette reformulation est invalidée (la qualité des références est trop faible).

REQUETES	TYPOLOGIE	RESULTAT	DIAGNOST
norme **(formulation initiale)**	M	u01020702 .12 .33 u01030202 .00 .33	qualité faible
normalisation ou qualité de documentation **(reformulation)**	M	u0102 1 .5 u0202 .2 1 u01010105 .1 .5	bon
établissement de la documentation technique de référence **(formulation intiale)**	S	aucune référence	peu
établissement ou établissement de documentation technique de référence et documentation de référence **(reformulation)**	S	u01040103 .02 .16	qualité faible
établissement de documentation technique et documentation de référence **(reformulation)**		invalidé	
établissement et documentation de référence **(reformulation)**	S	u01040103 .02 .16	qualité faible

Tableau 2.2

REQUETES	TYPOLOGIE	RESULTAT	DIAGNOST
logiciel ou matériel **(formulation initiale)**	S	u01040203 1 .04 u01010404 .11 .16 u01010202 .11 .01 u01010501 .11 .08 u01030202 .01 .13 u01030201 .06 .04 u01040304 .06 .04 u01020202 .05 .04	trop
logiciel **(reformulation)**		invalidé	
matériel **(reformulation)**	S	u01040203 1 .04 u01010404 .11 .16 u01010202 .14 .08 u01030202 .01 .13 u01040304 .06 .04 u01020202 .05 .04	trop
type de matériel ou liste de matériel ou installation de matériel **(reformulation)**	S	u01030202 .00 1 u01020501 .14 .5 u01010501 .11 .5 u01030201 .06 .5	bon

Tableau 2.3

La reformulation avec "matériel" diminue le nombre de références qui est encore élevé (6), on reformule donc avec les spécifiques de "matériel". On obtient alors un nombre de références conforme.
La qualité des références obtenues est très bonne (rappel et précision de 1).

Sur les requêtes expérimentées, nous avons obtenu de bonnes reformulations permettant d'améliorer à la fois le rappel et la précision. Cependant, le processus de reformulation reste encore à améliorer, notamment en considérant comme élément de reformulation non plus un simple terme mais un ensemble de termes. En effet, nous ne reformulons qu'un concept à la fois, et non pas une sous expression de l'équation de recherche. La reformulation d'une sous expression nous permettrait d'utiliser des relations contextuelles qui sont souvent très intéressantes. Par exemple, sur notre corpus "document" dans le contexte de "marché" peut être substitué par "bordereau de documentation", ce qui fait que la sous expression "document ET marché" est bien reformulée par "bordereau de documentation".

5. Performances en temps de IOTA

Nous donnons ici quelques mesures effectuées sur le temps d'unité centrale requis par le traitement des requêtes. Cette évaluation est très indicative et permet de donner des ordres de grandeur. Sur les requêtes présentées au 3, nous avons mesuré les temps suivants (en secondes d'UC) :

REQUETES	TEMPS UC
Numéro 1	5.07
Numéro 2	4.47
Numéro 3	16.95
Numéro 4	18.44
Numéro 5	13.95
Numéro 6	15.14
Numéro 7	15.1
Numéro 8	21.8
Numéro 9	21.93
Numéro 10	11.54
Numéro 11	7.54
Numéro 12	21.4
Numéro 13	24.47

Même s'il est difficile d'estimer si ces requêtes sont significatives, on peut dire que le temps de traitement n'est pas demesuré (environ 25 secondes au maximum).
Le coût approximatif d'un cycle de reformulation (reformulation proprement dite plus

interprétation de la nouvelle équation) est de l'ordre de 5 secondes. Ce temps doit être augmenté si l'équation est assez compliquée. En effet, si on essaye d'estimer plus finement le comportement de IOTA, on s'aperçoit que le traitement le plus coûteux en temps est le plus souvent l'étape de composition du résultat et l'accès aux textes. Si on veut obtenir un outil réellement performant, il serait donc souhaitable de faire porter les efforts d'optimisation sur cette partie du traitement.

Les performances observées devraient être améliorée de façon importante en développant une version compilée de IOTA. Cette amélioration devrait également nous permettre de gérer des corpus beaucoup plus volumineux.

6. Réflexions sur l'expérimentation

L'expérimentation de IOTA, en plus du fait qu'elle a permis de valider un certain nombre d'hypothèses faites, nous a fourni des renseignements intéressants quant aux différentes connaissances et heuristiques utilisées. En effet, si globalement celles ci ont été validées, nous avons pu nous apercevoir que la définition des seuils est très liée au corpus et qu'il ne parait très réaliste de les fixer a priori. L'ensemble des seuils doit donc être défini comme un paramètre du corpus. Par exemple, sur ce corpus les mesures de pertinence associées aux références lors de l'indexation sont assez faibles, ce qui nous a obligé à réduire considérablement les seuils de qualité.
De même, le nombre de références correspondant à chaque catégorie d'utilisateur a du être réduit, dans la mesure où pour de la documentation technique le nombre de références est rarement important (3 ou 4 références en moyenne).

7. Exemples de sessions

Nous donnons deux exemples de sessions sous IOTA (voir annexe), la première décrivant le traitement d'une requête formée d'un concept large "papier" qui nécessite de nombreuses reformulations, et la seconde le traitement d'une requête plus précise "établissement de la documentation technique de référence".
On peut noter que les requêtes précises nécessitent un gros effort de compréhension, et une interprétation souvent simple alors que pour des requêtes larges la proportion est inversée.

CHAPITRE VIII

CONCLUSION

D'une manière générale, le problème de la recherche d'informations est de trouver, à partir d'une requête donnée, une formulation (ou une reformulation) dans les termes du système équivalente à cette requête, et devant permettre de trouver un ensemble de références satisfaisant.
Ce problème d'ordre sémantique (équivalence entre deux formulations) est, malgré de nombreux travaux, difficile à résoudre dans un cadre général (il faudrait un modèle de connaissances du monde).

Les SRI classiques ne tentent pas de résoudre ce problème puisqu'ils traitent (à peu de choses près) les requêtes dans leur formulation initiale, sans chercher à les reformuler sous une forme plus efficace. Cette restriction importante oblige l'utilisateur à trouver de lui même une bonne formulation à sa requête, ce qu'il n'est pas capable de faire sans une connaissance approfondie du système et des bases gérées. Dans le plupart des cas, cette tâche revient donc au documentaliste qui est l'intermédiaire obligatoire de tels systèmes; cet inconvénient est aggravé par la nature même des langages d'interrogation qui ne favorise pas non plus une utilisation directe par le plus grand nombre.

Nous avons tenté d'apporter notre contribution à ces problèmes en utilisant un certain nombre d'outils d'intelligence artificielle permettant de soulager quelque peu l'utilisateur de ces contraintes en définissant un système acceptant des requêtes en langue quasi-naturelle, et en modélisant l'activité des documentalistes de la façon suivante :

- construction de l'équation initiale de recherche à partir de la formulation en langue naturelle,
- évaluation des réponses fournies,
- reconstruction éventuelle de l'équation de recherche en fonction des résultats de l'évaluation.

Cette modélisation nous a permis d'améliorer les performances globales de la recherche d'informations, à travers le rappel (recherche d'un ensemble de réponses plus conforme à notre attente grâce au processus de reformulation automatique), la précision (utilisation de mesures de pertinence requête - référence et de modèles d'évaluation des réponses), tout en

présentant une plus grande convivialité (utilisation d'un langage quasi-naturel, prise en compte de la typologie de l'utilisateur).

Sur le plan pratique, l'utilisation des techniques liées aux systèmes experts, nous a permis de réaliser un prototype très modulaire, présentant une répartition des tâches entre modules classiques et modules experts très nette. La mise au point des connaissances expertes peut donc se faire de façon incrémentale, ce qui est très important durant la phase intiale de validation.

On peut cependant dire que la puissance d'inférences des systèmes experts n'est pas actuellement très utilisée, ce qui peut mettre en cause la justification technique de tels outils. L'approche système expert s'est néanmoins avérée très riche comme méthode d'analyse de ce type d'application, dans la mesure où elle permet de bien séparer les problèmes de modélisation du documentaliste, des processus classiques de recherche d'informations (accès aux documents, composition des réponses), leur coopération étant de plus aisée à mettre en œuvre.

C'est pourquoi nous préférons parler de "système intelligent de recherche d'informations" au lieu de "système expert en recherche d'informations", le terme d'expert ne se justifiant pas complétement dans l'état actuel; il nous faudrait établir une base de connaissances beaucoup plus élaborée que celle dont nous disposons, et dont l'exploitation justifierait les mécanismes inférentiels sophistiqués propres aux systèmes experts.

Sur un autre plan, il nous semble que l'approche intelligence artificielle soit pour l'instant plus prometteuse que l'approche base de données. En effet les modèles de données classiques sont mal adaptés à la prise en compte des aspects sémantiques de la correspondance entre une requête et des documents. Cependant l'émergence de nouveaux modèles de données, dits sémantiques, couplés avec les bases de données déductives, peut conduire à revoir complétement ce point de vue, s'ils débouchent sur des systèmes aux possibilités inférentielles voisines des systèmes d'intelligence artificielle, et pouvant de plus gérer des masses de connaissances beaucoup plus importantes.

Une évaluation approfondie du prototype tant sur les aspects qualitatifs que quantitatifs est encore nécessaire. En ce qui concerne les aspects quantitatifs, il est important d'évaluer IOTA sur des corpus plus significatifs quant à leur taille (une collection de documents et non pas un chapitre comme c'est le cas actuellement) et à leur type (il est intéressant d'évaluer IOTA sur des corpus de natures différentes).

Il est toujours difficile d'évaluer les performances qualitatives en termes de taux de silence et de bruit, car nous ne disposons pas encore d'une collection de documents et de requêtes résolues. L'évaluation de IOTA ne peut donc être actuellement effectuée qu'au travers des utilisateurs classiques d'un SRI, à savoir les documentalistes et les usagers finaux.

Le prototype réalisé est, pour l'instant encore, très expérimental et de nombreuses améliorations d'ordre technique peuvent lui être apportées. Dans l'immédiat, deux améliorations significatives sont prévues :

- la compilation des fonctions de IOTA :
elle doit permettre une augmentation importante des performances du système (on espère un facteur de gain de l'ordre de 5),

- une gestion des données sur support externe :
toutes les données sont gérées actuellement en mémoire centrale ce qui interdit l'exploitation de corpus volumineux. Il est indispensable d'intégrer les primitives d'accès aux données externes depuis LELISP (mécanisme type fichier à accès direct à partir d'un nom d'atome), qui sont apparues dans les versions développées récemment.

Sur le plan fonctionnel, des extensions à IOTA sont prévues, notamment au niveau de l'interface utilisateur et des possibilités d'apprentissage. Nous développons dans le groupe un outil de compréhension des requêtes en langue naturelle ("understander") plus performant que celui actuellement utilisé dans le prototype. Les résultats attendus (et déjà partiellement observés) en sont une meilleure interprétation des opérateurs de recherche sous-jacents, de même qu'une meilleure identification des concepts à travers des constructions syntaxiques plus libres (relatives, ponctuation, paraphrasage, ...).
L'autre aspect très important concerne les possibilités d'apprentissage au niveau des données et des connaissances requises. Un certain nombre d'outils ont déjà été présentés ici, mais il reste à étudier des mécanismes plus riches et plus généraux (utilisation de règles d'accord en genre et en nombre et d'une matrice de précédence pour l'inférence de la catégorie grammaticale d'un mot inconnu, utilisation de relations sémantiques pour l'inférence de concepts inconnus).

Peu d'études ont été entreprises à ce jour dans le domaine des systèmes intelligents de recherche d'informations, et moins encore ont abouti à la réalisation de prototypes; il est donc malaisé de situer IOTA par rapport à un existant, comme on pourrait le faire dans d'autres domaines. Notre souhait est donc que ce travail ait apporté une contribution effective à un thème que nous estimons très prometteur.

BIBLIOGRAPHIE :

[ABR 74] J.R.ABRIAL :
"Data semantics"
IFIP TC2 Working Conference, Cargèse, avril 1974

[ADI 85] M.ADIBA :
"Les bases de données généralisées"
Actes Conférence RIAO 85, Grenoble, 18-20 mars 1985

[ALB 84] P.ALBERT :
"PROLOG et les objets"
Actes Congrès R.D.F.I.A, Paris, 1984

[BAR 77] J.A.BARRET, M.I.BERNSTEIN :
"Knowledge based system : a tutorial"
US Dept of Commerce, 1977

[BAT 83] M.BATES, R.J.BOBROW :
"Information retrieval using a transportable natural language interface"
Proc. 6th ACM SIGIR Conference Research and Development in Information Retrieval, Bethesda, 1983

[BEG 85] : J.O.BEGOUIN, J.P.COUSSOULET, S.MIRANDA :
"Interface uniforme d'accès à une base de données factuelle et textuelle"
Actes Conférence RIAO 85, Grenoble, 18-20 mars 1985

[BEN 85] A.BENSAID :
"SHIVA - un modèle de données relationnel étendu pour la mise en œuvre de bases de connaissances centrées objets"
Thèse Docteur Ingénieur, INPGrenoble, mai 1985

[BOG 82] B.K.BOGURAEV, K.SPARCK JONES :

"A natural language analyser for database access"
Information Technology : Research and Development 1982, 1 (23-39)

[BON 84] A.BONNET :
"L'utilisation des langages orientés objets (LOOBS) en intelligence artificielle"
Journées Systèmes experts, Avignon, 1984

[BRU 85] M.F.BRUANDET :
"Modèle partiel de connaissance pour un système de recherche d'informations"
Actes Conférence RIAO 85, Grenoble, 18-20 mars 1985

[CHA 85] J.CHAILLOUX :
"LELISP de l'INRIA : manuel de référence"
INRIA, février 1985

[CHI 86a] Y.CHIARAMELLA, D.KERKOUBA, M.F.BRUANDET :
"Intégration d'une fonction documentaire dans un atelier de logiciel"
Journées CONCERTO, Perros Guirrec, France, février 1986

[CHI 86b] Y.CHIARAMELLA, B.DEFUDE, D.KERKOUBA, M.F.BRUANDET :
"IOTA : a prototype of an information retrieval system"
ACM SIGIR Conference, Pisa, Italie, sept 1986

[COL 83] A.COLMERAUER, H.KANOUI, M.VAN CANEGHEM :
"PROLOG, bases théoriques et développements actuels"
TSI, vol 2, No 4, 1983

[COO 64] W.S.COOPER :
"Fact retrieval and deductive question-answering information retrieval systems"
JACM, 11, No 2, pp 117-137, 1964

[CRA 83] J.B.CRAMPES, C.Y.CHRISTMENT, G.ZURFLUH :
"The BIG project"
Proc. 2nd International Conference, ICOD, Cambridge, Sept 1983

[CRO 80] W.B.CROFT :
"A model of cluster searching based on classification"

