Modélisation et prototypage d'un système de recherche d'informations basé sur la proximité des occurences des termes de la requête dans les documents

Annabelle Mercier

N° d'ordre : 417 I

Ecole Nationale Supérieure des Mines
SAINT-ETIENNE

THÈSE

présentée par

Annabelle MERCIER

pour obtenir le grade de

Docteur de l'Ecole Nationale Supérieure
des Mines de Saint-Etienne

spécialité informatique

Modélisation et prototypage d'un système de recherche d'informations basé sur la proximité des occurrences des termes de la requête dans les documents

Soutenue à Saint-Etienne le 13 novembre 2006
en présence d'un jury composé de :

Gabriella PASI	Professeur à l'Université de Milan présidente et examinatrice
Catherine BERRUT	Professeur, Université Joseph Fourier, Grenoble rapportrice
Sylvie CALABRETTO	Maître de Conférences HDR, INSA de Lyon rapportrice
Christine LARGERON	Professeur, Université Jean Monnet, Saint-Etienne examinatrice
Jean-Jacques GIRARDOT	Maître de Recherche, Ecole des Mines directeur de thèse
Michel BEIGBEDER	Maître-Assistant, Ecole des Mines directeur des recherches

- **Spécialités doctorales :** **Responsable :**

SCIENCES ET GENIE DES MATERIAUX	**J. DRIVER** Directeur de recherche - Centre SMS
MECANIQUE ET INGENIERIE	**A. VAUTRIN** Professeur - Centre SMS
GENIE DES PROCEDES	**G. THOMAS** Professeur - Centre SPIN
SCIENCES DE LA TERRE	**B. GUY** Maitre de recherche
SCIENCES ET GENIE DE L'ENVIRONNEMENT	**J. BOURGOIS** Professeur - Centre SITE
MATHEMATIQUES APPLIQUEES	**E. TOUBOUL** Ingénieur
INFORMATIQUE	**O. BOISSIER** Professeur - Centre G2I
IMAGE, VISION, SIGNAL	**JC. PINOLI** Professeur - Centre CIS
GENIE INDUSTRIEL	**P. BURLAT** Professeur - Centre G2I
MICROELECTRONIQUE	**Ph. COLLOT** Professeur - Centre CMP

- Enseignants-chercheurs et chercheurs autorisés à diriger des thèses de doctorat (titulaires d'un doctorat d'Etat ou d'une HDR)

BENABEN	Patrick	PR 2	Sciences & Génie des Matériaux	SMS
BERNACHE-ASSOLANT	Didier	PR 1	Génie des Procédés	CIS
BIGOT	Jean-Pierre	MR	Génie des Procédés	SPIN
BILAL	Essaïd	MR	Sciences de la Terre	SPIN
BOISSIER	Olivier	PR 2	Informatique	G2I
BOUDAREL	Marie-Reine	MA	Sciences de l'inform. & com.	DF
BOURGOIS	Jacques	PR 1	Sciences & Génie de l'Environnement	SITE
BRODHAG	Christian	MR	Sciences & Génie de l'Environnement	SITE
BURLAT	Patrick	PR 2	Génie industriel	G2I
COLLOT	Philippe	PR 1	Microélectronique	CMP
COURNIL	Michel	PR 1	Génie des Procédés	SPIN
DAUZERE-PERES	Stéphane	PR 1	Génie industriel	CMP
DARRIEULAT	Michel	ICM	Sciences & Génie des Matériaux	SMS
DECHOMETS	Roland	PR 2	Sciences & Génie de l'Environnement	SITE
DELAFOSSE	David	PR 2	Sciences & Génie des Matériaux	SMS
DOLGUI	Alexandre	PR 1	Informatique	G2I
DRAPIER	Sylvain	PR 2	Mécanique & Ingénierie	CIS
DRIVER	Julian	DR	Sciences & Génie des Matériaux	SMS
FOREST	Bernard	PR 1	Sciences & Génie des Matériaux	SMS
FORMISYN	Pascal	PR 1	Sciences & Génie de l'Environnement	SITE
FORTUNIER	Roland	PR 1	Sciences & Génie des Matériaux	CMP
FRACZKIEWICZ	Anna	MR	Sciences & Génie des Matériaux	SMS
GARCIA	Daniel	CR	Génie des Procédés	SPIN
GIRARDOT	Jean-Jacques	MR	Informatique	G2I
GOEURIOT	Dominique	MR	Sciences & Génie des Matériaux	SMS
GOEURIOT	Patrice	MR	Sciences & Génie des Matériaux	SMS
GRAILLOT	Didier	DR	Sciences & Génie de l'Environnement	SITE
GROSSEAU	Philippe	MR	Génie des Procédés	SPIN
GRUY	Frédéric	MR	Génie des Procédés	SPIN
GUILHOT	Bernard	DR	Génie des Procédés	CIS
GUY	Bernard	MR	Sciences de la Terre	SPIN
GUYONNET	René	DR	Génie des Procédés	SPIN
HERRI	Jean-Michel	PR 2	Génie des Procédés	SPIN
JOYE	Marc	Ing. (Gemplus)	Microélectronique	CMP
KLÖCKER	Helmut	CR	Sciences & Génie des Matériaux	SMS
LAFOREST	Valérie	CR	Sciences & Génie de l'Environnement	SITE
LE COZE	Jean	PR 1	Sciences & Génie des Matériaux	SMS
LI	Jean-Michel	EC (CCI MP)	Microélectronique	CMP
LONDICHE	Henry	MR	Sciences & Génie de l'Environnement	SITE
MOLIMARD	Jérôme	MA	Sciences & Génie des Matériaux	SMS
MONTHEILLET	Frank	DR 1 CNRS	Sciences & Génie des Matériaux	SMS
PERIER-CAMBY	Laurent	MA1	Génie des Procédés	SPIN
PIJOLAT	Christophe	PR 1	Génie des Procédés	SPIN
PIJOLAT	Michèle	PR 1	Génie des Procédés	SPIN
PINOLI	Jean-Charles	PR 1	Image, Vision, Signal	CIS
SOUSTELLE	Michel	PR 1	Génie des Procédés	SPIN
STOLARZ	Jacques	CR	Sciences & Génie des Matériaux	SMS
THOMAS	Gérard	PR 1	Génie des Procédés	SPIN
TRAN MINH	Cahn	MR	Génie des Procédés	SPIN
VALDIVIESO	Françoise	CR	Génie des Procédés	SPIN
VAUTRIN	Alain	PR 1	Mécanique & Ingénierie	SMS
VIRICELLE	Jean-Paul	CR	Génie des procédés	SPIN
WOLSKI	Krzysztof	CR	Sciences & Génie des Matériaux	SMS
XIE	Xiaolan	PR 1	Génie industriel	CIS

Glossaire :

PR 1	Professeur 1ère catégorie
PR 2	Professeur 2ème catégorie
MA(MDC)	Maître assistant
DR 1	Directeur de recherche
Ing.	Ingénieur
MR(DR2)	Maître de recherche
CR	Chargé de recherche
EC	Enseignant-chercheur
ICM	Ingénieur en chef des mines

Centres :

SMS	Sciences des Matériaux et des Structures
SPIN	Sciences des Processus Industriels et Naturels
SITE	Sciences Information et Technologies pour l'Environnement
G2I	Génie Industriel et Informatique
CMP	Centre de Microélectronique de Provence
CIS	Centre Ingénierie et Santé

N° d'ordre : 417 I

Ecole Nationale Supérieure des Mines
SAINT-ETIENNE

THÈSE

présentée par

Annabelle MERCIER

pour obtenir le grade de

Docteur de l'Ecole Nationale Supérieure
des Mines de Saint-Etienne

spécialité informatique

Modélisation et prototypage d'un système de recherche d'informations basé sur la proximité des occurrences des termes de la requête dans les documents

Soutenue à Saint-Etienne le 13 novembre 2006
en présence d'un jury composé de :

Gabriella PASI	Professeur à l'Université de Milan présidente et examinatrice
Catherine BERRUT	Professeur, Université Joseph Fourier, Grenoble rapportrice
Sylvie CALABRETTO	Maître de Conférences HDR, INSA de Lyon rapportrice
Christine LARGERON	Professeur, Université Jean Monnet, Saint-Etienne examinatrice
Jean-Jacques GIRARDOT	Maître de Recherche, Ecole des Mines directeur de thèse
Michel BEIGBEDER	Maître-Assistant, Ecole des Mines directeur des recherches

à André Coullet,
mon premier professeur d'informatique du « TO7 » au pentium I,
mon professeur de Sciences Physiques au lycée Calmette,
mais surtout au père et grand-père de mes petits cousins,
en souvenir des bons moments passés avec Thérèse.

Remerciements

Je remercie très chaleureusement tous les membres de mon jury de thèse ainsi que toutes les personnes qui m'ont soutenu pendant ces longues études.

A travers son directeur, Robert Germinet, je remercie l'institution qu'est l'Ecole Nationale Supérieure des Mines de Saint Etienne, qui m'a permis de présenter mes travaux à diverses reprises dans des conférences nationales et internationnales et qui m'a donné les moyens de poursuivre mes recherches notamment pour la participation aux forums d'évaluation internationnaux, incontournables dans le domaine de la recherche d'informations.

Je remercie très vivement Gabriella PASI de m'avoir fait l'honneur d'être la présidente de mon jury de thèse et de s'être déplacée jusqu'à Saint Etienne. J'espère que ses remarques judicieuses me permettront très prochainement d'améliorer mon modèle pour de nouvelles expérimentations.

Je remercie aussi très vivement Catherine Berrut et Sylvie Calabretto d'avoir accepté d'être rapportices de mon travail et d'avoir consacré du temps à la lecture de mon mémoire. Ce n'est pas sans émotion que je me souviens de mes premiers pas en recherche d'informations que ce soit aux réunions de l'ISDN, à la conférence INFORSID 2003 ou à ESSIR 2003.

Je remercie également mon directeur de thèse, J.J. Girardot, et mon directeur de recherches, Michel Beigbeder pour l'implication qu'ils ont mis dans la direction de ces travaux pendant ces trois années. Je remercie tout particulièrement Michel de m'avoir encouragé à continuer dans la voie de la proximité.

Je remercie particulièrement Christine Largeron pour sa participation à mon jury de thèse mais aussi pour le soutien qu'elle m'a apporté au cours de cette dernière année. La liberté qu'elle m'a laissé dans mes recherches m'a permis d'améliorer et d'expérimenter mon modèle sur la collection TERABYTE en 2006 tout en me consacrant à la rédaction de ce mémoire.

Je remercie également Alexandre Dolgui, directeur du centre G2I, pour nous (les doctorants) avoir intégré dans l'organisation du Symposium INCOM'06. Ce fut une expérience très enrichissante tant sur le point scientifique que relationnel.

Je remercie également tout le personnel tant au niveau de la direction de la recherche qu'au niveau du centre G2I pour l'assistance quotidienne et minutieuse, surtout un gros merci à Marie Line pour son dynamisme et sa bonne humeur. Et pour tout ce qui est de l'informatique, un gros merci à Jean-François et Nilou. Merci aussi à ceux qui ont participé avec moi à la vie au Labo pendant ces années de thèse. Je pense notamment à tous les doctorants de l'équipe RIM Faiza, Camille, Amélie, Fabien, Xavier, Thanh-Trung, Hoan, Thierry mais aussi autres en dehors de l'équipe Sana, Julie, Olga, Natacha, Medhi et tous ceux que j'oublie.

Enfin, je remercie tous ceux qui m’ont permis de réaliser ces longues études. Je pense d’abord aux professeurs qui m’ont donné l’envie d’apprendre au collège, à ceux qui m’ont appris à réfléchir au Lycée et enfin à ceux qui m’ont initié à l’informatique pendant les cours d’algoritmique et de programmation à l’Université de Nice. Je remercie également tous les membres de l’équipe Objet et Composants Logiciels pour leurs conseils et leur aide précieuse, plus particulièrement, Philippe Collet qui m’a ouvert au monde de la recherche pendant mon stage de DEA.

Pour terminer, je remercie tout mon proche entourage de m’avoir soutenu et encouragé malgré la distance. Je pense à mes amis, à ma famille et particulièrement mes parents, ma grand-mère et Laurent pour le soutien qu’ils continuent de m’apporter.

Table des figures

Liste des tableaux

Table des matières

Chapitre 1

Introduction

Table des matières

Nous sommes aujourd'hui arrivés à l'âge d'or de la société de l'information et de la communication. S'informer en temps réel sur l'« Internet », tant pour ses besoins personnels que professionnels, est devenu pour la plupart des individus aussi naturel que de prendre le téléphone pour communiquer ou d'allumer la télévision pour s'informer et se divertir. Pour les générations futures, cela deviendra certainement un réflexe si la censure n'atteint pas trop ce médium. Pour tout un chacun, la voie de l'Internet s'est démocratisée avec le courrier électronique et les pages sur la Toile. D'ailleurs, les plus passionnés apportent leur contribution à l'édifice en y ajoutant leur page personnelle ou leur participation aux nombreux forums de discussion. La manière et le droit d'accès à l'information ont considérablement évolué depuis le siècle dernier ; l'anecdote suivante illustre bien cette évolution. Au moment de la grande crise financière de 1929 en Amérique, un notable pouvait être informé en 3 jours (de bateau) des événements et prendre les décisions nécessaires pour éviter de se trouver ruiné tandis que les informations n'arrivaient pas aussi rapidement aux oreilles de personnes moins influentes qui se trouvèrent dans la tourmente... Par contre, si de nos jours, une crise boursière se reproduisait, tout le monde serait au même niveau : par Internet le petit porteur accéderait à l'information de la même manière que le journaliste économique d'un magazine des finances de renom. Aujourd'hui, nul besoin d'être privilégié pour accéder à l'information : elle est partout sous forme électronique ou non (journaux, radio, télévision), elle est disponible instantanément, librement ou par abonnement. Pour

rechercher ou accéder à une information, l'utilisation des moteurs de recherche ou des annuaires sur la Toile est quasiment devenue un réflexe.

Le succès des nouvelles technologies avec l'utilisation quotidienne du courrier électronique et la mise en place de nouveaux sites Web génère une augmentation considérable de la quantité d'informations numériques. Par exemple, si les échanges de courrier par Internet étaient réunis dans une base documentaire pour y être retrouvés, plusieurs tera-octets de données devraient être traités. La croissance très rapide du nombre d'utilisateurs d'une part (bien que 6 milliards, le nombre d'habitants sur la terre soit la limite indéniable) et de la quantité d'informations d'autre part, conduisent à l'exploration de nouvelles techniques pour rechercher et traiter les différents visages de l'information, transcrits sous divers formats de fichiers informatiques. Si l'on remarquait précédemment que le volume généré par les fichiers textuels de courrier est considérable, que pourrait-on dire du volume généré par la vidéo si tous les programmes télévisés, les films du cinéma ou familiaux constituaient une base documentaire multimédia ? Récemment, l'Institut National de l'Audiovisuel a mis en place un accès aux programmes anciennement diffusés sur les antennes de télévision française, cette nouvelle application est introduite de la manière suivante sur la page d'accueil[1] :

> Depuis sa création en 1974, l'INA conserve et exploite les programmes produits par les chaînes publiques hertziennes, soit 60 ans de radio et 50 ans de télévision. Grâce à un ambitieux plan de sauvegarde numérique de ses archives engagé en 2001, l'INA est aujourd'hui en mesure de mettre en ligne à l'usage de tous près de 10 000 heures de ce patrimoine audiovisuel français. Découvrez ou redécouvrez ainsi près de 100 000 émissions. Feuilletons, séries, ... l'effervescence de plus d'un demi-siècle de mémoire, de création et d'émotion collectives.

Cette application montre que l'accès à l'information se révèle être de plus en plus facile, si bien qu'il faut rester vigilant quant à la possibilité de son utilisation. Par exemple, à raison de 8h de visionnage par jour, trois années seraient nécessaires pour regarder les 10000 heures d'archives proposées par l'INA, et combien faudrait-il de temps pour regarder toutes les pages Internet, lire tous les échanges de messages électroniques, etc. ? Des alternatives à l'exhaustivité doivent donc être recherchées. Face à ce surcroît d'informations souvent hétérogènes, face aux nouveaux usages de l'informatique et face à l'exigence des utilisateurs, il devient urgent de développer des méthodes et des techniques adaptées et performantes, tant au niveau de la rapidité que de la fiabilité.

Le défi consiste donc à définir de nouvelles idées voire à explorer de vieilles pistes sous des angles différents pour améliorer l'accès à l'information. Cependant, avec l'essor de la Toile et des moteurs de recherche, le domaine de l'informatique s'étend de nouveau sur une problématique bien connue depuis les années 50 c'est-à-dire la recherche d'informations. La section suivante fait une brève rétrospective des évènements clés reportés par J. Y. Nie dans l'introduction sur le domaine dans le livre *Assistance intelligente à la recherche d'informations* [Gaussier et Stéfanini, 2003].

[1] Introduction au téléchargement sur le site de l'INA : http ://www.ina.fr

1.1 Un peu d'histoire...

Le domaine de la recherche d'informations prend naissance après la seconde guerre mondiale à cause de l'explosion de la quantité d'informations due au « progrès » de la vie moderne : les enfants des classes moyennes peuvent aller à l'école, les journaux et les magazines sont rapidement transportés dans les provinces et exportés dans les autres pays et les télécommunications sont développées dans le monde entier. C'est à ce moment que dans un contexte universitaire, l'expression « information retrieval » (traduite en français par recherche d'informations) est introduite pour la première fois par Calvin N. Mooers dans son rapport de fin d'étude.

Les pionniers comme par exemple Cyril Cleverdon ou Brian Campbell posent les principes fondamentaux et tournent leurs préoccupations vers l'indexation pour la recherche des documents par des systèmes automatisés. La notion d'index n'est pas nouvelle, elle se retrouve dès la parution de la première encyclopédie de Diderot et D'Alembert [Diderot et D'Alembert, 1751] où deux volumes sont consacrés à l'index. Ce dernier permet de renvoyer à la page d'un article dans un des dix-sept volumes, ce qui est très utile car la recherche exhaustive n'est pas concevable avec une telle quantité d'information[2]. Aujourd'hui, la même information peut par exemple être sauvegardée dans une carte mémoire de quelques centimètres carrés pour être lue sur un PDA[3][4]. Étant la clé de voûte de la recherche documentaire, la notion d'index a été rapidement transposée pour être utile dans un environnement informatisé.

Lors de la première conférence en recherche d'informations (*International Conference on Scientific Information*) qui voit le jour en 1958 à Washington, Peter Luhn présente son premier système de recherche d'informations dans lequel il a mis en place quelques idées pour l'indexation et la sélection des documents. Le système KWIC fonctionne sur la sélection des documents en fonction de la fréquence des termes tout en éliminant les mots fonctionnels de la langue. Ces premières recherches trouveront leur application pour l'accès aux fonds documentaires des bibliothèques.

Pour continuer dans cette voie, plusieurs projets expérimentaux apparaissent et aboutissent à la création de nouveaux systèmes, dont certains sont encore utilisés aujourd'hui. Le projet CRANFIELD dirigé par Cleverdon [Cleverdon, 1967] est l'un des premiers projets pour lequel une collection de test réaliste est construite, constituée de 18000 articles et de 1200 requêtes. Les premiers résultats en termes de rappel/précision permettent d'évaluer la performance du système[5]. Au même moment, une collection de documents médicaux est mise en place par Lancaster dans le projet MEDLARS. Cette expérience met en échec le principe d'indexation manuelle avec un vocabulaire contrôlé par rapport à une approche automatique. A partir des années 60 la construc-

[2] Dans les premières éditions, un volume fait au moins 1 kilo, prend 50 cm × 30 cm dans une bibliothèque, l'ensemble des volumes réunit au total 18 000 pages.

[3] http://www.inrp.fr/vst/Dossiers/Savoir_encyclopedique/structuration/informatisation.htm

[4] http://portail.atilf.fr/encyclopedie/

[5] La définition de la mesure d'évaluation rappel précision est donnée en section 2.4.

tion du système de recherche d'informations SMART [Salton, 1971b, Salton et McGill., 1983] par Gérard Salton introduit l'utilisation du modèle vectoriel (cf. section 2.3.5). De nombreuses études sont réalisées pour améliorer l'architecture des systèmes et pour comparer les atouts de l'indexation automatique par rapport à l'indexation manuelle. Le système a été réécrit par Fox puis par Buckley et reste encore utilisé aujourd'hui. Les avancées du domaine y ont été intégrées au fur et à mesure, comme la rétroaction de pertinence ou les différentes possibilités de pondération des termes d'index. Un autre projet important, le projet STAIRS [Blair et Maron, 1985], permet de constituer une collection de grande taille pour l'époque, constituée de 40000 documents issus du domaine du droit. Les expériences issues de cette collection mettent en évidence que le nombre de documents retournés (rappel) par les systèmes n'est pas encore suffisant dans un domaine d'application comme le droit. D'ailleurs, en 2006, la recherche d'informations dans ce genre de documents a été remise au goût du jour dans la toute nouvelle tâche *« legal »* de la conférence TREC [6].

Parallèlement, l'évolution du réseau Internet et l'utilisation des protocoles pour le courrier électronique et les pages Internet fournissent une mine de documents pour construire des collections de données volumineuses et hétérogènes indispensables pour comparer les systèmes de recherche d'informations.

En 1992, la première édition de la conférence TREC [Harman, 1992] préfigure une importante expansion du domaine de la recherche d'informations. Il s'agit d'une campagne d'évaluation où les chercheurs du monde entier évaluent leur système sur des collections de test et peuvent ainsi comparer leurs résultats avec d'autres systèmes. Plusieurs collections sont développées pour et par les équipes en fonction de l'évolution du domaine. A chaque collection sont associées une ou plusieurs tâches de recherche d'informations. Progressivement, la tâche de recherche d'informations « adhoc » laisse la place à de nouvelles applications telles que les systèmes de question-réponse, la recherche multilingue ou bien la recherche dans les documents multimédia ou structurés. Il est stratégique pour les équipes de recherche de trouver des solutions pour anticiper les problèmes des systèmes de demain car de nos jours, la recherche d'informations tend à se populariser avec la croissance du nombre d'internautes.

1.2 Rechercher une information

Une récente étude de l'INSEE (numéro 1076, mai 2006) montre que 91% des internautes français ont déjà utilisé un moteur de recherche. Sur l'espace du Web, le moteur de recherche est souvent le point d'accès à l'information, et l'incitation à la recherche d'informations est forte. Outre les sites dédiés comme les annuaires et les moteurs de recherche, de nombreux portails d'entreprises ont ajouté la fonctionnalité de recherche à leur vitrine Internet. Par exemple, le site de La Poste vous incite à cliquer sur son bandeau publicitaire : « Rechercher une information, la poste.net trouve tout pour vous ? » Cliquez-dessus et laposte.net vous présente son interface de

[6] http://trec.nist.gov

recherche... Mais derrière chacune de ces interfaces se cache un nombre minimum d'éléments communs :

- **la base de documents** est constituée de documents provenant du site ou de l'entreprise, ou de documents récoltés sur la Toile ;
- **l'index** est interrogé par le moteur. Il fournit un accès rapide aux informations sur les documents, ces dernières sont utilisées pour les sélectionner et les classer par rapport à la requête soumise ;
- **le module de traitement des requêtes** transforme une requête exprimée par l'utilisateur en requête utilisable par la fonction de correspondance ;
- **la fonction de correspondance** calcule la similarité entre la requête et les documents de l'index et permet ainsi de retourner un ensemble de réponses à l'utilisateur.

En résumé, les systèmes de recherche d'informations intègrent dans leur index les informations sur les documents, comme la fréquence des mots et parfois même leurs positions, et à partir de ces données la fonction de correspondance sélectionne et établit un classement des documents par rapport à la requête. D'autres systèmes plus complexes prennent aussi en compte la structure de la Toile en utilisant les liens hypertextes ou la structure du document lui-même s'il en possède une.

En général, le processus de recherche d'informations est effectué en plusieurs étapes, par un utilisateur, car la première interrogation n'est pas toujours suffisante pour retrouver les documents correspondants au besoin d'informations. Contrairement à un instituteur face à la question d'un élève, le moteur de recherche n'est qu'un programme informatique et ne s'adapte pas à l'internaute devant son écran : si ce dernier n'est pas satisfait, il doit tout simplement reformuler sa question. Nous discuterons de ces points clés (processus, modèles et évaluation) dans le chapitre suivant.

1.3 Problématique de la thèse

1.3.1 Les systèmes de recherche d'informations classiques

Les modèles sur lesquels nous nous basons pour introduire notre problématique sont exposés de manière plus formelle et détaillée dans le chapitre suivant (cf. sections 2.3.2 et 2.3.5). Les systèmes de recherche d'informations contemporains reposent sur des modèles théoriques fondamentaux. Chacun d'eux définit, d'une part, la manière de représenter un document et une requête, et d'autre part, celle d'apparier les documents et les requêtes. Le premier modèle classique repose sur la théorie des ensembles et l'algèbre de Boole. Dans ce modèle booléen, l'usager formule son besoin d'information à l'aide d'une expression booléenne et les documents sont représentés par l'ensemble des termes qui le composent. Un système de recherche d'informations, implémentant ce modèle, effectue la sélection des documents selon qu'ils ont ou non la propriété formulée dans la requête. L'inconvénient de ce modèle est que son critère de décision de pertinence est binaire, un document est pertinent (1) ou ne l'est pas (0), le score attribué au document

est pris dans l'ensemble $\{0, 1\}$. Par conséquent, il n'y a pas de nuances dans la pertinence calculée par le système ce qui implique que les documents retournés ne peuvent pas être classés. L'interprétation d'une requête peut conduire à deux cas de figure en ce qui concerne l'ensemble de réponses retournées :

- **un ensemble de réponses trop petit**
  Utiliser des conjonctions[7] de termes permet d'augmenter la précision des résultats. Par exemple, pour une recherche sur l'organisation ETA en France[8] la requête suivante « `(eta OU (organisation ET terroriste ET basque))ET france` » peut renvoyer un ensemble de réponses réduit. Il se peut que des documents pertinents aient été écartés en ajoutant les mots « organisation » ou « france » car ces deux mots ne sont pas forcément contenus dans tous les documents pertinents. Une solution serait de retirer ces deux mots de la requête, mais dans ce cas, l'ensemble des réponses peut devenir un ensemble de réponse trop large ;
- **un ensemble de réponses trop large**
  Pour le même besoin d'informations, la requête « `(eta OU (terroriste ET basque))` », moins contraignante, peut être suggérée. Dans ce cas, l'effet souhaité (rappeler plus de documents) permet d'obtenir des réponses plus nombreuses mais celles-ci peuvent ne pas être pertinentes. En effet, un document satisfaisant la requête dont le contexte se situe en France est pertinent tandis qu'un document parlant de l'ETA en Espagne ne l'est pas.

Pour résumer, si un document répond partiellement à une requête, il n'est pas retourné et la solution qui tend à réduire le nombre de mots-clés dans une requête conjonctive pour élargir le champ de la recherche n'est pas satisfaisante.

Pour rendre le résultat de l'interrogation plus précis tout en fournissant un langage souple, l'opérateur d'adjacence, dénommé ADJ ou NEAR selon les systèmes, a été introduit dans le langage de requête booléen. Son utilisation est limitée car il est utilisé seulement entre deux mots d'une requête. Cet opérateur ne permet pas en soi de résoudre le problème de dualité rappel/précision induit par les requêtes booléennes mais permet par exemple d'exprimer la requête précédente avec « `eta ADJ 10 France` » afin de détecter avec plus de précision les documents, c'est-à-dire, ceux qui contiennent au plus dix termes entre `eta` et `France`. Néanmoins, il reste impossible d'exprimer intégralement la requête « `(eta OU (organisation ET terroriste ET basque)) NEAR (france OU français)` » avec l'opérateur de proximité puisque celui-ci s'applique seulement entre deux termes. De plus, sa généralisation à plus de deux termes n'a jamais été étudiée dans le cadre des modèles booléens, mais certaines approches qui reposent sur les intervalles de texte étendent la notion de proximité à plusieurs termes[9]. Enfin, dans les systèmes qui utilisent cette extension du modèle booléen, la sélection des documents reste binaire et ne permet pas de classer les documents.

[7] Dans cet exemple, la disjonction est utilisée pour donner un synonyme à l'abbréviation.

[8] Ce besoin d'informations a été exprimé pour la campagne d'évaluation CLEF 2006.

[9] Nous reviendrons sur ces approches ainsi que sur la possibilité ou non de généraliser l'opérateur de proximité aux nœuds d'un arbre de requête booléen dans le second chapitre de l'état de l'art (cf. sections 3.2 et 3.4).

Afin de répondre aux problèmes de classement des documents selon un niveau de pertinence et de proposer une solution au problème de la pertinence partielle d'un document par rapport à une requête, le modèle vectoriel [Salton, 1971b, Salton et McGill., 1983] a été défini et mis en place dans le système SMART. Les documents comme les requêtes y sont représentés par des vecteurs[10] dont les composantes correspondent au poids[11] de chaque terme du vocabulaire d'indexation. Pour une requête, une valeur de similarité entre la requête et le document est calculée en prenant le cosinus entre le vecteur représentant la requête et celui représentant le document. Ceci permet d'établir une liste triée des documents en fonction du score de similarité des documents. La proximité entre les termes dans les documents n'est pas prise en compte dans le calcul de la pertinence du document. Ce calcul ne reflète pas la proximité des termes de la requête, l'utilisateur ne peut utiliser qu'un simple ensemble de mots-clés pour spécifier son besoin d'informations.

Le modèle booléen présente donc l'avantage de permettre de spécifier des relations de proximité au niveau des requêtes, mais son inconvénient majeur est l'absence de classement dans la liste de réponses. Par contre, le modèle vectoriel, attribuant un score de pertinence aux documents pour une requête donnée permet ce classement mais ne prend pas en compte le fait que les termes de la requête se retrouvent proches ou non dans les documents sélectionnés. Le tableau 1.1 situe les modèles par rapport à l'utilisation de la proximité et le domaine de valeur du score calculé entre document et requête.

modèle	appartenance	fréquence	proximité	domaine du score
booléen	oui	non	non	$\{0, 1\}$
vectoriel	oui	oui	non	[0, 1]
booléen étendu (near)	oui	non	oui	{0,1}
booléen étendu (ens. flous)	oui	oui	non	[0,1]

TAB. 1.1 – Modèles classiques et proximité

L'idée fondamentale de nos travaux est d'utiliser la notion de proximité pour le classement des réponses tout en combinant les avantages du modèle de requête booléen.

Notre travail étudie l'hypothèse selon laquelle plus les termes de la requête sont retrouvés proches les uns des autres dans les documents, et plus ceci apparaît fréquemment dans un document, alors plus ce document est pertinent. Ceci permet d'une part de mettre en valeur les documents qui contiennent des expressions constituées des mots-clés de la requête, et, d'autre part, de détecter les documents possédant des morceaux qui contiennent les mots-clés de la requête proches les uns des autres. Ceci doit correspondre au besoin de l'utilisateur, c'est-à-dire aux mots-clés qu'il soumet pour représenter son besoin d'information.

[10] De dimension n, où n est la taille du vocabulaire d'indexation.

[11] Les poids sont fonction de la fréquence du terme dans le document et de la fréquence documentaire du terme.

1.3.2 L'organisation du sens dans le texte

L'auteur d'un texte choisit les mots en fonction des idées et des émotions qu'il veut transmettre ; la répartition et le choix du vocabulaire sont pour lui des moyens d'écrire et de structurer le texte. Lors de la première étape du processus de recherche d'informations, l'utilisateur transpose son besoin d'informations en une requête et s'attend à retrouver dans les documents retournés les mots-clés soumis au système moyennant des éventuelles variations lexicales ou sémantiques[12].

Au retour des réponses, l'utilisateur juge de la pertinence du résultat obtenu. Le facteur humain intervient à cette étape comme aux autres (construction de la requête notamment) et le jugement est en partie subjectif. Néanmoins, on peut imaginer qu'un certain nombre de critères sont à prendre en compte pour évaluer la pertinence. L'utilisateur juge le document selon la présence ou l'absence des concepts qu'il attendait, selon la représentation en termes de quantité de ces concepts mais aussi selon la disposition de ces concepts dans le texte. Il est clair qu'un utilisateur ne décide pas de la pertinence du document sur la satisfaction d'une requête booléenne ou sur la fréquence des mots-clés, mais qu'il prend aussi en compte le sens et le contexte dans-lequel apparaissent les termes de la requête. C'est ce que nous souhaitons modéliser à travers la proximité.

En général, les méthodes traditionnelles tiennent compte de l'absence, de la présence ou de la fréquence des termes pour déterminer la pertinence au niveau du système (cf. tableau 1.1). Notre approche tente de modéliser le troisième critère de décision concernant la disposition des termes en tenant compte de leur position dans un document.

1.3.3 L'examen des réponses par l'usager

L'idée de tenir compte de la répartition et de la proximité des termes de la requête offre l'avantage d'obtenir une trace concrète de la pertinence système. En effet, plutôt que d'être calculée globalement pour un document, la pertinence, dans notre approche, est calculée à toutes les positions du texte dans ce document. Une telle méthode offre donc la possibilité de représenter visuellement la pertinence système, ce qui permet à l'usager d'accéder rapidement aux endroits des documents jugés pertinents par le système. Cependant, d'autres méthodes basées sur la proximité n'ont pas l'objectif de faciliter l'examen des réponses à l'usager. Néanmoins, ces méthodes réputées pour être des méthodes à haute précision, c'est-à-dire contenant des documents pertinents parmi les premiers retournés, permettent, de ce fait, un examen rapide des résultats. En conclusion, notre travail, situé dans le cadre de la recherche d'informations textuelles, privilégie une approche à haute précision en exploitant la proximité des termes de la requête retrouvés dans les documents. Notre approche teste l'hypothèse que plus les mots-clés sont proches dans un

[12] polysémie, synonymie, hyponyme, hypéronyme, etc.

document, et cela le plus grand nombre de fois, alors plus ce document doit être placé en haut de la liste des réponses d'un système de recherche d'informations.

1.4 Contributions de la thèse

Nous pouvons présenter les contributions sur deux plans :

1. sur le plan *théorique*, nous définissons une méthode pour interpréter les requêtes booléennes, qui tient compte de la proximité des termes pour attribuer un score aux documents afin de les classer. L'originalité de la méthode réside dans l'alliance de l'expression des requêtes en langage booléen et du classement avec la proximité. Dans la littérature, nous pouvons trouver d'une part, des méthodes utilisant la proximité des termes pour classer les documents et d'autre part des méthodes étendant le modèle booléen avec la proximité pour sélectionner les documents pertinents, mais il n'existait pas de méthodes alliant l'expressivité des requêtes booléennes et le classement des documents selon la pertinence définie par le système de recherche d'informations.
2. sur le plan *pragmatique*, nos expériences de première année nous ont appris à manipuler et gérer les collections de données. Notre expérience préliminaire mesurant l'impact des paramètres classiques tels que la fréquence des termes et la fréquence documentaire a conforté l'idée que l'utilisation de la proximité peut améliorer la précision des réponses d'un système de recherche d'informations en utilisant la proximité comme un indicateur de pertinence. Ensuite, pour avoir une idée de l'amélioration de la qualité des résultats par rapport aux modèles traditionnels tels que le modèle vectoriel, nous avons mis en œuvre les trois premières méthodes à intervalles (cf. section 3.4) basées sur la notion de proximité. Enfin, nous avons étendu le système de recherche d'informations LUCY [13] en y intégrant la méthode que nous avons définie, basée sur la proximité floue. Pour ces expérimentations, nous avons commencé par travailler sur la collection WFR4 de 5 millions de pages du Web francophone, collectée au Laboratoire CLIPS-IMAG par Mathias Gery et Dominique Van Freydar pour l'étude sur les fréquences documentaires. Ensuite, pour analyser l'impact de l'utilisation de la proximité, nous avons utilisé la collection de test WT10G de la campagne d'évaluation TREC, auparavant utilisée pour montrer les résultats de la dernière thèse soutenue dans l'équipe. Néanmoins, notre équipe de recherche n'avait pas encore participé aux campagnes d'évaluations qui réunissent chaque année la communauté des chercheurs en recherche d'informations. Nous avons donc participé pour la première fois en 2005 à ces campagnes (CLEF pour l'Europe et TREC aux États-Unis) en utilisant l'implémentation du modèle de proximité floue et nous avons renouvelé l'expérience en 2006 avec quelques modifications pour la tâche adhoc en Français pour CLEF et pour la tâche adhoc TERABYTE pour TREC.

[13] http://www.seg.rmit.edu.au/lucy

1.5 Plan du mémoire

Après cette introduction, le chapitre 2 présente les notions de base comme les différentes étapes d'un processus de recherche d'informations, l'architecture d'un système, les modèles théoriques classiques et les mesures d'évaluation des systèmes. Ensuite, le chapitre 3 expose les différentes méthodes qui touchent de près ou de loin à la notion de proximité en recherche d'informations. Les deux chapitres suivants sont consacrés à nos contributions. Le modèle que nous avons élaboré pour utiliser la proximité des termes est décrit dans le chapitre 4. Le chapitre 5 expose l'ensemble de nos expériences, depuis les premières qui nous ont incitées à poursuivre nos recherches sur l'utilisation de la proximité en recherche d'informations jusqu'aux plus récentes où nous avons valorisé notre savoir-faire dans les campagnes d'évaluation CLEF et TREC en 2005 et 2006.

Chapitre 2

Recherche d'informations

Table des matières

2.1 Qu'est-ce qu'un système de recherche d'informations

Un *système de recherche d'informations* (SRI) est un système informatique fournissant un accès à l'information à travers des requêtes. Cette dernière est restituée sous la forme de documents numériques (ex. bibliothèques numériques, moteurs de recherche) ou bien sous la forme de références vers des documents physiques (ex. bibliothèques universitaires).

En mettant en correspondance l'ensemble des documents de sa base et la requête soumise par l'usager, un SRI permet en premier de retrouver les documents correspondant aux besoins d'informations de l'usager et en second de les lui présenter. Un SRI possède une architecture complexe, essentielle pour sauvegarder et restituer l'information.

Pour être opérationnel, un *système de recherche d'informations* a besoin d'une *base documentaire* qui est un espace de sauvegarde des documents, d'un *module d'indexation* pour construire la représentation du contenu des documents, d'un *module de traitement des requêtes* pour établir la correspondance entre les représentations des documents et des besoins d'informations, et d'une *interface homme/machine* pour la saisie des requêtes et la présentation des documents jugés pertinents par le système.

Dans la section suivante, nous présentons les objets et les acteurs du processus de recherche d'informations ainsi que ses particularités que ce soit du côté utilisateur ou du côté système.

2.2 Le processus de recherche d'informations

D'après [Baeza-Yates et Ribeiro-Neto, 1999], deux difficultés peuvent être rencontrées en recherche d'informations :

- celle liée à l'utilisateur dépend de sa capacité à déterminer ses besoins d'informations et à les transformer en requête. Cette dernière peut ne décrire que partiellement le besoin d'informations réel ;
- celle liée au système concerne la représentation et l'identification des documents pertinents à la requête.

Les trois aspects principaux qui se dégagent dans un système de recherche d'informations sont en premier lieu, le document, en deuxième lieu, la requête — ces deux premiers éléments constituent les données — en troisième lieu, le processus de traitement qui établit une correspondance entre les deux premiers éléments.