Information Systems, 5; pp 189-195, 1980

[CRO 83] W.B.CROFT, R.WOLF, R.THOMPSON :
"A network organization used for document retrieval"
Proc. 6th ACM SIGIR Conference, Research and Development in Information Retrieval, Bethesda, 1983

[CRO 85] W.B.CROFT :
"An expert assistant for a document retrieval system"
Actes Conférence RIAO 85, Grenoble, 18-20 mars 1985

[DAV 77] R.DAVIS, J.KING :
"An overview of production systems"
Machine Intelligence 8, pp 300-332, 1977

[DAV 80] J.M.DAVID :
"Simulation de l'activité d'un documentaliste"
Thèse de 3ème cycle, INPLorraine, Nancy, 1980

[DEB 82] F.DEBILI :
"Analyse syntaxico-sémantique fondée sur une acquisition automatique de relations lexicales-sémantiques"
Thèse d'état, Univ. Paris-Sud (Paris 11), 1982

[DEF 84] B.DEFUDE :
"Knowledge based system versus thesaurus : an architecture problem about expert system design"
Proc. 3rd ACM & BCS Symposium, Research and Development in Information Retrieval, Cambridge University Press (C.J Van Rijsbergen Ed.), 1984

[DEF 85] B.DEFUDE :
"Different levels of expertise for an expert system in Information Retrieval"
Proc. 8th International ACM SIGIR Conference on Research and Development in Information Retrieval, Montréal, Canada, june 1985

[DEL 82] C.DELOBEL, M.ADIBA :
"Bases de données et systèmes relationnels"

Dunod - Informatique, 1982

[DEO 85] J.S.DEOGUN, V.V.RAGHAVAN :
"Integration of information retrieval and data-bases management systems"
Actes Conférence RIAO 85, Grenoble, 18-20 mars 1985

[DEW 81] A.DEWEZE :
"Réseaux sémantiques"
Thèse d'état, Univ. C Bernard Lyon, 1981

[DUD 77] R.O.DUDA, P.E.HART, N.J.NILSONN, G.L.SUTHERLAND :
"Semantic network representation for rule based inference system"
Stanford Research Institute, Menlo Park, California, 1977

[EIS 85] K.P.EISELT :
"A parallel-Process Model of On-Line Inference Processing"
in Proc. 9th IJCAI 85, Los Angeles, 1985

[FLU 77] C.FLUHR :
"Algorithmes à apprentissage et traitement automatique des langues"
Thèse d'état, Univ. Paris Sud, Centre d'Orsay 1977

[GAL 84] H.GALLAIRE, J.MINKER, J.M.NICOLAS :
"Logic and databases : A deductive approach"
Computing Surveys, vol 16, No 2, juin 1984

[GAL 85] H.GALLAIRE :
"PROLOG et la programmation objet"
Séminaire LIFIA, Grenoble, Déc 1985

[GEO 83] M.P.GEORGEFF, U.BONOLLO :
"Procedural expert systems"
Proc. of the 8th IJCAI, Karlsruhe, West Germany 1983

[GUI 83] G.GUIDA, C.TASSO :
"IR-NLI : an expert natural language interface to online data bases"
Proc. Conf. on Applied Natural Language Processing, Santa Monica, 1983

[HAY 83] F.HAYES-ROTH, D.A.WATERMAN, D.B.LENAT :
"Building expert systems"
Addison Wesley, Reading, Massac., 1983

[KAU 75] A.KAUFMANN :
"Introduction à la théorie des sous ensembles flous"
Tome 1 à 3, Masson, Paris, 1975

[KER 84] D.KERKOUBA :
"Une méthode d'indexation automatique des documents fondée sur l'exploitation de leurs propriétés structurelles. Application à un corpus technique"
Thèse 3ème cycle INPGrenoble, 1984.

[KER 85] D.KERKOUBA :
"Indexation automatique et aspects structurels du texte"
Actes Conférence RIAO 85, Grenoble, 18-20 mars 1985

[KUI 75] B.KUIPERS :
"A frame for frames : representing knowledge for recognition"
in Bobrow & Collins (Eds) Representation and understanding, Academic Press, N.Y, 1975

[LAU 81] J.L.LAURIERE :
"Représentation et utilisation des connaissances"
TSI, vol 1, No 1 et 2, 1981

[LAU 84] J.P.LAURENT :
"La structure de contrôle dans les systèmes experts"
TSI, vol 3, No 3, 1984

[LEM 85] : J.LEMAITRE :
"VIDOC un langage relationnel pour non informaticiens"
Actes Conférence RIAO 85, Grenoble, 18-20 mars 1985

[LOP 79] M.LOPEZ :
"Communication en langue naturelle avec un système d'aide à la conception d'assemblages physiques : un essai d'utilisation des réseaux sémantiques partitionnés"
Thèse Docteur Ingénieur, INPGrenoble, 1979

[LUK 30] J.LUKASIEWICZ :
"Many-valued systems of propositional logic"
in S.McCall, Polish Logic, O.U.P, 1957

[MCC 82] M.C.McCORD :
"Using slots and modifiers in logic grammars for natural language"
Artificial Intelligence, 18, 1982

[MCL 85] : I.A.McLEOD :
"Three approaches to information retrieval"
Actes Conférence RIAO 85, Grenoble, 18-20 mars 1985

[MIC 83] R.S.MICHALSKI, J.G.CARBONELL, T.M.MITCHELL :
"Machine Learning : an artificial intelligence approach"
Tioga publishing Company, Palo Alto, California, 1983

[NIE 85] J.NIE :
"Compréhension de requêtes en langue naturelle. Pré-étude dans le contexte d'un système de recherche d'informations"
Rapport de DEA INPG/USMG, Grenoble, 1985

[PAL 85] P.PALMER, C.BERRUT :
"Etude d'un analyseur de surface de la langue naturelle pour un système de recherche d'informations"
Proc. 13th annual CAIS Conference, Montréal, juin 1985

[PER 80] F.C.N.PEREIRA, D.H.D.WARREN :
"Definite clauses grammars for language analysis - A survey of the formalism and a comparison with augmented transition networks"
Artificial Intelligence, 13, 1980

[POL 83] S.E.POLITT :
"End user touch searching for cancer therapy literature : a rule based approach"
Proc. of the ACM SIGIR Conference, SIGIR Forum, 17, 136-145, 1983

[RIC 79] E.RICH :
"Building and exploiting user models"

Ph.D Thesis, technical report, CMU-CS-79-119, CMU, 1979

[RIC 85] E.RICH et al :
"Panel session on user models"
Proc. 9th IJCAI 85, Los Angeles, 1985

[RIJ 79] C.J.VAN RIJSBERGEN :
"Information retrieval"
Second édition, Buttersworth London England 1979.

[SAL 71] G.SALTON :
"The SMART retrieval system"
Prentice-Hall, Englewood Cliffs, N.J, 1971

[SAL 83] G.SALTON, M.J.McGILL :
"Introduction to modern information retrieval"
Mcgraw Hill Book Company, New York, 1983

[SAL 85] G.SALTON :
"A note on information retrieval models and theories"
Actes Conférence RIAO 85, Grenoble, 18-20 mars 1985

[SCH 77] R.C.SCHANK, R.P.ABELSON :
"Scripts, Plans, Goals and understanding"
Lawrence Erlbaum Associates, Publishers, Hillsdale, New Jersey, 1977

[SCH 80] R.C.SCHANK, M.LEBOWITZ, L.BIRNBAUM :
"An integrated understander"
in American Journal of Computational Linguistics, vol 6, No 1, 1980

[SIC 83] : Club SICO de l'INRIA et groupe de travail sur l'intelligence artificielle du CNRS
"Rapport sur l'intelligence artificielle"
TSI, vol 2, No 5, 1983

[SMI 80] L.C.SMITH :
"Artificial intelligence applications in Information Systems"
Ann. Rev. Inform. Sci. Technol., 15, pp 67-115, 1980

[SPA 78] K.SPARCK JONES :
"Artificial intelligence : what can it offer to information retrieval"
in Informatics 3, Aslib, London, 1978

[STE 82] M.STEFIK, A.G.BELL, D.G.BOBROW :
"Rule-oriented programming in LOOPS"
Memo KB-VLSI-82-22 (working paper), Xerox Parc, 1982

[TON 85] R.M.TONG, V.N.ASKMAN, J.F.CUNNINGHAM, C.J.TOLLANDER :
"RUBRIC - An environment for Full Text Information Retrieval"
Proc. 8th International ACM SIGIR Conference on Research and Development in Information retrieval, Montréal, juin 1985

[VEL 84] : F.VELEZ :
"Un modèle et un langage pour les bases de données généralisées"
Thèse Docteur Ingénieur, INPGrenoble, sept 1984

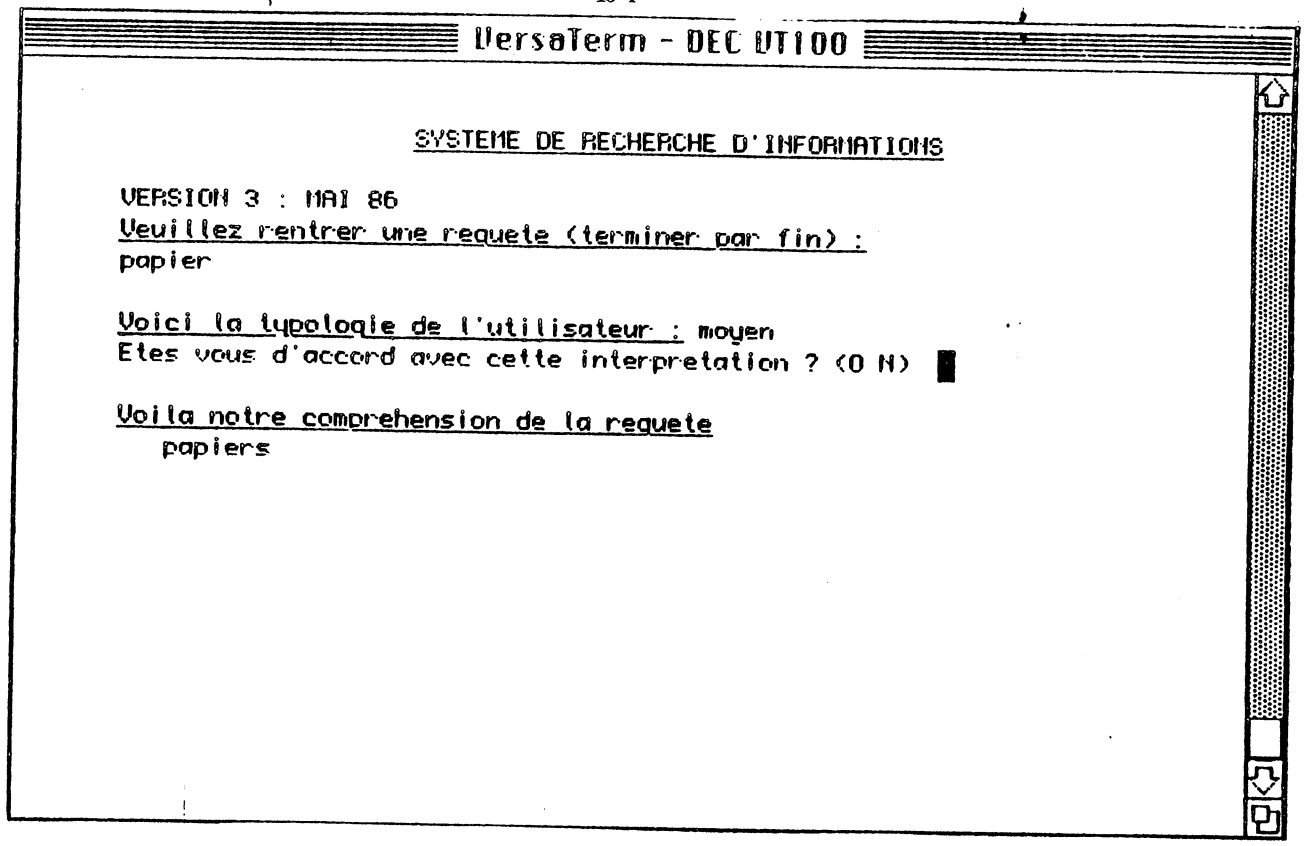
[WOO 70] W.A.WOODS :
"Transition network grammars for natural language analysis"
CACM, 13, pp 591-606, 1970

[YIP 81] M.K.YIP :
"An expert system for document retrieval"
M.S Thesis, Dept of Electrical Engineering and Computer Science, MIT, 1981

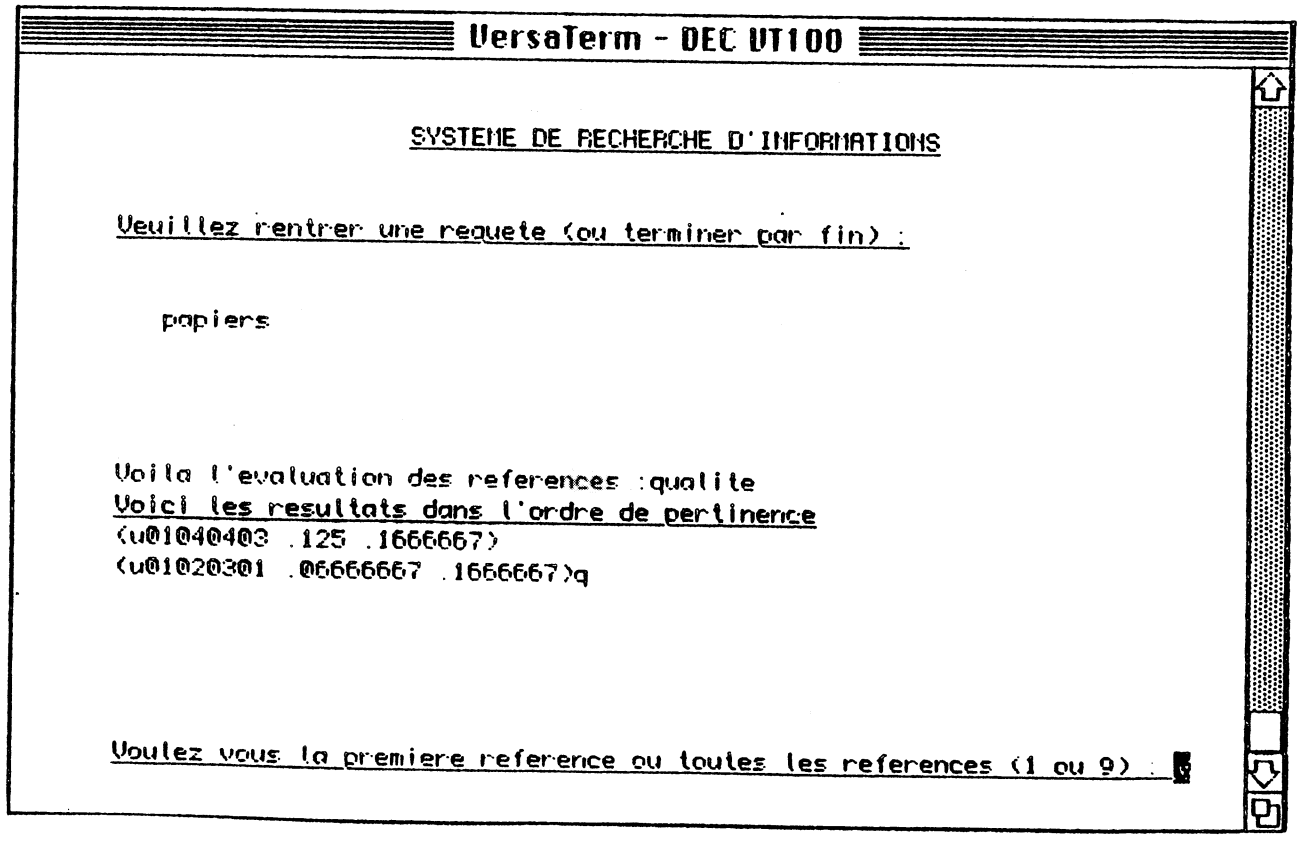
[ZLO 77] M.M.ZLOOF :
"Query-by-example : a data base language"
IBM Systems Journal, vol 16, No 4, 1977

ANNEXE 1

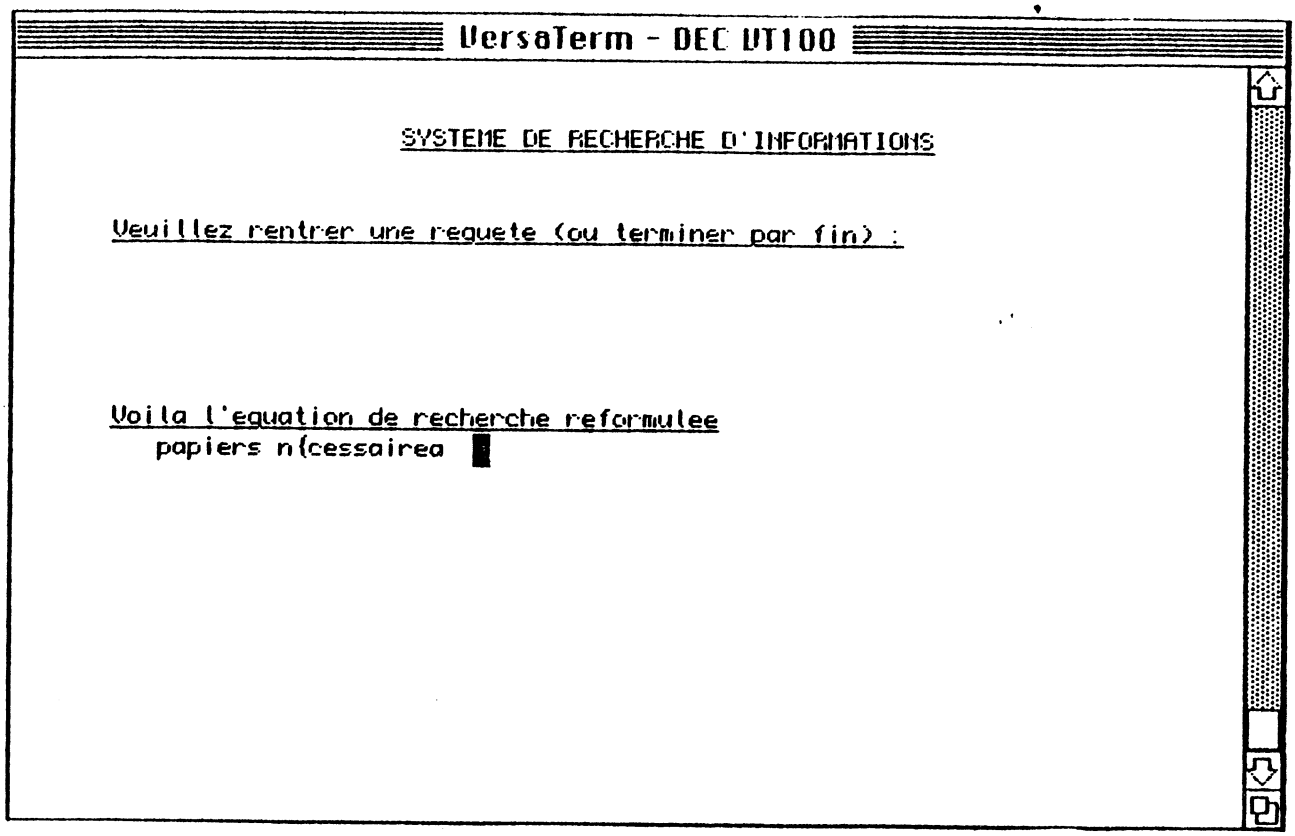
Nous donnons ici à titre d'exemple, deux exemples complets de sessions effectuées sous IOTA (traitement de deux requêtes avec reformulation).



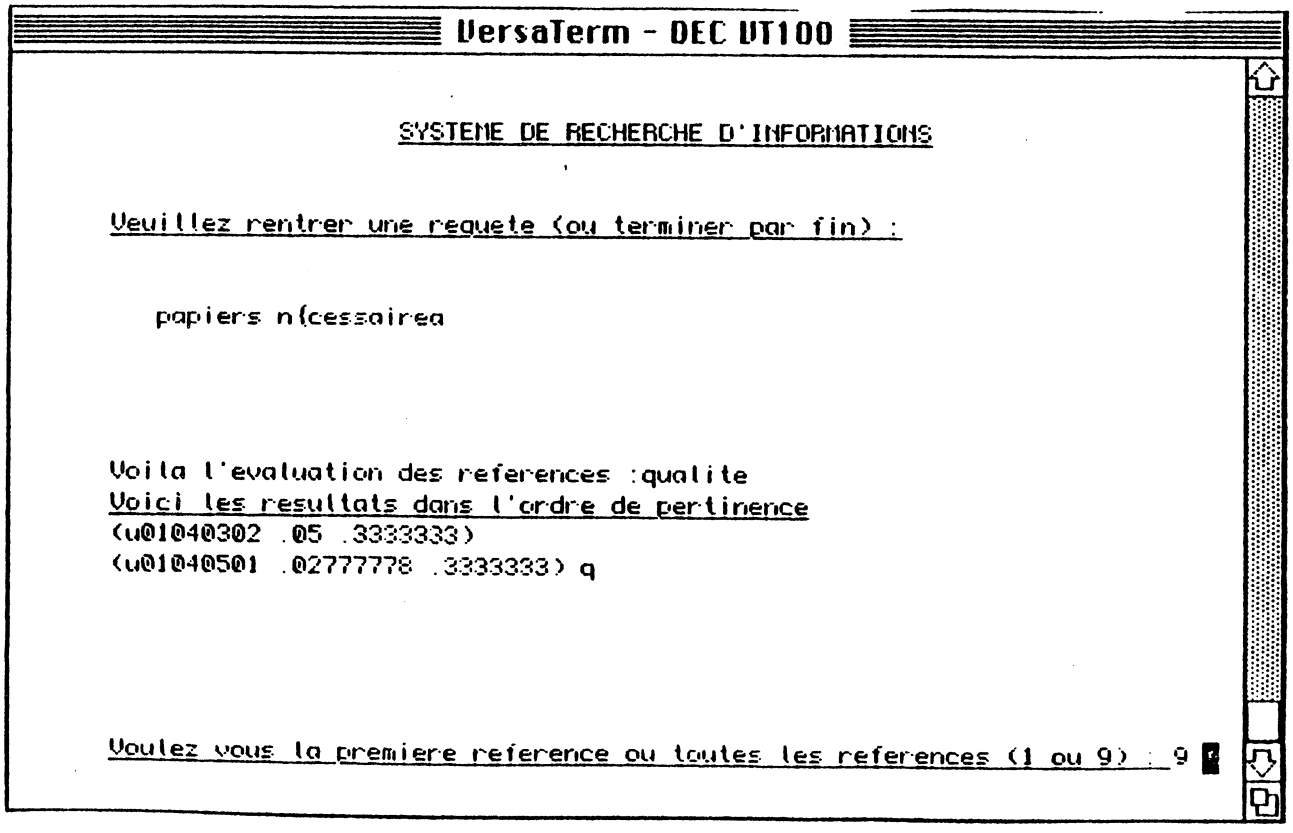
La requête initiale est saisie, et le système affiche sa compréhension à l'utilisateur (expression booléenne de lemmes, c'est à dire mot avec sa catégorie grammaticale accolée). ainsi que la typologie estimée



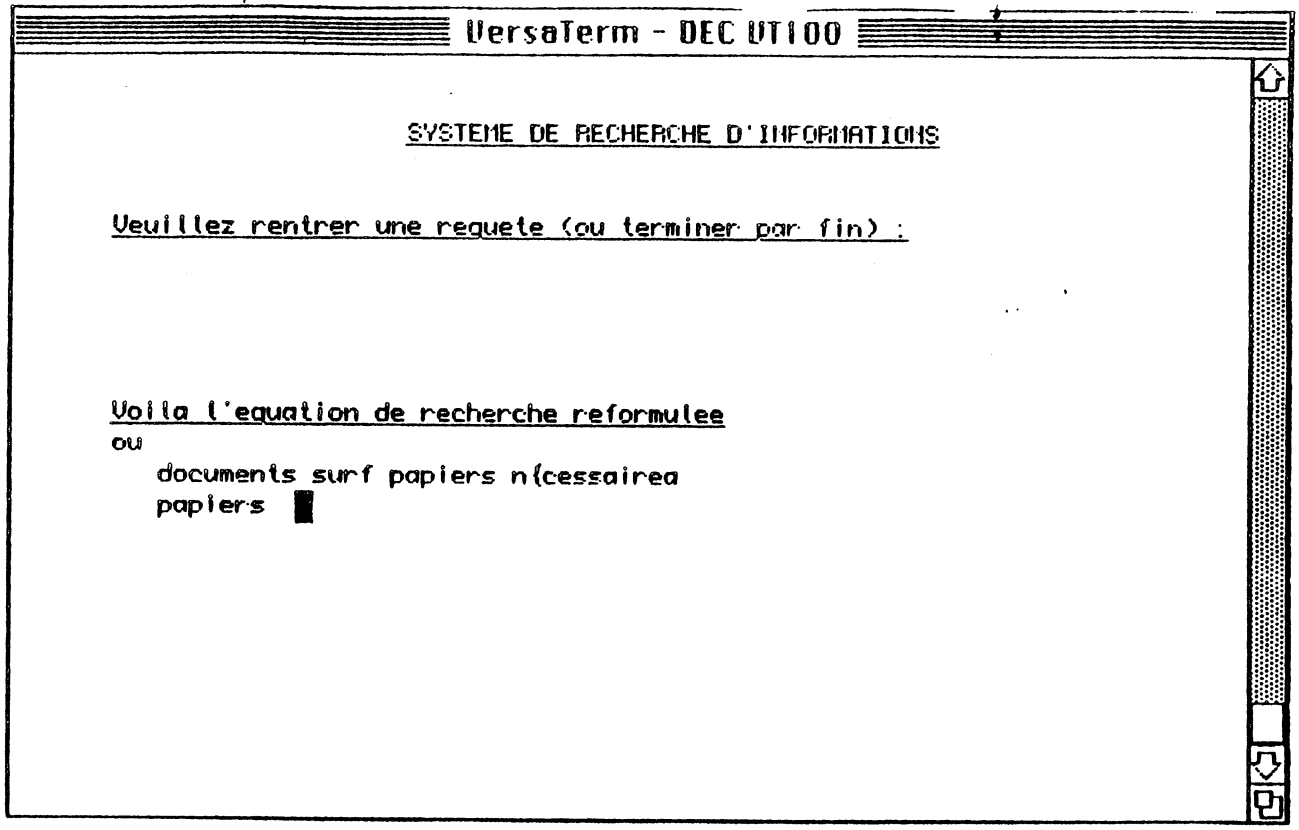
L'équation de recherche initiale est résolue et un premier ensemble résultat est proposé à l'utilisateur avec le diagnostic d'évaluation (ici faible qualité). L'utilisateur peut éventuellement visualiser ces références



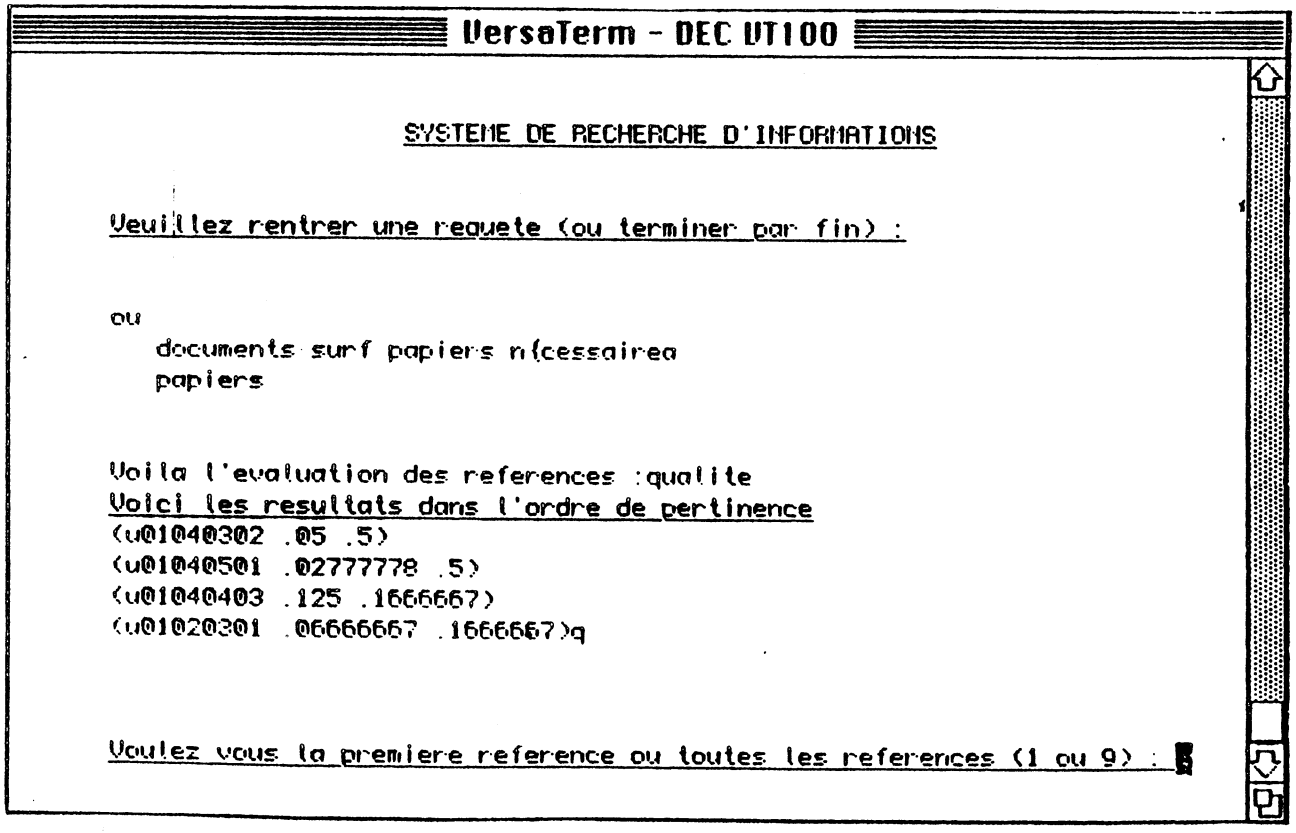
Une première reformulation est signalée à l'utilisateur (utilisation de la relation de voisinage)