2.2.1 Les objets et les acteurs du processus

Si les SRI sont apparus dans les années 50, c'est parce que la gestion de l'information est devenue un enjeu stratégique pour la société. L'écriture et la lecture existent depuis les premières civilisations de l'antiquité. Les textes, dont certains sont transcrits dans des formats numériques (projet gutenberg)[1], possèdent des caractéristiques statistiques qui sont exploitées dans les SRI. Ces caractéristiques ont été découvertes dès les premières recherches en théorie de l'information, elles traduisent les relations existantes entre les langues, les auteurs et les documents textuels. Si l'auteur a besoin de s'exprimer, son vocabulaire et son style sont les moyens de transmettre ses idées. Cependant, pour que ses textes soient retrouvés et lus par l'intermédiaire d'un SRI, l'utilisateur effectuant une interrogation doit être en phase avec le vocabulaire utilisé dans les documents indexés car de nos jours, peu de systèmes exploitent la notion de sémantique. En effet, le niveau conceptuel n'a pas encore été atteint par les systèmes : les documents sont sélectionnés par rapport à la terminologie.

Dans ce contexte, notre vision est la suivante : les auteurs des documents, les utilisateurs et le système lui-même sont les acteurs du processus alors que les documents, les requêtes qui traduisent les besoins d'informations, et les données utilisées par le système sont les objets du processus comme illustré dans le tableau 2.1. De plus, le besoin d'informations reste au centre du processus de recherche d'informations.

Selon Belkin [Belkin *et al.*, 1993], un utilisateur interroge un système de recherche d'informations s'il se trouve dans un état anormal de connaissance. Via le processus de recherche d'informations, il s'attend à recevoir un ensemble de documents qui répondent à son besoin pour combler ses lacunes.

[1] http ://www.guntenber.org

Objets	Acteurs
Document	Auteur
Requête	Utilisateur
Index	Fonction de correspondance
Interface du système	Concepteur

Besoin d'information

TAB. 2.1 – Notre vision synthétique des acteurs et des objets

Nous définissons un *document* comme une unité d'informations véhiculant un certain nombre d'idées, de connaissances ou de savoir-faire. Cette unité est retournée en réponse à l'utilisateur. Un système de recherche d'informations lui associe généralement une représentation interne et effectue sa sauvegarde sous des formats variés (texte, image, son, vidéo). Nos contributions relèvent de la recherche d'informations dans le cadre des documents textuels. Pour notre problématique, la donnée utile pour la représentation des documents est la posision des termes. Un ensemble de documents constitue un *corpus de documents*.

Les documents, et parfois les requêtes sont exprimés dans le *langage naturel*, c'est-à-dire le langage qui permet de s'exprimer couramment à l'oral comme à l'écrit. L'interface d'un système de recherche d'informations permet de saisir les requêtes, ces dernières sont traduites par le système dans son langage de représentation interne.

2.2.2 Du point de vue de l'utilisateur

Un processus de recherche d'informations (cf. figure 2.1(a)) est généralement itératif. Chaque étape se déroule en trois phases : il commence par *la formulation de la question*, côté utilisateur, (cf. figure 2.1 (b)), continue par *la construction de la réponse*, en accédant aux documents, côté système (cf. figure 2.1 (c)) et finit par *l'évaluation de la réponse*, côté utilisateur, (cf. figure 2.4 (A, B et C)).

La figure 2.1 illustre quelques éléments clés du processus de recherche d'informations :

(a) *une recherche ou une demande d'informations* s'accompagne d'une réponse fructueuse ou non. Dans la négative, le processus est itératif dans le sens où la question est reformulée dans l'attente d'obtenir une réponse convenable ;
(b) *le besoin d'informations* est formulé par l'utilisateur sous la forme d'une requête qui transcrit un ensemble d'images mentales ;
(c) *le corpus de documents* est accessible par l'index du système. Cet index constitue une représentation des documents de la base documentaire.

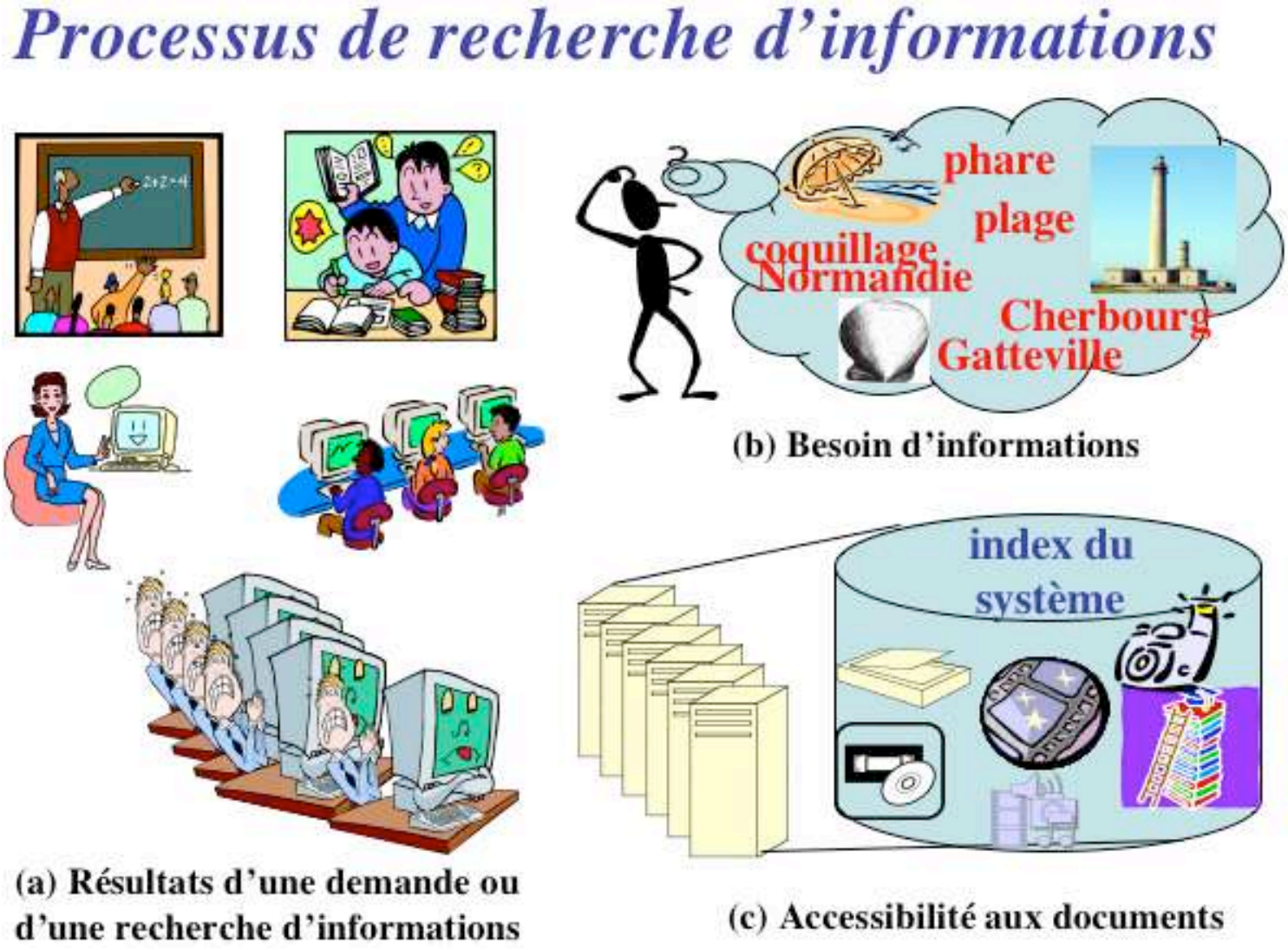


FIG. 2.1 – Différents points de vue du processus de recherche d'informations

2.2.2.1 Expression du besoin d'informations

Une session de recherche, dans un système de recherche d'informations, commence par la soumission d'une requête au système. Pour l'utilisateur, la première difficulté est d'exprimer, en langage naturel, l'image mentale qu'il se fait de son besoin d'informations. Ensuite, la seconde est de traduire cette dernière en requête lisible par un système.

Le choix des mots-clés est important car certains mots du vocabulaire sont à éviter[2]. Il s'agit, d'une part, des mots fonctionnels de la langue qui sont des mots de liaison et qui n'ont pas de signification, et d'autre part, des mots qui ne possèdent pas de pouvoir discriminant : soit globalement, les mots couramment utilisés (ex. maison, vie, manger) possédant une fréquence élevée dans n'importe quel corpus, soit localement, les mots internes au corpus interrogé.

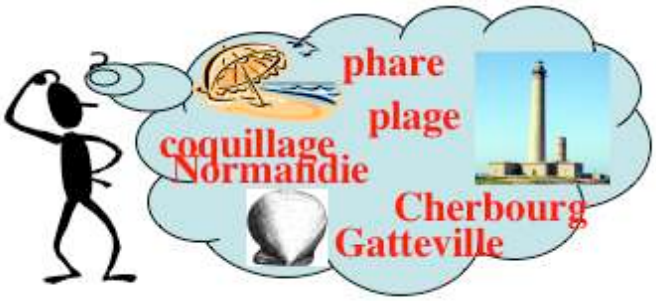
Un mot-clé peut être qualifié d'interne ou d'externe pour un corpus donné, cette distinction est inhérente aux corpus thématiques. Prenons, par exemple, le mot « bateau » pour un corpus sur les bateaux à voile : le mot-clé « bateau » permet d'accéder au corpus, c'est un mot-clé *externe* pour une requête à l'extérieur du corpus, par contre, à l'intérieur du corpus « bateau » est un mot-clé *interne*, car caractérisant tous les documents. Il perd son rôle de mot-clé discriminant et ce n'est plus un bon terme pour une requête. Un mot interne devient donc un mot *vide* (cf. section 2.2.3.1) à l'intérieur du corpus.

Une *requête* est le plus souvent formulée dans un *langage de requête* spécifique au système. Néanmoins, elle peut aussi être exprimée en langage naturel sous la forme d'une phrase ou d'un petit paragraphe. Mais de manière générale, il s'agit d'un ensemble de mots-clés qui en font souvent une description courte et ambiguë qui n'exprime pas intégralement le besoin d'informations. Plusieurs types de *langage de requêtes* peuvent être utilisés :

- simples
  - ensemble de mots ou sac de termes ;
  - une phrase ou un paragraphe en langage naturel ;
- complexes
  - expressions booléennes ;
  - expressions régulières ;
  - langage structuré précisant la valeur d'attributs tels que les noms d'auteurs, les mots du titre etc. ;
  - expression de relations de proximités pondérées entre les mots (cf. section 3.4.2) ;

La figure 2.2 illustre les transformations du besoin d'informations en langage naturel puis en requêtes. Dans le premier cas, les mots importants sont sélectionnés pour être envoyés sous la forme d'un sac de termes au système. Dans le second, tous ces mots sont reliés par des opérateurs pour constituer une expression booléenne. Par ailleurs, au niveau du système, la requête est transformée et possède une représentation interne dans un système de recherche d'informations.

[2] Des précisions sur le choix des termes côté système sont expliqués en 2.2.3.1.



Langage naturel : je voudrais partir en Normandie, profiter du beau temps à la plage, des restaurants du poisson frais, ramasser des coquillages, visiter le sous-marin à Cherbourg et le phare de Gatteville.

Sac de termes : normandie plage coquillage sous-marin Cherbourg Gatteville.

Expression booléenne : (normandie & (plage | coquillage))| (Cherbourg & sous-marin) | Gatteville

FIG. 2.2 – Un exemple de transposition du besoin d'informations

2.2.2.2 Un processus itératif : un dialogue entre l'utilisateur et le système

Pendant une session de recherche, il est important de préciser qu'un résultat concluant n'est pas forcément obtenu dès la première requête. Le processus de recherche d'informations est un processus itératif au cours duquel un dialogue s'installe entre l'utilisateur et le système. La figure 2.4 illustre les différents cas qui peuvent être rencontrés. Le plus décevant (cas A) est celui où le dialogue s'arrête par manque de satisfaction si la réponse est inexistante, incompréhensible ou à côté du sujet. Mais la recherche peut être aussi fructueuse et conduire à d'autres recherches connexes (cas B). Par ailleurs, si les informations requises ont été obtenues (cas C), le processus s'arrête sur une note positive.

Ce processus de recherche d'informations s'apparente donc à un dialogue. On peut faire l'analogie entre un élève posant une question à son professeur et une requête soumise à un système de recherche d'informations. Le dialogue débute dès la formulation de la première question. Le professeur, après avoir assimilé la question, formule une réponse à la portée de l'élève et adapte son discours en fonction des réactions qu'elle provoque (reformulation, demandes de précision sur la question). Par contre, pour un système de recherche d'informations, c'est le module de traitement des requêtes qui analyse, traite la « question » et renvoie automatiquement les documents jugés pertinents, ce dernier n'est qu'une machine qui ne peut pas modifier l'ensemble de réponses s'il s'avère incompréhensible pour l'utilisateur grimaçant !

Parfois, les informations collectées ne correspondent pas, si bien que le besoin d'affiner la réponse apparaît. Les réponses peuvent être hors sujet, incomplètes ou incompréhensibles et

demandent un approfondissement grâce à une extension de la recherche. Pour le dialogue entre l'élève et son professeur, l'expression du visage est un indicateur et le professeur se mettra au niveau de son élève pour compléter et donner une réponse plus compréhensible, par contre, pour une recherche à l'aide d'un système, l'utilisateur va devoir choisir de nouveaux mots-clés qui soit le mèneront à une réponse satisfaisante soit le feront dévier de son sujet initial.

Pour aider l'utilisateur qui ne trouve pas l'information satisfaisante reçue, certains systèmes permettent la *rétroaction de pertinence*[3]. Le système prend en compte les retours de l'utilisateur avant d'interroger à nouveau la base documentaire. L'objectif est de préciser le besoin d'informations pour accéder à des documents que l'utilisateur jugera pertinents. Plusieurs méthodes permettent de mettre en œuvre la rétroaction de pertinence. Par exemple, les liens hypertextes sont exploités pour les pages Web [Brin et Page, 1998]. D'autres approches reposent sur l'enrichissement des requêtes, de nouveaux mots-clés sont ajoutés par le système, soit automatiquement, soit en interaction avec l'utilisateur [Salton, 1971a, Attar et Fraenkel, 1977, Robertson et Walker, 1997].

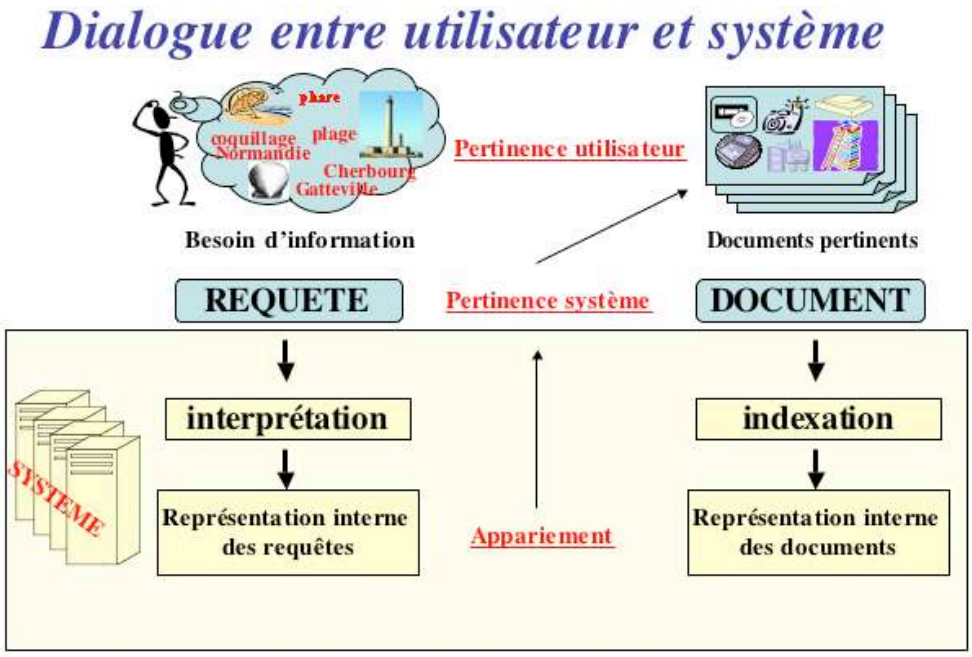


FIG. 2.3 – Lien entre le processus de recherche d'informations et le système

Enfin, une fois l'indexation réalisée, le système peut traiter les requêtes saisies par les utilisateurs. Les requêtes sont traitées de la même manière que les documents afin de pouvoir mettre en correspondance les représentations de chacun et sélectionner les documents.

[3] *Relevance feedback* en Anglais.

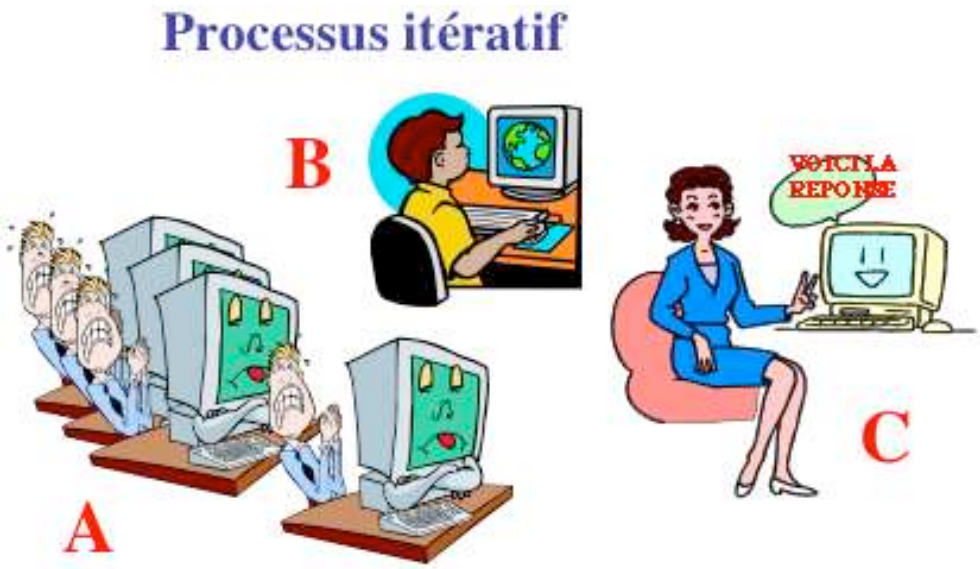


FIG. 2.4 – La recherche d'informations : un processus itératif

2.2.3 Du point de vue du système

Bien que l'expression « processus de recherche d'informations » désigne plutôt le chemin à parcourir par un utilisateur pour trouver une information, plusieurs étapes en amont, au niveau du système informatique, sont nécessaires. Dans un premier temps, un ensemble de documents communément appelé corpus doit être construit. Il existe deux cas de figure : soit la collection est « privée » (documents techniques, rapports d'activité d'une organisation) soit la collection est « publique », par exemple issue du Web. Dans ce dernier cas, un moteur de recherche constitue le système informatique de restitution des documents et un programme « aspirateur » de pages opère constamment sur le Web pour mettre à jour les pages collectées[4].

Ensuite, la deuxième phase consiste à effectuer l'indexation du corpus de documents (cf. section 2.2.3.1) selon la technique du modèle de recherche d'informations implémentée dans le système. L'indexation permet de construire les représentants des documents. Bien qu'il n'y ait pas de niveau sémantique dans les systèmes, les idées véhiculées dans les documents sont mémorisées dans le fichier inversé sous la forme de termes d'index. Le texte est d'abord segmenté en mots (simples ou expressions), ensuite les mots très fréquents sont éliminés, puis une phase de lemmatisation réduit les mots à leur racine, et enfin une pondération des mots est effectuée pour refléter leur importance dans le texte et à travers le corpus. Nous détaillons dans la suite les pré-traitements majeurs effectués sur les données pour en faciliter l'accès (indexation) et réduire leur taille (élimination des mots vides, lemmatisation).

[4]Chaque moteur couvre ainsi une partie des documents du Web, les moteurs de recherches forment un cas particulier de système de recherche d'informations où l'éclatement des documents en plusieurs fichiers représente une autre problématique de la RI.

2.2.3.1 Index

Le point (c) illustré dans la figure 2.1 se focalise sur l'index d'un système de recherche d'informations. Rappelons que si l'on peut retrouver un index pour clôturer chaque volume de l'Encyclopédie, c'est que ce concept est utilisé depuis les débuts de l'édition et de l'imprimerie. Un *index* permet l'accès rapide aux articles, et donc aux informations requises.

En recherche d'informations, la procédure d'*indexation* associe un ou plusieurs mots-clés à un document, c'est une étape nécessaire au vu de la quantité volumineuse de données d'un SRI. En effet, la solution triviale consistant à effectuer une recherche séquentielle des mots-clés est trop coûteuse en temps, et ne peut donc être envisageable sur des collections aussi volumineuses que celle de la Toile ou tout autre corpus aujourd'hui. Un index est généralement composé de fichiers inversés qui permettent à partir d'un terme d'index, de retrouver les documents auxquels il appartient.

L'efficacité de la recherche dépend du choix des termes d'index, c'est-à-dire des termes qui sont retenus pour faire référence aux articles. Dans ce cas, l'indexation peut être manuelle : les auteurs établissent leur index eux-mêmes à la fin des volumes (cf. exemple de l'Encyclopédie de Diderot). Quand les documents sont indexés par rapport à un ensemble de mots-clés prédéfinis, on parle d'indexation contrôlée sinon d'indexation libre. En passant à la numérisation des documents, la question du choix des termes d'index se pose différemment. Si la collection de documents est disponible sous forme numérique, alors l'indexation automatique devient une alternative à l'indexation manuelle. Dans ce cas, les mots-clés du texte fournissent souvent l'ensemble des termes d'index.

En résumé, l'indexation peut être manuelle, comme l'indexation dans les bibliothèques à l'aide de thésaurus, ou automatique, en se basant sur les mots-clés du texte. Cependant, dans les bibliothèques, les ouvrages peuvent être également indexés manuellement avec un vocabulaire libre. D'une autre manière, une indexation automatique peut être envisagée avec un vocabulaire contrôlé pour des corpus thématiques ou pour pallier le manque de ressources mémoires pour des collections de grande taille[5]. Dans le cas de l'indexation automatique, le choix des termes a été influencé par la théorie de l'information.

Loi de Zipf Les études et les observations de Zipf réalisées sur des ensembles de textes quelconques montrent que si l'on dresse une table de l'ensemble des mots du vocabulaire, classés par ordre de fréquences décroissantes, la fréquence d'un terme est inversement proportionnelle à son rang dans la liste. Il s'agit de la loi de Zipf qui peut être aussi exprimée par $tf \times rang = constante$, c'est-à-dire que le produit de la fréquence de n'importe quel mot par son rang est constant. Cette constante est atteinte de manière approximative. La loi de Zipf est vérifiable indépendamment des auteurs et de la langue. Elle stipule que la fréquence du second

[5] Nous avons utilisé une telle solution pour les expériences sur la collection TERABYTE de TREC 2006. Les termes du vocabulaire sont choisis parmi ceux qui expriment les besoins d'informations.

mot le plus fréquent est la moitié de celle du premier, la fréquence du troisième mot le plus fréquent, son tiers, etc., et dans ce cas s'exprime ainsi : $tf_n = \frac{tf_1}{n}$[6]. Par exemple, pour la langue française, les mots outils tels que les articles, prépositions, etc. constituent 50% des mots d'un texte alors qu'ils ne représentent que 0,5% de l'ensemble du vocabulaire. Le reste d'un texte est composé de mots significatifs, quelquefois représentés sous différentes formes et pourtant sémantiquement similaires. Les conclusions de la loi de Zipf ont eu un impact sur l'indexation en recherche d'informations, deux conséquences majeures en sont tirées :

- les mots outils sont mis dans un anti-dictionnaire[7] ce qui permet de réduire d'environ 30% la taille du fichier inversé. Une autre technique, la lemmatisation, permet de réduire la taille du vocabulaire (cf. 2.2.3.2) ;
- la fréquence d'apparition est un critère pour choisir les termes d'index.

Unité lexicale, termes Un mot (ou indifféremment un terme, une unité lexicale) est généralement défini par une séquence continue de lettres ou de chiffres. Le texte est analysé pour déterminer les « lemmes », c'est-à-dire les entrées du vocabulaire du SRI. Après la découverte de ces unités lexicales, plusieurs traitements comme la suppression des mots vides ou la lemmatisation sont appliqués pour réduire la taille de ce vocabulaire. Au moment du traitement du texte, le cas particulier du trait d'union peut être traité ou non. Par exemple, pour « mots-clés », trois entrées peuvent être intégrées à l'index : le terme composé « mot-clé », et, les termes simples « mot » et « clé ». Les syntagmes nominaux[8][Chevallet et Haddad, 2001] sont détectés par des techniques linguistiques [Strzalkowski et Carballo, 1996] ou statistiques [Lebart et Salem, 1994, Manning et Schütze, 1999]. L'utilisation de la proximité directement dans le système permet aussi de retrouver les expressions et de donner une importance à leurs apparitions dans un document.

Mots vides Si les données textuelles sont considérées comme un signal, les *mots vides* sont ceux qui produisent du bruit, linguistiquement, il s'agit des mots fonctionnels comme « le », « de », « elle » en Français qui apparaissent très fréquemment et qui ne sont pas directement porteurs de sens ; de plus ils se retrouvent généralement avec la même fréquence d'apparition dans tous les documents. Outre ces mots de liaison, d'autres, fréquemment utilisés, deviennent aussi des mots vides à cause de leur fréquence élevée. Concrètement, une recherche sur les mots « après, quand, être » n'est généralement pas entreprise car ces mots outils ne sont pas porteurs de sens donc les supprimer n'est pas un inconvénient.

Pondération Souvent, un index des termes les plus représentatifs des documents est construit en utilisant une pondération en fonction de la fréquence d'apparition du terme dans un document,

[6] Une applet sur la page `http://users.info.unicaen.fr/~giguet/java/zipf.html` de l'université de Caen permet de visualiser cette loi sur le texte qu'on lui fourni.

[7] Liste de mots vides ou *stoplist* en Anglais.

[8] Dans le langage courant expression ou *phrase* comme « bateau à voile ».

et de la fréquence d'apparition dans différents documents de la collection. Cette pondération permet de relativiser l'importance d'un terme dans un document – la première occurrence a plus d'importance que les suivantes – et dans la collection, un mot présent dans beaucoup de documents n'est plus discriminant.

2.2.3.2 Lemmatisation

Tout d'abord, la *racinisation* consiste à réduire un mot à sa forme canonique, c'est-à-dire supprimer toutes les variantes flexionnelles du mot liées à son usage (genre, nombre, personne). La forme canonique d'un mot correspond à son entrée dans le dictionnaire : infinitif pour les verbes, masculin singulier pour les noms. La racinisation peut être facilement utilisée dans le cadre de l'indexation manuelle. Par exemple, un documentaliste, expert en la matière, choisit les termes d'index sous leur forme canonique.

L'idée est reprise dans le cadre de l'indexation automatique, nous parlerons alors de *lemmatisation*. Cependant, les algorithmes couramment utilisés ne produisent pas exactement le résultat qui aurait été obtenu manuellement. Par exemple, pour les mots « informations » et « informateurs », la racinisation conduit au radical « inform » tandis la lemmatisation peut donner des lemmes différents « informat » ou, « information » et « informateur », selon l'algorithme utilisé ; de plus le lemme obtenu n'est pas forcément un mot de la langue. L'intérêt de l'élimination des variations morpho-syntaxiques est non seulement de permettre la mise en correspondance des formes variant en genre ou en nombre mais aussi de réduire la taille du vocabulaire. En effet, pour un être humain, cette taille est de l'ordre de 50000 mots mais elle s'accroît énormément pour les moteurs de recherche qui gèrent les documents en texte intégral, parfois même dans plusieurs langues, et avec des orthographes différentes.

La procédure de lemmatisation[9] ramène les mots à leur « radicaux ». Après l'analyse d'un document texte, chaque unité lexicale est traitée, deux aspects morphologiques différents sont pris en compte, le flexionnel pour ramener le mot à son singulier et les dérivés, entre nom et verbe, pour ramener théoriquement le mot à son radical commun. En réalité, comme l'exemple ci-dessus le montre, le résultat obtenu ne correspond pas forcément au mot obtenu par racinisation. Néanmoins, la lemmatisation favorise ainsi le rappel[10] plutôt que la précision[11]. En effet, bien que les transformations soient efficaces pour réduire la taille de l'espace de sauvegarde mais aussi le temps de réponse des systèmes, elles conduisent parfois à des contre-sens comme A et B (n'exprimant pas la même idée) radicalisés en C. Un même terme d'index peut ainsi faire référence à des termes sémantiquement différents. L'accentuation du phénomène de polysémie est l'inconvénient majeur de la lemmatisation car il conduit parfois à des réponses inexactes.

Plusieurs algorithmes appliqués à la langue anglaise effectuent ce traitement, tout d'abord

[9] *Stemming* en Anglais.
[10] Obtenir le plus de documents correspondant de la collection cf. section 2.4.3.1
[11] Les documents rappelés sont effectivement pertinents cf. section 2.4.3.1.

celui de Lovins en 1968 [Lovins, 1968] qui a identifié 260 suffixes, puis celui de Porter en 1980 [Porter, 1980] qui applique 60 règles de réduction. Nous pouvons aussi citer l'adaptation de Savoy [Savoy, 1999] pour le Français[12].

En conclusion, l'indexation peut être manuelle ou automatique, utiliser un vocabulaire contrôlé ou non, prendre en compte un anti-dictionnaire et fusionner les différentes formes des mots du vocabulaire. La phase d'indexation consiste à créer les représentations des documents.

2.2.3.3 Mise en correspondance

Pour beaucoup de systèmes de recherche d'informations, les documents et les requêtes sont représentés à l'aide des mots-clés qui les composent. Un *mot-clé* est utilisé soit comme terme d'index (descripteur d'un document), côté système, soit comme terme dans une requête, côté utilisateur.

Une fois l'index construit et la requête soumise, l'appariement entre les données peut être effectué. La *fonction de correspondance* (ou d'appariement) est le moyen de sélectionner à partir de la représentation du document et de celle de la requête les documents pertinents de la base. Le système attribue généralement un degré de pertinence aux documents : il s'agit de la « pertinence système ». Un système considère qu'un document est pertinent s'il répond aux critères spécifiés dans la requête. Ce jugement dépend des représentations attribuées aux documents et à la requête mais aussi de l'algorithme d'appariement. La notion de pertinence prend un autre sens côté utilisateur. Ce dernier juge un document pertinent par rapport à son besoin initial d'informations et bien évidemment, ce jugement est subjectif, variant d'un utilisateur à un autre pour une même requête. Nous parlerons de « pertinence utilisateur ».

La *pertinence* est un concept fondamental de la recherche d'informations, l'objectif d'un système est en effet de répondre avec le plus de pertinence possible, c'est-à-dire que les documents retournés doivent contenir les informations dont l'utilisateur a besoin. La notion de pertinence est une notion complexe qui peut être caractérisée par différents critères comme celui de « topicalité » (capacité du document à contenir les informations requises) ou d'« utilité » (le document apporte-t-il des informations supplémentaires ?).

Le but de tout système de recherche d'informations est de minimiser la perte d'informations pendant toutes les étapes du processus pour rapprocher la pertinence vue par l'utilisateur de celle vue par le système.

Nous venons de discuter des acteurs et des étapes d'un processus de recherche d'informations. Nous allons tout naturellement dans la section suivante montrer les différents modèles sur lesquels sont fondés la plupart des systèmes. Chaque modèle est caractérisé par sa manière de représenter les documents dans l'index, par son langage de requêtes et le module de

[12] Nous avons utilisé son implantation pour comparer l'impact de la lemmatisation sur les résultats d'une recherche.

traitement associé, par la fonction de correspondance et par sa façon de fournir les résultats [Roussey *et al.*, 1999]. Nous étudierons plus particulièrement, les modèles booléen, vectoriel, probabiliste et à ensembles flous.

2.3 Les modèles classiques

2.3.1 Notations

Dans chaque monographie traitant de recherche d'informations, un chapitre est dédié aux modèles, et chacun propose, au moins dans sa table des matières, une taxonomie de ces modèles. Bien qu'il n'y ait pas de consensus complet sur la classification des modèles, celle de Baeza-Yates et Ribeiro-Neto est assez commune [Baeza-Yates et Ribeiro-Neto, 1999]. Ils considèrent trois *modèles classiques*, respectivement le modèle *booléen*, le modèle *vectoriel* et le modèle *probabiliste*. Ensuite, dans leur vision, chacun de ces trois modèles se décline en variations et ils obtiennent trois familles : (i) *les modèles ensemblistes* (modèles à ensembles flous, et modèles booléens étendus), (ii) *les modèles algébriques* (modèle vectoriel généralisé, et indexation sémantique latente), et (iii) *les modèles probabilistes* (modèles basés sur les réseaux d'inférence, les réseaux bayésiens et les réseaux de croyance).

La figure 2.5 permet de visualiser où les différents éléments des modèles sont utilisés dans le processus de recherche d'informations. Le modèle de requête définit comment l'utilisateur exprime son besoin d'informations ainsi que la représentation interne de la requête dans le système ; le modèle de documents définit la représentation interne des documents ; englobe la notion d'index et enfin, la fonction de correspondance (ou d'appariement) utilise les représentations internes pour calculer la pertinence système.

Pour définir un modèle en recherche d'informations, trois éléments essentiels doivent être clairement définis :

- **un modèle de représentation des documents**, pour savoir quelles informations sont utiles pour l'indexation de la base documentaire ;
- **un modèle de représentation des requêtes** ;
- **le moyen de faire correspondre la représentation des documents et celle des requêtes** .

Ces trois éléments sont interdépendants à travers le modèle de recherche d'informations utilisé.

Nous allons présenter chacune de ces caractéristiques pour les modèles booléen, vectoriel et probabiliste ainsi que celui fondé sur la logique floue, certains aspects de ce dernier étant exploités par le modèle d'interprétation des requêtes que nous proposons dans la chapitre 4.

Nous utiliserons les notations suivantes :

un terme (ou indifférement mot ou mot-clé) est noté t, t', etc. éléments de T, l'ensemble des termes du vocabulaire d'indexation ou du dictionnaire des termes utilisables dans les requêtes,

un document est noté d, d', etc. éléments de D, l'ensemble des documents, généralement appelé collection ou corpus de documents,

une requête est composée de termes et, est notée q, élément de Q, l'ensemble des requêtes,

un poids $w_{t,d}$ (resp. $w_{t,q}$) est le poids du terme t dans le document d (resp. pour la requête q).

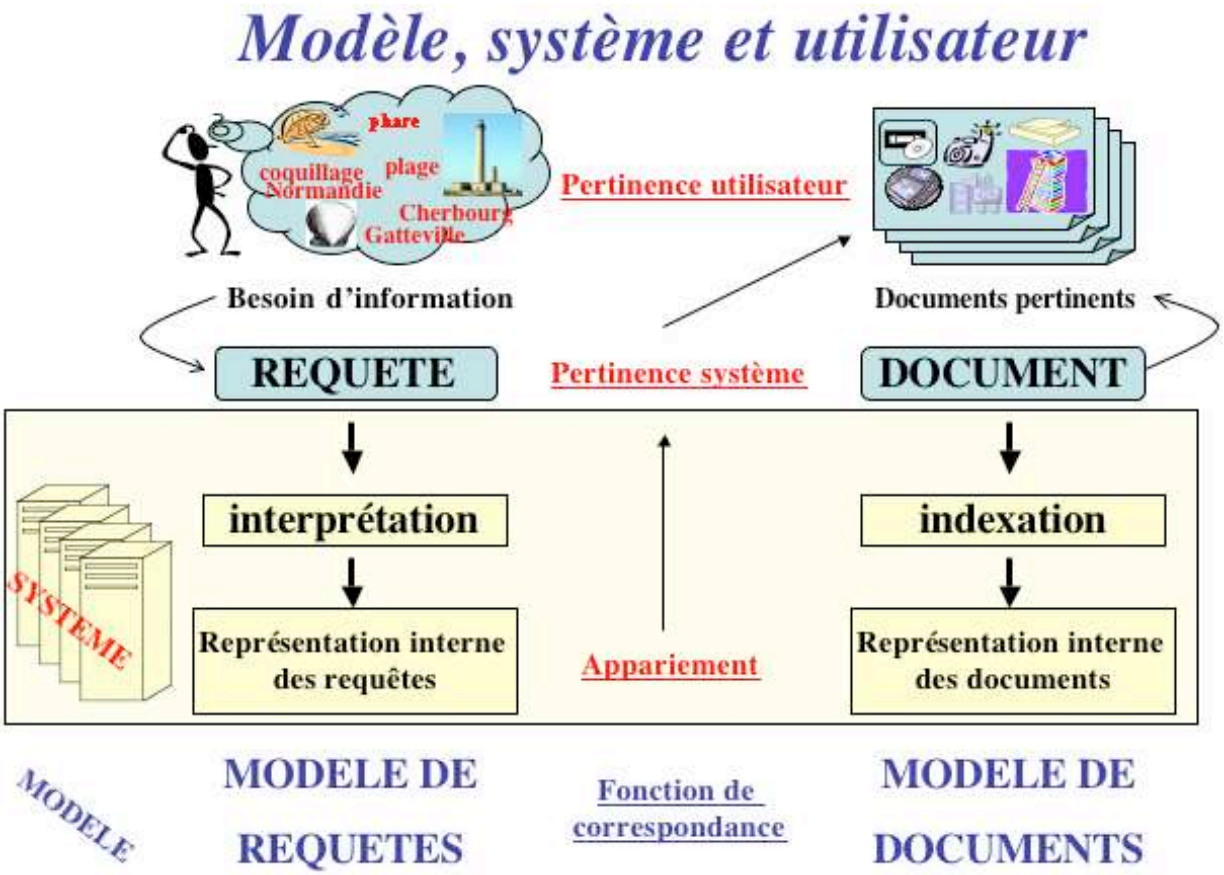


FIG. 2.5 – Lien entre l'utilisateur, le système et le modèle de recherche d'informations

2.3.2 Modèle booléen

Le modèle booléen est utilisé, dès les débuts de la recherche d'informations, sur des collections de résumés d'articles ou des collections indexées manuellement. De ce fait, le modèle booléen repose, à l'origine, sur deux hypothèses : (i) tout terme d'index est univoque, il possède un sens clair, et (ii) un document est « monotopique », il ne contient qu'un seul thème pouvant être décrit par un ensemble restreint de mots du vocabulaire [Blair et Maron, 1990].

Ces hypothèses, a priori évidentes, étaient acceptables au début de la recherche d'informations sur des corpus indexés manuellement ou sur des corpus de résumés d'articles, elles ne le sont plus de nos jours. En effet, la prise en compte du texte intégral produit, d'une part, des corpus volumineux où le problème de polysémie se rencontre facilement, et d'autre part, des documents aux multiples sujets. La simplicité de ces deux hypothèses remet en cause l'efficacité du modèle booléen, ce modèle a donc fait l'objet de diverses modifications que nous présenterons après avoir expliqué ses principes de base.

Dans le modèle booléen pur, un document est représenté par une conjonction de termes indépendants sans pondération, tandis qu'une requête est une expression booléenne composée de termes reliés par les opérateurs classiques OU, ET ou NON. Un document est jugé pertinent par le système si l'expression logique de la requête est satisfaite par ce document. Dans ce modèle, étant donné une requête, un jugement *binaire* de pertinence est attribué à chaque document par le système : selon lui, un document *est* ou *n'est pas* pertinent à la requête.