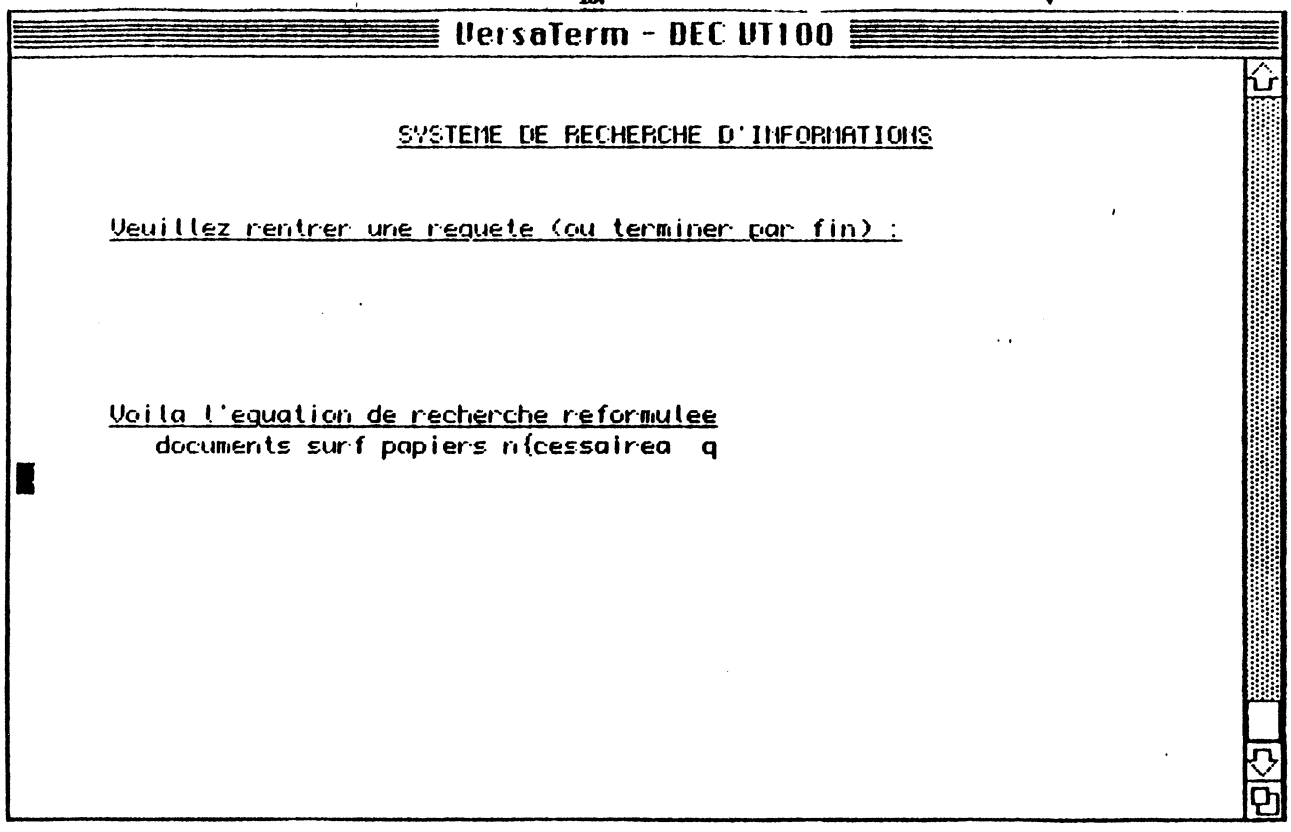
La nouvelle équation est résolue à son tour, et l'on affiche de même le résultat (la qualité est toujours faible)



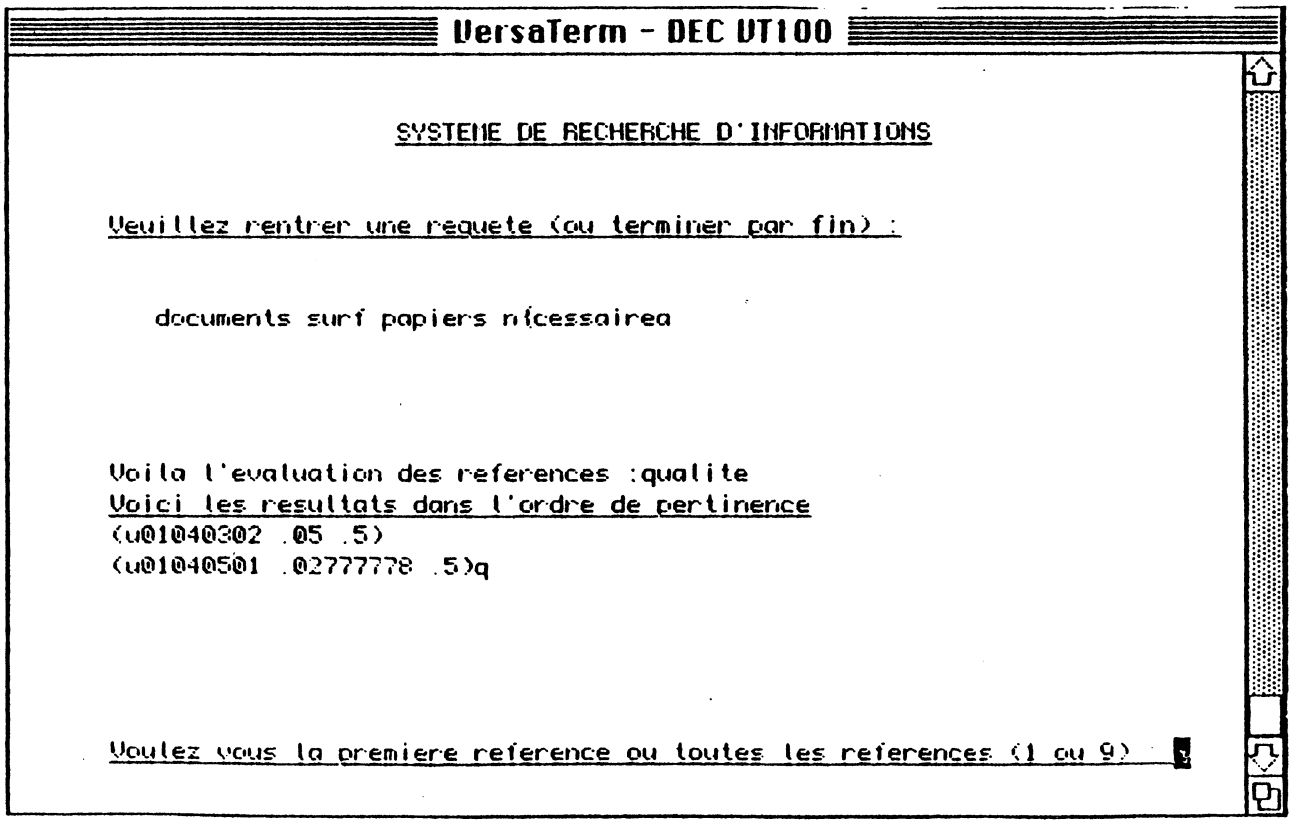
Une nouvelle reformulation est signalée à l'utilisateur (utilisation de la relation de voisinage)



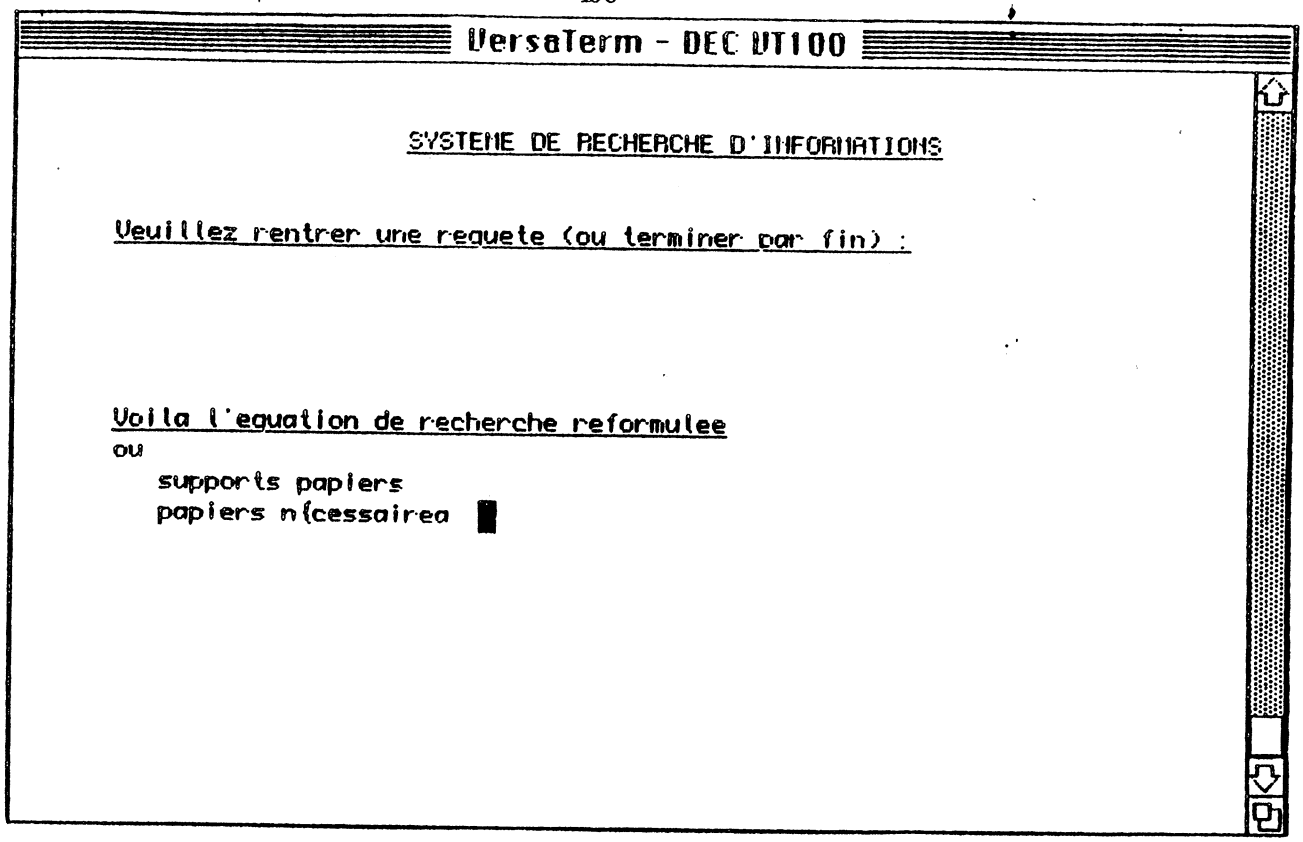
Le résultat est toujours de faible qualité



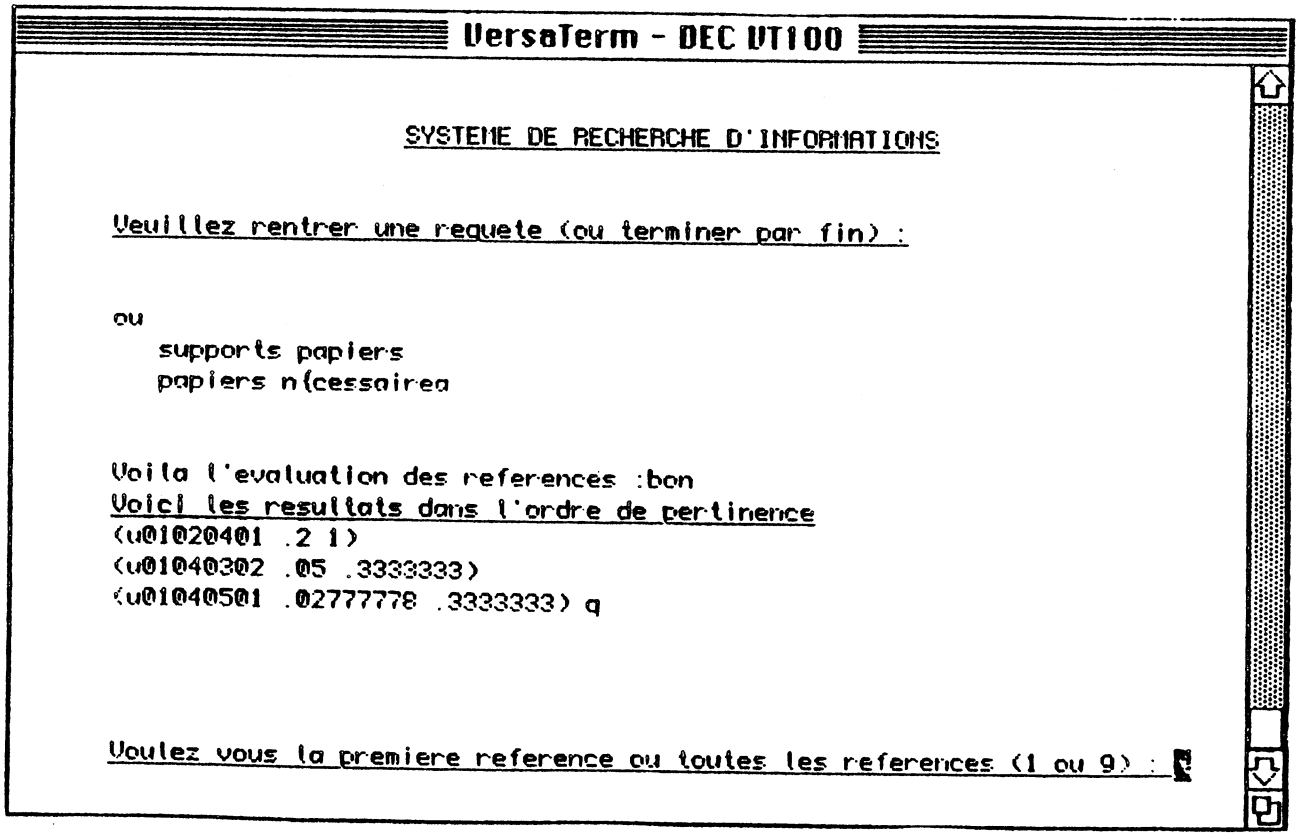
On reformule encore, en supprimant l'opérande du OU apportant les références de plus mauvaise qualité



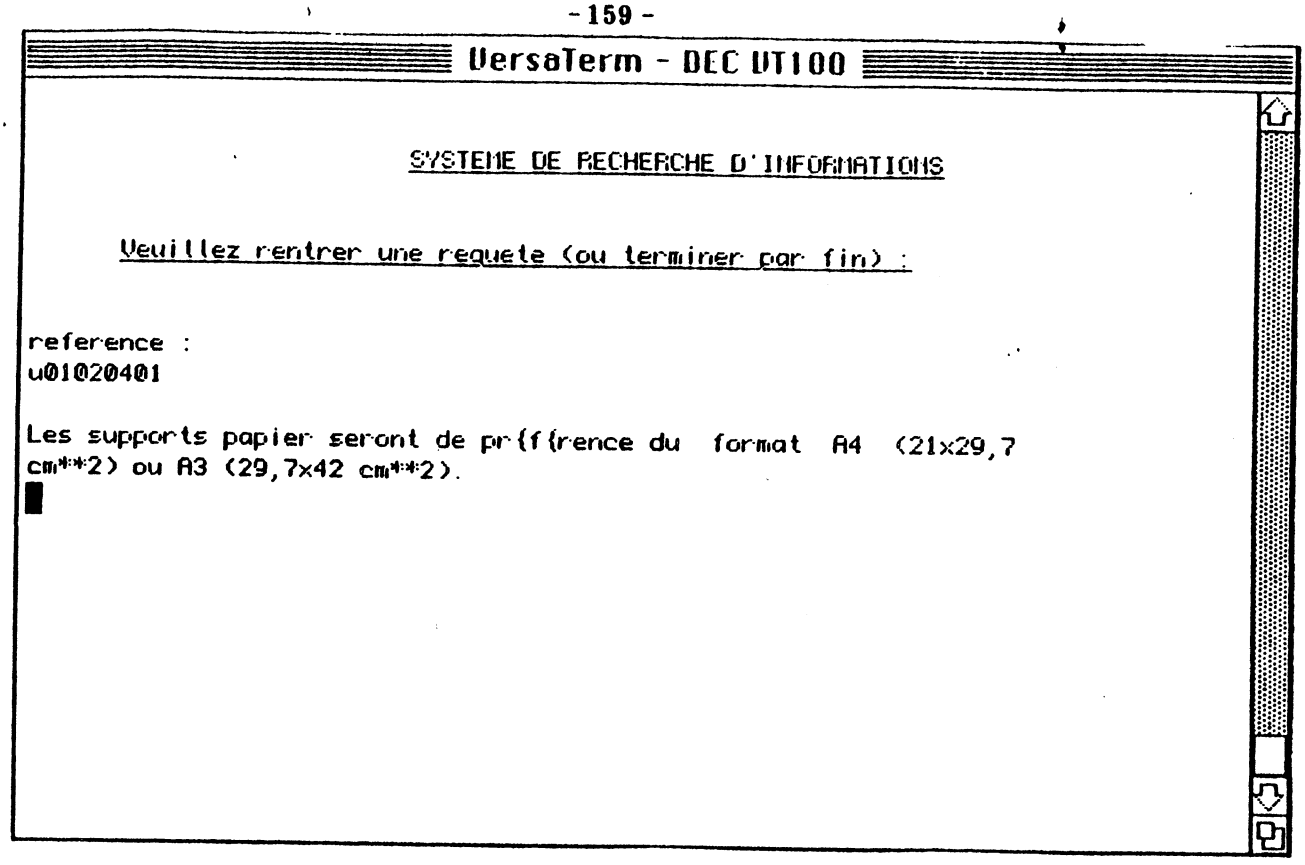
Le résultat demeure de faible qualité



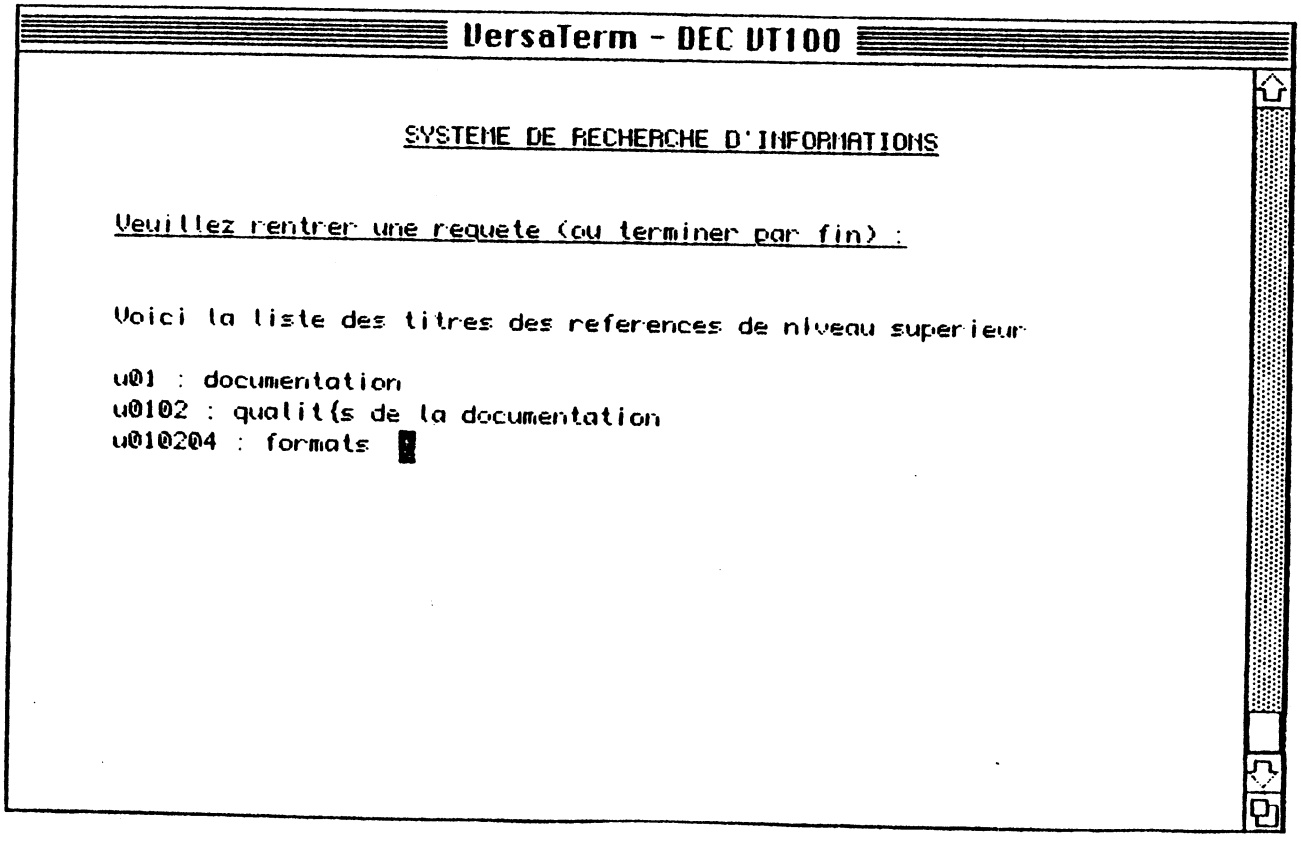
On reformule avec la relation de voisinage



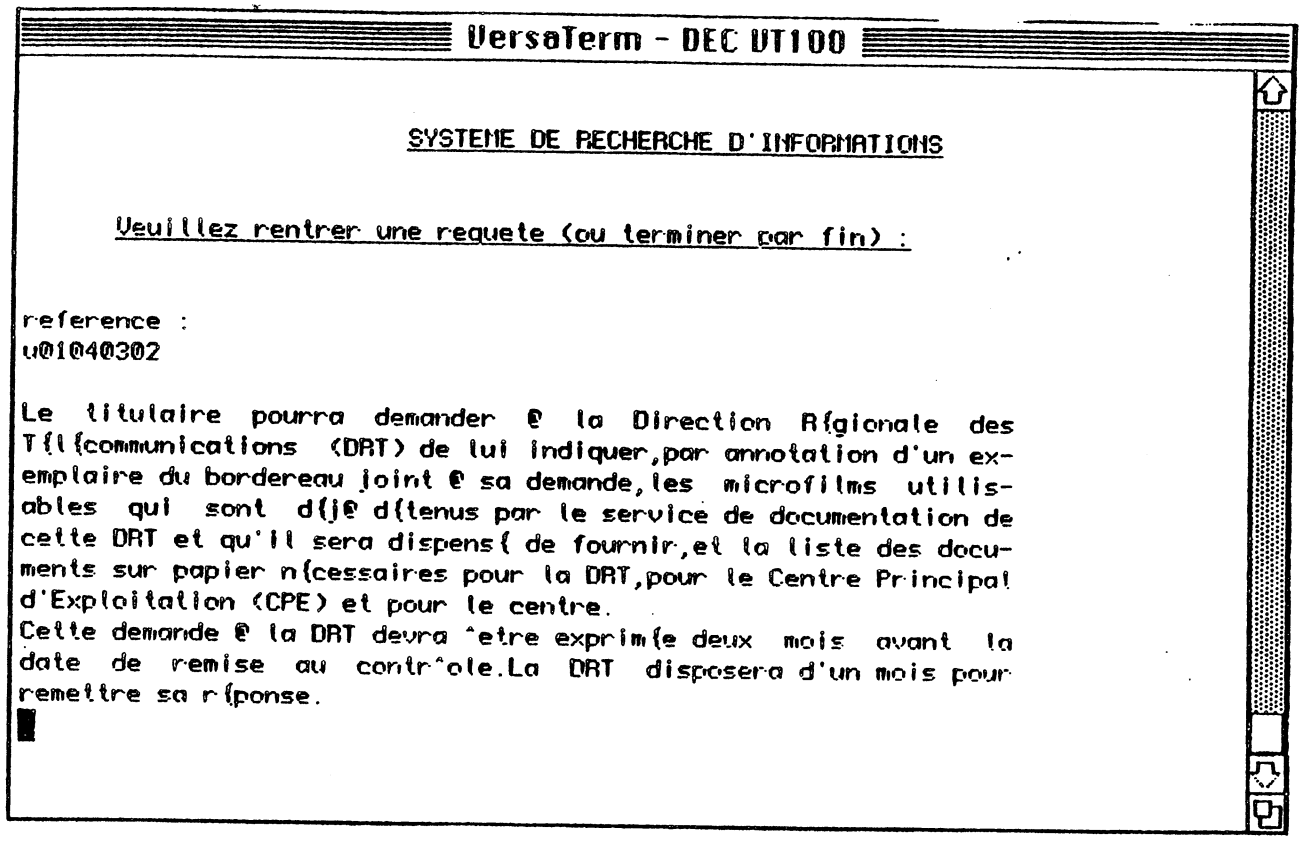
Le résultat obtenu est enfin évalué comme satisfaisant, et l'utilisateur peut alors choisir de visualiser les références résultat