Une des forces de ce modèle est que le langage de requête est assez expressif alors que sa principale faiblesse est son critère de pertinence binaire.

2.3.2.1 Modèle de documents

Pour le modèle booléen de base, un *document* est un ensemble de termes indépendants sans aucune pondération, ce qui peut être modélisé soit comme $d \subset T$ ou comme $d \in \{0,1\}^T$, pour un document d. Nous préférerons cette seconde forme parce qu'elle s'étend facilement vers celle utilisée par les modèles booléens étendus où : $d \in [0,1]^T$.

Avec une telle définition, un document d est une fonction $d : T \rightarrow [0,1]$. Bien que dans le modèle booléen pur, le poids soit inexistant (ou bien égal pour tous les termes décrivant un document), nous pouvons modéliser le poids du terme t dans le document d par $d(t)$, plus usuellement noté $w_{d,t}$ pour anticiper le cadre des extensions du modèle prenant en compte la pondération des termes d'index. Dans le cas du modèle booléen pur le poids $w_{t,d}$ d'un terme t dans le document d appartient à $\{0,1\}$.

2.3.2.2 Modèle de requête

Pour l'utilisateur, une requête est une expression booléenne qui peut se modéliser avec un arbre où les feuilles sont des termes (pondérés ou non), et les nœuds internes sont des opérateurs OU, ET ou NON (avec éventuellement un paramètre numérique p). Dans le modèle booléen de base, pondération des termes et paramétrisation des opérateurs n'existent pas et n'ont donc pas à être représentés dans l'arbre. Par exemple, la requête (**A** ET **B**) OU **C** est représentée par l'arbre de la figure 2.6.

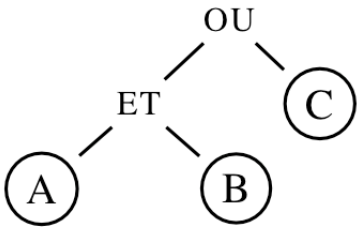


FIG. 2.6 – Arbre de la requête (**A** ET **B**) OU **C**.

Un élément de Q est soit une feuille soit un nœud interne. Une feuille est un couple $(t, w_{q,t}) \in T \times \mathbb{R}$. Un nœud interne est un triplet $(\mathrm{op}, (q_i)_i, p) \in \{et, ou\} \times \mathcal{P}(Q) \times \mathbb{R}$, où $(q_i)_i$ est un sous-ensemble fini de Q et p est le paramètre numérique pour ce nœud utilisé pour l'évaluation de la requête.

2.3.2.3 Modèle d'évaluation de pertinence

Une fonction d'évaluation de pertinence doit calculer un score $s(q, d)$ pour le document d par rapport à la requête q. La définition de cette fonction est récursive dans la mesure où la modélisation d'une requête est récursive elle aussi. L'évaluation est effectuée en partant de feuilles et remonte jusqu'à la racine de l'arbre. Si l'expression booléenne de la requête est évaluée à «vrai» pour le document d alors $s(d, q) = 1$, alors, le modèle booléen prédit que le document d est pertinent à la requête q sinon le document n'est pas pertinent et $s(d, q) = 0$.

Pour une implantation efficace, un fichier d'index permet de connaître la liste des documents de la collection qui contiennent un terme t. A partir des listes des termes de la requête, l'évaluation est effectuée avec l'union (resp. l'intersection) pour une disjonction (resp. conjonction) de termes. Pour la négation, la différence est utilisée.

Le modèle booléen apporte certains avantages :
- la modélisation mathématique, basée sur la théorie des ensembles, en fait un modèle simple à expliquer dont la réponse est facilement prévisible ;
- le temps de réponse est assez faible, et même pour les collections volumineuses les systèmes restent efficients ;
- en posant une requête précise, *re*-trouver des documents connus est facile ;
- le langage de requête est structuré, effectuer l'interrogation de manière précise avec une expression logique est très apprécié par les utilisateurs expérimentés ;

Néanmoins, le dernier avantage [Borgman, 1986, Cooper, 1988] peut se retourner en un inconvénient parmi d'autres :
- pour certains utilisateurs, la transformation du besoin d'informations en expression booléenne n'est pas une tâche aisée. En effet, l'utilisation du « et » en langage naturel doit parfois se transposer en OU dans une expression logique [Gupta et Padmini, 1987]. Par exemple, si je veux les documents expliquant les origines des ânes corses et celles des

ânes normands, ma requête sera « âne ET (corse OU normand) ». Un ensemble d'études [Howard, 1982, Fenichel, 1981, Hersh et Over, 2001] montre que seuls des individus expérimentés obtiennent de bons résultats ;
- l'appariement total implique le retour de documents complètement pertinents. Des documents répondant partiellement à la requête, mais pouvant être pertinents pour le besoin d'informations de l'utilisateur, sont écartés. Ce manque de souplesse induit un rappel insuffisant, lorsque la requête est très précise, ou très large, dans le cas contraire ;
- l'absence de classement entre les documents retournés, ou autrement dit, le manque d'un niveau de pertinence ne facilite pas l'examen des réponses ;
- l'absence de pondération des termes d'index (resp. de la requête), ne permet pas de relativiser l'importance des différents « concepts » dans les documents (resp. dans la requête).

Dans certains systèmes, une manière de préciser le besoin d'informations, est d'écrire une requête encore plus structurée en spécifiant que les termes apparaissent dans le titre, le résumé ou d'autres parties du document [Oldroyd et Schroder, 1982].

Par ailleurs, pour résoudre une partie des problèmes liés à l'appariement exact, deux symboles « joker » ont été ajoutés au langage de requête « ? » pour remplacer n'importe quel caractère ainsi que « $*$ » pour tronquer le début ou la fin des termes. L'objectif avec ces deux symboles est de mettre en relation les documents composés des différentes formes dérivées des mots-clés. Néanmoins, l'utilisation de ces jokers ne résout pas le problème des requêtes conjonctives qui contraignent fortement le besoin d'informations et ne permettent pas d'avoir un rappel important (cf. section 1.3.1). voire inexistant le cas échéant[13]. Si un document ne satisfait qu'une des deux parties de la requête, ce dernier ne sera pas retourné à l'utilisateur.

Une extension du modèle booléen fournit un opérateur d'adjacence ADJ au langage de requêtes, pour attribuer un sens plus strict à l'opérateur ET. L'utilisateur peut préciser que deux mots apparaissent l'un près de l'autre et peut même indiquer le nombre de mots maximum devant les séparer. Dans ce cas, l'index des documents doit inclure la position des occurrences de termes afin de fournir les réponses dans un temps raisonnable ; la proximité des termes n'est donc pas analysée « à la volée » mais sauvegardée dans l'index via les positions. De ce fait, la taille du fichier inversé augmente. Il n'en reste pas moins que ce modèle avec ADJ est booléen, donc une requête permet de sélectionner des documents en fonction de la proximité mais le système ne calcule pas la pertinence en fonction du nombre de fois où cette proximité apparaît mais sur l'absence ou la présence de cette proximité. Par conséquent, les systèmes basés sur ce modèle ne produisent pas de classement des documents considérés comme pertinents en fonction du nombre d'occurrences de relations de proximité trouvées dans le document.

[13] Pour une requête telle que $(t_1 \& t_2)$, un compromis est d'utiliser des disjonctions de termes sémantiquement proches $(t_1|t'_1)\&(t_2|t'_2)$.

2.3.3 Modèle du niveau de coordination

Pour contourner les difficultés liées à la transformation du besoin d'informations en expression booléenne, et pour fournir un niveau de pertinence système à l'utilisateur, la méthode du niveau de coordination permet d'écrire une requête, tout simplement, sous la forme d'un ensemble de mots-clés. La pertinence, dans le système *quorum search* [Cleverdon, 1984], est évaluée en comptant le nombre de mots-clés communs entre les documents et une requête. La requête est un ensemble de termes, les documents contenant les k termes puis $k-1$, etc. jusqu'à 1 terme sont présentés dans cet ordre. Notons que les documents qui possèdent le même nombre de termes seront classés arbitrairement, par exemple, selon leur place dans la collection ou leur date de création. Cette méthode retourne des classes de documents dont la caractéristique de regroupement est le nombre commun de mots-clés entre les documents et la requête. Des modèles prenant en compte la pondération des termes, que l'on expliquera par la suite, ont été développés pour pallier ce problème et permettent de distinguer les documents qui appartiennent à la même classe.

Pour obtenir le même cas de figure dans un modèle incluant la pondération des termes, il faudrait qu'une combinaison de termes de la requête ait la même fréquence dans plusieurs documents et à travers la collection : plus la collection et la requête sont longues, plus les risques d'un tel cas de figure diminuent. Ainsi, dans la liste des documents pertinents, l'écart entre les scores se creusant, plusieurs niveaux de pertinence sont définis. Une amélioration du niveau de coordination correspond au schéma de similarité classique qui prend en compte le nombre d'occurrences de termes dans un document plutôt que le nombre de termes en commun. Cette amélioration correspond au modèle $bnn.bnn$ utilisé dans le système SMART (cf. Annexe A).

2.3.4 Modèles booléens étendus

Il est couramment admis qu'un utilisateur préfère comme résultat d'une interrogation une liste triée. Pourquoi cette préférence ? Au-delà de la satisfaction exacte du document à la requête, l'utilisateur attend que le système compare les documents pertinents pour lui préparer un ensemble de réponses faciles à examiner. Il semble que certains critères comme le nombre de termes de la requête, leur fréquence dans le document, leur proximité ou leur éloignement, leur puissance discriminante dans la collection soient des indicateurs d'une part quantifiables par un système informatique, et d'autre part implicitement utilisés par les utilisateurs lors du jugement de pertinence. Ces idées sont exploitées pour établir une graduation de la pertinence côté système. L'utilisation de la logique floue permet une telle graduation en passant du cadre purement booléen à la modélisation de notions vagues.

2.3.4.1 Logique classique vs logique floue

Que ce soit avec la logique classique, la théorie classique des ensembles ou le calcul des probabilités, seule la manipulation d'objets précis clairement liés entre eux via des sous-ensembles, des fonctions caractéristiques ou des relations entre événement et complémentaire est réalisable. Cependant, le monde qui nous entoure est plus complexe et ces modèles stricts et précis ne permettent de représenter qu'une infime partie des problèmes et des applications de la réalité [Tong Tong, 1995].

En 1973, Zadeh formalise la complexité des systèmes de notre environnement sous la dénomination de principe d'incompatibilité [Zadeh, 1973] :

> Au fur et à mesure que la complexité d'un système augmente, notre capacité à formuler de manière précise et significative son comportement, diminue jusqu'à une limite, au-delà de laquelle la précision et la signification deviennent des caractéristiques pratiquement mutuellement exclusives.

En effet, la logique booléenne et ses ensembles ne nous permettent de modéliser les applications réelles que de manière trop simpliste et n'abordent pas la représentation d'objets complexes.

Le problème se situe au niveau du passage d'un état à l'autre dans la logique booléenne ou de son événement et de son complémentaire dans le calcul des probabilités. Par exemple, en recherche d'informations, un document peut passer de l'état pertinent à non pertinent selon le contexte d'apparition des mots-clés de la requête, mais un autre contexte pourrait aussi bien rendre le document partiellement pertinent, dans ce cas, dans quel ensemble le placer, l'ensemble des documents pertinents ou des non pertinents ? L'utilisateur est capable d'appréhender la complexité et de jouer avec les notions vagues et imprécises véhiculées par le langage naturel mais un système informatique basé sur une logique booléenne détermine un état $\{0, 1\}$ mutuellement exclusif selon l'évaluation des objets (dans notre cas, des mots-clés) qui lui sont présentés. Ici, le principe du tiers-exclu est remis en cause, la représentation du monde réel et de ses applications nécessite de la nuance et ne peut se réaliser au moyen d'un modèle idéal et simple comme la logique booléenne.

Un exemple de la vie courante Dans notre langage quotidien, nous utilisons souvent un vocabulaire imprécis ; par exemple, dans le dénombrement des quantités (plusieurs, beaucoup, approximativement, la plupart) ou la qualification d'individus, d'objets, ou de phénomènes (intelligent, chaud, pertinent, tiède). Le sens réel de ces mots découle de notre interprétation et il est parfois difficile de donner une définition précise. Le terme de variable linguistique est introduit [Zadeh, 1975], il s'agit de variables dont les valeurs sont des termes ou des expressions linguistiques.

Prenons par exemple les qualificatifs de « froid », « tiède » et « chaud » lorsque l'eau coule du robinet. La variable linguistique à considérer est la « température » et les valeurs sont « froid »,

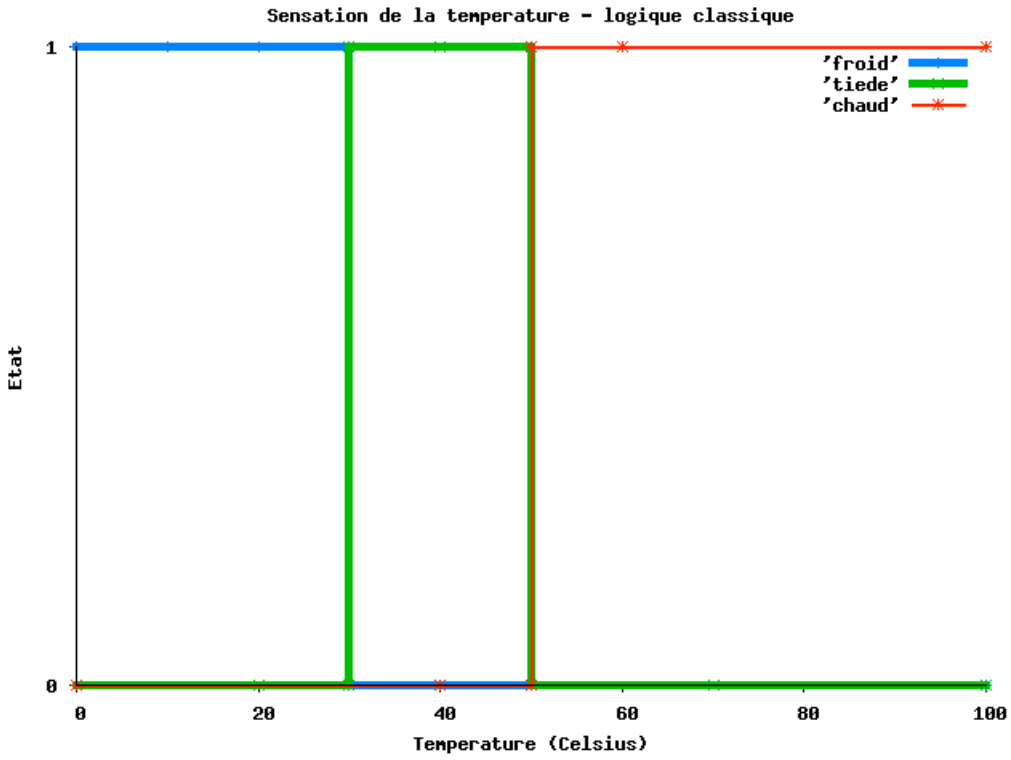


FIG. 2.7 – Représentation des classes de la sensation de la température en logique classique

« tiède » et « chaud ». La logique booléenne classique oblige à définir les frontières entre froid et tiède (par exemple à 30° Celsius) puis entre tiède et chaud (par exemple à 50° Celsius), c'est-à-dire que pour une température donnée, une seule sensation peut être ressentie comme illustré dans la figure 2.7. En réalité, ce n'est pas le cas, une eau à 25°C peut être ressentie comme froide et un peu tiède. Pour prendre en compte cette constatation, trois classes de température peuvent être représentées en utilisant la logique floue comme illustré dans la figure 2.8. En prenant la classe des températures « tièdes », nous remarquons que 22°, 34° ou 50° ne sont pas à mettre sur le même plan car les écarts sont significatifs et peuvent traduire des sensations différentes. Ici, l'utilisation de la logique floue introduit une notion d'échelle ou une graduation de la sensation de chaleur qui permet de restituer les nuances du concept flou « sensation de la température » inadaptée à la notion d'appartenance du modèle ensembliste classique.

De manière générale, la théorie classique des ensembles n'est pas adaptée aux notions vagues et imprécises. Or de nombreux domaines utilisent ce genre de notions ; pour prendre en compte la complexité du monde réel, L.A Zadeh a introduit en 1965 la théorie des ensembles flous. Le principe fondamental réside en l'inutilisabilité du concept classique d'appartenance à une classe définie par une notion vague, tout n'est pas blanc ou noir : toutes les nuances de gris ou de couleurs sont à utiliser. Il s'agit de définir une échelle ou degré d'appartenance qui permet de pondérer des objets par rapport à une propriété vague (comme une pertinence entre requête et documents).

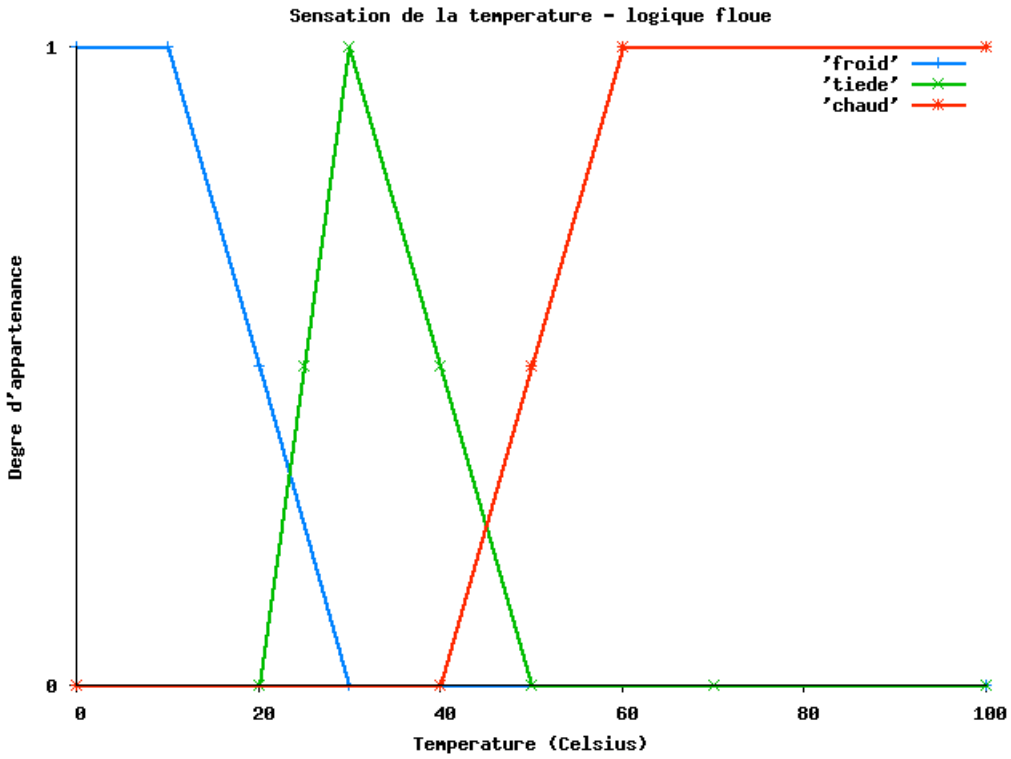


FIG. 2.8 – Représentation des classes de la sensation de la température en logique floue

L'idée à retenir pour le domaine de la recherche d'informations, c'est que la notion de pertinence est une notion vague et imprécise livrée à l'interprétation de tout un chacun. De cette manière, plutôt que de se restreindre à une pertinence binaire, des nuances peuvent être introduites avec une modélisation floue.

Les notations Nous reprenons ici les notations du livre *La logique floue, J.R Tong Tong, édition Hermes, 1995*. La caractéristique fondamentale d'un ensemble classique est la frontière abrupte entre deux catégories d'éléments : ceux qui sont dans l'ensemble et qui lui appartiennent, et ceux qui sont à l'extérieur et qui appartiennent à son complémentaire. Le concept d'appartenance est dans ce cas modélisé par une fonction ϕ_A de type tout ou rien appelée fonction caractéristique de l'ensemble A, sous-ensemble d'un ensemble de référence U appelé également univers ou ensemble universel.

Nous pouvons ainsi représenter le modèle booléen en recherche d'informations à l'aide de la fonction ϕ_P. Si par exemple U désigne une collection et que le sous-ensemble P est défini par la caractéristique d'être pertinent, trois possibilités existent selon les critères de sélection appliqué à l'Univers U :

1. A est vide il n'y a pas de documents pertinents,
2. $P = U$ tous les documents sont pertinents et $\phi_P(x) = 1\ \forall x \in U$ et,
3. P est un sous-ensemble propre ($P \neq U$ et $P \neq \emptyset$), dans ce cas, $\forall x \in U, \phi_P(x) = 1$ si x est pertinent et $\phi_P(x) = 0$ si x est non pertinent sont deux attributs mutuellement exclusifs définissant les deux sous-ensembles complémentaires P et $\overline{P}$.

La fonction caractéristique ϕ_P signifie donc que tout membre de la communauté U est soit pertinent, soit non pertinent, excluant par conséquent tout cas intermédiaire qui supposerait l'existence d'éléments x sur la frontière de P comme des documents partiellement pertinents. La théorie des ensembles flous sert à généraliser cette fonction d'appartenance pour des catégories vagues.

Soit un ensemble de référence (ou univers) U, un ensemble flou A dans U est défini par la donnée d'une application μ_P de U dans l'intervalle réel $[0, 1]$. A tout élément $x \in U$ est associée une valeur $\mu_P(x)$ telle que

$$0 \leq \mu_P(x) \leq 1\ (\mu_P : I \rightarrow [0, 1]).$$

L'application μ_P est appelée fonction d'appartenance de l'ensemble flou P, généralisant ainsi le concept d'appartenance et donc la notion de fonction caractéristique.

A tout élément x de U la valeur $\mu_P(x)$ associée n'est pas nécessairement égale à 0 ou à 1, elle est a priori quelconque et désigne le degré d'appartenance de x à l'ensemble P. Trois cas peuvent être distingués $\forall x \in U$:

1. $\mu_P(x) = 0$, x n'appartient pas à P, c'est-à-dire l'élément x ne satisfait pas du tout la propriété vague sous-entendu par P (par exemple x n'est pas pertinent selon le critère de définition de la catégorie P des documents pertinents),
2. $\mu_P(x) = 1$, x appartient à P, c'est-à-dire l'élément x satisfait pleinement la propriété vague définie par P (par exemple un document contenant tous les termes de la requête dans le bon contexte sémantique est pleinement pertinent),
3. $\mu_P(x) \in [0, 1]$, dans ce cas, le degré d'appartenance $\mu_P(x)$ est une valeur intermédiaire entre 0 et 1. x appartient partiellement à l'ensemble flou P, c'est-à-dire que x ne satisfait que partiellement à un certain degré $\mu_P(x)$ la propriété vague définie par P, on écrit $x \in P$ avec le degré $\mu_P(x)$ (par exemple, le document est partiellement pertinent, il ne recouvre qu'une partie de la requête).

De plus, plusieurs cas particuliers existent. Tout d'abord, l'ensemble flou vide $\emptyset$: sa fonction d'appartenance est partout nulle ou $\mu_\emptyset(x) = 0 \, \forall x \in U$. Ensuite, l'univers lui-même : sa fonction d'appartenance est la constante 1 ou $\mu_U(x) = 1 \, \forall x \in U$. Enfin, le singleton flou : $\mu_P(x) = 0$ sauf en un seul point x_0 tel que $\mu_P(x) = A, A \in [0, 1]$. Un sous-ensemble classique P : c'est un ensemble flou particulier dont la fonction caractéristique ne prend que deux valeurs 0 et 1. On a $\mu_P(x) = 1$ si $x \in P$ et $\mu_p(x) = 0$ si $x \notin P$.

Le terme ensemble flou est la traduction littérale de *fuzzy sets* en anglais. Cependant, le purisme mathématique français conduit à utiliser le terme de sous-ensemble flou. En effet, un ensemble flou n'existe pas par lui-même puisqu'il est défini par rapport à un référentiel explicite bien déterminé. Dans la suite, nous utiliserons le terme d'ensemble flou.

2.3.4.2 Modèle de document

Afin de graduer la pertinence système, une pondération des termes a été introduite aux modèles de représentation des documents. Le degré de représentativité du terme pour le document est intégré au système en le mémorisant dans le fichier inversé.

2.3.4.3 Évaluation de la pertinence

La première approche est une évaluation du score d'un document par la somme des poids des termes d'index :

$$s(d, q) = \sum_{t \in q} w_{d,t}.$$

Dans le même objectif, une solution basée sur la théorie des ensembles flous et une autre généralisant les opérations logiques ont été proposées.

Nous rappelons qu'une fonction d'évaluation de pertinence permet de calculer un score $s(q, d)$ pour le document d par rapport à la requête q. Cette dernière est triviale pour les feuilles de l'arbre de la requête :

$$s(q, d) = w_{q,t} \cdot w_{d,t} \text{ si } q = (t, w_{q,t}).$$

Pour les nœuds internes, il faut combiner les scores des fils de ce nœud.

Différentes formules sont utilisables à ce niveau, trois d'entre elles sont montrées dans la table 2.2 — le modèle *p-norm* cité dans cette table a été introduit dans [Salton *et al.*, 1983]. Le point essentiel dans ces formules est que si des fils (soit des feuilles, soit des arbres) sont ajoutés à un nœud OU, le score de ce nœud ne peut pas être plus faible. De même, si des fils sont ajoutés à un nœud ET, son score ne peut pas être plus élevé.

2.3.4.4 Modèle à ensembles flous

L'application de la logique floue au domaine de la recherche d'informations a conduit à la formalisation de nombreux modèles [Radecki, 1981, Bookstein, 1980, Bordogna et Pasi, 1995a, Bordogna et Pasi, 1995b, Kerre *et al.*, 1986, Lucarella et Morara, 1991, Bordogna et Pasi, 1998, Kraft *et al.*, 1999, Cross, 1994, Crestani et Pasi, 1999]. En recourant à la théorie des ensembles flous [Zadeh, 1988], le poids attribué à un terme d'indexation reflète le degré d'appartenance du terme au document [Radecki, 1976, Tahani, 1976, Buell, 1982]. Contrairement au modèle booléen, un terme peut décrire un document avec une certaine nuance. Le poids $w_{d,t}$ est pris dans l'intervalle $[0, 1]$ et non plus dans l'ensemble $\{0, 1\}$. Il représente l'importance du terme t dans le document d et traduit l'adhésion du concept sous-jacent au document. Une valeur unitaire (resp. nulle) représente une appartenance complète (resp. aucune appartenance) au sous-ensemble flou.

Dans plusieurs méthodes, le langage booléen est étendu pour permettre à l'usager de différencier l'importance des termes de la requête. Pour ce faire, un poids est attribué aux termes de la requête. Trois différents types de sémantique sont associés à ces poids numériques. La sémantique d'importance relative [Radecki, 1979, Bookstein, 1980, Yager, 1987, Sanchez, 1989] permet de relativiser l'importance des mots-clés les uns par rapport aux autres. Cependant, [Dubois et Prade, 1985] montre que l'utilisation du minimum pour l'évaluation d'une conjonction peut donner de l'importance au terme ayant le poids bas ce qui conduit à des résultats non intuitifs et contraire au désir de l'usager [Kraft et Buell, 1983]. La sémantique de seuil agit comme un sélecteur et ne permet pas de refléter l'importance des termes les uns par rapport aux autres [Buell et Kraft, 1981]. Avec la sémantique de poids idéal, une requête est une spécification des documents idéaux dans la collection [Cater et Kraft, 1987]. Dans l'approche définie dans [Bordogna *et al.*, 1991], le caractère booléen des requêtes est utilisé. Pour une requête sous sa forme normale disjonctive, chaque disjonction correspond à un ensemble de documents idéaux dans lequel l'importance d'un terme spécifié dans la requête est pris en compte mais l'importance des autres termes est aussi prise en compte. L'idée de séparation des documents en classes idéales est intéressante pour la résolution du problème de pertinence partielle inhérent au modèle booléen classique. Les quantifieurs linguistiques [Zadeh, 1983], permettant de généraliser ceux

	$q = (\text{ou}, (q_i)_i, p)$	$q = (\text{et}, (q_i)_i, p)$
sous-ensembles flous (version 1)	$s(q,d) = max_i s(q_i, d)$	$s(q,d) = min_i s(q_i, d)$
sous-ensembles flous (version 2)	$s(q,d) = 1 - \prod_i (1 - s(q_i, d))$	$s(q,d) = \prod_i s(q_i, d)$
modèle *p-norm*	$s(q,d) = (\frac{\sum_i s(q_i,d)^p}{\sum_i 1})^{1/p}$	$s(q,d) = 1 - (\frac{\sum_i (1-s(q_i,d))^p}{\sum_i 1})^{1/p}$

TAB. 2.2 – Différentes formules pour calculer $s(q,d)$ si q est soit un nœud ET, soit un nœud OU dans l'arbre de la requête des modèles booléens étendus.

de la logique classique, sont utilisés dans certaines approches. Par exemple, dans [Pasi, 1999], un language d'interrogation flexible utilisent des variables linguistiques pour la spécification par l'usager du poids des termes. Les requêtes sont évaluées à l'aide de ces quantificateurs linguistiques.

La recherche d'information dans le cadre de la logique floue s'est étendue à la rétroaction de pertinence [Pasi et Marques Pereira, 1999] et à l'utilisation de thésaurus. Une définition formelle du thésaurus flou est proposée dans [Miyamoto, 1990a]. Les thésaurus explicitent les relations entre les paires de termes dans les documents, il peuvent ainsi définir une proximité à l'échelle du document. Ils permettent de rendre l'indexation automatique plus efficace mais sont aussi utilisés pour l'aide à la formulation des requêtes [Radecki, 1976, Miyamoto, 1990b].

Le reproche qui peut être fait au modèle flou classique est le résultat de l'évaluation de la requête booléenne parfois imprévisible selon les opérations utilisées au niveau des nœuds. Néanmoins, ce modèle reste une alternative pour niveler la pertinence système et permettre le classement des documents.

2.3.4.5 Modèle *p-norm*

Dans la description du besoin d'informations, l'usager (ou le système) associe à chaque opérateur logique une valeur p influençant l'interprétation que le système doit faire de l'opérateur concerné. De même, une pondération peut être associée à chaque terme de la requête. Des interprétations plus ou moins strictes de l'opérateur sont effectuées selon la valeur attribuée à p. Par exemple, avec $p = 1$, les opérations pour ET et OU sont identiques. Enfin, d'après l'étude de Savoy [Savoy, 1997], les valeurs de p comprises entre 1 et 5 fournissent les meilleures performances.

Le manque de classement, principale limite du modèle booléen, a conduit d'une part à l'élaboration des modèles booléens étendus que nous venons de présenter, et d'autre part à celle du modèle vectoriel [Salton, 1971b].

2.3.5 Modèle vectoriel

Basé sur une intuition géométrique, le modèle vectoriel emprunte des éléments de la théorie des espaces vectoriels : l'ensemble des termes trouvés lors de l'indexation définit l'espace $< t_1, t_2, ..., t_n >$, un document et une requête sont représentés par un vecteur. Au sein de l'index, le poids des termes n'est pas binaire, ce qui permet d'établir un degré de similitude entre les documents et la requête. La correspondance entre les deux est réalisée en prenant le produit interne ou le cosinus entre les deux vecteurs. Les documents sont alors proposés à l'utilisateur selon l'ordre décroissant de degré de similitude afin de pallier le manque de classement du modèle booléen.

2.3.5.1 Modèle de document et de requête

Dans le modèle vectoriel, les documents et les requêtes sont des fonctions $T \rightarrow \mathbb{R}$. Dans le modèle d'origine, l'hypothèse d'indépendance entre les termes est faite et le poids d'un terme ne tient compte que de la fréquence tf de ce terme dans un document.

Document Un document d est représenté par un vecteur de termes de dimension n où n est la taille du vocabulaire. La valeur de chaque composante, pour un terme d'indexation t, dans ce vecteur, est constituée du poids $w_{t,d}$ attribué au moment de l'indexation du document d. Un document d peut être représenté par : $d = (w_{t_1}, w_{t_2}, ..., w_{t_n})$ dans laquelle chaque valeur w_{t_i} indique la pondération associée au terme d'indexation dans le document.

Requête De manière générale, une requête est écrite par l'utilisateur à l'aide d'un *sac* de mots, c'est-à-dire un ensemble de termes. Cependant, une requête peut être aussi une phrase ou un court paragraphe écrit en langue naturelle qui sera considéré comme un sac de termes. De plus, si le document recherché est déjà connu de l'utilisateur, son titre ou un extrait de l'ordre de quelques mots peut constituer une bonne requête.

Comme pour un document, la représentation interne d'une requête est faite sous la forme d'un vecteur de dimension n dont les composantes permettent de représenter le poids du terme dans la requête.

Dans la représentation des documents ou de la requête, beaucoup de valeurs $w_{t,d}$ ou $w_{t,q}$ sont nulles, car bien évidemment, chaque document ou requête ne peut contenir qu'une infime partie des termes du vocabulaire. Seules les valeurs différentes de zéro sont mémorisées dans le fichier inversé construit à l'indexation.

2.3.5.2 Evaluation de la pertinence

La fonction d'évaluation de la pertinence, qui combine les poids des termes dans la requête et dans les documents peut être :

– un produit interne

$$s(q,d) = \sum_t q(t) \cdot d(t),$$

– un cosinus

$$s(q,d) = \frac{\sum_t q(t) \cdot d(t)}{\sqrt{\sum_t q(t)^2} \cdot \sqrt{\sum_t d(t)^2}}.$$

D'autres mesures de similarité comme *Dice* et *Jaccard* sont aussi utilisées.

Pour implanter efficacement ces mesures dans le système de recherche d'informations, la sommation des termes peut se limiter à ceux dont la pondération n'est pas nulle, plutôt qu'à tous les termes du vocabulaire.

Le produit interne par rapport aux autres fonctions possède l'inconvénient de ne pas être normalisé. Bien souvent, la fonction s (valeurs du score) est plus généralement à valeurs dans $\mathbb{R}^+$, voire dans $\mathbb{R}$, que dans $[0,1]$. Tout de même, la conséquence directe est que les documents peuvent être classés même en fonction d'une partie des termes de la requête. La correspondance partielle est donc possible, ce qui est un avantage par rapport au modèle booléen. Dans certains systèmes, un seuil pose une limite sur le degré de similitude pour éliminer les documents de plus faible pertinence système.

Dans la première version du modèle vectoriel, seule la fréquence des termes est prise en compte pour le poids des termes dans les documents, mais par la suite, les valeurs $d(t)$ et $q(t)$, habituellement notées $w_{t,d}$ et $w_{t,q}$, sont calculées à l'aide de formules de $tf \cdot idf$. Greiff [Greiff, 2000] démontre d'ailleurs que cette fréquence d'occurrence représente un bon indicateur de la pertinence d'un document.

Bien que du point de vue théorique, l'apport des pondérations à l'indexation et dans la requête ne puisse être justifié, la pratique [Voorhees et Harman, 2000] a décelé que certaines caractéristiques devaient être prises en compte dans le poids $w_{t,d}$ pour la discrimination et le classement des documents pertinents :

– le nombre d'occurrences du terme t dans un document d c'est-à-dire la fréquence du terme $tf_{t,d}$. Elle permet de favoriser les documents possédant le plus grand nombre d'occurrences des termes de la requête, ils sont ainsi considérés potentiellement plus pertinents que les autres ;

– la fréquence documentaire df_t, c'est-à-dire le nombre de documents du système de recherche d'informations qui contiennent le terme t, ce facteur permet de discriminer les documents entre eux, en effet, un terme intervenant dans tous les documents ne permet pas d'établir un ordre de pertinence entre les documents, car il n'est pas assez discriminant ;

– la longueur des documents ;
– l'apparition des termes dans certaines parties logiques des documents (titre, résumé, titre de section, etc.).

Les deux premières caractéristiques ont conduit à de nombreuses formules de pondérations des index (cf. annexes A).

Pondération de la fréquence des termes Deux idées essentielles sont mises en œuvre pour la prise en compte de la fréquence des termes. Premièrement, la première apparition d'un terme dans un texte doit avoir plus d'importance que la deuxième, la deuxième plus que la troisième etc.. La seconde est que la différence entre une fréquence d'apparition nulle ou positive, qui reflète l'absence ou la présence du terme, doit être plus fortement marquée qu'une fréquence entre deux valeurs positive (ex. 27 et 30). Pour appliquer ces deux idées, deux techniques de pondération ont été proposées : l'équation $w_{t,d} = 0,5 + 0,5\ tf_{d,t}\ avec\ tf_{d,t} \neq 0$ attribue un poids plus important à la première occurrence, et, $w_{t,d} = \log(tf_{ij})$ [Salton et Buckley, 1988] en prenant le logarithme donne de moins en moins d'importance aux occurrences supplémentaires.

Pondération de la fréquence documentaire La fréquence documentaire df est introduite pour jouer le rôle d'un facteur discriminant. Par exemple, si 60 % des documents possèdent le même terme alors celui-ci ne permet pas de distinguer les documents potentiellement pertinents des autres. Pour éviter ce problème, le poids d'un terme est déterminé selon une formule de $tf \cdot idf$. La formule classique utilisée pour le facteur idf est $idf_i = \log \frac{n}{df_j}$, n étant le nombre à considérer pour ce terme. Ainsi, les termes présents dans peu de documents sont favorisés au détriment de ceux qui apparaissent très fréquemment à travers le corpus.

Méthode du pivot Les différentes campagnes d'évaluations ont permis la manipulation de données concrètes et hétérogènes, et ont conduit à l'ajustement des pondérations, comme le « lnu » en prenant en compte la troisième caractéristique concernant la taille du document :

$$w_{t,d} = \frac{\frac{1+\log tf_{t,d}}{1+\log \frac{l_d}{\mathrm{nt}_d}}}{(1-\mathrm{slope}).\mathrm{pivot} + \mathrm{slope}.\mathrm{nt}_d}$$

Dans la formule « lnu », $w_{t,d}$ indique le poids accordé au terme t dans le document d, la valeur attribuée aux constantes *pivot* et *slope* dépendent de la collection interrogée, nt_d indique la taille du document (mesurée en nombre de mots distincts) et l_d la longueur du document D (mesurée en nombre de mots).

Dans ce modèle, la constante *slope* indique l'importance attribuée à la longueur du document et *pivot* le nombre moyen de termes distincts apparaissant dans la représentation d'un document. Ce dernier modèle possède l'avantage de tenir compte de la longueur des documents

en cherchant à pénaliser les longs documents abordant généralement plusieurs sujets et qui répondent, en moyenne, moins bien aux attentes de l'usager.

2.3.5.3 Variantes du modèle vectoriel

Dépendance des termes Pour simplifier le calcul de similarité entre documents et requêtes, l'hypothèse d'indépendance entre les termes est faite. Cependant, cette dernière ne reflète pas la réalité car si l'on prend ne serait-ce que l'ensemble des expressions du vocabulaire, certains mots possèdent plus de chances d'apparaître directement après d'autres. Les utilisations de thésaurus [Pechoin, 1991], des co-occurrences [Church et Hanks, 1990, Losee, 2001, Van Rijsbergen, 1977, Spark Jones, 1971, Wong *et al.*, 1987], de la pseudo-classification [Salton, 1980] et de de la linguistique sont proposées dans certaines méthodes pour prendre en compte les différentes relations entre les termes.