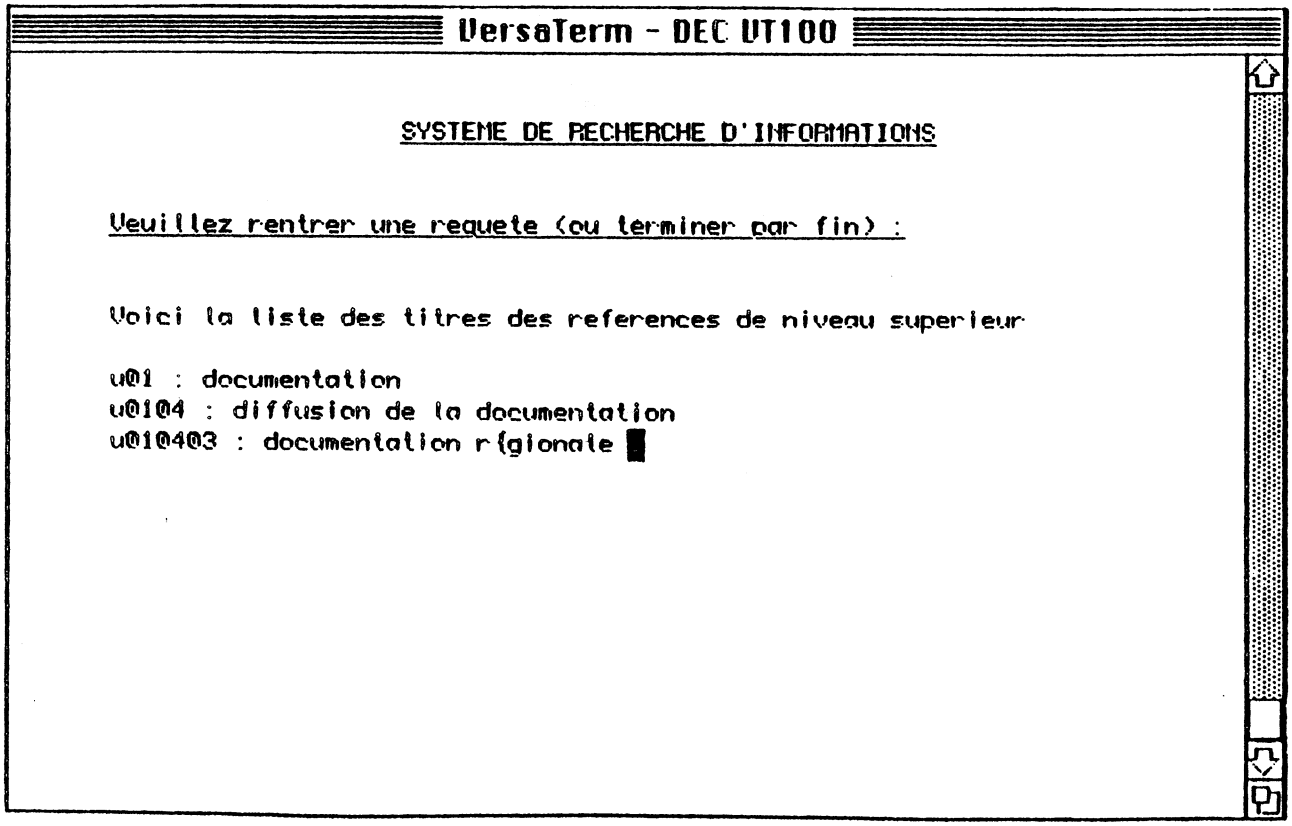


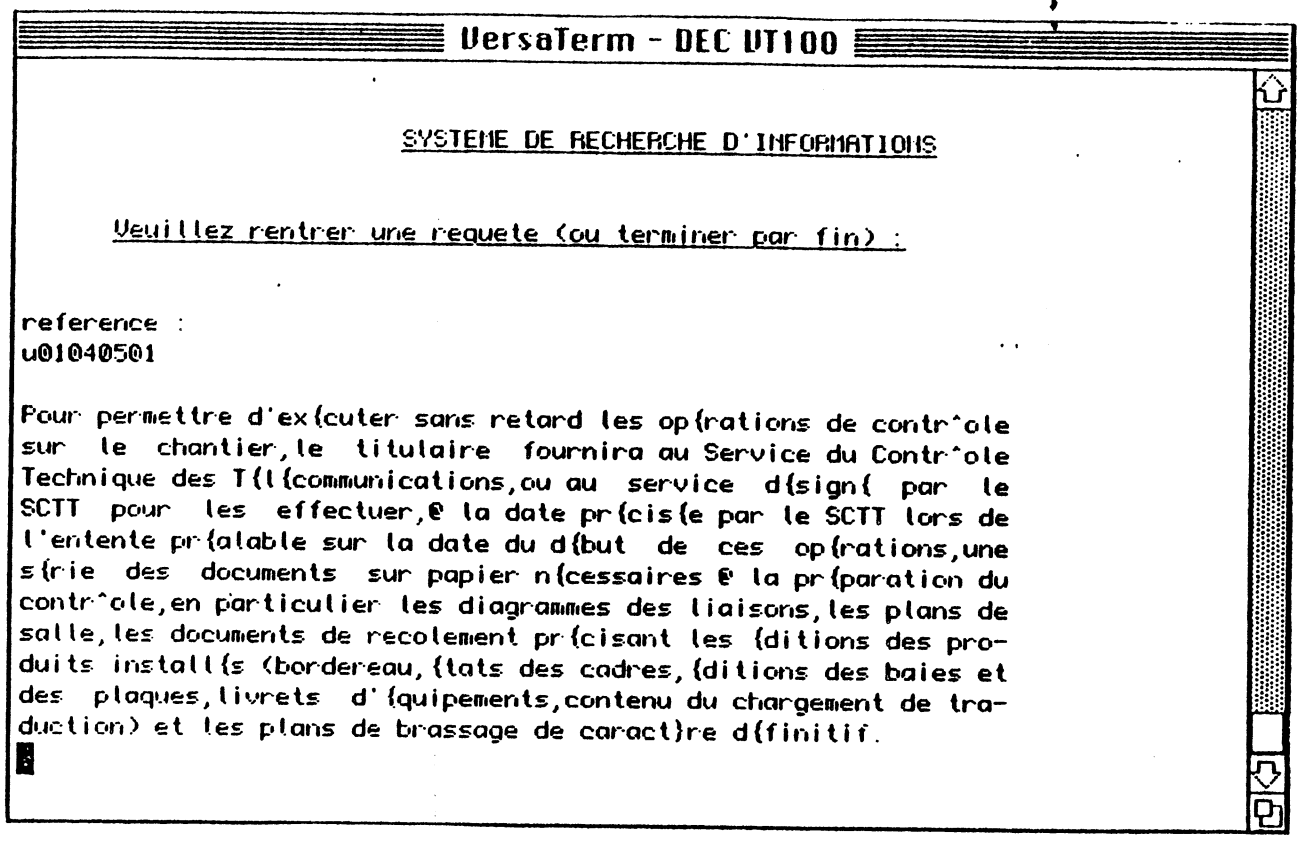
La référence est affichée avec son numéro, son titre éventuel et son texte

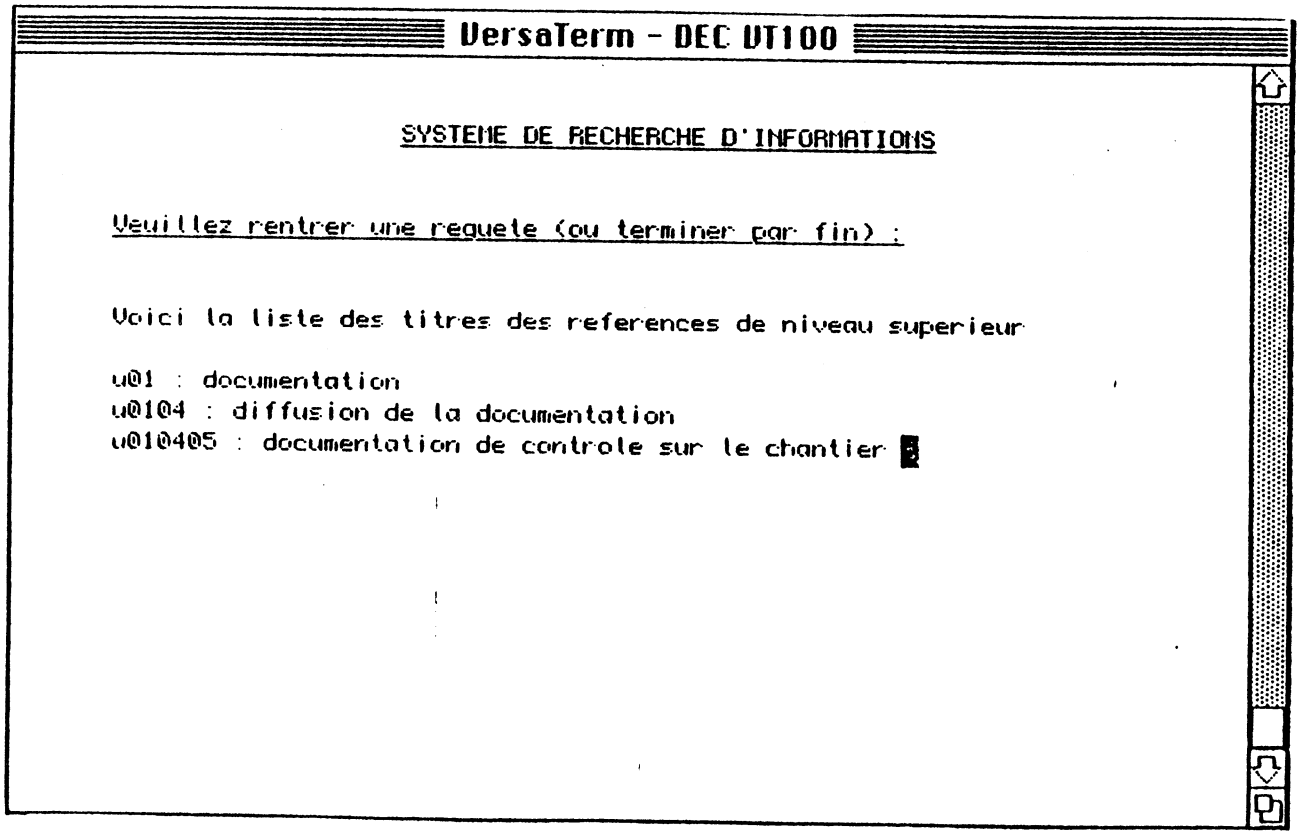


Les titres des entités de plus haut niveau dans la structure logique sont affichées pour permettre de situer l'unité résultat dans le document

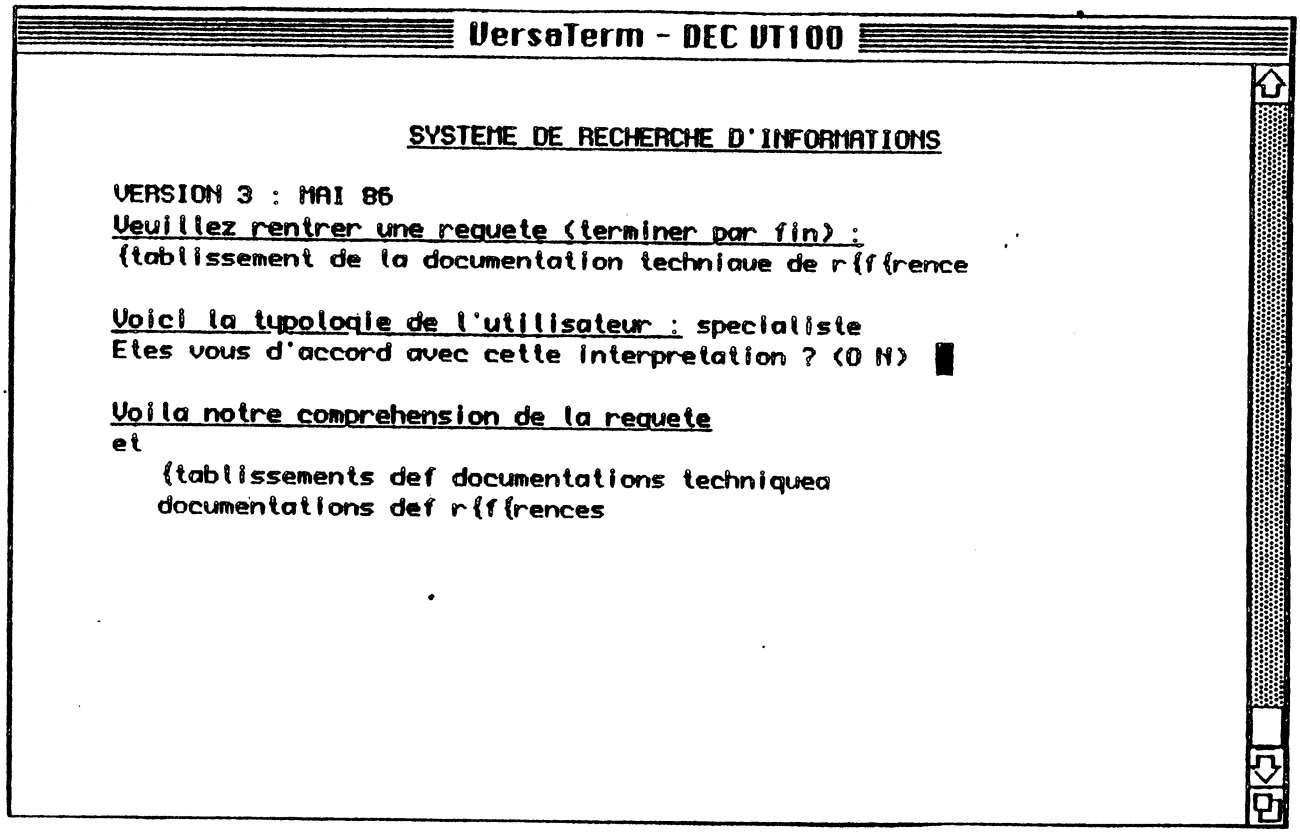

VersaTerm - DEC VT100
SYSTEME DE RECHERCHE D'INFORMATIONS
Veuillez rentrer une requete (ou terminer par fin) :
reference :
u@1040302
Le titulaire pourra demander @ la Direction R{gionale des T{l{communications (DRT) de lui indiquer,par annotation d'un exemplaire du bordereau joint @ sa demande,les microfilms utilisables qui sont d{j@ d{tenus par le service de documentation de cette DRT et qu'il sera dispens{ de fournir,et la liste des documents sur papier n{cessaires pour la DRT,pour le Centre Principal d'Exploitation (CPE) et pour le centre.
Cette demande @ la DRT devra ^etre exprim{e deux mois avant la date de remise au contr^ole.La DRT disposera d'un mois pour remettre sa r{ponse.


VersaTerm - DEC VT100
SYSTEME DE RECHERCHE D'INFORMATIONS
Veuillez rentrer une requete (ou terminer par fin) :
Voici la liste des titres des references de niveau superieur
u@1 : documentation
u@104 : diffusion de la documentation
u@10403 : documentation r{gionale