Latent Semantique Indexing Comme la taille du vocabulaire est importante pour un corpus, et que dans le modèle vectoriel, le poids des termes dans les composantes des vecteurs de documents et de requête sont très souvent nulles, la méthode *LSI* [Deerwester *et al.*, 1990] propose de réduire la taille de l'espace vectoriel.

Rétroaction de pertinence Une variante du modèle vectoriel exploite la caractéristique itérative du processus de recherche d'informations. Du côté utilisateur, si les réponses données par le système de recherche d'informations ne correspondent pas à l'attente, celui-ci va modifier sa requête en supprimant ou ajoutant de nouveaux mots-clés comme des synonymes, des qualificatifs ou bien des termes généralisant ou spécifiant le besoin d'informations. Du côté système, des techniques ont été développées pour enrichir automatiquement les requêtes en interaction ou non avec l'utilisateur. Le dialogue entre le système et l'utilisateur permet à l'utilisateur d'évaluer les documents, et d'indiquer le résultat de son évaluation au système. La technique de rétroaction de pertinence de Rocchio [Salton, 1971a] prend en compte les termes des documents jugés pertinents et non pertinents par l'utilisateur. Une nouvelle requête est construite en diminuant le poids des termes des documents non pertinents et en augmentant le poids des termes provenant des documents jugés pertinents. Sur ce principe de base, d'autres méthodes ont été mises en œuvre [Salton et Buckley, 1990, Qui et Frei, 1993] et apportent une amélioration de 10 à 30 % en termes de précision. D'autres méthodes procèdent automatiquement, les premiers documents, considérés pertinents, fournissent des termes pour l'enrichissement de la requête [Buckley *et al.*, 1995, Robertson, 1990, Robertson et Walker, 1997]. Celui-ci peut aussi être réalisé à l'aide de thésaurus, de termes sémantiquement reliés comme des synonymes [Nie et Brisebois, 1996] ou de termes provenant de l'analyse du corpus [Church et Hanks, 1990, Small, 1973, Smadia, 1993, Spark Jones, 1971] (co-occurrences).

2.3.6 Modèle probabiliste

Au début des années 60, le premier modèle de recherche d'informations défini dans le cadre probabiliste est proposé dans [Maron et Kuhns, 1960]. La notion de pertinence et de non-pertinence d'un document à une requête est vue en terme de probabilité. Par la suite, Robertson précise ce modèle avec le « principe de classement probabiliste[14] » [Robertson, 1977]. Le modèle BIR (*Binary Independence Retrieval*) est introduit dans [Robertson et Sparck Jones, 1976]. Par ailleurs, on peut trouver les détails de ce modèle dans [van Rijsbergen, 1979, Belew, 2000]. Pour une requête q et un document d de la base documentaire, ce modèle essaie d'estimer la probabilité que l'utilisateur trouve le document pertinent. Pour une requête donnée, il existe parmi les documents de la collection un ensemble idéal de documents pertinents noté $Pert$ et tous les autres documents constituent l'ensemble des documents non pertinents $\overline{Pert}$. Les poids des termes d'index sont binaires $w_{t,d} \in \{0,1\}$ et $w_{t,q} \in \{0,1\}$. Donc, une requête q et un document d peuvent être vus comme un sous-ensemble de termes d'index. $P(Pert|\overrightarrow{d})$ est la probabilité que le document d soit pertinent pour la requête q et $P(\overline{Pert}|\overrightarrow{d})$, la probabilité qu'il ne soit pas pertinent. La fonction de similitude qui permet de ranger les documents par ordre décroissant de probabilité de pertinence est :

$$s(d,q) = \frac{P(Pert|\overrightarrow{d})}{P(\overline{Pert}|\overrightarrow{d})}$$

ce qui donne en utilisant la loi bayesienne :

$$s(d,q) = \frac{P(\overrightarrow{d}|Pert) \cdot P(Pert)}{P(\overrightarrow{d}|\overline{Pert}) \cdot P(\overline{Pert})}$$

où $P(\overrightarrow{d}|Pert)$ est la probabilité que le document d soit tiré de l'ensemble des documents pertinents et $P(Pert)$ la probabilité qu'un document pris dans la collection entière soit pertinent. Or $P(Pert)$ et $P(\overline{Pert})$ possèdent les mêmes valeurs dans toute la collection donc :

$$s(d,q) \ \alpha \ \frac{P(\overrightarrow{d}|Pert)}{P(\overrightarrow{d}|\overline{Pert})}$$

Pour poursuivre les calculs dans ce modèle, l'hypothèse d'indépendance des termes est faite, c'est-à-dire que si le terme t apparaît, un terme t' qui suit souvent le terme t (car t et t' forment une expression de la langue) possède la même probabilité d'apparition qu'un terme t'' qui ne suit pas couramment t dans cette langue. En effet, prenons un document qui contient l'expression « bateau à voile », on sait que si « bateau » apparaît dans le document alors « voile » a plus de chances d'apparaître ensuite, que la plupart des autres mots du langage. Cette distinction n'existe pas dans ce modèle, « voile » et, par exemple, « voiture », possèdent la même probabilité d'apparition. Il reste à estimer ces probabilités : le document est décomposé en un ensemble d'évènements représentant la présence ou l'absence d'un terme, par exemple, $P(\overrightarrow{d}|Pert) =$

[14] *Probability Ranking Principle - PRP.*

$P(w_{1,d}\, w_{2,d} \ldots w_{n,d}$. En prenant le logarithme, la fonction de similitude utilisée pour le classement des documents est donnée par :

$$s(d,q)\alpha \sum_{i=1}^{t} w_{t,q} \cdot w_{t,d} \cdot (\log \frac{P(t|Pert)}{1-P(t|Pert)} + \log \frac{1-P(t|\overline{Pert})}{P(t|\overline{Pert})})$$

Pour démarrer et permettre le calcul des scores de pertinence des documents vis-à-vis de la requête, il existe plusieurs méthodes [Baeza-Yates et Ribeiro-Neto, 1999] pour calculer les valeurs initiales de $P(t|Pert)$ et $P(t|\overline{Pert})$ comme par exemple $P(t|Pert) = 0.5$ et $P(t|\overline{Pert}) = \frac{n_t}{N}$ où n_t est le nombre de documents qui contiennent le terme t et N le nombre total de documents de la collection. Les systèmes relatifs au modèle 2-poisson [Robertson et Walker, 1994] et au modèle Okapi [Robertson *et al.*, 1994] sont les plus utilisés dans le cadre de la recherche d'informations probabiliste. La section suivante expose la fonction de correspondance Okapi que nous avons mis en œuvre dans nos expériences.

2.3.6.1 Fonction de correspondance Okapi

Parmi les méthodes qui utilisent la proximité des termes (cf. chapitre 3), certaines basent le score à la fois sur la proximité et sur le modèle Okapi. Nous détaillons ci-dessous la similarité[15] utilisée dans [Rasolofo et Savoy, 2003] ; la seconde issue de l'approche de Song et *al.* repose sur un score Okapi similaire.

Le poids d'un mot t dans le document est donné par :

$$w_t = (k1+1) \cdot \frac{tf_t}{K + tf_t}$$

où

$$K = k \cdot [(1-b) + b \cdot \frac{l}{avdl}]$$

avec l longueur du document, $avdl$ longueur moyenne des documents et, b, k, k_1 des constantes.

Par ailleurs, le poids d'un mot dans la requête par :

$$qw_t = \frac{qtf_t}{k3 + qtf_t} \cdot \log \frac{N - df_t}{df_t}$$

où qtf_t fréquence d'apparition de t dans la requête, df_t nombre total de documents contenant t et N nombre total des documents.

Le score final du modèle Okapi est obtenu par

$$RSV_{Okapi}(d,q) = \sum_i w_i \cdot qw_i.$$

[15] Il s'agit de la partie RSV_{OKAPI} du score dans [Rasolofo et Savoy, 2003].

Par conséquent, la mesure Okapi est la combinaison de trois paramètres : la taille du document, la fréquence des termes et la fréquence documentaire. De plus, la mesure BM25 basée sur le modèle Okapi est connu pour donner de très bons résultats en recherche d'informations.

2.3.6.2 Probabilité vs degré d'appartenance

Les différents modèles de recherche d'informations sont fondés sur des modèles mathématiques précis : modèle booléen, vectoriel, flou et probabiliste. Nous avons vu pourquoi un modèle flou a été introduit en recherche d'informations. Dans cette section, nous expliquons brièvement la différence entre probabilité et degré d'appartenance. Bien qu'il s'agisse dans les deux cas d'un nombre réel de l'intervalle $[0, 1]$ traduisant une incertitude, la finalité est différente.

Une probabilité est associée à une notion d'événement, le nombre traduit les chances que l'occurrence d'un événement se produise, la réalisation de cet événement est d'autant plus probable que la valeur de la probabilité est grande. Par contre, pour un degré d'appartenance, il s'agit d'une mesure de croyance par rapport à une notion vague « tiède, pertinent, beaucoup ». Par exemple, de manière consciente ou non, pour un utilisateur un document possède un degré de pertinence variable en fonction de la fréquence des termes de la requête qui le compose. Le niveau de pertinence est évalué par rapport à l'image mentale que l'utilisateur se fait sur ce qu'est un terme fréquent ou peu fréquent. De plus, pour chaque terme de la requête, cette interprétation de la fréquence est croisée entre les termes : deux termes peu fréquents et un très fréquent peuvent impliquer que l'utilisateur A trouve le document très pertinent et qu'au contraire l'utilisateur B le trouve moyennement pertinent. Il s'agit ici de quantifier la pertinence par rapport à une notion vague (nombre de termes) plutôt que d'estimer la chance qu'il a d'être pertinent.

2.4 Mesures standard d'évaluation pour comparer les systèmes

2.4.1 Collections de test

L'évaluation en recherche d'informations s'effectue en testant les systèmes sur des corpus de test. Un *corpus de test* (ou collection de test) est composé de trois éléments :

- une **collection** c'est-à-dire un ensemble de documents ;
- des **besoins d'informations** exprimés sous différentes formes, ensemble de mots-clés, paragraphes en langue naturelle, fichiers structurés, etc. ;
- des **jugements de pertinence** pour tous les besoins d'informations sur l'ensemble de la collection.

Plusieurs corpus de test sont disponibles pour mettre en œuvre des expériences. Tout d'abord en 1960, Cleverdon réalise un corpus de 1400 papiers de recherche, puis dans les années 80 pour tester le modèle vectoriel Salton et ses étudiants construisent un ensemble de corpus

CACM, CISI, INSPEC, MED et NPL. Les collections de la conférence TREC [16] sont les corpus de test de plus grande taille (18Go pour .gov, environ 500Go pour .gov2). Pour ces corpus de test, plusieurs tâches de recherche d'informations ont été créées. Par exemple, en 2005, nous avons participé à la tâche *robuste* qui s'occupe de comparer les résultats des différents systèmes en ce qui concerne les requêtes « difficiles » issues des évaluations de la tâche adhoc, précurseur en la matière lors des campagnes plus anciennes. Dans une tâche de type adhoc, la collection de test est tout d'abord indexée avec un système, ensuite, créées de manière automatique ou non, les requêtes correspondant aux besoins d'informations sont soumises, enfin les résultats obtenus sont envoyés au logiciel TREC_EVAL pour être analysés puis comparés avec d'autres systèmes. A l'occasion de la campagne d'évaluation CLEF 2005, nous avons participé à une telle tâche sur une collection en Français. Par ailleurs, il existe d'autres initiatives, comme la campagne INEX[17] qui développe des collections de test pour la recherche dans les documents structurés avec le langage XML.

2.4.2 Notion de pertinence

La pertinence est à la fois :

- une notion précise et déterministe au niveau système. Un algorithme décide de la pertinence d'un document pour une requête par un ensemble de calculs généralement fondés sur les termes contenus dans ces deux entités. Dans ce cas, la pertinence est binaire ou est donnée par une valeur dans $\mathbb{R}$ ou $[0, 1]$;
- une notion vague, complexe et subjective au niveau utilisateur. Plusieurs critères, qui ne sont pas toujours connus de l'usager lui-même, sont pris en compte. L'utilisateur apprécie la *topicalité* et *l'utilité* du document. En effet, dans le premier cas, le document doit permettre d'accéder à une information complémentaire, et dans le second cas, l'utilisateur doit retrouver les sujets de sa requête dans le document. Outre l'analyse du sens, l'utilisateur, souvent inconsciemment, applique un algorithme pour juger de la pertinence. Pour une première lecture rapide, il recherche si les termes sont présents ou non et dans quelle proportion, il peut aussi chercher si les termes utilisés dans sa requête sont proches dans le document [18].

Les jugements de pertinence des corpus de test nécessitent l'intervention d'experts de confiance, qui, pour un besoin d'informations donné, jugent de la pertinence des documents du corpus. Parfois, plusieurs experts évaluent les mêmes documents pour un même besoin d'informations, ce qui permet de modéliser les aspects subjectifs des jugements.

Cette dernière notion est exploitée par certains moteurs de recherche qui permettent aux utilisateurs de donner leur jugement pour les documents qui leur sont présentés en réponse, ainsi

[16] http://trec.nist.gov

[17] Initiative for the Evaluation of XML retrieval (INEX), http://qmir.dcs.qmw.ac.uk/INEX

[18] Dans certains moteurs de recherche, une visualisation de la page en cache est proposée avec un affichage des mots surlignés de couleurs différents.

la pertinence d'un document n'est pas établie par une seule personne mais par un ensemble d'utilisateurs qui recherchent des informations. La notion de pertinence consensuelle permet donc d'exploiter un ensemble de points de vue provenant de plusieurs individus pour aboutir à un jugement statistiquement plus solide.

Cependant, même avec une mesure de pertinence consensuelle, l'évaluation reste une tâche difficile car elle nécessite le parcours de tout le corpus par de nombreux experts pour être significative. Par exemple, au moment des premières expériences, comme celle de Lancaster en 1968 sur 1400 documents avec 221 questions, la vérification de la pertinence pouvait être accessible à une personne car celle-ci pouvait appréhender la totalité du corpus. Mais avec l'augmentation de la taille des collections, les nouveaux systèmes ne peuvent pas suivre de telles vérifications. La méthode du *pooling* est traditionnellement utilisée en recherche d'informations [Sparck Jones et van Rijsbergen, 1975, Voorhees et Harman, 2000]. Pour une requête sur plusieurs systèmes indépendants, l'union des k premiers résultats obtenus est réalisée. Les documents sont répartis afin que plusieurs personnes puissent les juger et vérifier, si nécessaire, les différences de jugements. Cette méthode est utilisée pour la campagne d'évaluation TREC[19] avec $k = 100$.

2.4.3 Comparaison des systèmes

De nombreux critères permettent l'évaluation et la comparaison des systèmes de recherche d'informations. Par exemple, le temps de réponse pour une requête, le temps nécessaire à l'indexation, la taille de l'index (souvent exprimée en pourcentage de la taille de la collection) sont des critères quantitatifs relativement utilisés. Mais le critère traditionnellement retenu pour évaluer et comparer les différents systèmes, est qualitatif et vise à vérifier la correction des réponses retrouvées par les systèmes. Cette évaluation utilise les corpus de test pour comparer les documents obtenus avec les jugements de pertinence. Les résultats de deux systèmes sont comparables s'ils sont effectués sur le même corpus de test (collection, besoins d'informations et jugements).

2.4.3.1 Rappel et précision

Les mesures de précision et de rappel permettent l'évaluation qualitative des systèmes de recherche d'informations [Kent *et al.*, 1955]. Dans un corpus de test, les documents pertinents pour une requête de test donnée sont connus, cet ensemble de documents pertinents est noté $Pert$. Quand un système évalue une requête, l'ensemble des documents retrouvés est noté $Retr$ et l'ensemble des documents retrouvés qui sont effectivement pertinents est noté $Pert_R = Pert \cap Retr$. Les mesures de *rappel* et de *précision* introduites par Kent en 1955 et décrites ci-dessous définissent un critère d'évaluation des systèmes de recherche d'informations :

[19] Text REtrieval Conference, subventionnée par le NIST, elle se déroule aux Etats-Unis.

le *taux de rappel* mesure la proportion de documents pertinents retrouvés par rapport à l'ensemble des documents pertinents connus :

$$\text{Rappel} = \frac{|Pert_R|}{|Pert|}$$

Le complémentaire de la mesure de rappel est le silence qui traduit la proportion de documents pertinents manqués :

$$\text{Silence} = 1 - \text{Rappel} = \frac{|Pert| - |Pert_R|}{|Pert|}$$

le *taux de précision* mesure la proportion de documents pertinents retrouvés par rapport à l'ensemble des documents retrouvés :

$$\text{Precision} = \frac{|Pert_R|}{|Retr|}$$

Le complémentaire de la mesure de précision est le bruit qui traduit la proportion de mauvais documents retrouvés :

$$\text{Bruit} = 1 - \text{Precision} = \frac{|Retr| - |Pert_R|}{|Retr|}$$

L'objectif de tout système est que l'ensemble $Pert \cap Retr$ approche au mieux l'ensemble des documents pertinents $Pert$.

Il est important d'utiliser ces deux mesures simultanément. En effet, une valeur de rappel maximale est obtenue en donnant en réponse, tous les documents de la base mais la précision est alors médiocre. Les deux mesures de rappel et de précision sont fortement liées, lorsque le rappel augmente, la précision diminue et inversement, de ce fait, il est important de caractériser un système en utilisant les deux mesures à la fois.

2.4.3.2 La courbe rappel/précision à 11 points

Si les résultats sont obtenus avec le même corpus de test alors des systèmes différents peuvent être comparés. Pour cela, la courbe qui donne le taux de précision à un certain niveau de rappel pour chaque requête ou en moyenne sur un ensemble de requêtes est tracée. Tous les dix pour cent, la précision moyenne est calculée, traditionnellement, on parle de la courbe de rappel précision à 11 points [Lesk et Salton, 1969]. Pour obtenir la valeur à zéro pour cent de rappel, la courbe doit être interpolée c'est-à-dire que pour deux points de rappel i, j avec $i < j$, si au point i la précision est inférieure à celle du point j alors la précision du point i devient celle du point j.

Un système de recherche d'informations ordonne l'ensemble des réponses selon le niveau de pertinence des documents[20] qu'il a calculé grâce à la fonction de correspondance :

$$rang(d) \ < rang(d') \Leftrightarrow sim(q,d) > sim(q,d')$$

avec $sim(q,d) \in \mathbb{R}$ et $Rang(d) \in \mathbb{N}^+$. Dans ce contexte, un système de recherche d'informations est plus performant s'il présente ses réponses dans un ordre qui se rapproche le plus de celui de la pertinence utilisateur. Le meilleur des cas intervient si les documents pertinents sont dans les plus hauts rangs de la liste et le cas le plus défavorable est atteint si les documents pertinents sont les derniers ou pire, inexistants dans la liste. Pour limiter le nombre de réponses, un seuil peut être défini, cependant, il doit être judicieusement choisi : trop haut il ne laisserait pas assez de documents et, trop bas il donnerait une liste trop longue de réponses. En utilisant les jugements du corpus de test et la liste ordonnée des réponses, la courbe rappel précision peut être tracée ; si le rappel est égal à 1 à la fin puisque tous les documents sont récupérés, tous les documents pertinents ont été trouvés, par contre, la précision tend vers 0 : plus le taux de rappel augmente, plus celui de la précision diminue.

2.4.3.3 R-Précision

Les systèmes peuvent être aussi comparés par rapport à une seule valeur : la R-précision. Il s'agit de la moyenne des précisions calculées après chaque document pertinent retourné par le système. Une valeur égale à 1 reflète une performance maximale pour le rappel et la précision. Si tous les documents pertinents ne sont pas retournés, la moyenne s'effectue quand même par rapport au nombre de documents pertinents qui auraient dû être retrouvés.

2.4.3.4 Précision à X documents

La précision initiale est la précision au niveau de rappel 0. C'est une valeur interpolée. Elle reflète la qualité de la partie haute de la liste de réponses. De même, cette qualité peut être appréciée à d'autres niveaux par exemple, 5, 10, 30 et 100.

2.4.3.5 P_{tr} à X documents ou taux de documents pertinents toutes requêtes à X documents

Pour certaines de nos expérimentations, nous avons introduit une mesure, P_{tr} à X, qui est le taux de documents pertinents pour toutes les requêtes de la collection de test comme par exemple les cinquante requêtes de TREC 9. Cette mesure s'exprime par :

$$P_{tr} = \frac{Pert_R\_nb\_req}{|Pert\_nb\_req|} \text{ avec } Pert\_nb\_req > 0$$

[20] On parle de *hitlist* en Anglais.

Pour ce faire, nous calculons ce taux sur trois tailles de liste de réponses. Ces tailles ne sont pas prises au hasard. Nous calculons ainsi P_{tr}

- à 100, ce qui correspond au nombre maximum de documents évalués selon la méthode du *pooling*,
- à 1000, ce qui correspond à la taille des *runs*[21] dans les campagnes d'évaluation,
- à « tous », ce qui correspond au nombre de réponses retournées par une méthode si la *run* n'est pas coupée au millième document.

Cette mesure a pour objectif de traduire le caractère de haute précision d'une méthode.

2.5 Bilan

Tout d'abord, nous avons présenté les différentes étapes du processus de recherche d'informations. La collection de données, côté système, est indexée après avoir subie un ensemble de traitements notamment sur le texte (recherche d'unités lexicales, élimination des mots vides, lemmatisation, pondération des termes, etc.). Le besoin d'informations, côté utilisateur, est transposé en langage naturel puis soumis au système sous forme de requête compréhensible par le SRI. Après la création de la représentation interne de celle-ci, l'appariement calculé par la fonction de correspondance entre la représentation interne des documents et des requêtes permet de renvoyer les documents pertinents au niveau du système. L'ensemble des documents retournés est jugé par l'utilisateur, ce dernier itère le processus autant de fois qu'il le désire pour aboutir à l'information qu'il recherche. Ensuite, nous avons expliqué les principaux modèles sur lesquels reposent les systèmes de recherche d'informations (booléen, booléen étendu, vectoriel et probabiliste). Nous avons finalement présenté les mesures que nous utiliserons dans le chapitre 5 pour analyser les résultats de nos expérimentations. Le chapitre suivant expose l'ensemble des méthodes fondées sur la proximité que nous avons étudiées avant de construire notre propre modèle.

[21] Une *run* contient le résultat d'une expérience. Il s'agit de la liste des réponses classées par ordre de pertinence système.

Chapitre 3

La notion de proximité en recherche d'information

Table des matières

3.1 Introduction

La proximité, notion transdisciplinaire, est exploitée dans de nombreux domaines (économie, sociologie, mathématiques, etc.) [Bellet *et al.*, 1998]. L'informatique et en particulier la recherche d'informations, domaine pionnier en la matière, l'utilisent. Dès les premières applications jusqu'à nos jours, elle est perçue comme un moyen pragmatique permettant d'accroître la précision des systèmes. Dans différentes approches, associée au calcul de pertinence, elle constitue un critère supplémentaire. L'intuition que la proximité entre les termes retrouvés dans un document peut affecter la pertinence de celui-ci date de 1958 quand Luhn a écrit [Luhn, 1958] :

> « It is here proposed that the frequency of word occurrences in an article furnishes a useful measurement of word significance. *It is further proposed that the* ***relative position*** *within a sentence of words having given values of significance furnishes a useful measurement for determining the significance of sentences*. The significance factor of a sentence will therefore be based on a combination of these two measurements. »

Aujourd'hui encore, les systèmes commerciaux tels que Yahoo ou Google l'intègre dans leur moteur de recherche. Rappelons que la recherche d'informations trouve son application la plus populaire dans les moteurs de recherche du Web. Du point de vue des usages de la Toile, les internautes ne regardent en détails que les premiers résultats présentés [Spink *et al.*, 2001]. Dans ce contexte, les systèmes à haute précision sont donc préférables. Un tel système favorise le retour de documents pertinents en haut de liste, même s'il y en a peu, plutôt que le retour (ou le rappel) de *tous* les documents pertinents. L'utilisation de la proximité, peut tout à fait trouver sa place, dans ce genre de systèmes, en tant que facteur discriminant les documents, afin que ces systèmes atteignent un haut degré de précision. Par exemple, le moteur de recherche Google, connu principalement pour la notion de « popularité »[1], indique dans la documentation[2] destinée aux utilisateurs :

[1] Implémenté par l'algorithme PageRank [Brin et Page, 1998].
[2] http://www.google.fr/intl/fr/help/basics.html.

> « Par ailleurs, Google privilégie les pages dans lesquelles vos termes de recherche apparaissent aussi près que possible les uns des autres. »

Pour notre part, nous nous intéressons à l'aspect « lexical » de la proximité, notre hypothèse est la suivante : **plus les mots de la requête sont proches dans un document, plus ce document doit être jugé pertinent par le système.**

Cette utilisation est, cependant, critiquée, certains prétendant que si la fonction de correspondance utilisée au départ est efficace alors celle-ci ne peut être améliorée que dans une faible mesure [Buckley *et al.*, 1995]. Cette observation nous conduirait à constater que l'utilisation d'une stratégie supplémentaire, comme par exemple la proximité, ne permettrait pas forcément l'optimisation qualitative de la liste de réponses sauf si, à l'origine, une fonction de correspondance de mauvaise qualité était utilisée. Néanmoins, dans le cadre du Web, les stratégies utilisant les liens hypertextes ont permis d'améliorer la performance des systèmes. De plus, considérant le changement de nature des bases documentaires, notamment en termes d'accroissement du nombre des documents et d'hétérogénéité de ceux-ci, nous continuons à préconiser que la proximité demeure un atout pour la discrimination des documents. En effet, cette discrimination a fait ses preuves sous différentes formes dans des collections constituées de documents en texte intégral. Si dans une collection, les documents sont de tailles variables alors les occurrences de termes peuvent être largement dispersées dans un long document et apparaître ainsi dans un contexte sémantique complètement différent. La normalisation, prenant en compte la taille des documents pour la pondération des termes, tente de résoudre ce problème mais provoque souvent des biais, notamment celui de préférer les documents courts par rapport aux longs.

Dans ce chapitre, nous exposons les diverses approches qui se réfèrent de près ou de loin à cette notion de proximité. Nous explorons le champ de la proximité à travers différentes méthodes qui l'utilisent soit explicitement (opérateur NEAR et intervalles) soit implicitement (cooccurrences, passages, signal).

3.2 L'opérateur NEAR dans les systèmes booléens

Si l'on se tourne vers l'histoire des méthodes de recherche documentaire, l'utilisation d'opérateurs de proximité est assez ancienne. En effet, pour sélectionner des documents dans des bases de données bibliographiques, un opérateur de proximité a été introduit dans de nombreux systèmes dans les années 60.

Le principe du NEAR est d'ajouter une contrainte à l'opérateur ET. Dans le modèle booléen standard, un document est représenté comme un ensemble de mots. Un document d répond à la requête A ET B si et seulement si l'ensemble E_d des termes qui le représente contient à la fois les mots A et B, autrement dit ssi : $\{A, B\} \subset E_d$. Avec un critère de proximité dans la requête qui peut se formuler par exemple sous la forme : A NEAR B, il faudra qu'il existe au moins une occurrence de A et une occurrence de B dans le texte du document qui sont « proches ».

Le mot « proche » peut avoir différentes significations selon les systèmes : même phrase (au sens grammatical), même paragraphe (au sens typographique), distance (exprimée en nombre de mots) inférieure à un seuil, etc. Toutefois, du point de vue de la modélisation mathématique, cet opérateur n'est pas homogène avec les opérateurs strictement booléens ET, OU ; il s'applique aux mots mais on ne peut le généraliser de façon *consistante* à des locutions [Mitchell, 1973]. Nous emploierons le terme locution pour se démarquer de ce que l'on appelle expressions (*phrase* en Anglais). Une locution est définie comme un groupe de mots constituant un syntagme figé. Cette dernière peut être traduite dans une requête en utilisant l'opérateur NEAR comme dans « Les aventures de Tintin et Milou » qui n'est pas proprement dit une expression du vocabulaire mais implique fortement que les mots doivent être retrouvés proches dans un document. Nous serions tentés d'écrire la requête suivante : « aventures NEAR (Tintin NEAR Milou) », cependant, comme nous allons l'expliquer ci-dessous, il n'est pas naturel d'appliquer l'opérateur de proximité pour des éléments de la requête qui ne soient pas des termes.

3.2.1 Démonstration de l'inconsistance de l'opérateur NEAR

La confusion vient de la similarité entre les écritures associées aux opérateurs NEAR, ET et OU. En effet, le langage de requêtes permet d'écrire :

1. A ET B
2. A OU B
3. A NEAR B

mais,

1. dans les deux premiers cas, la notation A fait référence à la proposition « le document d contient le terme A », proposition qui a une valeur de vérité vraie ou fausse ;
2. dans le dernier cas, la notation A ne fait référence à aucune proposition, la proposition logique est liée à toute l'expression A NEAR B, et est « le document d contient au moins une occurence de A et une occurence de B qui sont à moins de X termes l'une de l'autre ». Cette proposition peut elle-même avoir une valeur vraie ou fausse.

En fait l'opérateur NEAR ne s'applique pas à des locutions mais seulement à des termes, et produit une proposition booléenne. Mais la similarité de notation peut laisser penser que des expressions plus complexes faisant intervenir tous ces opérateurs peuvent être écrites. La question est de donner du sens à ces expressions complexes. À noter que Mitchell en 1973 a relevé une inconsistance par rapport à l'utilisation de ces opérateurs. Mais le contre-exemple qu'il donne n'est pas très convaincant.

Prenons un premier exemple (A OU B) NEAR C qui pourrait se dire en français, « le document contient A ou B proches de C ». Ici (A OU B) est une expression booléenne s'appliquant aux deux propositions A et B. Par contre l'opérateur NEAR n'apparaît pas entre deux termes. Une idée serait de rendre NEAR « distributif » sur OU (comme Mitchell le suggère), et donc d'évaluer l'expression : (A NEAR C) OU (B NEAR C). Dans cette expression, l'opérateur NEAR apparaît

bien chaque fois entre deux termes, produisant deux expressions booléennes : (A NEAR C) et (B NEAR C). Ces deux expressions peuvent sans souci être combinées avec le connecteur booléen OU. L'expression « (A NEAR C) OU (B NEAR C) » a donc bien un sens. Cette dernière pourrait être traduite en Français par « le document contient A proche de C ou B proche de C ».

Pour le deuxième exemple, remplaçons le OU par un ET : (A ET B) NEAR C qui pourrait se dire en français, « le document contient A et B proches de C ». Comme précédemment (A ET B) est une expression booléenne s'appliquant aux deux propositions A et B. Et de même l'opérateur NEAR n'apparaît pas entre deux termes. La même idée de « distributivité » nous fait considérer l'expression (A NEAR C) ET (B NEAR C). Dans cette expression, l'opérateur NEAR apparaît bien chaque fois entre deux termes, produisant deux expressions booléennes : (A NEAR C) et (B NEAR C). Ces deux expressions peuvent de nouveau être combinées avec le connecteur booléen ET. L'expression "(A NEAR C) ET (B NEAR C)" a bien un sens et pourrait se dire en français « le document contient A proche de C et B proche de C ».

Malheureusement, ces deux « distributivités » sont inconsistantes. Voici un contre-exemple avec l'expression (A OU B) NEAR (C ET D). Ici encore une fois, la confusion vient de ce que l'opérateur NEAR est appliqué (à tort) non à des termes mais à des expressions booléennes. Essayons de « distribuer » NEAR sur ET : ((A OU B) NEAR C) ET ((A OU B) NEAR D) puis NEAR sur OU : ((A NEAR C) OU (B NEAR C)) ET ((A NEAR D) OU (B NEAR D)). Cette expression a du sens, ne faisant intervenir NEAR qu'entre des termes. Par exemple, le document :

...AC...BD...

vérifie cette expression.

Voyons ce qui se passe si on commence par « distribuer » NEAR sur OU : (A NEAR (C ET D)) OU ((B NEAR (C ET D)) puis NEAR sur ET : ((A NEAR C) ET (A NEAR D)) OU ((B NEAR C) ET (B NEAR D)) ; nouvelle expression qui a du sens, ne faisant intervenir NEAR qu'entre des termes. Mais le même document

...AC...BD...

ne la vérifie pas. D'où l'inconsistance de ces « distributivités ».

3.2.2 Utilisation du NEAR dans les systèmes

Salton *et al.* présentent ces opérateurs dans le contexte des systèmes commerciaux de recherche documentaire [Salton et McGill., 1983]. Différents travaux ont abordé les problèmes d'implémentation posés par ces opérateurs et leurs conséquences sur les index. Plusieurs expériences ont été effectuées pour évaluer l'utilité de cet opérateur afin d'augmenter la précision. Keen a tenté quelques idées pragmatiques mais sans s'appuyer sur un formalisme mathématique [Keen, 1991b, Keen, 1991a]. Il a en fait été guidé par ce qui pouvait être implémentable au dessus d'un système booléen avec un opérateur de proximité, ce qui a limité les possibilités.

Dans une autre étude, il présente la performance obtenue en terme de précision/rappel par différents logiciels utilisant l'opérateur de proximité. Ces travaux indiquent que l'usage de l'opérateur permet d'améliorer la précision pour l'ensemble des documents retournés tout en restant dans un cadre purement booléen [Keen, 1992a, Keen, 1992b].

L'opérateur NEAR a été utilisé plus récemment dans un autre cadre. Le système INQUERY [Callan *et al.*, 1992], basé sur un réseau d'inférence dans lequel différents indices participent au calcul du score de pertinence, l'un d'eux utilisant la notion de proximité. Les requêtes sont exprimées dans un langage qui fournit des opérateurs faisant intervenir les positions des occurrences de mots. Par exemple : `#3 (A, B)` indique que 3 mots, au plus, peuvent séparer une occurrence de A d'une occurrence de B.

3.3 Co-occurrences

Une co-occurence est l'apparition de deux mots dans le même document, ou dans une portion de texte plus courte. Leur calcul [Van Rijsbergen, 1977, Peat et Willett, 1991] permet de savoir si deux mots sont souvent présents ensembles dans les mêmes documents. Traditionnellement, les co-occurences sont des valeurs booléennes dans le sens où deux termes sont tous les deux ou non dans la même portion de texte. Un travail récent de Miyamoto [Miyamoto, 2003] introduit un poids dans chaque co-occurence en prenant en compte la distance (en nombre de mots intermédiaires) entre les deux occurences des termes.

Certaines approches utilisent le calcul des co-occurrences pour étendre les requêtes automatiquement. L'expansion des requêtes basée sur les co-occurrences n'est pas toujours efficace car les mots similaires qui sont retrouvés souvent ensemble ne sont généralement pas assez discriminants [Peat et Willett, 1991]. Ces mots ne permettent pas d'améliorer la recherche et conduisent à rappeler des documents tout aussi peu pertinents que les documents initialement retrouvés.

La notion de co-occurence, fournissant en quelque sorte une « proximité » au niveau du document, ne nous paraît pas assez précise pour définir la proximité entre les occurrences de mots, c'est pourquoi nous avons concentré notre étude sur d'autres approches notamment celles basées sur les intervalles délimités par les mots de la requête que nous présentons dans la section suivante.

3.4 Méthodes basées sur les intervalles de mots

Dans cette section, nous présentons d'abord trois méthodes à intervalles assez proches avant d'exposer les principes de deux autres méthodes qui tentent d'étendre ces dernières.

Dans la littérature, trois approches similaires concernant la sélection des intervalles qui contiennent les occurrences de mots de la requête, et, l'utilisation de ces intervalles dans la fonction de correspondance pour le calcul du score des documents, sont proposées :

- Clarke *et al.* [Clarke *et al.*, 2000, Clarke et Cormack, 1996, Clarke *et al.*, 1995] ;
- Hawking et Thistlewaite [Hawking et Thistlewaite, 1995, Hawking et Thistlewaite, 1996] et ;
- Rasolofo et Savoy [Rasolofo et Savoy, 2003].

Pour ces méthodes, deux modèles différents de requêtes sont utilisés : Clarke *et al.* et Rasolofo *et al.* définissent une requête comme un sac de mots (ex. « bateau voile ») alors que Hawking *et al.* l'expriment à l'aide d'un ensemble de relations de proximité associées à un coefficient d'importance, par exemple, « ((bateau, voile), 2), ((bateau, moteur), 12), ((bateau, moteur, jeanneau), 20) ».

De même, les intervalles sont sélectionnés de manières différentes : Clarke *et al.* considèrent les intervalles qui contiennent les k mots de la requête, puis ceux contenant $k-1$ mots et ainsi de suite jusqu'à 1 ; Hawking et *al.* élaborent les intervalles à partir des relations de proximité explicitement formulées dans la requête, et Rasolofo *et al.* construisent d'abord l'ensemble des paires de mots de la requête, puis considèrent les intervalles pour chaque instance de ces paires.

Chaque intervalle produit un score partiel (ou contribution) en fonction de sa longueur. Les contributions issues de chaque intervalle interviennent dans le score final attribué au document.

Dans ce qui suit, les caractéristiques relatives à la construction des intervalles et au calcul du score du document sont détaillées pour chaque méthode.

3.4.1 Méthode de Clarke et *al.*

Cette méthode a été baptisée *Cover Density Ranking* par Clarke *et al.* [Clarke *et al.*, 2000, Clarke et Cormack, 1996, Clarke *et al.*, 1995] parce qu'elle utilise pour le classement des documents « la densité de couverture » des mots-clés de la requête. Tout document contenant au moins un mot de la requête est considéré, l'ensemble des documents obtenus est classé dans l'ordre décroissant du nombre de mots de la requête retrouvés avant que l'analyse de proximité soit effectuée, la méthode sélectionne donc les documents qui possèdent des intervalles contenant les k termes de la requête, puis $k-1$ termes et ainsi de suite jusqu'à 1. Lorsque le nombre de documents, fixé d'avance, est renvoyé, le processus précédent se termine.

3.4.1.1 Découpage en intervalles

Un intervalle, noté $[p, q]$, commence à la position p et termine à la position q. L'ensemble des intervalles obtenus est noté I_{Clarke}. L'exemple ci-dessous montre à partir d'un extrait de

document quels sont les intervalles de proximité qui sont sélectionnés. Les intervalles $[2, 4]$ et $[3, 5]$ sont sélectionnés et contribuent au score du document.

Exemple : Requête = {T G V}

```
1  2  3  4  5
      ,-------,
T  T  G  V  T
   '------'
```

FIG. 3.1 – Intervalles de Clarke et *al.* pour la requête TGV.

A partir des n mots de la requête retrouvés dans le document, les intervalles sont sélectionnés de la manière suivante. Seuls les plus petits intervalles contenant les n mots sont conservés. Dans l'exemple suivant, pour la requête {T, G, V}[3], l'intervalle $[1, 4]$ contient tous les termes mais c'est le plus petit intervalle $[2, 4]$ qui est retenu.

Deux conséquences découlent de ce critère : en premier, *deux intervalles ne peuvent pas s'emboîter* et en second, *à partir d'une position, seul 1 intervalle est pris en compte,* par exemple, l'intervalle $[2, 5]$ est exclu.

Par ailleurs, *deux intervalles peuvent se chevaucher* : le mot G en position 3 dans le document appartient aux intervalles ($[2, 4]$ et $[3, 5]$). De plus, aucun seuil n'est fixé, aucune longueur maximale n'est imposée.