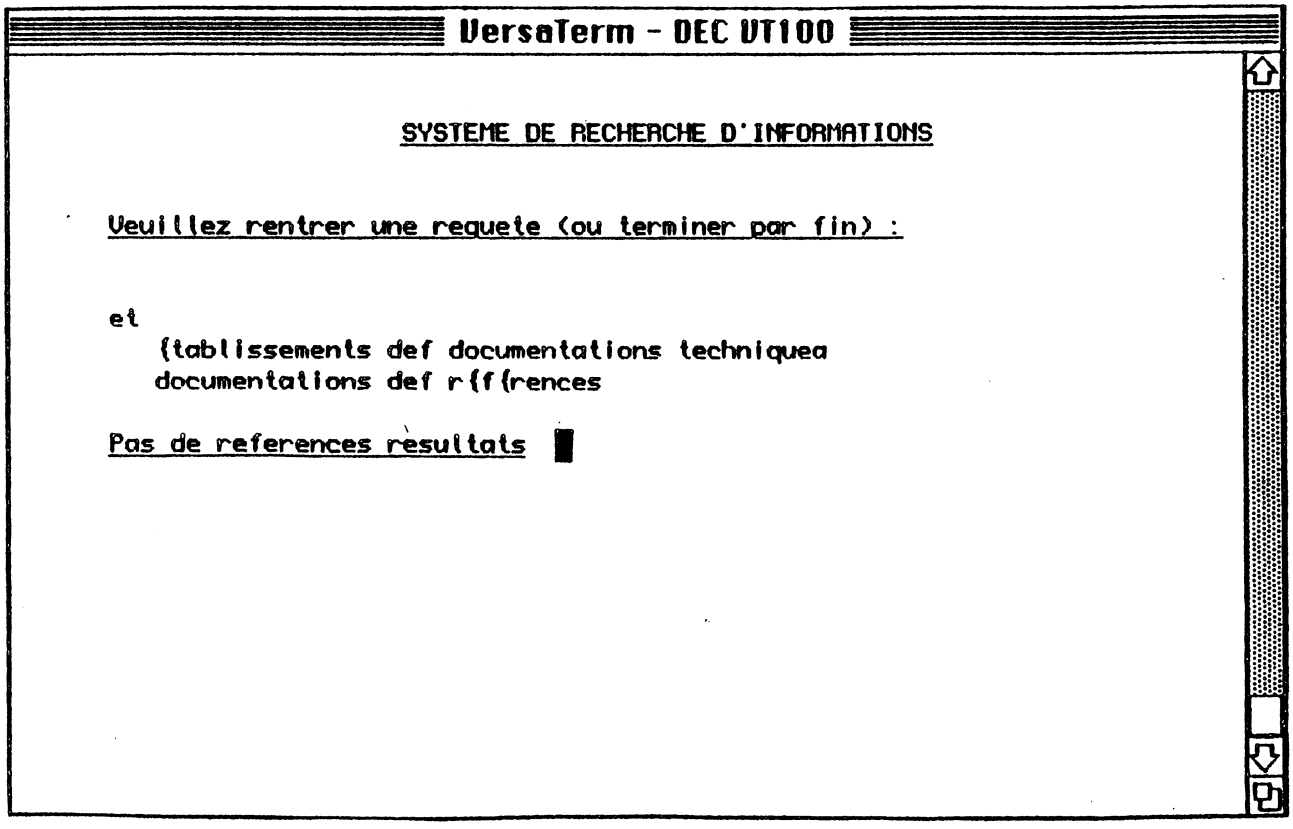




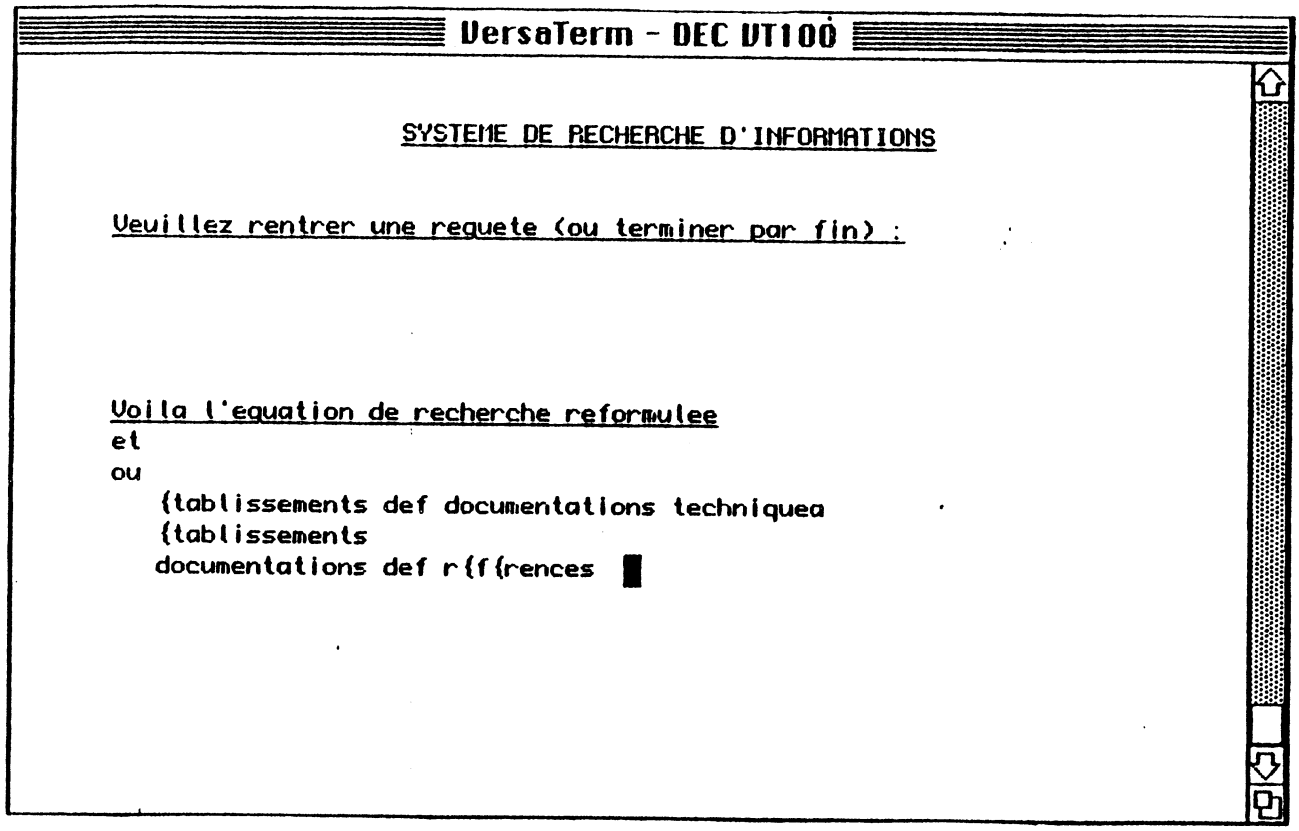
Le traitement de la requête est terminé



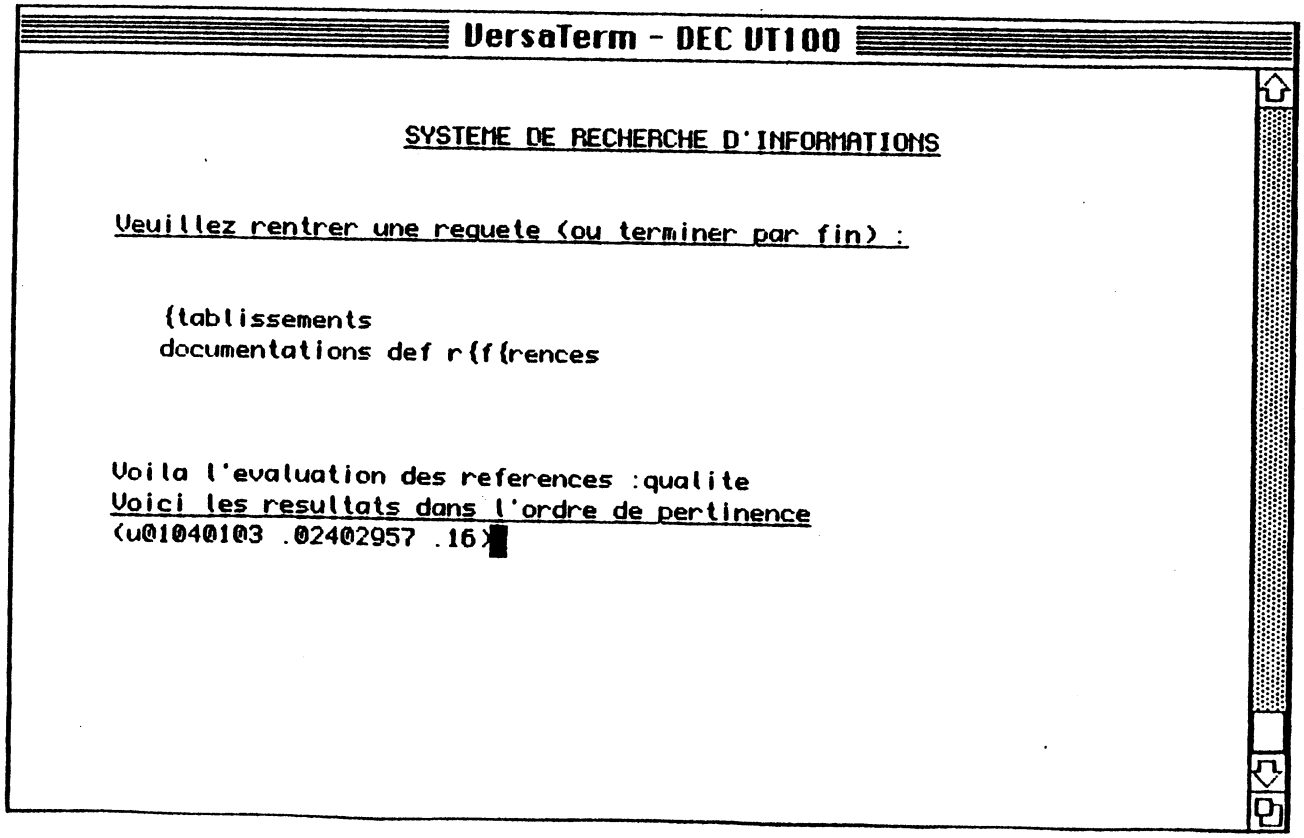
La requête initiale est saisie, le système affiche sa compréhension, et la typologie estimée. Le groupe initial trop long a été transformé par le processus de correspondance en une conjonction de groupes plus simples



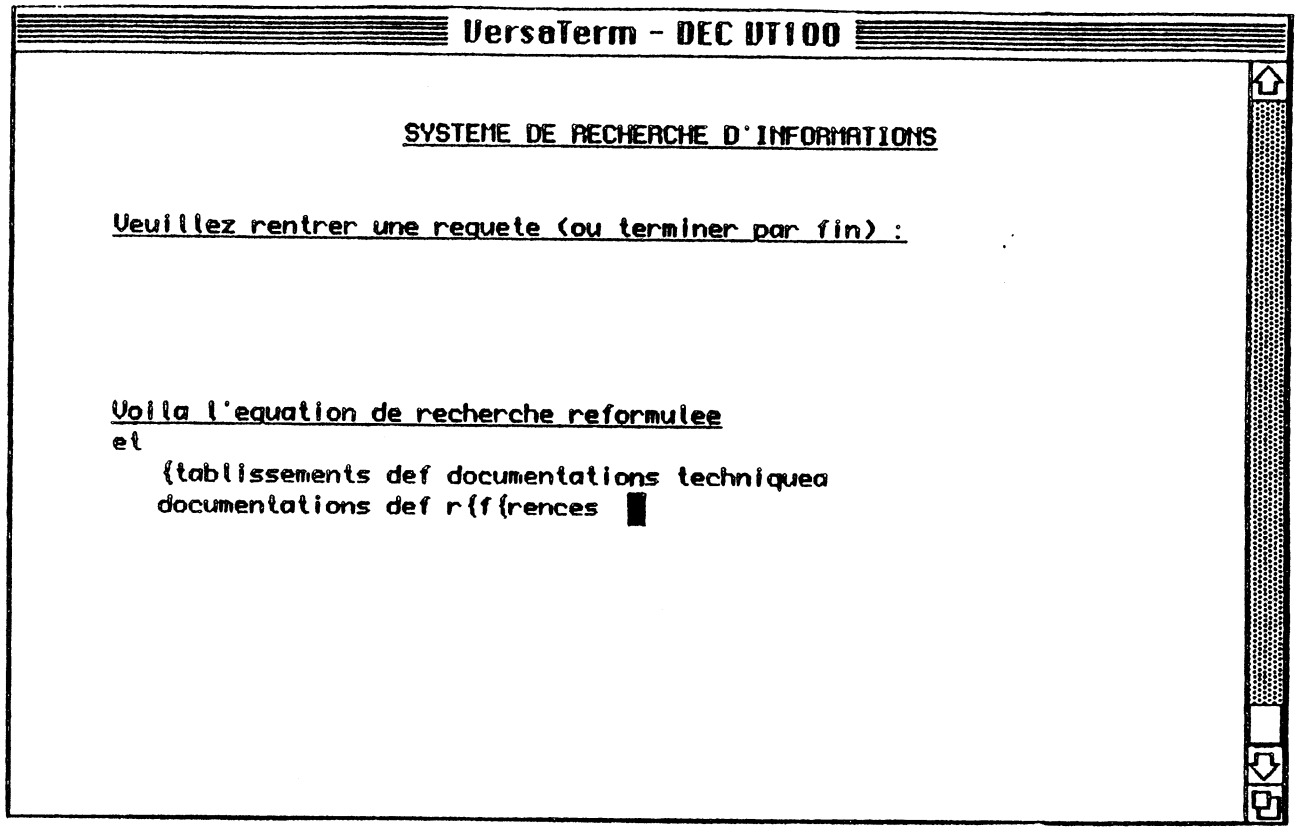
Il n'y a pas de résultats correspondant à l'équation de recherche