3.4.1.2 Calcul et attribution de score

La contribution d'un intervalle $[p, q]$ est calculée de la manière suivante :

$$c_{([p,q])} = \frac{K}{(q-p+1)} \qquad \text{si} \quad q-p+1 > K, \qquad 1 \quad \text{sinon}$$

La constante K est utilisée en tant que seuil sur la longueur d'un intervalle. Si la longueur de l'intervalle est plus petite que la valeur attribuée à cette constante (dans les expériences $K = 4$ ou $K = 16$), la contribution vaut 1. La contribution calculée est d'autant plus élevée que tous les termes sont proches.

Le score du document est finalement déterminé par :

$$S_{Clarke}(d) = \sum_{[p,q] \in I_{Clarke}} c_{([p,q])}$$

Les documents-réponses sont classés dans l'ordre décroissant du nombre de termes trouvés puis dans l'ordre décroissant des scores pour les documents possédant le même nombre de termes.

[3] Pour une question de présentation le mots-clés sont représentés par leur première lettre T, G, et V pour « train », « grande » et « vitesse ».

3.4.2 Méthode de Hawking et Thistlewaite

Une méthode similaire [Hawking et Thistlewaite, 1995, Hawking et Thistlewaite, 1996] a été développée par David Hawking et Paul Thistlewaite à peu près au même moment.

3.4.2.1 Découpage en intervalles

Une requête est un ensemble de couples, que nous notons (r, α), composés d'une relation de proximité et d'un coefficient d'importance.

Une relation de proximité r est représentée par un ensemble de termes, $\mathcal{I}(r, d)$ désigne l'ensemble des intervalles du document d qui satisfont la relation. Dans l'analyse de la requête {({T, G, V}, 1)}, nous cherchons les intervalles contenant les mots T, G et V ; un intervalle étant construit à partir de chaque occurrence d'un mot de la relation, cette méthode choisit *les plus petits intervalles*, l'intervalle $[2, 4]$ est sélectionné parmi $[2, 5]$ et $[2, 4]$ dans l'exemple ci-dessous.

Par conséquent, contrairement à la méthode précédente, *l'emboîtement est autorisé* puisque tous les intervalles possibles sont considérés (en retenant celui de longueur minimale) à partir d'un point de départ donné, ainsi l'intervalle $[2, 4]$ bien qu'étant emboîté dans $[1, 4]$ est retenu.

Par contre, nous retrouvons une similitude avec la méthode de Clarke et *al.*, *le chevauchement est autorisé* ($[2, 4]$ et $[3, 5]$). La longueur d'un intervalle est *limitée par un seuil*, exprimé selon des métriques différentes (nombre de mots ou nombre de caractères). La figure 3.2 montre les intervalles obtenus pour un extrait de document :

Exemple : Requête = `((T, G, V), 1)`

```
                1  2  3  4  5
                      ,-------,
                T  T  G  V  T
                   '------'
                '---------'
```

FIG. 3.2 – Intervalles de Hawking et *al.* pour la requête TGV.

Les intervalles $[1, 4]$, $[2, 4]$ et $[3, 5]$ participent à l'attribution du score du document.

3.4.2.2 Calcul et attribution de score

Pour une requête Q, la pertinence du document d pour la relation r s'exprime ainsi :

$$s_r(d) = \sum_{[p,q] \in \mathcal{I}(r,d)} \frac{1}{\sqrt{q - p}}$$

Pour un document d, l'ensemble des relations de proximité (α, r) contribue au score de la manière suivante :

$$S_{Hawking}(d) = \sum_{(r,\alpha)\in Q} \alpha \cdot s_r(d)$$

Les expériences menées avec le système PADRE, créé par les auteurs, utilisent des requêtes construites manuellement. L'attribution de valeurs aux coefficients α et le remplacement de certains mots de la requête par une liste de termes synonymes permet difficilement de reproduire les expérimentations originales.

3.4.3 Méthode de Rasolofo et *al.*

Développée à l'université de Neuchâtel par Yves Rasolofo et Jacques Savoy, cette dernière méthode [Rasolofo et Savoy, 2003] est une extension du modèle Okapi (cf. section 2.3.6.1) qui prend en compte la proximité des occurrences des paires de mots de la requête retrouvés dans les documents.

3.4.3.1 Découpage en intervalles

Tout d'abord, à partir des mots de la requête Q, l'ensemble des paires de mots, noté S_q est construit. Ensuite, les intervalles contenant les occurrences des deux mots de chaque paire sont sélectionnés sous certaines conditions : pour une paire s, l'ensemble des instances de cette paire de mots est noté $\mathcal{I}(s)$. *Un seuil sur la longueur maximale d'un intervalle* est défini : au maximum 4 mots séparent les deux mots de la paire pour qu'il soit retenu. *La position d'un terme peut être le point de départ de plusieurs intervalles* ($[1,3]$, $[1,4]$ et $[1,5]$). Les intervalles peuvent aussi *se chevaucher* ($[2,4]$ et $[3,5]$) et être *emboîtés* ($[1,3]$ et $[1,4]$). Pour la requête T G V, l'ensemble des paires de termes qui sont cherchées est : $\{\{T,G\},\{T,V\},\{G,V\}\}$. Les intervalles contenant les instances $\{t_i, t_j\}$ des paires de termes sont considérés.

Exemple : Requête = `T G V`

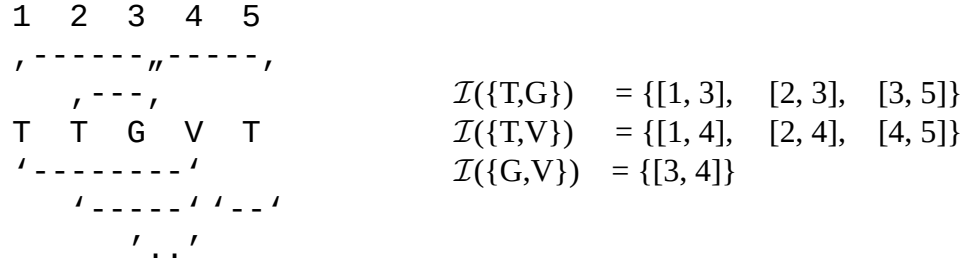


FIG. 3.3 – Intervalles de Rasolofo et *al.* pour la requête TGV.

3.4.3.2 Calcul et attribution de score

Nous détaillons ci-dessous, le score calculé par le module de proximité. Dans un document retrouvé, chaque instance d'une paire de mots s est considérée et contribue à la valeur du score du document pour cette paire de mots :

$$\forall [p,q] \in \mathcal{I}(s), \qquad w([p,q]) = \frac{1}{(q-p)^2}$$

Le poids du document pour une paire de termes est fonction de la somme du poids calculé pour chaque instance :

$$w_d(s) = (k1+1) \cdot \frac{\sum_{[p,q]\in\mathcal{I}(s)} w([p,q])}{K + \sum_{[p,q]\in\mathcal{I}(s)} w([p,q])}$$

Le score de proximité du document, noté $TPRSV$, dépend de la somme des poids obtenus pour chaque paire de termes :

$$TPRSV(d,q) = (k1+1) \sum_{s\in S_{req}} min_{x\in s}(qw(s)) \cdot \frac{\sum_{[p,q]\in\mathcal{I}(s)} \frac{1}{(q-p)^2}}{K + \sum_{[p,q]\in\mathcal{I}(s)} \frac{1}{(q-p)^2}}$$

Le score[4] d'un document est la somme du score attribué par le modèle Okapi et de celui calculé par l'analyse de la proximité, soit :

$$RSV_{NEW}(d,q) = RSV_{Okapi}(d,q) + TPRSV(d,q)$$

Finalement, les documents sont classés en fonction de la proximité des termes mais aussi grâce à la similarité Okapi qui prend en compte la pertinence d'un document dans sa globalité. Les deux méthodes que nous allons présenter dans ce qui suit prennent aussi en compte la proximité des termes en l'incluant dans un calcul global de similarité.

3.4.4 Méthode de Monz

Monz définit une méthode sur les intervalles permettant de localiser la partie d'un document où peut être trouvée une réponse dans un système de Questions/Réponses [Monz, 2003, Monz, 2004]. Il ne s'agit donc pas de la tâche traditionnelle de recherche d'informations qui retourne un ensemble de documents mais de celle qui à partir d'une question posée en langage naturel fournit une réponse. Par exemple, pour la question « Qui a posé le premier pas sur la Lune ? », le système doit renvoyer la réponse : « Neil Armstrong ». A partir d'une telle question, les mots non vides sont extraits pour constituer une requête du type « sac de termes ».

[4] RSV comme *« relevance score value »* et TPRSV *« term proximity relevance score value »*

3.4.4.1 Sélection de intervalles

Partant du constat que pour la plupart des réponses les termes de la question se trouvent très proches dans les documents, la proximité, pour Monz, est exprimée par la distance entre les termes, c'est-à-dire le nombre de mots qui apparaissent entre eux. Définir ainsi la proximité est trivial pour deux termes mais quand il s'agit de la définir pour plus de deux termes, les méthodes à intervalles que nous venons de présenter proposent une solution. Monz étend ces approches en proposant la sélection d'un seul intervalle par document : celui de taille minimale qui possède tous les termes communs entre la requête et le document. Dans ce contexte, un *matching span* est défini comme un ensemble de positions qui contient au moins une position pour chaque terme commun. C'est parmi cet ensemble de positions qu'est extrait un *minimal matching span* (mms). Ce dernier est défini comme le plus petit intervalle contenant tous les termes communs. Cette nouvelle approche que nous appelerons *« Minimal span weighting »* utilise également des propriétés générales sur le document pour le calcul du score.

3.4.4.2 Calcul du score d'un document

Le score d'un document repose sur les trois éléments suivants :

- la similarité globale entre le document et la requête qui est donnée par la fonction de correspondance décrite dans [Buckley *et al.*, 1995] (*document similarity*). Le résultat est normalisé par rapport au plus grand score obtenu par requête :

$$\mathrm{RSV}_n(q,d) = \frac{\mathrm{RSV}(q,d)}{\max_d \mathrm{RSV}(q,d)} \; ;$$

- la proportion entre le nombre d'occurrences de termes de la requête et le nombre total d'occurrences dans l'intervalle (*span size ratio*) et ;
- la proportion de termes communs entre le document et la requête (*matching term ratio*).

Les deux derniers élements sont pris en compte lorsque le nombre de termes communs est supérieur à 1 sinon le score correspond à la similarité globale du document. La proportion d'occurrences de termes (respectivement de termes différents présents) de la requête est contrôlée par le paramètre α (resp. β) tandis que les similarités globales et locales sont contrôlées par λ. Pour une requête q et un document d, le score s'exprime ainsi :

$$\mathrm{RSV}'(q,d) = \begin{cases} \lambda\,\mathrm{RSV}_n(q,d) + (1-\lambda)\left(\frac{|q\cap d|}{1+max(mms)-min(mms)}\right)^{\alpha}\left(\frac{|q\cap d|}{|q|}\right)^{\beta} & \text{si } |q\cap d| > 1 \\ \lambda\,\mathrm{RSV}_n(q,d) & \text{si } |q\cap d| = 1 \end{cases}$$

Dans les trois méthodes à intervalles précédentes, deux clés de classement, le nombre de termes communs entre requête et document puis le score sont utilisées, tandis que dans l'approche de Monz, ce nombre est directement intégré dans le calcul du score de proximité[5]. De

[5] A travers les proportions *span size ratio* et *matching term ratio*.

plus, le cas d'un « intervalle » avec un seul terme est traité comme un cas particulier, seul le score global du document est pris en compte.

Les performances du *mms* sont meilleures par rapport à la seule utilisation de la similarité globale, cependant l'application est réalisée pour les systèmes de Questions/Réponses donc aucun résultat n'est présenté pour la recherche d'informations traditionnelle, même si nous percevons intuitivement l'amélioration en terme de haute précision.

3.4.5 Méthode de Song et *al.*

L'approche de Song, Wen et Ma (1995) investit le champ de la proximité en sélectionnant un ensemble d'intervalles par document, chacun d'eux produit une contribution au score basé sur la proximité qui, lui-même, est intégré dans la formule Okapi à la place du tf [Song *et al.*, 2005]. Si l'on regarde l'évolution des méthodes, la notion de proximité n'est plus utilisée seule pour classer les documents. En effet, dans les méthodes plus récentes, le calcul du score inclut la proximité pour affiner la pertinence donnée par une similarité globale. En effet, d'abord, Monz en plus de la similarité globale entre le document et la requête insère un calcul en fonction du plus petit intervalle contenant les termes de la requête, ensuite, Rasolofo et *al.*, ont ajouté un score supplémentaire à la méthode Okapi pour refléter la proximité des occurrences des paires de termes de la requête. Enfin cette méthode propose une approche pour introduire la proximité à la place du facteur tf dans un calcul de similarité du type Okapi. Cette approche repose sur trois hypothèses :

1. plus les mots-clés apparaissent proches dans le document (la longueur de l'intervalle tend vers le nombre de mots-clés), plus le texte correspondant est pertinent pour la requête ;
2. plus le nombre d'intervalles est grand, plus le document est pertinent ;
3. plus un intervalle contient de termes différents de la requête et plus ces termes sont importants, plus le document est pertinent.

Les deux premières hypothèses caractérisent toute méthode à intervalles, tandis que la dernière est particulière à leur modèle. De plus, celle-ci est en accord avec la méthode des « phrases restreintes » de Salton [Salton *et al.*, 1993] qui, pour la construction de passages constitués des phrases clés d'un texte, exclut celles dont l'un des mots de la requête contribue à plus de 80% de la similarité. Cette dernière hypothèse se rapproche également de l'indicateur *matching term ratio* utilisé dans l'approche de Monz.

3.4.5.1 Sélection des intervalles

Cette approche propose un compromis entre les trois premières méthodes et celle de Monz car d'une part, plusieurs intervalles sont sélectionnés comme dans les méthodes de Clarke et *al.*, Hawking et *al.* et Rasolofo et *al.*, et d'autre part, la notion d'intervalle étendu est introduite

puisqu'un intervalle doit contenir le plus de mots différents comme pour la méthode de Monz, avant d'atteindre un seuil sur sa longueur. La sélection d'une position dans un intervalle s'arrête dans les cas suivants :

- si la distance entre la position du 1^{er} mot de l'intervalle et celle du mot courant est supérieur à un seuil ;
- si le mot à la 1^{ere} position est identique à celui de la position courante ;
- si le mot courant a déjà été retrouvé dans une position de l'intervalle, le plus petit intervalle est conservé.

Dans l'ensemble ainsi obtenu, les intervalles étendus ne se chevauchent pas, ne s'emboitement pas et sont limités par un seuil. Le critère majeur de sélection est qu'un intervalle puisse contenir le plus de mots différents possible : si un seuil est dépassé, même s'il s'agit d'un mot différent des précédents, un nouvel intervalle est considéré.

3.4.5.2 Calcul du score d'un document

Suite à la première et à la troisième hypothèses, la contribution d'un intervalle est calculée de la manière suivante :

$$f(t, espan_i) = (\frac{n_i}{Width(espan_i)})^x \cdot (n_i)^y$$

où t est un terme, $espan_i$ est l'intervalle étendu qui contient t, n_i est le nombre de termes qui apparaissent dans cet intervalle. $Width(espan_i)$ est la longueur de cet intervalle, cependant, la méthode retient aussi comme intervalle étendu un seul mot, dans ce cas, le seuil est utilisé comme valeur pour $Width(espan_i)$. L'exposant x permet de réduire la croissance de cette valeur et y sert à mettre en valeur les cas où le nombre de termes différents est important. Pour suivre la deuxième hypothèse, les contributions de chaque intervalle sont accumulées :

$$rc(t) = \sum_i f(t, espan_i)$$

Finalement, le facteur tf de la fonction de similarité Okapi [6] est remplacé par rc :

$$\sum_{t \in Q} w^{(1)} \frac{(k_1 + 1) \cdot rc(t)}{K + rc(t)}$$

L'approche est comparée à celle de Rasolofo et *al.* et à Okapi pour les collections de test TREC 9, 10 et 11. La précision moyenne et la précision à 5 et à 10 documents sont toujours meilleures que celles obtenues pour le modèle Okapi. Par contre, par rapport à la méthode de Rasolofo, les résultats ne sont pas forcément meilleurs mais parfois équivalents. On peut regretter que ce récent rapport ne positionne pas la méthode par rapport à celle de Monz qui possède en commun l'idée de considérer les intervalles qui possèdent le plus de termes de la requête.

[6] Nous rappelons ici la fonction de correspondance $\sum_{t \in Q} log \frac{N-n+0.5}{n+0.5} \frac{(k_1+1).tf}{K+tf}$ où $K = k_1.[(1-b) + b.\frac{1}{\text{avdl}}]$.

3.4.6 Bilan

Les approches basées sur les intervalles que nous venons de présenter, adoptent une stratégie différente pour la sélection des intervalles ainsi que pour le calcul des contributions et du score du document. Le tableau 3.1 récapitule les caractéristiques du découpage en intervalles, essentiel pour le calcul du score des documents. Ce dernier dépend des contributions (scores partiels) de chaque intervalle et finalement permet de classer les documents en fonction de la proximité des termes de la requête.

Parmi toutes ces méthodes, seule celle de Monz ne sélectionne qu'un intervalle, cependant, cette dernière repose aussi sur la proximité globale du document. D'ailleurs, nous pouvons remarquer que la tendance pour les approches les plus récentes est d'intégrer une similarité globale dans le calcul du score. Nous l'expliquons par le fait que les méthodes à base de proximité sont très sélectives donc ne permettent pas un rappel suffisant. En effet, en analysant la formule de la fonction de correspondance, nous constatons que les méthodes de Rasolofo et *al.* et de Song et *al.* complètent la liste de réponses avec des documents retenus par la méthode Okapi.

Par la suite, nous mettrons en œuvre les trois premières méthodes pour analyser l'impact de l'utilisation de la proximité en fonction de la taille des collections de documents. Nous pourrons ainsi étudier les différentes méthodes de construction de l'ensemble des intervalles ainsi que les fonctions utilisées pour le calcul des contributions. De plus, dans de futures expériences, il serait intéressant de combiner les différentes méthodes de découpages aux différentes méthodes de calcul et d'étudier dans quelle mesure nous pouvons utiliser ces méthodes avec une description booléenne du besoin d'informations.

Critère	Clarke	Hawking	Rasolofo	Monz	Song
Plus petit intervalle	Oui		Non	Oui	Oui
Chevauchement	Oui			-	Non
Emboîtement	Non	Oui		-	Non
1 intervalle par position	Oui		Non	Non	Non
Seuil de distance	Non	Oui	Oui	Non	Oui
1 intervalle par document	Non	Non	Non	Oui	Non

TAB. 3.1 – Comparatif des méthodes à intervalles (sélection)

3.5 Méthodes à passage

Parmi les deux catégories d'approches qui se basent sur des extraits de texte dans les documents pour calculer un score, nous venons de développer la première concernant les *méthodes à intervalles* où ces extraits dépendent des intervalles dans le texte contenant les mots de la requête. Pour la seconde, les méthodes travaillent sur des extraits de texte construits *a priori*,

c'est-à-dire indépendamment de la requête, dans ce cas les extraits sont appelés des *passages*, et nous parlerons de *méthodes à passage*.

3.5.1 Des bases bibliographiques au texte intégral

La motivation d'origine pour ces méthodes à passage est le changement d'échelle de la taille des documents et des collections. En effet, dans les débuts de la recherche d'informations, les tests étaient menés sur de « petites collections » telles que Adi, Cacm et Cisi où les documents étaient homogènes à l'intérieur d'une collection, en particulier en ce qui concerne leur longueur. Cependant, depuis une décennie, les capacités de stockage augmentent très vite : c'est l'essor du tout numérique. Des documents de toutes sortes sont entièrement conservés sous forme électronique : des courts (notices bibliographiques ou dépêches) ainsi que des longs (livres ou encyclopédies). Par conséquent, dans ces nouvelles bases documentaires, l'homogénéité de longueur a disparu avec l'augmentation de la taille des collections, plusieurs travaux se sont intéressés à la *recherche de passages*. Or, la recherche d'informations dans les documents longs pose deux problèmes majeurs [Salton *et al.*, 1993] : *la baisse de performance* en termes de rappel/précision, et, *la surcharge de l'utilisateur* due à la perte d'énergie et de temps pour accéder à l'information à l'intérieur du document.

Le problème de similarité avec la requête Pour ce problème, la baisse de performance est expliquée par les limites du modèle vectoriel largement utilisé jusqu'à l'apparition de ces nouveaux types de collections. En effet, le modèle vectoriel est bien adapté aux résumés d'articles qui contiennent les mots-clés relatifs aux idées générales, ces mêmes mots-clés se retrouvant d'ailleurs dans les requêtes. Par contre, avec l'augmentation la taille des documents, un paragraphe contenant les mêmes mots-clés possède moins de poids dans un document long que dans un document court. En utilisant la méthode vectorielle, le document long risque de ne pas être renvoyé à l'utilisateur, ce qui implique une baisse des performances. Autrement dit, lorsque les extraits de texte sont disponibles dans la base documentaire, la similarité avec la requête est souvent plus forte pour la portion de document que pour le document entier [Salton *et al.*, 1993].

Les méthodes basées sur les passages posent donc la question de l'amélioration de la recherche si la similarité avec la requête est calculée par rapport à une partie du document (*i.e.* un passage) plutôt que par rapport au document entier.

Faciliter la lecture des portions pertinentes Par exemple, un utilisateur ne souhaite pas qu'un système de recherche d'informations lui retourne un livre entier (document trop long) car il aurait encore à rechercher où se situe l'information qu'il désire. De plus, s'il choisit de faire une recherche séquentielle et exhaustive dans ce livre, le temps passé serait considérable ; la perte de temps qui en résulte est variable en fonction de la taille du document. Le système de recherche d'informations doit mettre en œuvre une nouvelle stratégie pour que l'utilisateur ait un accès

direct aux passages du livre qui ont été jugés pertinents par le système : c'est le second volet des méthodes à passages. L'utilisateur n'a pas à faire face à une masse importante d'informations et peut concentrer instantanément son attention sur le contenu des passages proposés. L'avantage est de réduire la masse d'informations pour que l'utilisateur puisse concentrer son attention sur le contenu pertinent. Pour ce faire, plusieurs méthodes abordent le problème du découpage d'un document en passages.

3.5.2 Construction des passages

La définition d'un passage est la clé de voûte de ces différentes méthodes. Comment définir le concept de passage, est-ce une section, un paragraphe, une fenêtre de n mots, de n phrases, un ensemble de phrases sémantiquement cohérentes ? Croft présente un tour d'horizon de toutes ces approches [Croft, 2000]. Les méthodes utilisées pour le découpage des documents peuvent être classées dans trois catégories différentes :

- **discours**, unités du discours (phrase, paragraphe, etc.),
- **fenêtre**, fenêtres de texte comportant un nombre de mots de taille fixe ou non,
- **sémantique**, segments de texte homogène du point de vue du sens. Les passages sémantiquement cohérents sont obtenus en segmentant le texte aux endroits où un changement du sujet apparaît.

Pour chacune de ces catégories, nous allons présenter les principales caractéristiques de la décomposition des documents en passages.

Discours Certains travaux utilisent les marqueurs textuels classiques comme les sections ou paragraphes [Salton *et al.*, 1993, Zobel *et al.*, 1995]. Les passages peuvent aussi être heuristiquement déduits de marques syntaxiques (passage à la ligne, ligne blanche, point final de phrase, etc.). Dans certains cas, comme celui des documents structurés, ils sont explicités grâce à des balises spécifiques [Wilkinson, 1994].

Fenêtres Dans [Kaszkiel et Zobel, 1997], les passages sont des fenêtres de taille fixe, recouvrantes ou non. Dans ce dernier cas, la taille des fenêtres est un paramètre de configuration et certaines méthodes utilisent même des fenêtres variables ou non de mots [Callan, 1994].

Sémantique Enfin, pour la troisième catégorie, la méthode la plus connue est celle du *TextTiling* [Hearst, 1997, Hearst, 1993, Hearst et Plaunt, 1993]. Elle permet de détecter les passages sur un critère thématique, critère lui-même déterminé en fonction du vocabulaire utilisé. Le texte intégral d'un document est découpé en blocs, un paramètre k contient la taille des blocs en nombre de phrases[7]. La méthode procède en deux étapes pour construire les passages similaires : (1)

[7] Choix heuristique possible : la moyenne du nombre de phrases de tous les paragraphes.

pour chaque paire de blocs adjacents, une valeur de similitude est attribuée, (2) une représentation graphique est construite avec ces valeurs, l'examen visuel des pics et des vallées de ce graphe permet de déterminer les frontières entre les passages. Des valeurs de similitude élevées indiquent que les blocs adjacents possèdent une bonne cohérence et ont tendance à former des pics sur la représentation graphique. Des valeurs de similitude basses présagent d'une frontière entre les passages et créent des vallées. Chaque bloc de k phrases est considéré comme une unité du texte où la fréquence d'un terme dans le bloc est comparée à celle dans le document entier pour déterminer s'il s'agit d'un terme local ou global, un poids fort est donné au terme s'il est fréquent dans le bloc, un poids un peu moins important lui est donné s'il est fréquent à la fois dans le bloc et dans le document, et, enfin un poids faible lui est attribué s'il est peu fréquent dans le bloc. Ainsi, deux blocs adjacents qui partagent des termes de poids importants possèdent une cohérence. En quelques sortes, la méthode fait une analyse de la répartition des termes et repose en partie sur la notion de proximité. L'examen de la similitude du vocabulaire employé permet d'aboutir à des passages de texte cohérents qui ne sont pas découpés arbitrairement.

Mitra [Salton *et al.*, 1996] définit aussi des passages et des ensembles de passages partageant des sujets similaires dans un document. D'une part, un segment de texte (*text segment*) est un morceau contigu de texte qui est lié intérieurement mais en grande partie non relié au texte adjacent par exemple, un paragraphe à caractère introductif ou le développement d'une idée et d'autre part, l'ensemble du texte d'un thème (*text themes*) est un ensemble de portions de texte sémantiquement homogènes qui traitent du même sujet. Un document peut contenir plusieurs ensembles de tels morceaux de texte, ces derniers ne sont pas nécessairement adjacents.

3.5.3 Les expériences de Wilkinson

Pour une requête donnée, les méthodes à passage permettent d'attribuer un score de pertinence pour chaque passage disponible. Wilkinson [Wilkinson, 1994] propose un ensemble de formules pour calculer le score d'un document à partir de ceux des différents passages, nous nous en sommes inspirés dans l'une de nos expériences [Mercier *et al.*, 2005]. Ces métriques combinent le score de tous les passages d'un même document pour attribuer un score au document lui-même et retourner ce dernier plutôt qu'un passage. Dans ces travaux, Wilkinson utilise un sous-ensemble de documents de la collection TREC pour étudier l'influence du découpage d'un document en passages et compare la mesure de similarité cosinus appliquée dans le cadre d'une recherche traditionnelle retournant des documents entiers, à différentes combinaisons entre les scores des passages pour attribuer le score final (cf. figure 3.4). Les expériences reposent sur les données suivantes : la fréquence des termes dans les documents et les sections, le type des sections et les inclusions entre les sections et les documents.

Le premier type d'expériences (exp. 1-5) permet de répondre à la question suivante : la recherche de documents entiers peut-elle être améliorée si les sections sont utilisées ? Un résultat de référence est construit en appliquant une mesure $tf \cdot idf$ (exp. 1) sur le texte intégral, puis les documents sont découpés en section, la mesure $tf \cdot idf$ est appliquée aux sections et les documents

La mesure du cosinus est comparée aux stratégies suivantes :

Exp. n°	Description
1	documents entiers classés selon une mesure cosinus.
2	passages découpés via les sections, le score de chaque section est obtenu par la méthode du cosinus. Le classement des documents dépend du score maximum obtenu pour une section dans chaque document.
3	expérience numéro 2 avec en plus, un poids pour le type de section.
4	expérience numéro 2, mais le n^{ieme} meilleur score est utilisé à la place du maximum.
5	combinaison linéaire des expériences 3 et 4.
6	combinaison des rangs.
7 à 8	prise en compte la normalisation.
10 à 12	prise en compte du contenu voire aussi du type des sections.
13	classement des sections via le document puis via l'expérience 11 pour les sections.
14	expérience numéro 13 avec en plus un seuil sur la taille des passages.
15 à 18	combinaison linaire des expériences 1 et 11.

FIG. 3.4 – Stratégies de combinaison des scores dans les expériences de Wilkinson

sont classés en fonction du score maximal des sections dans un document (exp. 2). Une variante (exp. 3) permet de prendre en compte le type de section (résumé, sujet, misc ou autre). Dans l'expérience 4, au lieu de prendre le score maximal des passages d'un document, les n premiers scores sont considérés. L'expérience 5 dont le score est une combinaison linéaire des expériences 3 et 4 obtient le meilleur résultat en précision à 5 documents, ensuite, l'approche traditionnelle (exp. 1) devance les autres. Inversement, le troisième type d'expériences, en utilisant le contenu (exp. 10) et le type (exp. 11) des sections, et en utilisant le document comme indicateur pour le classement (exp. 12 à 14) montre que l'utilisation des sections et de leur type (exp. 10 et 11) est presque une aussi bonne stratégie que l'approche standard.

Le deuxième et le dernier types d'expériences mettent en relief une partie de notre problématique : Wilkinson se demande si l'utilisation d'informations, globales ou locales, peut améliorer la recherche soit des sections soit des documents entiers. Les expériences 6 à 9 portent sur la combinaison des rangs et sur la normalisation des mesures de similarité pour retrouver les documents, de nettes améliorations par rapport aux premières expériences sont à noter. Par contre, les dernières expériences (exp 15 à 18), qui sont des combinaisons linéaires du résultat de référence (exp 1) et du classement des sections selon leur type et leur contenu (exp 11) ne donnent pas de meilleurs résultats. Cependant, les résultats sont à prendre avec un certain recul car seule la précision est analysée, les jugements de pertinence n'étant pas complets, Wilkinson ne fait pas de commentaires sur les mesures de rappel.

D'autres méthodes utilisent les scores des passages ainsi que le score du document lui-même, pour attribuer le score final au document [Callan, 1994] pour une requête donnée. De telles stratégies permettent d'améliorer les performances en termes de rappel/précision. Néanmoins, si le problème de similarité avec la requête, inhérent aux documents longs, est résolu par

la prise en compte de la similarité avec les passages, le problème de l'assistance de l'usager pour l'examen des documents subsiste.

3.5.4 Synthèse sur les méthodes à passages

Nous présentons une brève comparaison des différents résultats obtenus par les méthodes à passages. Nous nous posons les questions suivantes :

– quel est l'impact de la méthode de construction des passages ?
– quelle catégorie de passages pour quels types de collections ?
– finalement, l'utilisation des passages a-t-elle un sens ?

La méthode de construction des passages a bien sûr un impact sur les résultats. Par exemple, Callan et Salton et *al.* ont réalisé chacun des expériences en utilisant le découpage par section. Pour le premier, les résultats ne sont pas bons tandis que pour le second, l'amélioration est perceptible. Callan propose une explication incriminant la nature de la collection. Dans le premier cas, il s'agit d'une partie de Federal Register de TREC et dans l'autre d'une collection d'articles encyclopédiques où les auteurs prennent soin de faire correspondre structure (forme du texte) et fond (changement de sujet). Cela est sans doute moins clair pour la collection de TREC, Callan suppose que bien souvent les auteurs des documents utilisent les changements de sections par souci d'esthétique, la structure ne sert donc pas à consolider le contenu du discours. La taille des sections n'est pas consistante et ces dernières deviennent difficilement utilisables. Par conséquent, Callan utilise les sections en les fusionnant ou les divisant pour obtenir des passages de l'ordre de 300 mots et obtient ainsi de meilleurs résultats. Ceci rejoint les expériences de Kraskiel et Zobel qui montrent que l'utilisation de fenêtres de taille fixe, cette taille étant optimale entre 150 et 200, apporte de meilleurs résultats. Ces derniers ont aussi comparé cette approche avec l'implantation disponible des *Tiles* et contrairement à ce que l'on pourrait penser, la méthode basée sur le contenu n'apporte pas de résultats supérieurs. Toutefois, Callan essaie une dernière méthode, qui obtient d'ailleurs les meilleures performances, conciliant l'utilisation de fenêtres avec la notion de sémantique : le premier passage de taille n commence au premier mot de la requête trouvé, le suivant commence $\frac{n}{2}$ mots avant et ainsi de suite pour les suivants. Le recouvrement des passages réduit les risques de couper le texte au milieu d'une zone pertinente et concilie ainsi le découpage brut des méthodes basées sur les fenêtres de texte et celui plus fin de la méthode *Textiling*.

Finalement, vues les différentes méthodes de construction et la disparité des résultats, l'utilisation de passages pourrait-elle être remise en cause ? Nous ne le pensons pas car, bien que les techniques de construction des passages puissent être critiquables, les méthodes à passages permettent de résoudre les deux problèmes que nous avons évoqués ci-dessus à savoir, en premier, rendre leur poids aux portions de texte pertinentes et les faire apparaître dans les listes de réponses et, en second, faciliter le dépouillement des documents à l'utilisateur.

En conclusion, la recherche de passages peut être perçue comme une sorte d'utilisation de la proximité puisqu'elle sélectionne les passages qui concentrent de nombreuses occurrences du

maximum des termes de la requête de l'utilisateur. Ces occurrences apparaissent dans le même passage et sont donc proches les unes des autres. Par conséquent, la recherche de passages peut s'interpréter comme la découverte des endroits dans le texte où la densité des termes de la requête est la plus élevée. La section suivante présente les approches reposant sur l'idée de densité.

3.6 Méthodes basées sur la densité des mots de la requête

Les deux méthodes suivantes, qui se situent également dans le cadre de la recherche de passages possèdent la particularité de prendre en compte la répartition locale des mots de la requête dans le texte.

Les motivations de ces méthodes sont communes aux méthodes à passages. Si de nos jours, les performances des systèmes en termes de rappel/précision, efficacité et rapidité sont honorables, la présentation des résultats et l'assistance à l'utilisateur dans l'examen des réponses posent toujours problème. En introduisant des méthodes basées sur la densité des mots-clés dans les documents, ces deux approches ont pour objectif de faciliter l'examen des réponses et de surcroît, d'améliorer l'efficacité du système de recherche d'informations associé.

3.6.1 Méthode de De Kretser et Moffat

De Krester et Moffat proposent une approche basée sur la *localité* dans laquelle la similarité avec la requête est calculée pour chaque position du texte [de Kretser et Moffat, 1999a, de Kretser *et al.*, 1998]. L'avantage considérable de la méthode réside dans la présentation des documents. Du point de vue de l'efficience, les auteurs proposent aussi une nouvelle manière d'indexer les documents qui permet d'améliorer les performances pour le stockage de la position des occurrences de mots dans le texte nécessaires aux calculs de densité.

Pour que le système basé sur la localité détermine la ou les positions où il y a une grande similarité avec les mots de la requête, la collection est considérée comme une longue séquence de mots plutôt qu'un ensemble de documents. La fonction de correspondance est basée sur les deux hypothèses suivantes :

– les occurrences de mots exercent une influence sur les positions voisines,
– en une position, l'influence des différentes occurrences de mots s'additionne.

L'idée générale est donc de passer de la contribution globale d'un terme traduite par la valeur du tf dans le modèle vectoriel à une contribution locale prenant en compte les occurrences de termes influant les unes sur les autres. La similarité avec la requête est calculée localement en chaque position du texte en fonction des occurrences des mots voisins.

L'influence d'une occurrence de mot est modélisée par une fonction fenêtre $c_t(x, l)$ (ou contributive) dont trois caractéristiques sont paramétrables : la forme, la hauteur et l'étendue. Le

tableau 3.2 montre quelques exemples de fonctions utilisables, $d = |x - l|$ est la distance en mots entre l'occurrence du terme x et la position l à partir de laquelle la contribution est évaluée, et, $c_t(x, l)$ est égale à zéro quand $d = |x - l| > s_t$.

triangulaire	**cercle**
$c_t(x,l) = h_t.(1 - \frac{d}{s_t})$	$c_t(x,l) = h_t.\sqrt{1 - (\frac{d}{s_t})^2}$
cosinus	**arc**
$c_t(x,l) = h_t.\frac{(1+cos(\pi \frac{d}{s_t}))}{2}$	$c_t(x,l) = \frac{h_t}{2}.(1 - \frac{d}{s_t} + \sqrt{1 - (\frac{d}{s_t})^2})$

TAB. 3.2 – Fonctions contributives

Dans leurs propositions, les auteurs suggèrent de paramétrer ces fonctions grâce à h_t et s_t en les faisant dépendre de la fréquence documentaire.

L'étendue s_t ou largeur de la fonction détermine la plus grande distance jusqu'à laquelle le mot exerce une influence positive. Elle est normalisée par la fréquence moyenne des mots.

La hauteur h_t détermine la valeur au niveau de l'occurrence du terme. C'est une fonction de la fréquence du mot dans la requête et de la fréquence du mot dans la collection. L'insertion du facteur sur la fréquence des mots dans la requête permet d'obtenir de meilleurs résultats par rapport aux expériences antérieures [de Kretser et Moffat, 1999b].

Enfin, *la forme* des différentes fonctions c_t contributives dépend de la position l d'une occurrence de mot de la requête, de la position courante x pour laquelle est calculée la contribution, de la hauteur h_t affectée à un mot et de s_t l'étendue d'un côté du mot.

Le score de la position x est défini par $C_Q(x) = \sum_{t \in Q} \sum_{l \in I_t} c_t(x, l)$ où Q est la liste des mots de la requête et I_t est l'ensemble des positions auxquelles le mot apparaît dans la collection. Après le calcul du score pour toutes les occurrences de termes, les meilleurs positions sont sélectionnées ; une extension permet aussi de retourner des documents plutôt que des passages.

Recherche des meilleures positions A la première lecture de la requête, une table est mise en mémoire, la hauteur h_t et l'étendue s_t sont calculées pour chaque mot de la requête. Cette table pré-calculée est utilisée pour déterminer $c_t(x, l)$ qui reflète l'influence du mot sur les autres occurrences de mots de la requête. Pour réduire les coûts en termes de calcul, l'évaluation de la fonction de pertinence $C_Q(x)$ est réduite aux seules positions x où les termes de la requête apparaissent. Pour la lecture du fichier inverse, les bornes du document sont aussi mises en mémoire la liste des positions I_t pour chaque mot $t \in Q$ est lue et un tableau permet de conserver les informations sur la position du mot, son numéro et un accumulateur. Le tableau est ensuite trié par position de mot. Quand les positions sont extraites pour tous les mots de la requête, un tri partiel est réalisé et conduit à la sélection d'un nombre fixé de réponses.

Le principal inconvénient, en termes d'efficience, de la méthode basée sur la localité par rapport à celle axée sur l'approche par document réside en la mise à jour d'un plus grand nombre d'accumulateurs correspondant aux différentes positions. En effet, si nous prenons l'exemple d'une requête à trois termes, avec la méthode vectorielle, on aura simplement besoin de variables pour la fréquence documentaire et la fréquence du terme, par contre, pour une approche basée sur la localité, il faut beaucoup plus d'accumulateurs (pour chaque position au voisinage d'une occurrence) mis à jour pour toute nouvelle occurrence de terme proche d'une autre.

Recherche des meilleurs documents Pour chaque région détectée, une fenêtre centrée sur chaque position importante est présentée à l'utilisateur. Le résultat d'une session de recherche pour l'utilisateur est l'ensemble des r positions dans le texte ayant les plus grandes valeurs de $C_Q(x)$.

Les modifications nécessaires pour appliquer la méthode dans un cadre de recherche d'unité documentaire sont envisagées. Une fonction utilitaire $doc(w)$ retourne un identificateur unique pour le document qui contient le mot apparaissant à la w-ieme position dans la collection. L'influence d'une occurrence de terme doit être limitée au document $doc(x) = doc(l)$ mais aussi par $|l - x| \leq s_t$. La deuxième somme de l'équation est restreinte au domaine $l \in I_t$ et $|l - x| \leq s_t$ et $doc(x) = doc(l)$.

La méthode attribue un score pour certaines positions dans les documents. Pour obtenir un score par document, la plus grande valeur pour une position est sélectionnée puis ajoutée à l'accumulateur correspondant à son document. Ce processus termine lorsque un nombre fixé à l'avance de documents est sélectionné, les documents sont présentés selon l'ordre décroissant de ce score.

Pour comparer les résultats, il faut remonter au niveau du document. Deux alternatives ont été proposées par les auteurs : la première consiste à prendre le maximum parmi les valeurs a aux différentes positions x dans un document et a été mise en place dans des expériences préliminaires, et la seconde approche, considère la somme à la place du max et obtient de meilleurs résultats. Avec la somme, cette approche favorise ainsi l'accumulation des pertinences plutôt qu'une pertinence locale prise via le max. Les expériences réalisées montrent d'aussi bons résultats que pour l'approche par documents.

3.6.2 Méthode de Kise et *al.*

Les motivations des auteurs rejoignent celles des approches de recherche de passages. D'une part, réduire les inexactitudes des méthodes traditionnelles quand il s'agit de retrouver les documents parmi des documents longs à l'aide de requêtes courtes, et d'autre part, réduire le «fardeau» de l'utilisateur quand il s'agit de fouiller dans le document pour accéder à l'information qui l'intéresse. Pour se faire, la méthode *Density Distribution* (DD) [Kise *et al.*, 2001,

Kise *et al.*, 2004] permet de segmenter le texte en portions de taille adaptée à la réponse montrant parfois même de très petits passages de l'ordre de 10 à 50 mots.

Le problème de la pertinence partielle des documents est évoqué. Parfois, les documents ne répondant qu'à une partie de la requête, on peut se demander comment les classer pour les différents cas de figure. Prenons l'exemple d'une requête possédant trois sujets différents, R=S1, S2, S3, et deux documents D1= S1, S2 et D2=S2, S3, comment classer D1 et D2 ? Si des termes sont prépondérants dans les documents, la similarité avec la requête peut être plus grande, comme pour un document couvrant un sujet par rapport à un document couvrant deux sujets. Pour éviter ce type de réponses, le calcul du score est fondé dans cette méthode sur la répartition de tous les sujets (à travers les mots-clés correspondant) dans les documents. Pour situer les performances de leur méthode, les auteurs la compare au LSI et à la rétroaction de pertinence bien connus comme amélioration du modèle vectoriel.

L'idée fondamentale est que les parties du document qui contiennent une forte densité des mots de la requête sont pertinentes. La méthode de Kise et *al.*, comme la précédente, permet d'attribuer un score en fonction de la densité des termes de la requête. Un score est attribué à chaque position du texte où l'occurrence d'un mot de la requête apparaît. Ce score dépend de la fréquence documentaire et de la fréquence du terme. Soit $a_j(l)_{(1\leq l\leq L_{=j})}$, un mot à la position l dans le document d_j avec L_j est la longueur du document d_j en mots. La distribution pondérée $b_j(l)$ des mots de la requête q est définie par

$$b_j(l) = \begin{cases} w_{iq}.idf_i \text{ si } a_j(l) = t_{iq} \\ 0 \text{ sinon.} \end{cases}$$

Pour calculer la *Density Distribution* des mots de la requête, une fonction de lissage $dd_j(l)$ est utilisée sur $b_j(l)$ pour combiner les scores des différentes occurrences de mots dans le document :

$$dd_j(l) = \sum_{x=-\frac{w}{2}}^{\frac{w}{2}} f(x)\, b_j(l-x)$$

où $f(x)$ est la fonction fenêtre de taille w. La fonction de Hanning employée est :

$$f(x) = \begin{cases} \frac{1}{2}(1+\cos 2\pi\frac{x}{w}) \text{ si } |x| \leq \frac{w}{2} \\ 0 \text{ sinon} \end{cases}$$

Le score est obtenu en prenant la valeur maximale :

$$score(d_j, q) = \max_l dd_j(l).$$

La différence majeure par rapport à l'approche précédente est que le score d'un document est attribué en prenant le maximum des valeurs de pertinence attribuées aux positions du texte. Les documents sont donc classés en fonction de cette valeur maximum reflétant la densité des termes de la requête retrouvés dans les documents.

Les expérimentations présentées par les auteurs sont particulièrement intéressantes car plusieurs collections de test sont construites pour regarder l'impact de DD en fonction de la longueur des documents et des requêtes. Tout d'abord, les documents de la collection sont séparés en deux ensembles disjoints, (a) les documents pertinents à au moins une requête, et (b) les documents non pertinents à toutes les requêtes. Ensuite, l'ensemble des documents est fractionné en trois ensembles disjoints, les documents de petite, moyenne et grande longueur. Enfin, à partir de là, trois sous-collections sont construites : petits documents et non pertinents, documents moyens et non pertinents et, grands documents et non pertinents. En conclusion, DD est plus interessante pour les documents longs avec des requêtes courtes. Dans les travaux futurs, les auteurs envisagent d'appliquer la recherche de passages sur une collection de documents web pour développer une méthode qui détermine automatiquement la taille des fenêtres de texte.

3.6.3 Méthode de Tajima et *al.*

La méthode de Tajima propose une fonction de correspondance utilisant les liens hypertextes et la densité des termes [Tajima *et al.*, 1999]. Le problème de la définition d'une unité documentaire et de la résolution des requêtes « conjonctives » (*i.e* composées de plusieurs mots-clés) y est abordé. Pour beaucoup de systèmes, c'est le fichier HTML physique qui est considéré comme un document. Or, dans la pratique, du point de vu informationnel, un document est éclaté en plusieurs pages HTML : les requêtes composées de plusieurs mots-clés souffrent de ce phénomène car l'index, s'il ne regroupe que les informations relatives à un fichier HTML, ne contient pas les informations pour satisfaire une telle requête. Idéalement, un document devrait être une unité logique regroupant un ensemble de pages HTML.

La fonction de correspondance est composée de deux parties. La *première* prend en compte l'éclatement des documents en plusieurs fichiers physiques. En effet, la méthode permet d'extraire des sous-graphes c'est-à-dire un ensemble de pages reliées entre elles par des liens de « routage » [8] qui contiennent les mots de la requête. Ensuite, ces sous-graphes sont ordonnés par rapport à leur structure et sont normalisés par rapport à leur taille. Le score est donné par la fonction $F(G) = \sum_{v \in V}(K(v) + C)^{-1}$ où C est une constante, V est un ensemble de pages dans le sous-graphe G et $K(V)$ est le nombre de mots-clés contenus dans la page v. La *seconde*, qui nous intéresse plus particulièrement, exploite la localisation des mots-clés dans les pages des sous-graphes en s'inspirant de la méthode de la « densité apparente ». L'analyse de la densité des termes est un avantage dans le contexte du Web. Par exemple, dans certains cas, retrouver deux mots-clés dans le même fichier HTML ne constitue pas un indicateur de pertinence suffisant pour bien noter la page. En effet, une page ne véhicule pas forcément un contenu sémantique mais peut proposer un ensemble de liens ; dans ce contexte, si deux mots sont éloignés, le document possède peu de chances de correspondre au besoin d'informations formulé par l'utilisateur, d'où l'intérêt d'analyser la proximité et d'intégrer son calcul dans la fonction de correspondance.

[8] Permettant à l'internaute d'exercer une navigation standard entre les pages (liens dans le même répertoire, sous-répertoire).

Pour chaque occurrence de terme, un score normalisé est calculé en fonction de la fréquence des termes dans une fenêtre autour de sa position. L'influence d'un terme est modélisée à l'aide d'une fenêtre de Hanning :

$$h_l = \begin{cases} \frac{1}{2}(1 + cos2\pi\frac{i-l}{W}) \text{ si } |i-l| \leq \frac{W}{2} \\ 0 \text{ sinon} \end{cases}$$

où W est la taille de la fenêtre. La densité apparente d'un terme en une position du texte est définie par :

$$d_i(i) = \sum_{j=1}^{L} h_i(j).a_t(j)$$

avec L la dernière position du texte, et $a_t(x)$ égal à 1 si le terme apparaît à la position x, 0 sinon. Pour relativiser l'importance des différentes valeurs pour les occurrences par rapport aux autres positions dans le document, les valeurs sont normalisées pour que les « pics » de $d_t(i)$ soient égaux à 1 dans un document :

$$d'_t(i) = \frac{d_t(i)}{\max_{1\leq i\leq L} d_t(j)}$$

Une fois la valeur de densité de chaque terme de la requête calculée, le degré de relation entre deux termes est déterminé : $r(t_1, t_2) = \max_{1\leq i\leq L} \min(d'_{t1}(i), d'_{t2}(i))$. Les termes sont analysés deux à deux, pour chaque paire de termes, la valeur maximale est prise parmi les valeurs minimales de chaque position et constitue le degré de relation entre deux termes. La somme de ces degrés de relation constitue la partie « proximité » de la fonction de correspondance. Finalement, le score dépend, à la fois, de la nature du sous-graphe des pages extraits mais aussi de la densité des termes de la requête :

$$F(G) = \sum_{v\in V}(A.K(v) + B. \sum_{t_i,t_j\in T(v)} r(t_i, t_j) + C)^{-1}$$

avec $T(V)$ l'ensemble des termes contenus dans v et A, B, C des constantes.

Comme pour les deux méthodes précédentes, une fonction fenêtre permet de représenter la zone d'influence des termes. Cependant, la densité des termes est analysée par rapport aux paires de termes et rejoint sur ce point la méthode de Rasolofo et *al.* sélectionnant les intervalles où se trouvent les paires de termes. Par ailleurs, une analogie entre les sous-graphes de cette méthode et les sujets secondaires de la méthode à passages de Hearst peut être faite. En effet, dans la première, les unités logiques sont regroupées en utilisant les liens hypertextes tandis que dans la seconde, les zones de texte des sujets secondaires sont extraites afin d'y rechercher les informations. En conclusion, l'originalité de la méthode est d'allier l'utilisation de la proximité tout en exploitant la structure des pages Web afin d'apporter une réponse aux requêtes composées de plusieurs termes.

3.7 Méthode basée sur la transformée de Fourier

Cette approche [Park *et al.*, 2001] traite d'un regard différent le sujet de la recherche d'informations. L'accroissement incessant du volume d'informations accessibles numériquement et la popularité de la recherche sur le Web incitent des équipes de recherche provenant d'horizons différents à s'intéresser aux problématiques de la recherche documentaire afin d'y appliquer leur savoir-faire. La méthode *Fourier Domain Scoring* (FDS) possède une large connotation signal. Chaque occurrence de mot dans le texte est vue comme un signal, des opérations sur ces « signaux » conduisent au calcul du score entre documents et requêtes. En partant de la constatation classique que la pertinence système ne correspond pas toujours à la pertinence utilisateur, une méthode pour filtrer et classer les documents est proposée afin d'éliminer au mieux les documents qu'un utilisateur ne trouverait pas satisfaisants. La méthode de filtrage[9] considère un ensemble de documents et effectue une analyse des termes de la requête dans ces derniers pour isoler les documents non pertinents et classer les autres en fonction de la localisation des termes de la requête. Pour chaque document, un score est calculé à partir de la répartition spatiale des termes de la requête. Un document est découpé en plusieurs intervalles de taille identique (*bins*), un vecteur, construit pour chaque terme de la requête, contient pour chaque *bin*, la fréquence du terme dans ce dernier. Une fonction de comptage est ainsi définie :

$$cf(d,t) = c_1(d,t)/\beta \, c_2(d,t)/\beta \, ... \, c_B(d,t)/\beta$$

où $c_b(d,t)$ est la fréquence du terme t dans le *bin* b ($1 \leq b \leq B$) du document d et β est le nombre de mots par *bin*. Par exemple, pour des *bins* de taille 3, la requête `T G V` et le document `|T x x | T G V | x T x | G x G|` sont obtenus :

- le vecteur `< 1 1 1 0 >` pour le terme T,
- le vecteur `< 0 1 0 2 >` pour le terme G et,
- le vecteur `< 0 1 0 0 >` pour le terme V.

Pour examiner la position relative des termes, les vecteurs sont mis en correspondance avec un domaine de fréquences, la transformée de Fourier est appliquée sur la fonction précédente $C(d,t) = F(cf(d,t))$ soit :

$$C(d_i,t_j) = H_1^{i,j} e^{i\phi_1^{i,j}} + H_2^{i,j} e^{i\phi_2^{i,j}} + ... + H_B^{i,j} e^{i\phi_B^{i,j}}$$

où F est la transformée de Fourier, $H_B^{i,j}$ la magnitude et $e^{i\phi_b^{i,j}}$ la phase du b^{ime} *bin* du document i et du terme j.

Ce calcul permet d'observer le spectre des termes. Pour chaque terme, l'amplitude et la phase du signal, calculées à partir du vecteur, permettent d'attribuer le score de pertinence du document. Un meilleur score est attribué aux documents dont les termes apparaissent périodiquement ou bien sont proches les uns des autres en comparaison avec ceux qui sont disséminés dans tout le document. Par exemple, une expression (ensemble de termes proches) est retrouvée

[9] Il ne faut pas confondre ici la méthode avec la tâche de filtrage en recherche d'informations qui consiste à distribuer un ensemble de documents dans diverses classes en respectant des règles de filtrage.

dans le texte si les termes qui la composent, apparaissent proches les uns des autres et périodiquement, c'est-à-dire qu'un même signal périodique est observé pour l'ensemble des termes avec en plus un décalage de phases entre les termes, la proximité ainsi que leur régularité est alors détectée. La méthode FDS est appliquée à deux besoins ponctuels d'informations sur un ensemble de documents de la Toile récupérés à partir de moteurs de recherche et sur la collection de test TREC. Les résultats fournis, comparés à la similarité cosinus, produisent quelques améliorations.

L'approche FDS, reposant sur la localisation des mots-clés dans les documents, utilise directement la notion de proximité des termes. En effet, plus les mots de la requête sont proches, plus le score attribué au document est élevé, ce dernier est positionné en tête de la liste des documents retournés ce qui conduit le système à favoriser la haute précision. Le niveau de proximité mis en relief dans cette approche correspond à celui que nous voulons modéliser dans nos travaux.

3.8 Bilan

Nous avons réalisé un tour d'horizon des approches de recherche d'informations utilisant la notion de proximité. Nous pouvons dégager trois axes principaux utilisant :

- **l'opérateur NEAR.** Implanté dans certains systèmes booléens, cet opérateur ajoute une contrainte supplémentaire à l'opérateur ET pour augmenter la précision des résultats. Cependant, cet opérateur ne peut s'appliquer qu'entre deux termes et les systèmes ne renvoient que des réponses binaires, ce qui constitue deux inconvénients majeurs ;
- **les intervalles.** Ces méthodes permettent de généraliser la notion de proximité à plusieurs termes en sélectionnant puis donnant un score aux intervalles de texte contenant les mot-clés de la requête. Pour ces méthodes, la tendance est de compléter les schémas de similarité classique avec un score additionnel reposant sur la proximité ;
- **les passages/densité des termes.** Ces méthodes définissent la proximité au niveau du passage. Elles permettent de sélectionner des parties pertinentes dans les documents longs et d'assister l'utilisateur pour la lecture des documents. Certaines méthodes calculent le score des documents en fonction de la densité des mots-clés.

La section suivante expose notre approche et montre un exemple d'utilisation sur un besoin d'informations extrait d'une campagne d'évaluation.

Chapitre 4

Interpréter la requête à l'aide de la proximité

Table des matières

4.1 Motivations

4.1.1 Utilisation de la proximité

Dans le précédent chapitre, nous avons vu que depuis la naissance du domaine de la recherche d'informations, la notion de proximité était indiquée comme une piste pour déterminer la « pertinence système » des documents [Luhn, 1958]. Son utilisation est variable et plusieurs moyens sont mis en œuvre pour la prendre en compte. Tout d'abord, les *intervalles* reflètent la proximité des termes qu'ils contiennent et sont intrinsèquement liés au calcul du score des documents (méthodes de Clarke et *al.*, Hawking et *al.* et Rasolofo et *al.*), ensuite, l'utilisation de *passages* dans les documents détermine de manière implicite la proximité à l'échelle du passage et enfin, le type de *calculs sur la densité* des termes est un autre moyen d'exprimer la proximité des termes.

La présence de ces nombreuses méthodes utilisant de manière explicite ou non la notion de proximité conforte notre idée sur son efficacité en termes qualitatifs pour l'amélioration du processus de recherche d'informations. Nous situons cette amélioration tant sur le plan de la qualité des réponses que sur la prise de connaissance des résultats, pouvant être visuellement assistée.

4.1.2 La pertinence : une notion vague

Comme nous l'avons déjà dit, un objectif majeur en recherche d'informations est de refléter la pertinence d'un document. Dans la plupart des cas, la fonction de correspondance, basée sur la fréquence des termes intra-document et la fréquence documentaire, permet d'attribuer un degré global de pertinence. Cette dernière étant nuancée par ces deux facteurs n'est pas binaire. Les systèmes mettent en œuvre des algorithmes pour reproduire le jugement humain, par exemple, si un document devait être noté par rapport à un besoin d'informations, il nous serait difficile de donner une note « zéro » ou « vingt » : toute sorte de graduations dans cet intervalle conduirait à une note intermédiaire. Il est en effet plus intéressant, pour l'utilisateur, de donner au document un degré de pertinence à la place d'une pertinence « tout ou rien » ; de la même manière, les systèmes attribuent un score aux documents dans le but de les classer. La pertinence est une notion vague et subjective car de nombreux facteurs influent sur le jugement final et, comme pour toute notion vague, une modélisation floue pourrait la caractériser.

4.1.3 Similarité globale ou locale

Pour la recherche d'informations, nous pouvons qualifier la pertinence comme une notion vague. Tout d'abord, du point de vue de l'utilisateur, elle est subjective et nous supposons que

certains aspects comme l'impression générale sur le texte, la précision ou au contraire l'élargissement du besoin d'informations, la proximité entre les termes, le nombre de termes de la requête (peu, nombreux, fréquents), la présence d'exemples, etc. peuvent participer à l'évaluation de la pertinence. Plusieurs travaux cherchent à inspecter comment sont réalisés les jugements de pertinence mais nous ne nous focaliserons pas sur ces derniers car nous situons notre travail au niveau de la pertinence système et non au niveau utilisateur[1]. Considérons l'ensemble des documents, un document n'est pas, « pertinent », ou, « non pertinent », mais peut être qualifié de pertinent à un certain degré traduisant sa pertinence globale. Pour prendre en compte ces considérations, la fonction de correspondance d'un système de recherche d'informations a déjà été modélisée de manière vague (cf section 2.3.4.4). A l'opposé de la pertinence globale, pour indiquer comment le système décide du degré de pertinence d'un document, la question sur la pertinence des parties du document peut se poser. Un document est-il retourné s'il est pertinent au début, à la fin, près d'une occurrence de terme, près de l'occurrence du premier terme de la requête, etc. ? Dans ce contexte, un degré de pertinence peut être attribué à toutes les positions du texte. Une telle stratégie se rapproche ainsi des méthodes à passages (cf. section 3.5 et 3.6) et permet d'extraire les portions de texte pertinentes plutôt que l'intégralité des documents.

4.2 Choix du langage de requête

De nombreux langages tels que les « sacs de termes », le langage naturel, les expressions booléennes, les langages structurés et d'autres, plus spécifiques, adaptés aux systèmes dans lesquels ils sont utilisés, permettent d'exprimer les besoins d'informations dans les SRI. Parmi ces derniers, le langage « sac de termes » reste majoritairement choisi pour les approches basées sur la proximité (intervalles ou densité des termes). Le cas particulier de la méthode de Hawking utilise un langage plus complexe où la requête est composée de relations pondérées de proximité dans le but d'être en adéquation avec la méthode de calcul du score. Parmi les outils offrant la possibilité d'écrire comme requête des expressions booléennes, certains implantent l'opérateur NEAR pour que l'utilisateur précise si les termes (spécifiés avec NEAR deux à deux) doivent être retrouvés proches dans les documents mais l'inconvénient majeur de ces systèmes est le caractère binaire du résultat retourné. Comme d'une part, la formulation du besoin d'informations à l'aide d'expressions booléennes est assez expressive, et que d'autre part, les requêtes du type « sac de termes » sont faciles à utiliser, la création de langages hybrides ajoutant au « sac de termes » des aspects booléens a été mise en œuvre dans les moteurs de recherche. Par exemple, « Altavista » proposait un langage à « deux niveaux » permettant de sélectionner un premier ensemble de documents grâce à une requête booléenne étendue pouvant utiliser l'opérateur NEAR puis cet ensemble était reclassé avec une fonction de correspondance basée sur le modèle vectoriel. De nos jours, les moteurs Google ou Yahoo proposent, via une interface, un mode avancé d'interrogation pour que l'usager structure les informations de sa requête (cf. figure 4.1). Google offre

[1] De nombreuses études ont été réalisées, il s'agit d'un aspect important du domaine de la recherche d'informations [Korfhage, 1997, Su, 1992].

FIG. 4.1 – Extrait de l'interface de moteur Google - partie « informations » du besoin d'informations.

un langage pseudo-booléen, l'opérateur « − » permet d'exclure les documents contenant le mot qui suit, les expressions décrites entre guillemets peuvent être spécifiées ou bien un ensemble de mots facultatifs (*i.e* pouvant apparaître dans les documents) sont pris en compte. Pour le besoin d'informations concernant les activités de l'ETA en France, le système Google traduit l'expression du besoin d'informations de la figure 4.1 par la requête suivante « eta france OR francais "organisation terroriste basque" -espagne -espagnol ». Il s'agit ici de la partie informationnelle, d'autres éléments concernant les autres propriétés du document (site, format, etc.) peuvent être fournis.

De manière naturelle, nous nous tournons vers l'utilisation du NEAR pour la prise en compte de la proximité des termes. Cependant, son utilisation initiale étant dans le cadre booléen, nous allons essayer de le transposer pour quantifier le degré de proximité entre deux termes, le but étant d'intégrer la valeur obtenue dans le calcul du score d'un document.

Proximité floue Nous souhaitons « flouïfier » la notion de proximité : une première approche est de donner une interprétation floue à NEAR. Pour cela, nous modélisons un document d comme une suite finie de longueur l de termes de T, $(t_0, t_1, t_2, \ldots, t_{l-1}) \in T^l$, c'est-à-dire, une fonction $d : \mathbb{N} \rightarrow T$ dont l'ensemble de définition est un intervalle de $\mathbb{N}$ commençant en 0. Avec cette notation, $d^{-1}(t)$ désigne l'ensemble des positions où apparaît le terme t.

Si nous cherchons par exemple A et B proches, nous donnons une valeur de proximité à la requête NEAR(A,B) dans le document d avec

$$\mu_{\text{NEAR}(A,B)}(d) = \max_{\substack{i \in d^{-1}(A) \\ j \in d^{-1}(B)}} (\max(\frac{k - |j - i|}{k}, 0))$$

où k est une constante fixant la portée d'une occurrence. La valeur que nous attribuons ainsi est liée à la distance séparant les deux plus proches occurrences de A et B dans le document d. La valeur maximale est atteinte lorsque la valeur absolue $|j - i|$ est minimale. Comme A et B ne peuvent pas apparaître à la même position, on a forcément $i \neq j$. La valeur minimale de $|j - i|$ est donc 1 et est atteinte lorsqu'il y a une occurrence de A qui est voisine d'une occurrence de B dans le texte.

Opérateur de proximité et arbre de requête L'opérateur NEAR que nous venons de présenter dans son usage binaire habituel et que nous avons étendu vers une notion de proximité floue, s'applique à deux termes.

Nous avons vu dans la section 3.2 que l'opérateur NEAR ne peut pas être généralisé et ne peut pas intervenir n'importe où dans un arbre de requête du modèle booléen au même titre que les opérateurs ET et OU. Cet opérateur peut donc intervenir entre des termes mais non entre des sous-arbres. Notre modèle ne va donc pas transposer l'opérateur NEAR décrit ci-dessus. En revanche, nous pourrions généraliser la notion de proximité aux opérateurs ET et OU afin de la propager dans l'arbre de requête.

Opérateur de proximité et sac de terme Une autre solution est d'introduire cet opérateur dans un langage du type « sac de termes ». Comme nous l'avons déjà dit, il est facile de concevoir son utilisation entre deux termes mais son extension à plus de termes n'a jamais été envisagée. Les méthodes que nous avons présentées dans l'état de l'art permettent de prendre en compte la proximité des termes par le biais des intervalles ou de la densité des termes mais ne généralisent en aucun cas cet opérateur. En utilisant les sacs de termes, nous devrions nous limiter à un opérateur de proximité reliant seulement deux termes. Or dans la plupart des cas, en regardant les besoins d'informations des campagnes d'évaluation nous remarquons que dès la formulation du besoin d'informations au sein du champ « titre » plus de deux mots-clés peuvent être utilisés. Comme nous désirons appliquer la proximité à plus de deux termes, nous écartons ce type de langage. De plus, si nous utilisions ce langage, il faudrait le transformer pour permettre à l'utilisateur de spécifier la proximité ce qui changerait la nature du langage.

4.2.1 Notre choix

Si nous tirons profit des différents types d'approches, idéalement, notre langage devrait permettre d'exprimer facilement les besoins d'informations à l'aide d'expressions booléennes tout en fournissant un opérateur de proximité NEAR. Malheureusement, la généralisation de l'opérateur d'adjacence a été démontrée inconsistante dans la section 3.2.1. Par conséquent, nous utiliserons pour nos requêtes des expressions booléennes pures dans lesquelles nous allons propager un degré de proximité.

L'arbre de requête est constitué de nœuds internes auxquels sont associés les opérateurs ET ou OU, et de feuilles portant les informations sur les termes. La notion de proximité est prise en compte dans la représentation des termes dans l'arbre de requête ainsi qu'au moment de l'évaluation des nœuds. Dans ce contexte, sans l'opérateur NEAR, la notion de proximité n'est pas exprimée directement par l'utilisateur, mais reste néanmoins intégrée dans le calcul de la pertinence système. L'idée fondamentale est de fournir aux documents un score qui prend en compte la proximité des termes, par conséquent, notre fonction de correspondance n'attribue pas

directement le score en fonction d'indicateurs généraux de pertinence tels que tf ou idf mais tient compte de la pertinence à la requête en chaque position du texte.

Par exemple, la requête (**A** ET **B**) OU **C** est représentée par l'arbre de la figure 4.2.

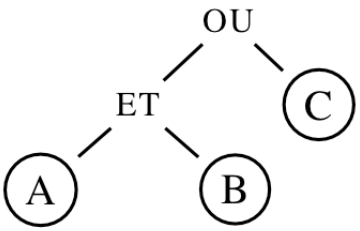


FIG. 4.2 – Arbre de la requête (**A** ET **B**) OU **C**.

4.3 Fonction de correspondance basée sur la proximité

De manière générale, la logique floue est utilisée pour modéliser des systèmes ou des phénomènes complexes (assurrance, robotique, météorologie, trafic aérien) à l'aide de systèmes à base de commande floue. Le domaine de la recherche d'informations s'attache en particulier à la modélisation et la construction de systèmes capables de décider, tout aussi bien qu'un être humain, de la pertinence de documents par rapport à un besoin d'informations. Du point de vue de l'utilisateur, beaucoup d'aspects sont pris en compte pour déterminer le niveau de pertinence d'un document. Certains critères statistiques, comme la fréquence des termes sur laquelle de nombreux systèmes reposent, peuvent en faire partie. Etant donnée la multiplicité des aspects pris en compte par l'utilisateur, on pourrait imaginer de manière utopique, un système à base de commande floue pour modéliser les critères de la pertinence utilisateur. Cependant, il appartient à chacun d'appliquer une stratégie pour décider de la pertinence et beaucoup trop d'aspects, pouvant parfois être contradictoires ou non orthogonaux, seraient à fédérer. Par conséquent, il est impossible de trouver une fonction unique qui modélise le jugement de la pertinence. De plus, seules des informations statistiques ou discrètes sont manipulables par les sytèmes. Dans notre approche, nous nous concentrons sur le critère de « proximité des termes » pour juger de la pertinence et définir notre fonction de correspondance [Mercier et Beigbeder, 2006].

4.3.1 Proximité floue à une occurrence d'un terme

Dans les modèles classiques, le critère de sélection d'un document est fondé sur l'appartenance (respectivement la fréquence) d'un terme de la requête pour le modèle booléen (respectivement vectoriel). Ces modèles procèdent avec une approche **globale** de l'influence des occurrences d'un terme sur la pertinence d'un document à une requête en utilisant le terme t. Ce qui

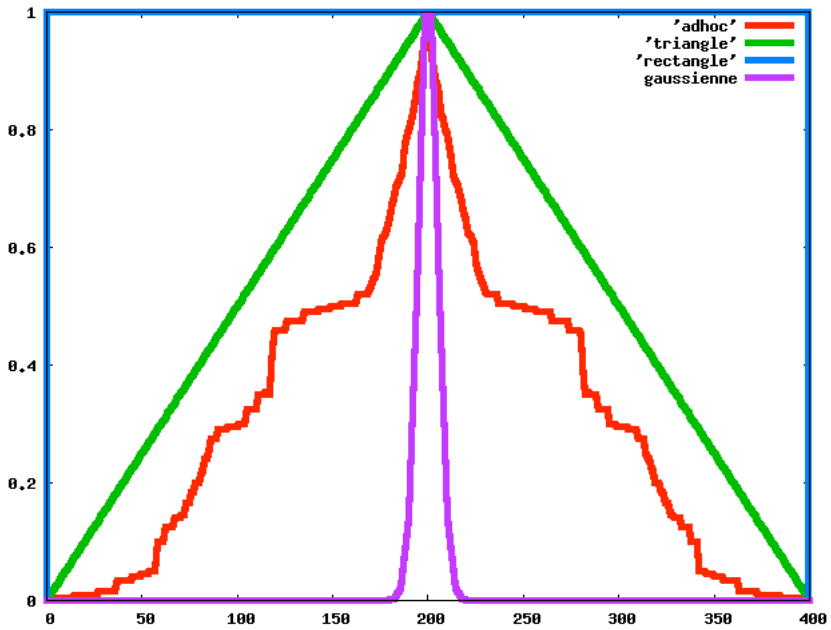


FIG. 4.3 – Fonctions d'influence (rectangle, triangle, gaussienne, adhoc)

revient à dire que la distribution des termes de la requête à l'intérieur d'un document n'intervient pas dans le calcul du score de pertinence de ce document. Cependant, le sens du texte dans un document ne dépend pas seulement du vocabulaire employé mais aussi de l'agencement des termes de ce vocabulaire et donc de la distribution de ces termes. C'est pourquoi nous adoptons une approche **locale** dans le sens où elle modélise une *influence* des occurrences. Nous définissons cette influence comme **une proximité au terme** qui permet de savoir si en un endroit du texte, on est proche d'une occurrence de ce terme. Cette proximité est graduée, et nous emploierons le terme de *proximité floue*.

Pour représenter l'influence d'un mot nous utilisons une *fonction d'influence*. Nous appelons ainsi une fonction définie sur $\mathbb{R}$, à support borné, prenant ses valeurs dans $[0, 1]$, symétrique, croissante sur $\mathbb{R}^-$, et décroissante sur $\mathbb{R}^+$. Différentes fonctions d'influence comme les fonctions de Hamming, fonctions de Hanning, fonctions gaussiennes, fonctions rectangulaires, triangulaires, etc... peuvent être utilisées (cf. figure 4.3). Dans la fonction « adhoc », trois paliers permettent de définir avec plus de précision l'impact de la proximité. Dans la partie très proche de l'occurrence la pente de la fontion est plus grande, puis elle est faible pour le quart suivant et redevient plus forte jusqu'à la valeur 0.

Nous appelons k le paramètre qui permet de contrôler la largeur de la zone d'influence. Pour une occurrence d'un terme à la position i, la translation $g(x) = f(x - i)$ d'une fonction d'influence f sert à modéliser la proximité floue. Par exemple, pour une fonction triangulaire, la valeur au point x est égale à 1 puis décroit de $\frac{1}{k}$ aux positions voisines jusqu'à atteindre la

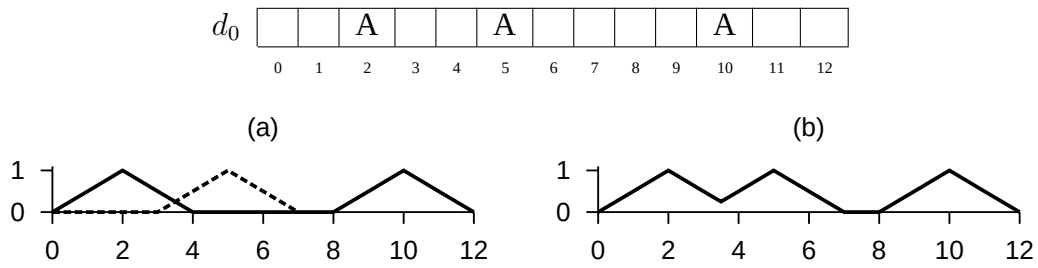


FIG. 4.4 – (a) les proximités floues aux trois occurrences, (b) la proximité floue au terme.

valeur 0. Dans ce cas, la fonction d'influence s'exprime ainsi :

$$g(x) = \max(\frac{k - |x|}{k}, 0).$$

4.3.2 Proximité floue à l'ensemble des occurrences d'un terme

Nous définissons la valeur de la proximité floue à un terme t en une position x d'un document comme la valeur de proximité floue de la plus proche occurrence du terme t. Par exemple, pour la position $x = 3$ de la figure 4.4, il est tout à fait compréhensible d'attribuer comme valeur de proximité floue celle provenant de la fonction d'influence de l'occurrence de terme la plus proche soit celle à la position 2 et non pas celle de la position 5. Comme les fonctions d'influence définies dans la figure 4.3 sont décroissantes par rapport à la distance des occurrences, en une position x du texte cela revient à prendre la valeur de proximité floue maximale et on peut poser :

$$p_t^d(x) = \max_{i \in Occ(t,d)} f(x - i)$$

où $Occ(t, d)$ est l'ensemble des positions des occurrences du terme t dans le document d et f la fonction d'influence choisie. De plus, dans un cadre général, différentes fonctions d'influence pourraient être associées à des termes différents.

Comme nous avons choisi un modèle de requête booléen, notre requête est représentée par un arbre dont les feuilles sont associées aux termes et les nœuds aux opérateurs ET et OU. Chaque feuille de l'arbre de requête représente un terme et porte donc la fonction de proximité correspondant à ce terme. Par exemple, pour la requête (A OU B), nous avons les fonctions p_A^d et p_B^d qui représentent la proximité floue des termes A et B à toutes les positions d'un document d, comme illustré par les deux premières courbes des figures 4.5 et 4.6.

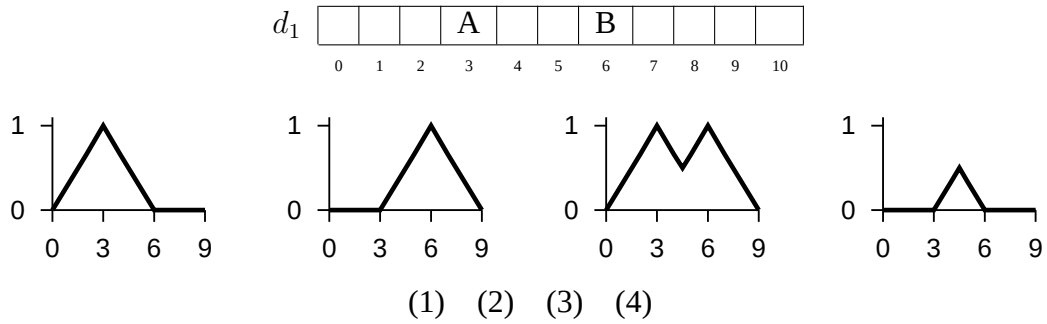


FIG. 4.5 – Document 1 – Représentation de p_A^{d1}, p_B^{d1}, $p_{A\,ou\,B}^{d1}$ et $p_{A\,et\,B}^{d1}$.

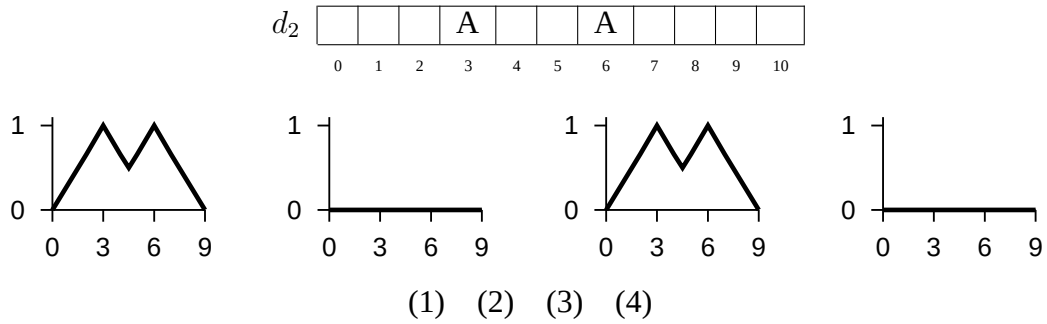


FIG. 4.6 – Document 2 – Représentation p_A^{d2}, p_B^{d2}, $p_{A\,ou\,B}^{d2}$ et $p_{A\,et\,B}^{d2}$.

4.3.3 Proximité floue à une requête

4.3.3.1 Disjonction et conjonction de termes

Nous généralisons maintenant ces fonctions sur les nœuds. Pour un nœud OU, considérons d'abord le cas de la requête (A OU B) avec deux documents, l'un contenant les deux termes A et B une fois aux positions 3 et 6 (cf. figure 4.5) et l'autre contenant deux occurrences de A aux mêmes positions (cf. figure 4.6).

Une telle requête suggère que l'utilisation de A ou de B dans le texte ait la même signification. Par conséquent, nous souhaitons obtenir la même fonction de proximité pour ces deux documents avec une requête disjonctive (comme le montre la troisième courbe des figures 4.5 et 4.6). En posant :

$$(\forall x)(p^d_{A\,\mathrm{OU}\,B}(x) = \max(p^d_A(x), p^d_B(x)))$$

cette contrainte est vérifiée et nous généralisons ceci à la requête en posant :

$$p^d_{q\,\mathrm{OU}\,q'} = \max(p^d_q, p^d_{q'})$$

pour un nœud OU, où les fils ne sont pas simplement des termes. Ceci correspond à l'opération faite dans le modèle flou classique. Par analogie, pour un opérateur ET, nous posons :

$$p^d_{q\,\text{ET}\,q'} = \min(p^d_q, p^d_{q'}).$$

Nous prenons ainsi les fonctions classiquement utilisées pour interpréter les opérateurs de logique floue. Notre modèle pourrait aussi s'adapter aux autres fonctions (cf. figure 4.1) appliquées en logique floue pour les opérateurs ET et OU. Avant d'introduire l'interprétation de l'opérateur de négation, nous expliquons comment une requête est évaluée et comment nous obtenons le score de similarité basé sur la proximité floue.