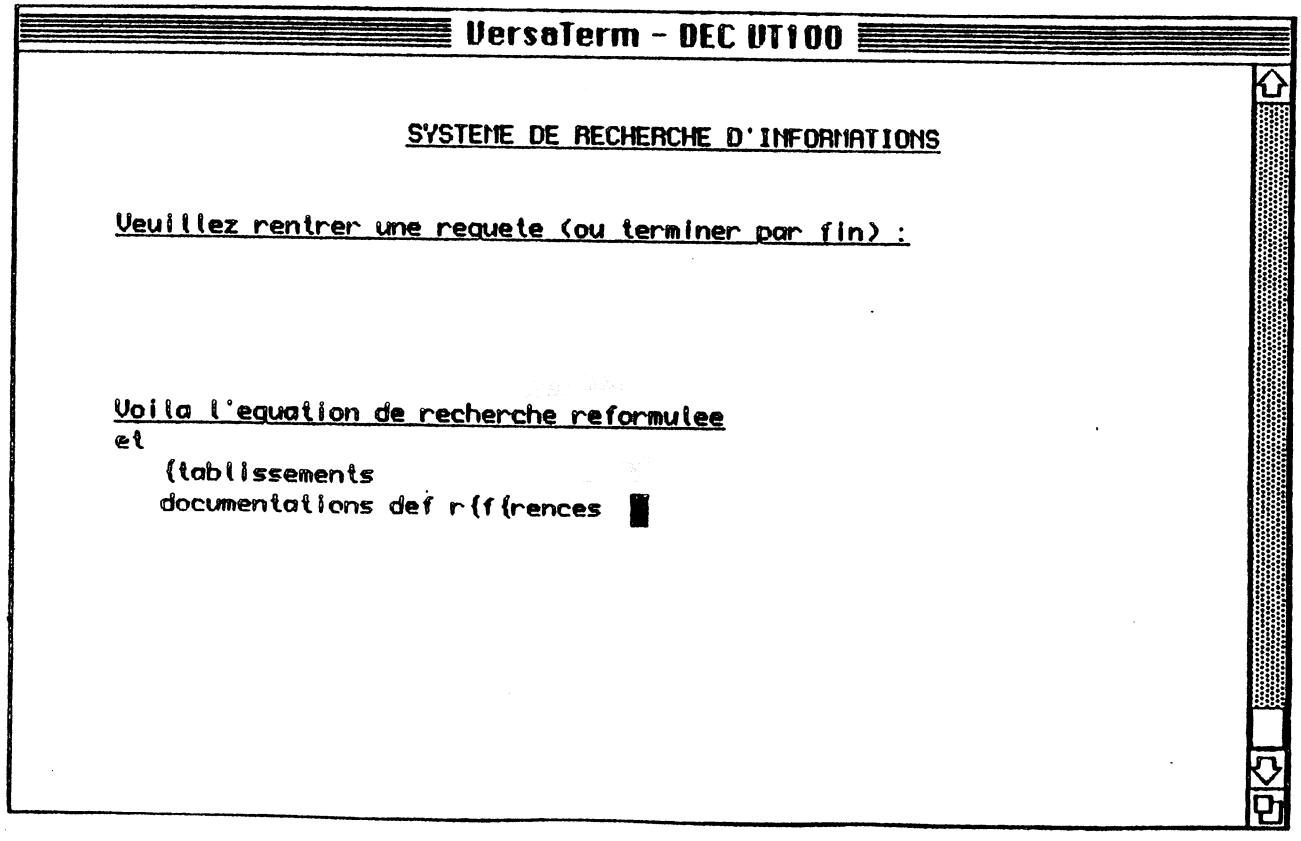
On reformule par les génériques



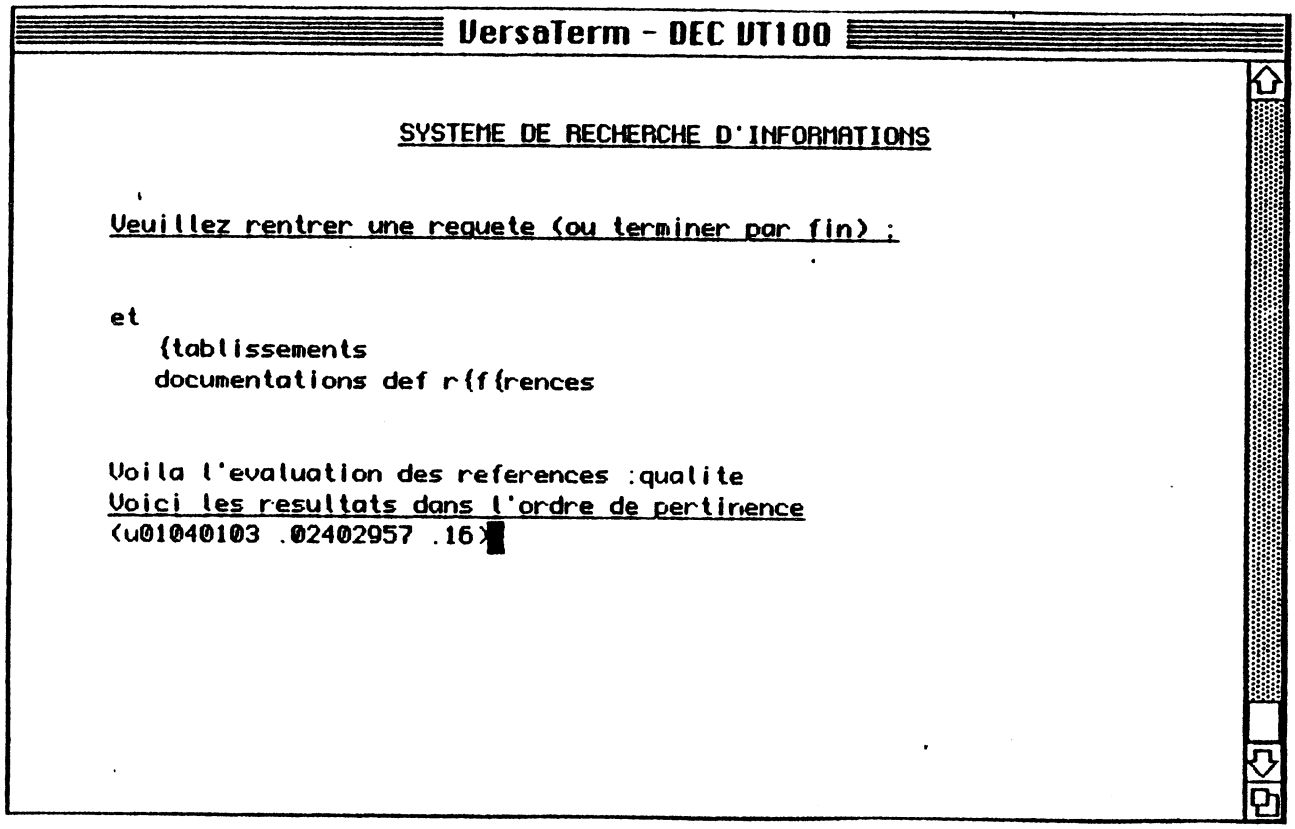
Un résultat est obtenu mais avec une qualité trop faible



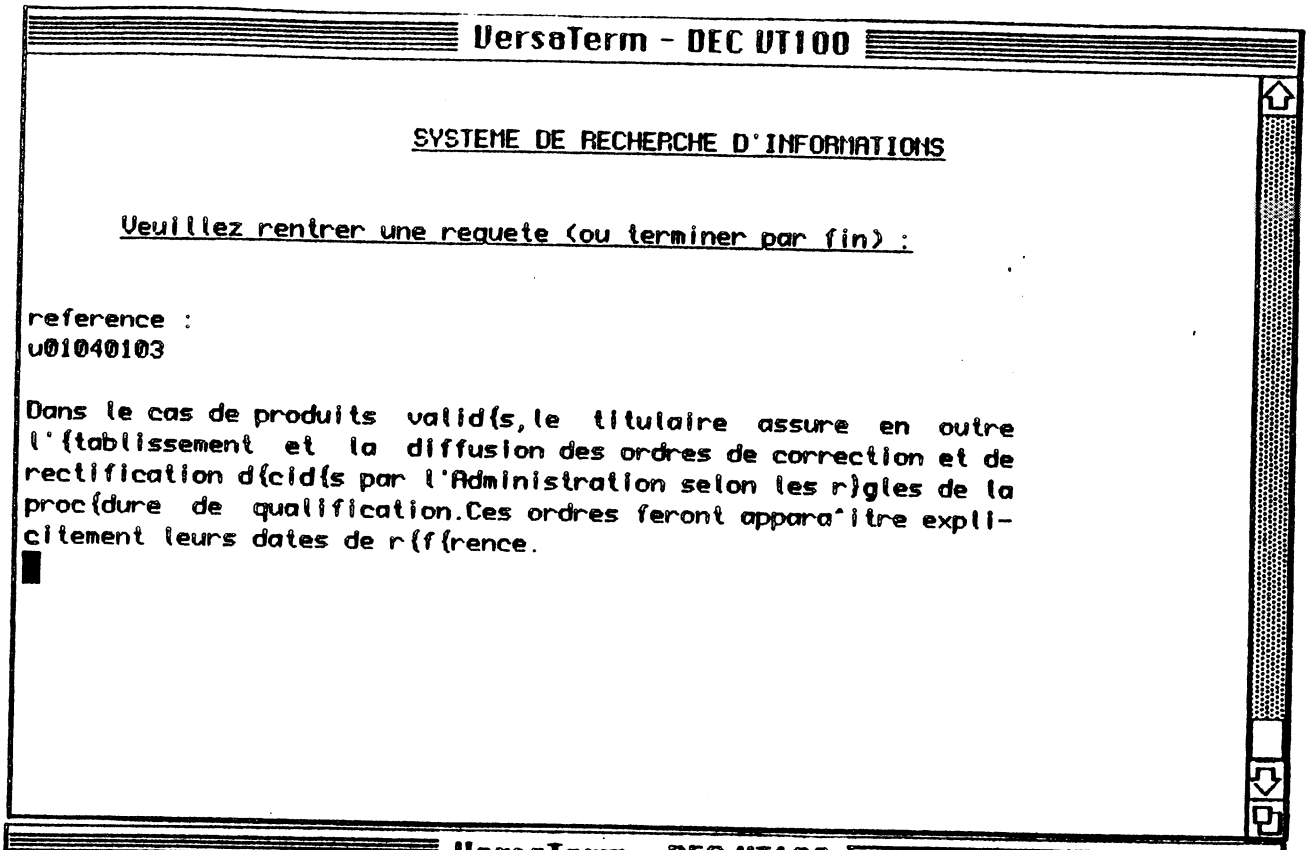
On reformule donc en supprimant l'opérande du OU fournissant les références de plus mauvaise qualité

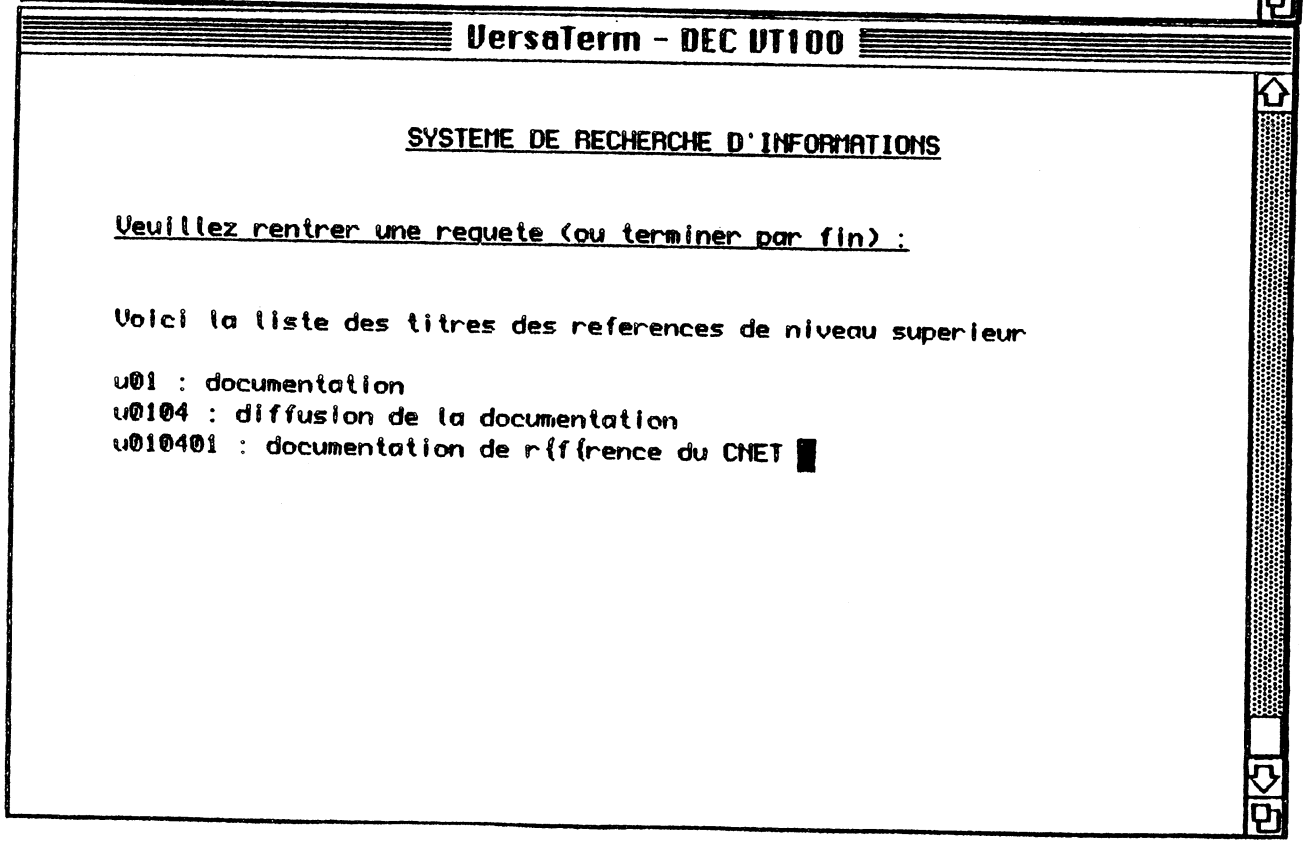


La reformulation est invalidée, on reformule en supprimant l'autre opérande du OU



Le résultat obtenu a toujours une qualité insuffisante, mais plus aucune reformulation n'est possible (cas d'échec, on donne la reformulation donnant le résultat de meilleure qualité)





On visualise la référence

ANNEXE 2

ETAT DE LA BASE DE CONNAISSANCES EXPERTES

Nous donnons l'état actuel des connaissances expertes utilisées pour le corpus étudié (un chapitre des NEF). Nous ne donnons que les règles concernant l'évaluation.

Evaluation locale d'une référence :

SI reptd(ref) > 0.9 ET repdt(ref) > 0.9
ALORS <référence bonne>

SI reptd(ref) < 0.1 ET repdt(ref) < 0.1
ALORS <référence de mauvaise qualité>

SI reptd(ref) ≥ bornesup(typologie) ET repdt(ref) ≥ bornesup(typologie)
ALORS <référence bonne>

SI reptd(ref) < borneinf(typologie) ET repdt(ref) ≠ 1
ALORS <référence de mauvaise qualité>

SI repdt(ref) < borneinf(typologie) ET reptd(ref) ≠ 1
ALORS <référence de mauvaise qualité>

SI reptd(ref) = 1
ALORS <référence bonne>

SI repdt(ref) = 1
ALORS <référence bonne>

SI reptd(ref) ≥ bornesup(typologie) ET repdt(ref) < bornesup(typologie)
ET repdt(ref) ≥ borneinf(typologie) ET reptd(ref) ≠ 1
ET <forte dégradation>
ALORS <référence bonne>

SI reptd(ref) ≥ bornesup(typologie) ET repdt(ref) < bornesup(typologie)
ET repdt(ref) ≥ borneinf(typologie) ET repdt(ref) ≠ 1
ET <faible dégradation>
ALORS <référence de mauvaise qualité>

SI repdt(ref) ≥ bornesup(typologie) ET reptd(ref) < bornesup(typologie)
ET reptd(ref) ≥ borneinf(typologie) ET repdt(ref) ≠ 1
ET <forte dégradation>
ALORS <référence bonne>

SI repdt(ref) ≥ bornesup(typologie) ET repdt(ref) < bornesup(typologie)
ET reptd(ref) ≥ borneinf(typologie) ET repdt(ref) ≠ 1
ET <faible dégradation>
ALORS <référence de mauvaise qualité>

où borneinf et bornesup sont deux fonctions, qui à partir de la typologie de l'utilisateur, renvoient les seuils de représentativités désirés, et reptd et repdt sont les représentativités terme - document, document - terme.

Evaluation globale des réponses :

SI nbrefbonnes < 1 ET nbref > 0
ALORS <résultat de mauvaise qualité>

SI nbrefbonnes ≥ 1 ET nbref ≤ 2 ET <utilisateur spécialiste>
ALORS <résultat bon>

SI nbrefbonnes ≥ 1 ET nbref > 5 ET <utilisateur spécialiste>
ALORS <trop de références>

SI nbrefbonnes ≥ 1 ET 2 < nbref < 5 ET <utilisateur spécialiste>
ET <faible dégradation>
ALORS <trop de références>

SI nbrefbonnes ≥ 1 ET 2 < nbref < 5 ET <utilisateur spécialiste>
ET <forte dégradation>
ALORS <résultat bon>

SI nbrefbonnes ≥ 1 ET 2 ≤ nbref < 5 ET <utilisateur moyen>
ALORS <résultat bon>

SI nbrefbonnes ≥ 1 ET nbref < 2 ET <faible dégradation> ET
<utilisateur moyen>
ALORS <peu de références>

SI nbrefbonnes ≥ 1 ET nbref < 2 ET <forte dégradation> ET
<utilisateur moyen>
ALORS <résultat bon>

SI nbrefbonnes ≥ 1 ET nbref > 5 ET <faible dégradation> ET
<utilisateur moyen>
ALORS <trop de références>

SI nbrefbonnes ≥ 1 ET nbref > 5 ET <forte dégradation> ET
<utilisateur moyen>
ALORS <résultat bon>

SI nbrefbonnes ≥ 1 ET nbref ≥ 5 ET <utilisateur débutant>
ALORS <résultat bon>

SI nbrefbonnes ≥ 1 ET nbref < 5 ET <faible dégradation> ET
<utilisateur débutant>
ALORS <peu de références>

SI nbrefbonnes ≥ 1 ET nbref < 5 ET <forte dégradation> ET
<utilisateur débutant>
ALORS <résultat bon>

où nbrefbonnes représente le nombre de références jugées bonnes par l'évaluation locale, et nbref le nombre de références retrouvées.

AUTORISATION de SOUTENANCE

VU les dispositions de l'article 15 Titre III de l'arrêté du 5 juillet 1984 relatif aux études doctorales

VU les rapports de présentation de Messieurs
. E. CHOURAQUI, Maître de recherche
. H. GALLAIRE, Directeur du ECRC

Monsieur Bruno DEFUDE

est autorisé à présenter une thèse en soutenance en vue de l'obtention du diplôme de DOCTEUR de L'INSTITUT NATIONAL POLYTECHNIQUE DE GRENOBLE, spécialité "Informatique".

Fait à Grenoble, le 20 mars 1986

Le Président de l'I.N.P.-G

D. BLOCH
Président
de l'Institut National Polytechnique
de Grenoble

P.O. le Vice-Président.