Méthode	ET	OU	NON
Zadeh	$\min(x, y)$	$\max(x, y)$	$1 - x$
Probabiliste	xy	$x + y - xy$	$1 - x$
Lukasiewicz	$\max(0, x + y - 1)$	$\min(1, x + y)$	$1 - x$
Weber	$\begin{cases} x \text{ si } y = 1 \\ y \text{ si } x = 1 \\ 0 \text{ sinon} \end{cases}$	$\begin{cases} x \text{ si } y = 0 \\ y \text{ si } x = 0 \\ 1 \text{ sinon} \end{cases}$	$1 - x$

TAB. 4.1 – Fonctions en logique floue utilisées pour l'interprétation des opérateurs

4.3.3.2 Évaluation d'une requête

L'évaluation d'une requête est effectuée en partant des feuilles. Tout d'abord, nous calculons pour chaque terme de la requête — c'est-à-dire pour les feuilles de l'arbre — la valeur de pertinence locale à chaque position x du document, c'est-à-dire la fonction p^d_t. Ensuite, nous évaluons ces valeurs au niveau de chaque nœud de l'arbre en appliquant (toujours pour chaque position x dans le document) les fonctions correspondant aux deux opérations (ET ou OU). Finalement, en remontant jusqu'à la racine, nous obtenons la fonction p^d_q qui permet de déterminer le score du document pour une requête donnée.

La figure 4.7 illustre l'évaluation des requêtes en prenant un document où apparaissent les trois termes **A**, **B**, et **C**. Les valeurs des fonctions p_A, p_B, et p_C ainsi que celles de la proximité floue de chaque requête y sont données pour les positions entre 0 et 11. Une visualisation de la proximité floue aux occurrences des termes **A**, **B**, et **C** ainsi que des requêtes **A ET B** et **(A ET B) OU C** est disponible dans la figure 4.8.

4.3.4 Attribution du score aux documents

Après le calcul de la proximité floue p^d_q, la dernière étape consiste à déterminer le score de pertinence $s(q, d)$ pour le document d et la requête q. Dans le cas du modèle booléen le score est

x	0	1	2	3	4	5	6	7	8	9	10	11	
d		A		B			C		A	B	C	C	...
$p_A^d(x)$	0.9	1.	0.9	0.8	0.7	0.7	0.8	0.9	1.	0.9	0.8	0.7	
$p_B^d(x)$	0.7	0.8	0.9	1.	0.9	0.8	0.7	0.8	0.9	1.	0.9	0.8	
$p_C^d(x)$	0.4	0.5	0.6	0.7	0.8	0.9	1.	0.9	0.8	0.9	1.	1.	
$p_{A\,\mathrm{ET}\,B}^d(x)$	0.7	0.8	0.9	0.8	0.7	0.7	0.7	0.8	0.9	0.9	0.8	0.7	
$p_{(A\,\mathrm{ET}\,B)\,\mathrm{OU}\,C}^d(x)$	0.7	0.8	0.9	0.8	0.8	0.9	1.	0.9	0.9	0.9	1.	1.	

FIG. 4.7 – Un document et les valeurs des fonctions de proximité.

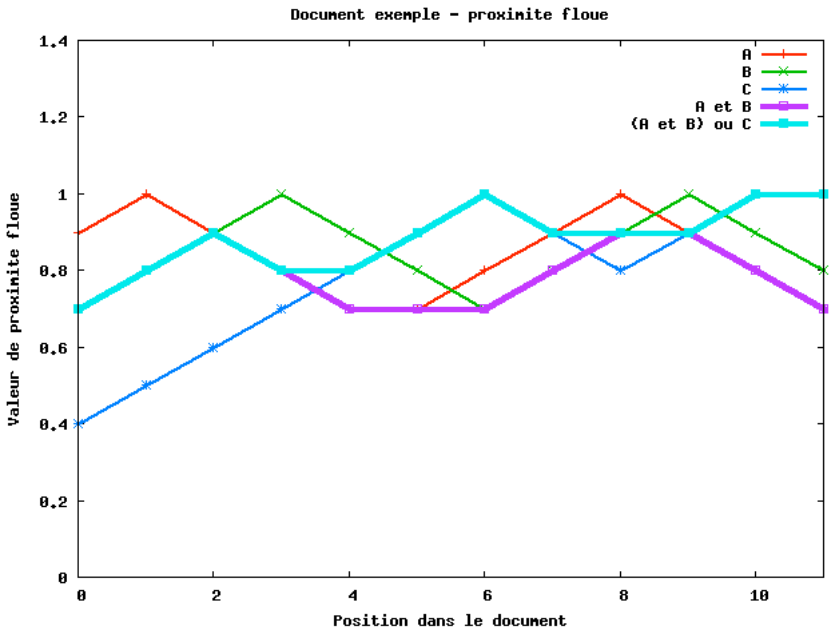


FIG. 4.8 – Visualisation des valeurs de proximité floue pour le document de la figure 4.7

binaire, c'est le résultat de l'évaluation de la requête pour le document d, par contre, pour notre approche locale, il doit refléter la proximité des termes.

Dans le modèle vectoriel, les formules de calcul de pertinence sont des produits scalaires ou des cosinus qui comportent une sommation pouvant s'interpréter comme une accumulation d'éléments de pertinence. Cette notion d'accumulation pour le calcul du score d'un document est utilisée pour prendre en compte les valeurs de proximité floue à chaque position d'un document. Les méthodes du calcul intégral permettent de mettre en œuvre cette idée en calculant la surface en dessous d'une courbe. Pour notre méthode, le score peut être représenté par une courbe (cf. figure 4.8) prenant les valeurs de proximité floue à chaque position du document, et serait exprimé de la manière suivante :

$$s(q,d) = \int_{-\infty}^{+\infty} p_q^d(x)\ \mathrm{d}x.$$

Cependant, dans nos implantations, nous calculons une valeur p_q^d pour chaque position dans le texte. Autrement dit, nous interprétons à chaque position du document la proximité locale à la requête donnée par valeur de la fonction p_q^d comme un élément de pertinence. Le score d'un document y est défini comme la sommation de tous ces éléments de pertinence, nous avons pour un document d et une requête q :

$$s(q,d) = \sum_{x\in\mathbb{Z}} p_q^d(x).$$

requête	score du document
A	10,1
B	10,2
C	9,5
A ET B	9,4
(A ET B) OU C	10,6

TAB. 4.2 – Scores de proximité floue pour le document de la figure 4.7.

Le tableau 4.2 montre les scores obtenus pour les requêtes **A**, **B**, **C**, **A ET B** et **A ET B OU C** appliqué au document de la figure 4.7.

Finalement, le score obtenu appartient ainsi à $\mathbb{R}^+$ et permet de classer les documents par ordre décroissant en fonction de la proximité des termes de la requête.

4.3.5 Ajout de la négation de termes

L'opérateur NON offre la possibilité d'exclure une partie des objets d'un ensemble qui répondent à un certain critère. Il s'agit de l'utilisation naturelle de cet opérateur. Par exemple,

pour le besoin d'informations vu dans la section 1.3.1, si pour un ensemble de documents traitant de l'organisation, les documents relatifs à ses activités en Espagne sont à enlever, ce besoin est précisé dans la requête suivante : « `!espagne & (eta | (organisation & terroriste & basque)) & france` ». Cet opérateur permet ainsi d'augmenter la précision du système. Par contre, lorsqu'il est tout seul, l'utilisation de l'opérateur NON, devient moins naturelle.

En effet, cet opérateur n'a de véritable sens que s'il permet d'éliminer un certain nombre de documents parmi d'autres pour favoriser la précision. Son utilisation seule pose en effet problème car le résultat d'une requête comme $\neg A$ peut renvoyer un ensemble de réponses tellement nombreuses qu'un utilisateur ne puisse pas les mettre à profit. De plus, pour une telle requête, une fonction de classement des documents ne pourrait pas être basée sur un plan informationnel puisque l'utilisateur demande de la « non information ». En effet, comment ordonner les documents qui répondent à une requête du type $\neg A$? Peut-être pourrait-on les classer sur les critères physiques des documents comme la taille, le nombre de mots distincts ou le nombre de paragraphes mais il est impossible de les classer sur des aspects plus sémantiques. La seule application d'une telle requête pourrait se comprendre si le seul critère d'une recherche est une « non information » à condition que la masse de données recherchées puisse être analysée par un ensemble d'individus. Par exemple, dans une enquête de renseignement, un individu fiché pourrait être recherché sur le seul critère négatif de ne pas avoir les yeux bleus. Dans le modèle booléen classique, seul le problème du nombre important de documents retournés se pose puisque, la réponse étant binaire, le problème du critère pour le classement n'existe pas.

De la même manière, la requête ($\neg A$ OU $\neg B$) et plus généralement toute disjonction de négations de terme ne possède pas réellement de sens puisque l'évaluation de la négation de chaque terme nous ramène au cas précédent. Pour l'introduction de l'opérateur de négation, nous ne retenons que l'intérêt de la première utilisation, c'est-à-dire la possibilité d'amélioration de la précision des réponses.

Bien que cet opérateur ne soit pas couramment utilisé par les utilisateurs des moteurs de recherche du Web, les utilisateurs expérimentés en font un réel usage. Cependant, il nous manque la connaissance sur la transformation du besoin d'informations en expression booléenne, ce qui serait intéressant pour la construction automatique de requêtes booléeennes dans le cadre des campagnes d'évaluation. Il est dommage que la négation ne soit pas plus utilisée car le langage booléen possède une telle expressivité que l'utilisateur courant pourrait transposer avec précision et rigueur son besoin d'informations. Intuitivement, permettre l'utilisation de cet opérateur se justifie largement car de nombreux besoins d'informations dans le domaine juridique ou technique nécessitent efficacité et fiabilité en termes de précision. Par exemple, les recherches de jugements de référence ou de lois nécessitent des critères précis car dans ce domaine, le travail de documentation prend une part très importante dans le traitement des dossiers. Peu de systèmes dédiés existent, or les besoins d'informations sont très spécifiques, l'introduction de l'opérateur NON permet de préciser des éléments de la requête et de s'adapter étroitement au besoin d'informations. La tâche *legal* sera inscrite dans la nouvelle édition de la campagne d'évaluation TREC 2006, un tel système pourra y être testé. Par ailleurs, l'utilisation de l'opérateur NON peut se

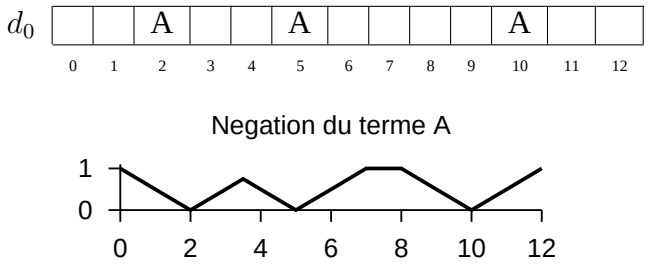


FIG. 4.9 – Proximité floue de la négation du terme A.

révéler efficace pour atténuer le problème de la polysémie comme par exemple dans la requête « java ET (! langage) ».

Dans notre cas, un autre problème se pose pour l'ajout de cet opérateur. En effet, pour déterminer le score d'un document, nous utilisons l'intégrale, or pour $\neg A$ comme requête simple la valeur tend vers l'infini. Ce problème peut être résolu en se bornant de chaque côté à la longueur du document additionné de $\frac{k}{2}$. Néanmoins, une telle solution implique que le document vide soit classé comme moins pertinent qu'un document ayant une ou plusieurs occurrences du terme devant être exclu. Mais cela a-t-il une importance puisque par définition le document vide ne véhicule aucune information exploitable mis à part le fait qu'il est vide. Cependant, pour éviter cette configuration, l'intégrale peut être calculée aux limites du document lui-même. La fonction $l(d)$ permet d'obtenir sa longueur, le score s'exprime alors : $s(q,d) = \int_0^{l(d)} p_q^d(x)\ \mathrm{d}x$ ce qui donne dans notre implantation pour un document d et une requête q :

$$s(q,d) = \sum_{x=0}^{l(d)} p_q^d(x).$$

Nous introduisons donc l'opérateur de négation à notre modèle. Un nœud NON est évalué avec le complément, classiquement utilisé en logique floue :

$$\text{NON}\, p_q^d = 1 - p_q^d.$$

La figure 4.9 montre la pertinence floue pour la négation du terme A dans le document d_0. Les figures 4.10 et 4.11 reflètent la négation des termes A et B ainsi que des requêtes (A OU B) et (A ET B) pour les documents d_1 et d_2.

4.4 Exemple pour un besoin d'informations de la collection CLEF 2005

Nous avons jusqu'ici choisi de présenter notre modèle à l'aide de documents simples (cf. figures 4.7, 4.4, 4.5 et 4.6) pour montrer d'une part, comment est calculée la proximité floue

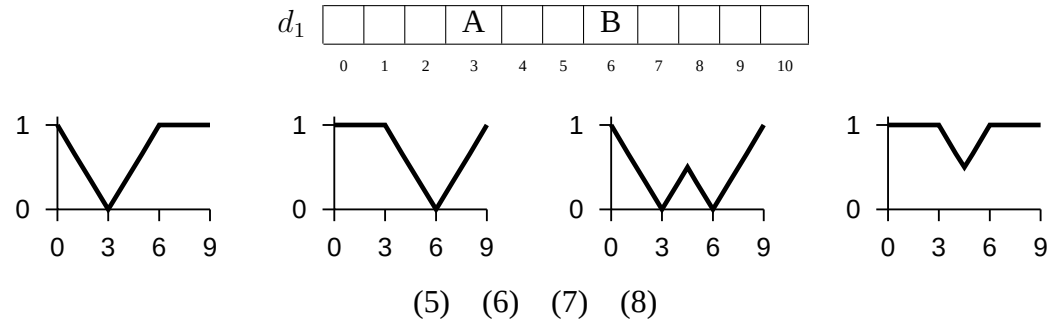


FIG. 4.10 – Document 1 – Représentation de $p^{d1}_{\neg A}$, $p^{d1}_{\neg B}$, $p^{d1}_{\neg (A\,ou\,B)}$ et $p^{d1}_{\neg (A\,et\,B)}$.

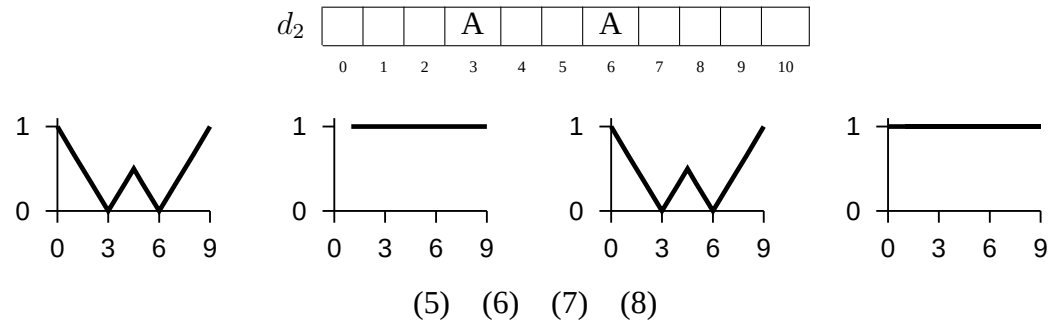


FIG. 4.11 – Document 2 – Représentation $p^{d2}_{\neg A}$, $p^{d2}_{\neg B}$, $p^{d2}_{\neg (A\,ou\,B)}$ et $p^{d2}_{\neg (A\,et\,B)}$.

d'un terme dans un document et d'autre part, comment est évaluée une requête. Cependant, cette exemple pédagogique ne reflète pas la réalité ; afin de montrer comment pourraient être présentés les résultats à l'utilisateur, nous avons choisi une requête de la campagne d'évaluation CLEF 2005 pour illustrer notre modèle. Nous avons orienté notre choix vers la requête 298 car celle-ci présente un fort taux de rappel (1000 documents retournés) et une précision parfaite au premier niveau de rappel si tout le potentiel de notre méthode est exploité. Dans ce cas, la variation de la constante k n'affecte pas trop le nombre de documents pertinents retournés. Pour la valeur 200, $93,43\%$ des documents retournés sont pertinents. Toutes les réponses sont intégralement retournées par la méthode de proximité floue. Le tableau 4.3 montre le résultat de la précision interpolée aux différents niveaux de rappel et la précision.

Rappel			Précision		
	$k = 200$	$k = 50$		$k = 200$	$k = 50$
0	100	100	5	60	20
10	34	34.15	10	40	30
20	24.94	28.86	15	26.67	33.33
30	24.94	28.86	20	25	40
40	24.94	24.10	30	20	33.33
50	24.94	24.10	100	23	24
60	24.94	24.10	200	22	24.50
70	24.94	22.97	500	22	20.20
80	22.92	13.61	1000	12.80	12.70
90	13.52	13.09			
100	0	0			

TAB. 4.3 – Rappel/Précision pour la requête 298 ($k = \{200, 50\}$, jeu lemme)

Le sujet abordé dans la requête 298 concerne les centrales nucléaires. Deux jeux de requêtes sont construits :

- des requêtes conjonctives (jeu titre : t\$$k$\$pf ou tOkapi) constituées des mots du titre « centrales & nucleaires ». Dans ce cas, 386 documents sont retournés dont 26 pertinents.
- des requêtes booléennes (jeu lemme : l\$$k$\$pf ou lOkapi) construites manuellement avec des mots du titre, de la description et de la partie narrative « (centrale | centrales) & (nucleaire | nucleaires) ». Ici, 1000 documents sont retournés parmi lesquels sont retrouvés 128 documents sur les 137 jugés pertinents.

Ainsi, nous exposons un exemple concret, les documents qui sont présentés dans la liste des réponses peuvent être résumés (ou visualisés) en fonction de la proximité à la requête à chaque position dans le document.

Le tableau 4.4 montre dans l'ordre des réponses le résumé « visuel » des premiers documents retournés, avec pour chaque graphique, en ordonnée la valeur de proximité floue et, en abscisse la position dans le document[2]. Le tableau C.1 de l'annexe C présente l'ensemble des

[2]Pour faciliter la comparaison, la valeur maximale en abscisse est celle du document le plus long.

documents pertinents retournés pour la requête 298 avec une valeur k égale à 200.

4.5 Intégration des modèles classiques

Les valeurs extrêmes de variation du paramètre k qui contrôle l'étendue de la zone d'influence d'un terme permettent de ramener notre modèle d'interprétation des requêtes aux modèles vectoriel ou booléen [Mercier et Beigbeder, 2005].

4.5.1 Modèle vectoriel

Le niveau de coordination, l'un des premiers modèles de recherche d'information, permet de classer les documents par rapport au nombre de termes communs entre la requête et le document. Une des améliorations de ce modèle, qui permettra d'évoluer naturellement vers le modèle vectoriel, est de calculer le score de pertinence d'un document par la somme de la fréquence de tous les termes de la requête apparaissant dans le document. Dans notre modèle, nous pouvons reproduire cette dernière méthode :

1. en prenant une fonction d'influence rectangulaire de largeur 1 (donc $k = \frac{1}{2}$) et de hauteur 1 comme dans la figure 4.12 si bien que la zone d'influence de toute occurrence de terme est limitée à l'occurrence elle-même et que les zones d'influence ne se recouvrent pas, et,
2. en utilisant une requête disjonctive.

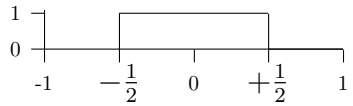


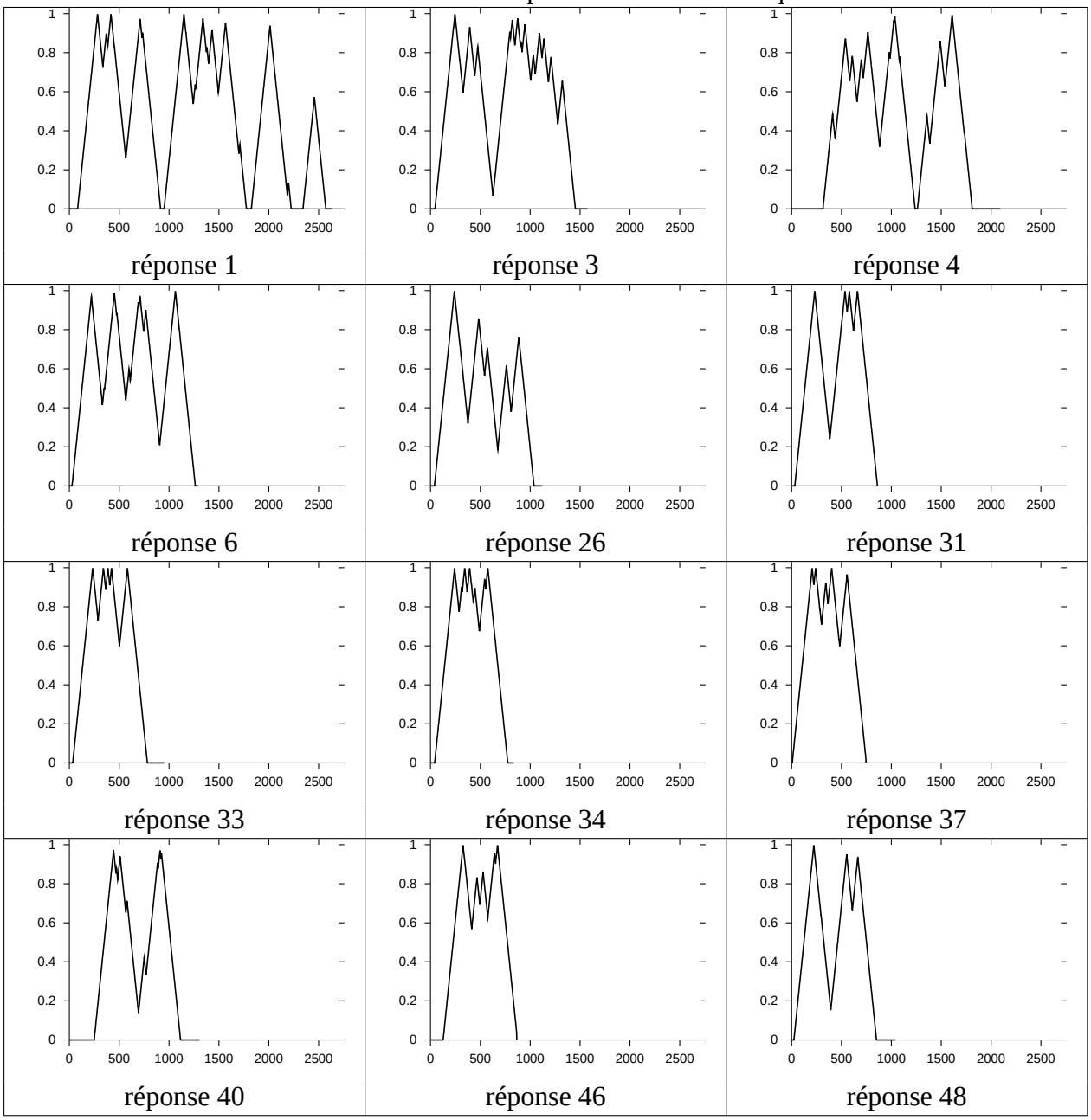
FIG. 4.12 – Fonction d'influence avec une zone d'influence très limitée $k = \frac{1}{2}$.

Le premier point permet de prendre en compte la fréquence des termes pour calculer la valeur de similarité entre un document et une requête tandis que le second permet de considérer les documents dès qu'ils contiennent au moins un terme de la requête.

De plus, nous pouvons nous rapprocher du comportement du modèle vectoriel en appliquant différents types de pondérations pour déterminer le score à chaque position dans le document. Par exemple, le comportement du modèle vectoriel peut être reproduit en affectant une hauteur de la fonction d'influence dépendant de la fréquence documentaire[3] aux positions d'apparition des termes. De cette manière, les scores des documents dépendent de la fréquence documentaire et de la fréquence des termes ce qui nous renvoie bien aux principes du modèle vectoriel.

[3]Nous pouvons appliquer différentes fonctions idf ayant une valeur normalisée.

TAB. 4.4 – Visualisation de la proximité floue à la requête 298



4.5.2 Modèle booléen

Si nous étendons la zone d'influence au document tout entier dans notre modèle de proximité floue, ce qui correspond au cas où le paramètre k tend vers l'infini, alors notre calcul d'appariement entre documents et requêtes se ramène à celui du modèle booléen. Nous en faisons ci-dessous la démonstration.

Tout d'abord, prenons une fonction d'influence rectangulaire de largeur $2k$ et de hauteur $\frac{1}{2k}$ comme illustrée dans la figure 4.13 :

$$\frac{1}{2k} \cdot \mathbb{1}_{[-k,k]}$$

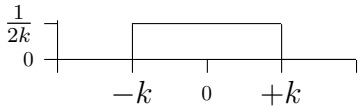


FIG. 4.13 – Fonction d'influence rectangulaire.

Étant donné un terme t et un document d de longueur $l+1$, nous majorons la fonction p_t^d pour n'importe quelle position x :

$$p_t^d(x) = \max_{i \in Occ(d,t)} f(x-i) \leq \max_{i \in [0,l]} f(x-i) \leq \frac{1}{2k} \cdot \mathbb{1}_{[-k,l+k]}(x).$$

Étant donnée une requête q, cette majoration est vraie pour chaque feuille, donc elle est aussi trivialement vraie pour tous les nœuds de l'arbre. En utilisant cette majoration à la racine, nous avons :

$$s_k(q,d) = \sum_{x \in \mathbb{Z}} p_q^d(x) \leq \sum_{x \in \mathbb{Z}} \frac{1}{2k} \cdot \mathbb{1}_{[-k,l+k]} = \frac{l+2k}{2k}$$

et

$$\lim_{k \to +\infty} s_k(q,d) \leq \lim_{k \to +\infty} \frac{l+2k}{2k} = 1.$$

Dans notre modèle, une requête q est un arbre qui porte les termes au niveau des feuilles et les opérateurs booléens ET et OU au niveau des nœuds. En développant une telle requête par distribution de l'opérateur ET sur l'opérateur OU, une forme normale disjonctive est obtenue $q = q_1 \, OR \, q_2 \, OR \, ... \, OR \, q_n$ où tous les termes[4] conjonctifs $(q_i)_{1 \leq i \leq n}$ sont des conjonctions d'éléments de T. Un tel document satisfaisant la requête est évalué à 1 avec le modèle booléen et nous allons prouver que $\lim_{k \to +\infty} s_k(q,d)$ est égale à 1.

[4]Ici, « terme » est utilisé dans le sens algébrique.

Considérons un document satisfaisant cette requête booléenne. Un tel document satisfait au moins un des $(q_i)_{1\leq i\leq n}$, soit q_{i_0}. Nous avons en particulier :

$$p_q^d = \max_{1\leq i\leq n} p_{q_i}^d \geq p_{q_{i_0}}^d.$$

Sachant que q_{i_0} est une requête conjonctive, nous pouvons l'écrire $t_1\ AND\ t_2\ AND\ ...\ AND\ t_k$ pour $(t_j)_{1\leq j\leq k} \subset T$. Comme d satisfait (q_{i_0}), chaque terme t_j, pour $1 \leq j \leq k$, apparaît dans le document d. La fonction $p_{q_{i_0}}^d$ est l'« intersection » de chaque fonction d'influence et donc est aussi l'« intersection » des deux plus éloignées. Notons u (resp. v) la première (resp. la dernière) position où une occurrence d'un terme pris dans $(t_j)_{1\leq j\leq k}$ apparaît, soit :

$$u = \min \bigcup_{1\leq j\leq k} Occ(t_j, d) \text{ et } v = \max \bigcup_{1\leq j\leq k} Occ(t_j, d).$$

Comme nous avons :

$$p_{q_{i_0}}^d = p_{t_1\ AND\ ...\ AND\ t_k}^d = \min_{1\leq j\leq k} p_{t_j}^d$$

cette fonction est encore égale à $\min(p_{t(u)}^d, p_{t(v)}^d)$ pour le terme $t(u)$ qui apparaît à la position u dans d et pour le terme $t(v)$ qui apparaît à la position v dans d (cf. figure 4.14). Comme $p_{t(u)}^d = \frac{1}{2k} \cdot \mathbb{1}_{[u-k,u+k]}$ et $p_{t(v)}^d = \frac{1}{2k} \cdot \mathbb{1}_{[v-k,v+k]}$ alors :

$$\min(p_{t(u)}^d, p_{t(v)}^d) = \mathbb{1}_{[v-k,u+k]}$$

comme illustré dans la figure 4.14.

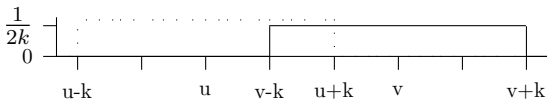


FIG. 4.14 – La surface de l'intersection entre les rectangles représente le score du document. Les deux occurrences des termes sont à la position u et à la position v.

Par conséquent, nous obtenons :

$$p_q^d(x) \geq p_{q_{i_0}}^d(x) = \frac{1}{2k} \cdot \mathbb{1}_{[v-k,u+k]}(x)$$

et donc :

$$s_k(q, d) = \sum_{x\in\mathbb{Z}} p_q^d(x) \geq \sum_{x\in\mathbb{Z}} \frac{1}{2k} \cdot \mathbb{1}_{[v-k,u+k]}(x)$$

avec

$$\sum_{x\in\mathbb{Z}} \frac{1}{2k} \cdot \mathbb{1}_{[v-k,u+k]}(x) = \frac{1}{2k} \cdot ((u+k) - (v-k)) = \frac{2k + (u-v)}{2k}$$

donc

$$\lim_{k\to+\infty} s_k(q,d) \geq \lim_{k\to+\infty} \frac{2k-u+v}{2k} = 1.$$

Comme nous avons précédemment prouvé que cette limite est plus petite que 1, elle est donc égale à 1.

Réciproquement, considérons un document d qui ne satisfait pas la requête booléenne. Dans ce cas, d ne satisfait aucun $(q_i)_{1\leq i\leq n}$. Étant donné i, $1 \leq i \leq n$, q_i est une requête conjonctive :

$$t_1\ AND\ t_2\ AND\ ...\ ANDt_k$$

et au moins un des (t_j), $1 \leq j \leq k$, disons t_{j_0}, n'apparaît pas dans le document d, alors :

$$(\forall x)\ p^d_{t_{j_0}}(x) = 0$$

et donc :

$$p^d_{q_i}(x) = 0$$

d'où :

$$p^d_q = \max_{1\leq i\leq n} p^d_{q_i} = 0.$$

Par conséquent, la sommation vaut zéro pour n'importe quelle valeur de k et sa limite est aussi égale à zéro.

Nous avons donc démontré que :

1. $\lim_{k\to+\infty} s_k(q,d) = 1$ si un document d satisfait une requête booléenne q ;
2. $\lim_{k\to+\infty} s_k(q,d) = 0$ puisque $s_k(q,d) = 0$ pour n'importe quelle valeur de k si le document d ne satisfait pas la requête q.

Nous pouvons ainsi retrouver le comportement du modèle booléen classique en considérant la limite du score calculé par notre méthode lorsque le paramètre k tend vers l'infini.

Chapitre 5

Expérimentations et résultats

Table des matières

5.1 Méthodologie

Dans cette section, nous exposons la méthodologie générale que nous avons adoptée pour mener nos expérimentations à leurs termes. Nous présentons, d'une part, les outils que nous avons utilisés et d'autre part, les moyens que nous avons mis en œuvre pour adapter et construire les données utiles à nos expérimentations.

5.1.1 Outils

5.1.1.1 MG

Dans un premier temps, nous avons choisi le système MG [Witten *et al.*, 1999] car il s'agit d'un système ouvert, composé de plusieurs outils annexes pouvant être utiles pour nos expériences. Il offre deux modèles de base : un modèle vectoriel et un modèle booléen. L'outil MG, du moins dans la version 1.2.1 que nous avons utilisé, ne gère, pour les termes, que les caractères sur un octet de l'ASCII. Un traitement préalable a donc été effectué pour remplacer tous les caractères accentués par leur équivalent sans accent et, tout autre caractère spécial comme par exemple le symbole © par un espace. Ce traitement est bien sur adapté pour les documents écrits en français dans lesquels les mots accentués sont fréquents, mais aussi pour les autres documents issus de diverses collections de test où figure l'usage de caractère spéciaux. Par ailleurs, MG dispose d'une procédure de lemmatisation pour la langue anglaise que nous avons utilisée pour les collections issues d'une image de la Toile.

Implantation des méthodes à intervalles L'application des méthodes basées sur la proximité, nous a conduit à construire un fichier inverse supplémentaire permettant d'obtenir la position des termes dans les documents, donnée essentielle pour le calcul du score des documents basé sur les intervalles de textes. De plus, comme la méthode de Rasolofo et *al.* repose sur le score Okapi, nous avons construit un autre fichier inverse pour accéder directement à la fréquence documentaire de chaque terme à partir de notre module simulant les multiples méthodes à intervalles. Nous avons choisi de créer de nouveaux fichiers pour éviter de modifier la lecture des index dans MG afin de pouvoir utiliser notre module de proximité avec les éventuelles mises à jour de cet outil.

Pour obtenir des résultats émanant des méthodes à base de proximité, nous effectuons une première interrogation de MG avec des disjonctions de mots pour chaque requête. Un ensemble de documents contenant au moins un mot de chaque requête est ainsi récupéré, ces documents sont par la suite classés en fonction du score calculé par les méthodes à intervalles pour chaque document. Les numéros des documents et ceux des termes, valeurs internes du système MG, sont sauvegardés dans un fichier pour chaque requête. Pour chaque méthode et chaque requête, le score des documents est calculé par un module auxiliaire qui prend en paramètre ce fichier de réponse ainsi que le fichier inverse des positions. Ces deux fichiers sont utilisés pour la sélection des intervalles, le calcul des contributions aux intervalles correspondant et le calcul du score des documents. Pour la méthode Okapi et la méthode de Rasolofo, un poids est calculé pour chaque terme de la requête. Ce dernier dépendant de la fréquence documentaire du terme peut être négatif. De ce fait, nous avons adapté la méthode de Rasolofo pour n'utiliser que des valeurs positives. Par conséquent, si le poids d'un terme calculé selon la méthode Okapi est négatif, il est considéré comme nul et les paires ne sont pas construites avec ce terme[1]. Nous avons simplifié la méthode de Hawking en ne considérant qu'une seule relation de proximité non pondérée, constituée de la liste des mots-clés de la requête. Les documents initialement retournés par MG sont ainsi reclassés et sont à nouveau retournés dans l'ordre décroissant des scores nouvellement calculés. Les résultats de la section 5.2.1 sont produits par l'implantation des différentes caractéristiques des méthodes basées sur la proximité comme la sélection et le calcul du score de chaque intervalle. Nous n'avons pas la prétention de revendiquer ces résultats comme ceux des méthodes originales mais nous essayons en reproduisant leurs principes fondamentaux d'observer leur comportement sur les sous-collections uniformes de taille croissante de la collection WT10g[2]. L'utilisation de ces sous-collections permet, entre autres, de mettre en relief le caractère de « haute précision » de ces méthodes.

Du point de vue de la taille des collections, nous avons pu facilement conduire des tests sur les collections *adi*, *cisi* et *cacm*, en revanche, MG ne nous a pas permis d'indexer la collection WT10g en entier. Nous nous sommes alors tournés vers le système de recherche d'informations LUCY.

5.1.1.2 LUCY

Nous avons facilement indexé avec LUCY la collection WT10g ; cet outil a d'ailleurs été développé à l'origine pour de grandes collections de données. Le système LUCY, que nous avons utilisé dans sa version 0.5.2[3], implémente le modèle *BM25* de Okapi, étant lui-même une extension du modèle probabiliste [Robertson *et al.*, 1994]. Depuis nos premières expériences en 2004, le système LUCY a beaucoup évolué, il est désormais connu sous le nom de ZETTAIR [4] . L'équipe

[1] Ce cas de figure se présente plus souvent pour les collections *adi*, *cacm* et *cisi* vu le faible nombre de documents dans ces collections. Dans ce cas, les résultats sont présentés sous la dénomination *pRasolofo*, *p* signifiant positif.

[2] `http://es.csiro.au/TRECWeb/wt10g.html`

[3] http ://www.seg.rmit.edu.au/lucy/

[4] http ://trec.nist.gov

développant ZETTAIR participe à la tâche TERABYTE de TREC et a amélioré cet outil[5] pour gérer la collection volumineuse qui lui est associée. Un atout supplémentaire de ces deux outils est qu'ils sont adaptés au traitement des documents et des fichiers de requêtes des collections TREC [6].

De même que pour MG, les documents sont indexés sans accents et sans caractères spéciaux. En revanche, l'inconvénient de LUCY (qui n'existe plus aujourd'hui dans ZETTAIR) est le manque d'une procédure de lemmatisation. Cependant, nous avons choisi d'étendre ce système pour y introduire notre méthode de proximité floue car, d'une part il permet de gérer des collections de la taille de la WT10g et d'autre part il possède déjà les positions des occurrences de termes dans son index, le langage de requêtes offrant la possibilité de spécifier des expressions.

Implantation de la proximité floue Nous avons ajouté un module et une option en ligne de commande[7] pour interroger l'index avec notre méthode. Pour résumer, nous avons intégré un nouveau traitement des requêtes booléennes afin de pouvoir calculer le score des documents selon la méthode de proximité floue.

5.1.2 Préparation des documents

Comme nous l'avons déjà dit, nous remplaçons dans les documents les caractères accentués par leur correspondant sans accents ainsi que les caractères spéciaux par des espaces. De plus, les documents des collections WT10g et TERABYTE sont issus du Web et sont donc au format HTML. Comme ni les outils MG ou LUCY, ni les approches qui nous intéressent, ne tiennent compte de la structure interne des documents, nous avons converti ces derniers en utilisant le programme LYNX[8] dans sa version 2.8.5 avec les options `-dump` et `-force_html` pour enlever les balises et pour remplacer les entités du langage HTML. Par ailleurs, pour les documents disponibles sous format XML dans les collections AQUAINT et CLEF, nous ne retenons que le contenu de certaines balises comme il est demandé dans les campagnes d'évaluation. Nous n'avons ainsi qu'à indexer le contenu textuel informationnel des documents. Enfin, pour la collection CLEF, les documents sont lemmatisés à l'aide du programme de Jacques Savoy adapté à la langue française. Ce pré-traitement supplémentaire avant l'indexation permet de réaliser une expérience particulière (cf. section 5.3.3). Nous étudions l'impact de la lemmatisation sur notre méthode et nous vérifions si l'utilisation de requêtes booléennes complexes améliore réellement les résultats.

[5] ZETTAIR implante diverses approches : Okapi (bm25), cosinus, *pivoted cosine*, hawkapi (mesure de D. Hawking) et dirichlet (mesure *dirichlet-smoothed LM*). Il fait partie des outils recommandés sur la page Web officielle de la tâche TERABYTE.

[6] http ://trec.nist.gov

[7] lucy -p -[k $entierlargeur$] pour préciser l'utilisation de la proximité avec éventuellement la largeur de la zone d'influence

[8] `http://lynx.browser.org/`.

5.1.3 Préparation des requêtes

Différents jeux de requêtes sont construits, d'une part pour correspondre à la nature de la requête à soumettre à un système donné et, d'autre part, pour traduire le besoin d'informations avec plus ou moins de précision de manière manuelle ou automatique. Rappelons que nous avons besoin de requêtes :

- **plates** pour la méthode vectorielle dans MG et la version Okapi dans LUCY ;
- **booléennes disjonctives** pour la sélection des documents à classer avec les méthodes à intervalles ;
- **booléennes** pour notre méthode.

Afin de pouvoir comparer équitablement les méthodes et notamment par rapport à la méthode de référence, nous utilisons toujours plusieurs jeux de requêtes par expérience. Par exemple, pour un besoin d'informations traduit par les mots « A, B, C », nous avons :

- `A B C` comme requête plate pour le modèle vectoriel et Okapi ;
- `A | B | C` comme requête disjonctive pour les méthodes à intervalles ;
- `A & B & C` comme requête conjonctive « automatique » pour la proximité floue ;
- `(A | A′) & (B | B′ | B″) & (C | C′)` comme requête booléenne « manuelle » pour la proximité floue ;

5.1.3.1 Choix des termes

Pour les premières collections que nous avons utilisées, *adi*, *cisi* et *cacm*, les besoins d'informations sont en fait formulés comme des questions en langage naturel (cf. figure 5.1). Pour construire une requête à partir d'un tel besoin d'informations, nous extrayons manuellement les termes importants. L'ensemble des mots extraits forme une requête du type « sacs de termes » (ou requête plate). L'interrogation des collections avec des requêtes plates est tout à fait adaptée au modèle vectoriel : même un court paragraphe peut constituer une requête dans ce modèle. Par contre, pour les méthodes à intervalles, le calcul du score devient complexe si la quantité de mots de la requête devient importante. Pour ces méthodes, nous souhaitons obtenir un ensemble initial de documents contenant au moins un des termes de la requête afin de classer les documents en fonction de l'analyse de la proximité. Pour ce faire, nous utilisons une disjonction des termes de la requête plate pour sélectionner un large ensemble de documents (cf. section 1.3.1) à classer par la suite à l'aide des multiples méthodes à intervalles. Pour les expériences relatives à la collection WT10g, en moyenne trois termes du champ titre des « topiques » sont choisis pour constituer les requêtes.

Nos dernières expérimentations ont été réalisées pour les campagnes d'évaluation TREC et CLEF. Nous avons construit différents jeux de requêtes pour comparer notre méthode à la méthode Okapi de LUCY. En premier, nous construisons de **manière automatique** deux jeux de requêtes ; l'un avec les termes du champ titre, l'autre avec les termes du champ description. Chacun de ces jeux est décliné d'une part en requêtes plates comme par exemple (A B C), et d'autre part en requêtes booléennes comme (A & B & C). En second, nous construisons des

```
.I 1
.W
What problems and concerns are there in making up descriptive titles?
What difficulties are involved in automatically retrieving articles
from approximate titles?
What is the usual relevance of the content of articles to their titles?
.I 2
.W
How can actually pertinent data, as opposed to references or entire
articles themselves, be retrieved automatically in response to
information requests?
.I 3
.W
What is information science?  Give definitions where possible.
.I 4
.W
Image recognition and any other methods of automatically
transforming printed text into computer-ready form.
.I 5
...
...
...
.I 35
.W
Government supported agencies and projects dealing with information
dissemination.
```

FIG. 5.1 – Extrait des besoins d'informations exprimés dans la collection *adi*.

requêtes de manière manuelle pour utiliser tout le potentiel de notre approche nous passons ainsi de la requête (A & B & C) à une requête plus précise telle que (A | A' | A") & (B | B') & (C | C'). La requête plate manuelle correspondante est aussi créée (A A' A" B B' C C').

Prenons par exemple un besoin d'informations de la collection *Adhoc fr* CLEF *2005*. Chaque sujet est composé d'un numéro et de trois balises pour le décrire : `<FR-title>`, `<FR-desc>`, `<FR-narr>`. Pour effectuer nos tests, les trois jeux de requêtes sont construits.

Pour les **requêtes construites automatiquement** (2 jeux), un jeu est composé des termes contenus dans le texte du champ `<FR-desc>`, l'autre des termes du champ `<FR-desc>`, les mots vides[9] sont retirés.

Les **requêtes construites manuellement** (1 jeu) sont constituées des termes du champ `<FR-title>` et de quelques termes du champ `<FR-desc>` voire aussi du champ `<FR-narr>`. L'idée générale est de pallier le manque de lemmatisation de LUCY en utilisant des conjonctions de disjonctions de termes. Pour ce faire, nous essayons de regrouper dans une ou plusieurs disjonctions les termes synonymes ou sémantiquement proches ainsi que les diverses formes des termes (pluriel, nom, verbe etc.). Pour l'évaluation avec l'outil LUCY, nous retirons les opérateurs booléens. Pour les méthodes à bases d'intervalles, nous sélectionnons les documents qui contiennent au moins un mot-clé de la requête puis nous appliquons la méthode.

L'**exemple du sujet 249** ci-dessous montre les étapes de construction d'une requête `<FR-desc>` :

```
<num> C249 </num>
<FR-title> Championne du 10.000 mètres féminin </FR-title>
```

Nous obtenons :

```
249 championne 10000 metres feminin.
```

A partir de cette requête, nous avons ses variantes **automatiques** :

```
lucy                    249 championne 10000 metres feminin
proximité floue         249 championne & 10000 & metres & feminin
méthodes intervalles    249 championne | 10000 | metres | feminin
```

ou **manuelles** :

```
lucy                  249 championne championnes 10000 metre metres feminin
feminins
proximité floue       249 (championne | championnes) & 10000 & (metre | metres)
```

[9] Les mots vides présents dans le fichier de « topiques » sont 'à', 'aux', 'au', 'chez', 'et', 'dans', 'des', 'de', 'du', 'en', 'la', 'les', 'le', 'par', 'sur', 'uns', 'unes', 'une', 'un', 'd', 'l'.

```
& (feminin |feminins)
méthodes intervalle  249 championne | championnes | 10000 | metre metres
| feminin | feminins
```

Nous procédons de la même manière pour la même tâche de CLEF 2006, pour la tâche robuste de TREC 2005 et pour la tâche TERABYTE de TREC 2006.

5.1.4 Production des listes de résultats

Notre méthode est très sélective ce qui implique que la liste des résultats est très courte (surtout quand des requêtes totalement conjonctives sont utilisées). Dans l'état de l'art, nous avons vu que les méthodes à intervalles ont tendance à intégrer la proximité dans un calcul de similarité globale (cf. Méthode de Monz, Rasolofo et *al.* et Song et *al.*). Par ailleurs, la méthode Okapi BM25 est réputée pour être l'une des plus performantes. Par conséquent, nous complétons notre liste de réponses à la longueur des *runs* de TREC (1000 documents) avec les documents renvoyés par la méthode Okapi de LUCY et n'ayant pas déjà été retournés par notre méthode de proximité floue. Pour ce faire, nous utilisons deux requêtes, l'expression booléenne traitée par notre méthode et la requête plate correspondante (c'est-à-dire contenant seulement les mots de la requête booléenne) traitée par la méthode de référence Okapi.

5.1.5 Évaluation des méthodes

A chaque expérimentation, nous utilisons une méthode de référence connue pour situer nos résultats. Nous comparons les méthodes à intervalles au modèle vectoriel implanté dans MG (mgVect) et notre méthode de proximité floue à l'implantation Okapi de LUCY (Okapi).

5.2 Méthodes à intervalles et passage à l'échelle

5.2.1 Etudes préliminaires

Dans [Mercier, 2004], nous avons étudié le comportement des différentes méthodes à intervalles sur trois collections de recherche d'informations assez peu gourmandes en mémoire : *adi*, *cacm* et *cisi*.

Pour ces petites collections, nous remarquons dans la figure 5.2 que le modèle vectoriel de MG produit de meilleurs résultats que les méthodes à intervalles dans la plupart des cas. Cependant, la méthode de Clarke et *al.* fait exception aux premiers niveaux de rappel pour les collections *cacm* et *cisi* et la courbe méthode de Hawking se rapproche aussi de celle du vectoriel.

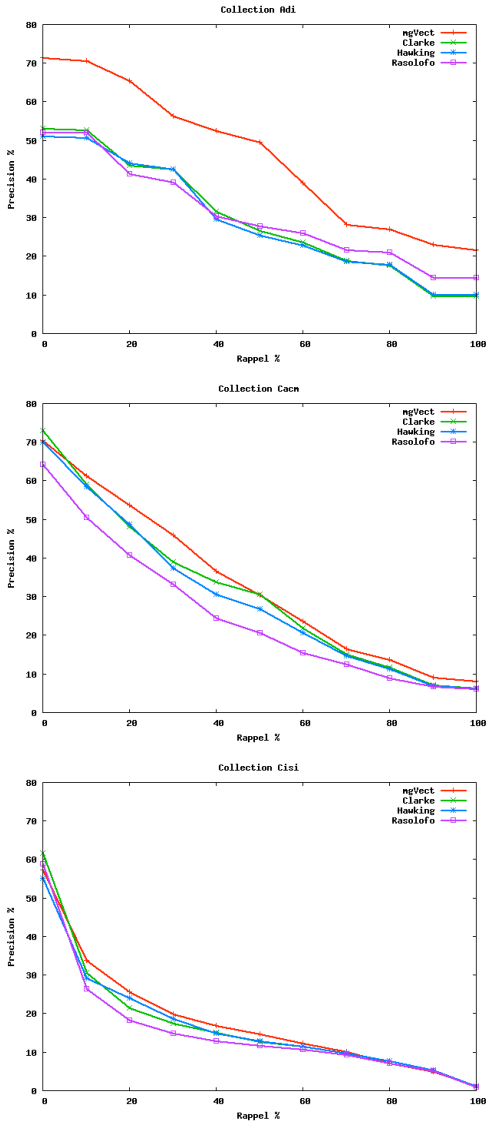


FIG. 5.2 – Rappel/Précision pour les collections Adi, Cacm et Cisi

Nous notons ainsi que des améliorations peuvent être apportées si des méthodes à base de proximité sont utilisées. Parmi celles-ci, nous constatons que celles de la méthode de Clarke sont les meilleures. Ces résultats s'expliquent par la taille des collections et le nombre de mots contenus dans les requêtes. La collection *adi* contient moins de documents que les deux autres. De plus, au fil de nos expérimentations, nous avons parfois réduit le nombre de mots dans les requêtes. Ces deux dernières explications vont dans le sens des résultats. Pour les plus grandes collections (*cacm* et *cisi*) en soumettant des requêtes adaptées aux méthodes, les résultats sont presque équivalents au modèle vectoriel connu pour être performant pour ce genre de collections[10]. Le résultat obtenu par la méthode de Clarke et *al.* est prometteur dans le sens où il traduit une meilleure performance sur les plus grandes collections. Néanmoins, nous ne pouvons conclure de manière tranchée sur l'apport qualitatif des méthodes de proximité avec ces expériences parce que :

1. la taille des collections ne correspond plus à celles des collections volumineuses utilisées aujourd'hui ;
2. les requêtes que nous avons construites contiennent beaucoup de mots-clés car nous n'avons pas voulu écarter certains aspects des questions posées (cf. section 5.1.3.1). Par conséquent, les requêtes ne sont pas forcément adaptées aux méthodes de proximité. En effet, ces dernières ont été testées par les auteurs sur la collection WT10g où l'on peut facilement, en utilisant seulement le champ titre, obtenir des requêtes courtes de l'ordre de trois mots pour les expérimentations [Clarke *et al.*, 2000].

Ces deux raisons, nous ont encouragé à observer le comportement de ces méthodes sur la collection WT10G puisque celle-ci est une collection assez volumineuse pour valider nos résultats. De plus, cette dernière a été utilisée pour différentes éditions de la campagne d'évaluation TREC.

En parallèle de nos expériences sur la proximité, nous avons réalisé une étude sur le passage à l'échelle [Beigbeder et Mercier, 2003]. Nous employons le terme de passage à l'échelle pour montrer le pas entre collections de tailles croissantes. Ce genre d'étude vise à observer le comportement des méthodes ou bien l'évolution de constantes connues en fonction de l'échelle de la collection. L'échelle peut se calculer en terme de nombre de documents ou de taille physique de collection.

Cette étude sur le passage à l'échelle nous a conduit à formuler l'hypothèse [11] que « plus une collection contient de documents, plus le facteur df, discriminant traditionnel des documents, perd de son pouvoir de discrimination ». Karbasi et Tamine ont en quelque sorte poursuivi cette étude en regardant l'impact des différents facteurs (tf, idf, longueur du document) sur les résultats obtenus avec le système MERCURE [Karbasi et Tamine, 2005]. Elles aussi constatent que le passage à de grandes collections amplifie le problème de la discrimination des termes puisque le nombre de termes fréquents n'augmente pas énormément et la proportion de termes discriminant diminue. Cependant, elles montrent, par une expérience annulant (ou non) le facteur idf dans la

[10] Petite collection avec des documents du type résumé et des requêtes de l'ordre de quelques mots, de la phrase ou du paragraphe.

[11] De manière intuitive aux vues des résultats obtenus pour l'évolution des facteurs tf et idf en fonction de la taille des collections.

formule d'appariement, que ce dernier joue bien un rôle de discrimination dans les collections de plus grande taille et ne dégrade pas les résultats. Néanmoins, ce résultat ne permet pas de contredire l'hypothèse que nous avions faite car leur étude analyse seulement l'impact de la présence ou de l'absence du facteur df dans la formule mais ne donne pas d'indication sur l'évolution de son pouvoir de discrimination en fonction de la taille de la collection. Une étude plus approfondie sur ce sujet pourrait mettre en place un contexte expérimental pour déterminer le pouvoir effectif de discrimination du facteur df quand le nombre de documents (et par conséquent la taille du vocabulaire) augmente.

L'expérience présentée dans la section suivante constitue un point de départ pour, d'une part, étudier les méthodes existantes basées sur la proximité et d'autre part, étayer notre idée sur le pouvoir discriminant du facteur df afin de justifier l'utilisation de la proximité comme un moyen alternatif à l'utilisation de ce facteur. Dans le cadre du passage à l'échelle, nous commencerons par présenter dans la section suivante, les résultats classiques de Rappel/Précision pour les méthodes à intervalles sur les sous-collections uniformes de taille croissante de la collection de test WT10G, avec les besoins d'informations [12] 451-500 provenant de la campagne d'évaluation de TREC 9 et, les jugements de pertinence[13] correspondant.

5.2.2 Étude du passage à l'échelle

Dans [Beigbeder et Mercier, 2003], nous avons abordé le problème du passage à l'échelle en essayant d'évaluer l'effet de l'augmentation du nombre de documents sur les distributions des valeurs de tf et df qui sont utilisées dans les représentants des documents dans les différentes implantations du modèle vectoriel. Pour faire des tests de distributions à grande échelle obtenus avec les différentes formules de pondération, nous avons eu besoin d'un ensemble de documents analogues à ceux que l'on trouve sur la Toile. La meilleure source est donc de prendre un extrait représentatif de celle-ci. Nous avons accès à un ensemble, que nous appellerons WFR4, de 5057642 pages[14] collectées sur la Toile par le Laboratoire CLIPS[15] de l'université de Grenoble en décembre 2000 grâce à un robot[16] qu'ils ont développé. Toutes les pages collectées sont dans des domaines d'origine géographique francophone, ce qui ne signifie pas que tous les documents sont en langue française, en effet de nombreux sites proposent plusieurs versions de leurs documents dans plusieurs langues. Pour étudier le passage à l'échelle, nous avons découpé WFR4 en six sous-ensembles de tailles différentes, WFR4.1 à WFR4.6, chaque sous-ensemble a un cardinal 10 fois plus grand que le précédent. WFR4.1 est composé de 10 documents, WFR4.2 de 10^2 documents et ainsi de suite.

[12] http://trec.nist.gov/data/topics_eng/topics.451-500.gz
[13] http://trec.nist.gov/data/qrels_eng/qrels.trec9.main_web.gz
[14] http://www-mrim.imag.fr/membres/mathias.gery/Robot/WebFr4_01_12_2000/domaines.html
[15] http://www-clips.imag.fr/
[16] http://www-mrim.imag.fr/membres/mathias.gery/CLIPS-Index/

Nous avons formulé deux perspectives pour compléter cette étude :

1. nous suggérons d'étudier les distributions des fonctions complètes de pondération, c'est-à-dire en prenant en compte les facteurs $tf \cdot idf$, mais aussi le facteur de pondération sur chaque vecteur $\vec{d}$ représentant le document d : $\vec{d} = (d_t)_{t \in T} = (tf(d,t) \cdot idf(t))_{t \in T}$.
2. sur le plan statistique, l'échantillonnage de nos collections est critiquable. Une solution pour résoudre ce problème est de construire un ensemble d'échantillons pour une taille de collection, et d'observer l'évolution de la moyenne des valeurs de tf et df pour chaque taille de collection. Cette dernière perspective, s'écartant de notre sujet de la proximité, a été reprise dans le sujet de thèse de l'année qui a suivie notre étude. [17].

Plutôt que de seulement comparer l'efficacité des méthodes à intervalles par rapport aux méthodes de références (vectorielle et Okapi), nous souhaitons observer leur comportement par rapport au nombre de documents que contient une collection. De cette manière, notre objectif est de déceler un éventuel comportement différent pour les méthodes à base de proximité. Pour ce faire, nous utilisons plusieurs sous-collections où les documents jugés pertinents ont été uniformément répartis dans les différentes sous-collections, leur construction est brièvement décrite ci-après.

La collection « uniforme » est construite à partir de la collection WT10G originale. Sur cette nouvelle collection, les documents pertinents sont répartis de façon à ce qu'en sélectionnant n'importe quelle portion de collection, la même distribution de documents pertinents soit obtenue ; ainsi le nombre de documents pertinents par besoin d'informations et pour tous les besoins est proportionnel à la taille de la portion [Imafouo et Beigbeder, 2005, Mercier *et al.*, 2005].

5.2.3 Les méthodes à intervalles sur les collections issues de WT10g

5.2.3.1 Les méthodes comparées

Notre module auxiliaire implanté dans l'outil MG, nous permet de comparer au modèle vectoriel, les méthodes à base de proximité suivante :

- *mgVect*, méthode vectoriel implantée dans MG,
- *hawking*, notre simulation de la méthode de Hawking et *al.*,
- *clarke*, notre simulation de la méthode de Clarke et *al.*,
- *rasolofo*, notre simulation de la méthode Rasolofo et *al.* en enlevant les termes de poids négatifs comme expliqué avant.

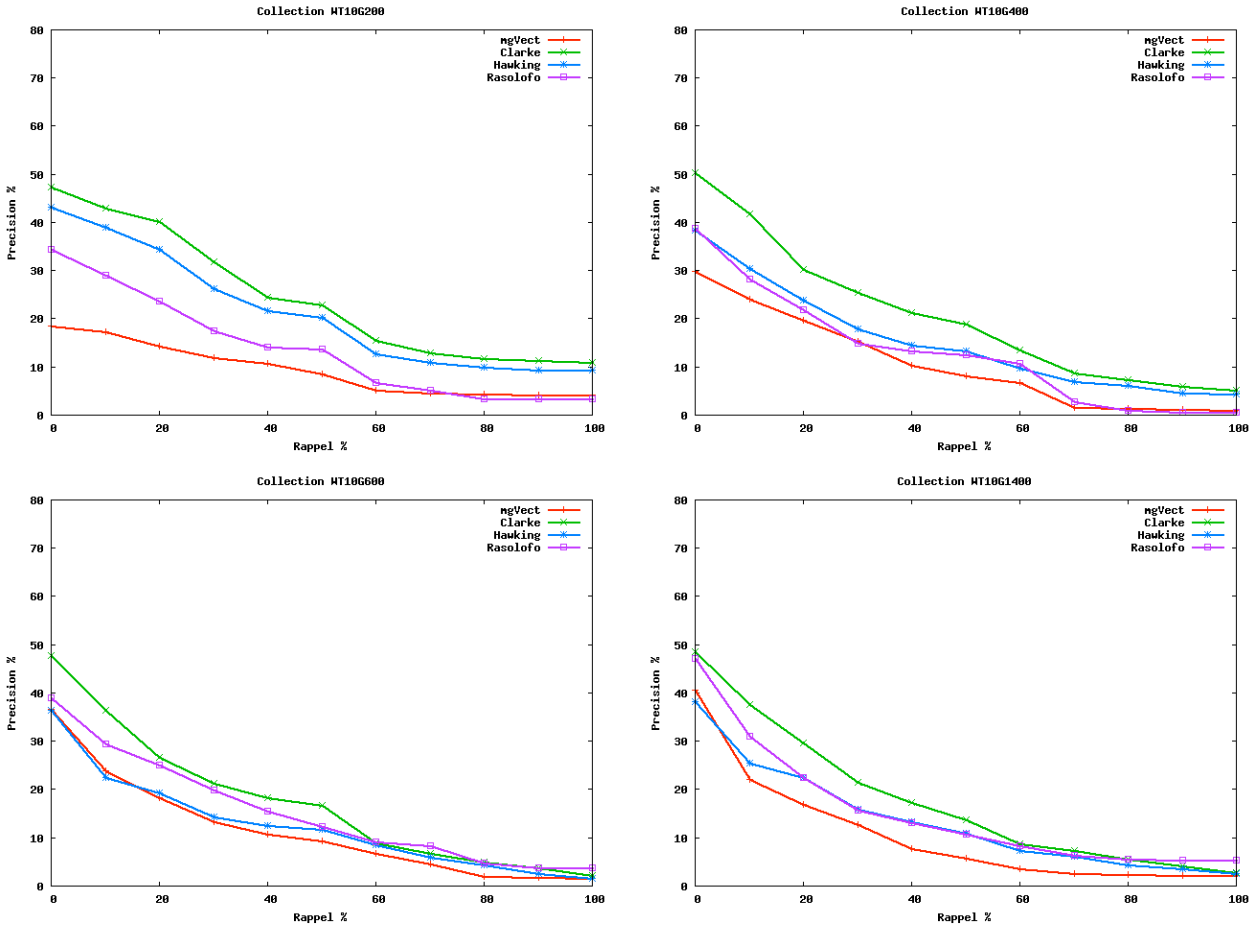


FIG. 5.3 – Précision - rappel à 11 points (Clarke, Hawking, Rasolofo (cf. section 5.1.1.1) et vectoriel) pour les différentes sous-collection de la WT10g

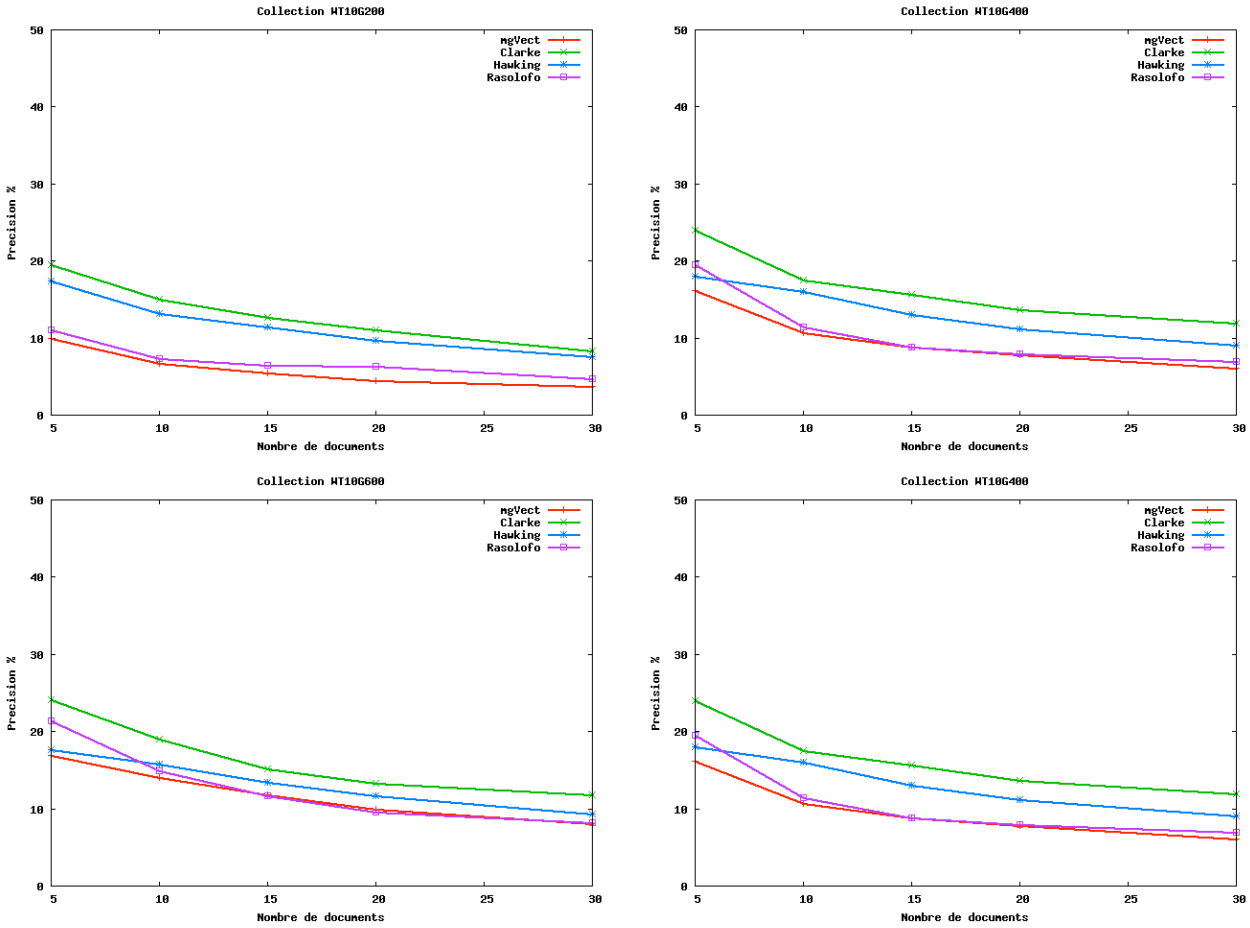


FIG. 5.4 – Précision à 5, à 10, à 15, à 20 et à 30 documents retournés (Clarke, Hawking, Rasolofo et vectoriel) pour les différentes sous-collection de la WT10g

5.2.3.2 Rappel/Précision en fonction de la taille de la collection

La figure 5.3 montre les courbes de rappel/précision[18] pour les collections de différentes tailles. La méthode de Clarke et *al.* est dominante pour tous les points de rappel et toutes les collections. Pour les collections à $200mille$ et $400mille$ documents, elle est suivie par la méthode de Hawking et *al.* Cependant, à partir de la collection à $600mille$ documents, c'est la méthode de Rasolofo et *al.* qui se place en deuxième position. Cette dernière étant basée en partie sur le modèle Okapi, l'accroissement du nombre de documents permet d'obtenir de meilleurs résultats. En effet, une partie de la formule utilisée produit des poids négatifs pour les termes se retrouvant dans au moins la moitié des documents. Nous avons exclu ces derniers du calcul des paires : en augmentant le nombre de documents, ce cas de figure se présentant moins souvent, la méthode de Rasolofo obtient de meilleures performances. Pour les collections à $800mille$, $1million$ et $1,2million$ documents, l'allure des courbes ressemble à la collection à $600mille$ avec un net détachement de la méthode de Clarke et un rapprochement des autres méthodes. De manière générale, la méthode vectorielle se place en dessous des méthodes à intervalles. Pour la collection à $1,4million$ documents[19], nous remarquons que la méthode de Rasolofo, dont la précision est de l'ordre de 50% est très proche de celle de Clarke au premier niveau de rappel puis retombe à 30% au deuxième niveau et descend à 15% de précision en rejoignant de la méthode de Hawking à partir de 30% de rappel. Quant aux performances du modèle vectoriel, elles sont au dessus de la méthode de Hawking au premier niveau de rappel (40% vs. 38%) mais deviennent dès 20% de rappel moins bonnes que pour les trois autres méthodes. En conclusion, la méthode de Clarke est une borne supérieure pour la précision à tous les niveaux de rappel tandis la méthode vectorielle est une borne inférieure pour pratiquement tous les points de rappel.

5.2.3.3 Précision@n en fonction de la taille de la collection

La figure 5.4 montre la précision à n documents pour les coupures à 5, à 10, à 15, et à 30. La précision à 5 documents est de l'ordre de 19% pour la méthode de Clarke et monte jusqu'à 30% pour la plus grande collection. Nous constatons que plus la taille de la collection augmente, plus la valeur de précision au niveau du même nombre de documents augmente. La méthode de Clarke est la plus performante pour toutes les collections. En revanche, la précision de la méthode de Rasolofo augmente plus que celle de la méthode de Hawking et devient la deuxième plus haute précision à partir de la collection à $400mille$ documents.

Nous nous sommes intéressés aux résultats classiques de rappel et précision obtenus par les méthodes à intervalles ; la section suivante analyse sous l'angle d'un indicateur simple, la

[17] Dans ses travaux, A. Imafouo a étudié sous deux axes différents (uniformisation et collections aléatoires) l'échantillonnage des collections dans le cadre de la problématique du passage à l'échelle.

[18] Les résultats ont été obtenus avec l'outil TREC_EVAL version 8

[19] Nous nous focalisons sur cette collection car elle contient le plus grand nombre de documents et se rapproche de la collection WT10g en entier

proportion de documents retrouvés que nous appelons *rappel@X* les performances des méthodes en fonction de la taille de la liste de réponses et de celle de la sous-collection.

5.2.3.4 P_{tr} : taux de documents pertinents toutes requêtes en fonction de la taille de la collection

Pour mettre en relief les différences entre le modèle vectoriel et les méthodes à intervalles, nous utilisons ici un indicateur simple reposant sur le taux de documents pertinents retrouvés pour toutes les requêtes. Cette mesure a été introduite dans la section **??**. Dans la figure 5.5 en colonne de gauche, nous représentons, en abscisse la taille de la collection et en ordonné le taux de documents pertinents retrouvés pour toutes les requêtes par rapport au nombre de documents jugés pertinents. Nous avons choisi de couper les listes de réponses à trois niveaux :

- 100, le nombre de documents utilisés dans la méthode du *pooling*,
- 1000, le nombre de documents pris en compte pour évaluer les performances des systèmes dans les campagnes d'évaluation, et
- « tous », le nombre total de documents retournés par une méthode. Dans nos explications, nous utiliserons le terme « tous » comme représentant le nombre réel de documents retournés. Le tableau 5.1 montre ce nombre de documents pour la méthode vectorielle de MG et les méthodes à intervalles.

taille collection	vectoriel	méthodes à intervalles
200000	1817515	1283522
400000	3829860	2787841
600000	6224980	4653514
800000	8230431	6157354
1000000	10607591	7963482
1200000	12598640	9430904
1400000	14671545	10981033

TAB. 5.1 – Nombre total (« tous ») de documents retrouvés selon la méthode et la taille de collection

Dans la figure 5.5 en colonne de droite, les trois autres courbes montrent la différence entre ces pourcentages. Les différences sont calculées entre deux paliers c'est-à-dire entre 100 et 1000, entre 1000 et « tous » et, entre 100 et « tous ».

Pour les collections à $200mille$ et à $400mille$, P_{tr} à 100 documents retournés est presque deux fois plus grand pour les méthodes basées uniquement sur les intervalles que pour modèle vectoriel. Plus la taille de la collection augmente, plus cet écart se réduit, cependant autant les performances du modèle vectoriel (de 26.48% à 12.87%) que celles des méthodes à intervalles (clarke 47.92% à 24.72% ; hawking 47.17% à 20.38%) baissent lorsque des documents non jugés ou non pertinents sont ajoutés pour constituer de plus grandes collections. La méthode de Rasolofo, extension du modèle Okapi, se situe entre les deux autres méthodes à intervalles et le

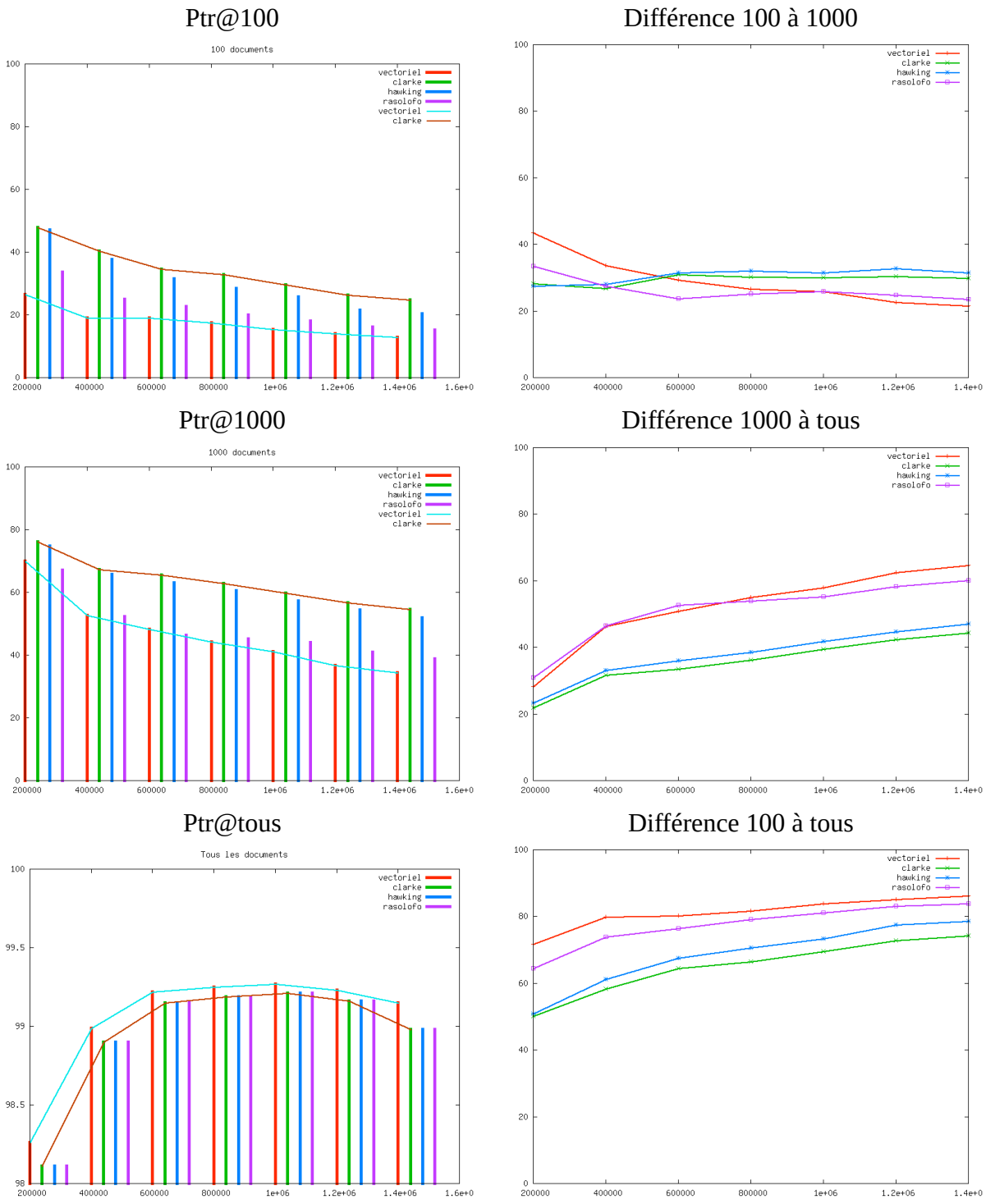


FIG. 5.5 – A gauche, P_{tr} aux niveaux 100, 1000 et « tous » et, à droite, différence entre ces taux aux niveaux 1000 et 100, tous et 1000 et tous et 100

modèle vectoriel. Sur la partie de gauche de la figure 5.5, pour P_{tr} à 1000, nous pouvons faire les mêmes constatations mais nous pouvons noter que le taux de documents pertinents est plus grand entre 76% et 43%. Par contre, P_{tr} à « tous » (troisième figure), se situe entre 98.11% et 99.27% et dans ce dernier cas, c'est le modèle vectoriel qui retourne le plus de documents pertinents. Nous pouvons ainsi dire d'une part, que les méthodes à intervalles favorisent la précision puisqu'elles détectent largement plus de documents pertinents à 100 documents retournés (ainsi qu'à 1000) et sont un peu moins performantes pour le rappel à « tous », et, d'autre part, que le modèle vectoriel favorise le rappel puisque ses performances se rapprochent le plus de 100%. Enfin, nous constatons que le choix de couper la liste de réponses à 1000 documents dans les campagnes d'évaluation constitue un bon compromis entre rappel et précision.

Les courbes de droite de la figure 5.5 permettent de visualiser d'une autre manière les remarques qui ont été faites ci-dessus. La méthode de Hawking et celle de Clarke sont très proches et sont en dessous des autres pour les différences avec « tous ». Pour ces méthodes, la différence entre le palier à « tous » et les deux autres premiers paliers est la plus petite, cela signifie que ces méthodes rappellent proportionnellement plus de documents dès les 100 premiers retournés. Pour la différence entre tous les documents et 1000 documents retournés, nous remarquons deux groupes distincts : (1) le vectoriel, la méthode de Rasolofo et Okapi (2) les deux autres méthodes à intervalles. Enfin, si nous prenons la différence entre le palier à 100 et tous les documents, nous pouvons constater que les valeurs sont les plus fortes (resp. plus basses) pour le modèle vectoriel (resp. méthodes à intervalles) et que l'amplitude n'est pas très importante (environ 15%) contrairement à celles des méthodes à intervalles (environ 25%). Cette dernière constatation montre encore que les méthodes à intervalles sont très performantes dès la première centaine de documents renvoyés.

5.3 Expérimentations du prototype basé sur la proximité floue

5.3.1 Tâche robuste TREC 2005

Nous reportons ci-dessous les résultats de notre méthode obtenus sur la tâche Robuste de la campagne d'évaluation TREC 2005. Nous avons utilisé l'outil LUCY pour indexer la collection de manière à obtenir des résultats issus du modèle Okapi BM-25[20] implanté dans LUCY et issus de notre méthode basée sur la proximité floue.

5.3.1.1 Préparation de la collection de test

Pour cette tâche, le corpus à utiliser est AQUAINT qui est composé d'articles de journaux numériquement sauvegardés dans un format XML. Le tableau 5.2 montre le nombre de docu-

[20]La méthode BM25 repose sur le modèle probabiliste

ments par journal et par année d'édition.

Journal	1996	1997	1998	1999	2000
APW			107882	77876	53818
NYT			85817	104698	90829
XIN	93458	95563	103470	104698	82244

TAB. 5.2 – Nombre de documents indexés par journal et par année d'édition.

Pour chaque document (balise `<DOC>`), les champs `<DOCNO>` avec le tag et le numéro de document contenu à l'intérieur ainsi que le contenu textuel des balises `<TEXT>`, `<P>`, `<HEADLINE>`, `<DOCTYPE>` sont indexés avec LUCY. Pour l'évaluation des méthodes à l'aide du programme TREC_EVAL, nous utilisons les topiques et les jugements de pertinence correspondant du corpus AQUAINT.

Construction des requêtes Comme nous l'avons expliqué dans la section 5.1.3.1, nous utilisons plusieurs jeux de requêtes construites automatiquement ou manuellement. Dans cette expérience, pour le deuxième jeu de requêtes utilisant le champ « description », les termes sont extraits automatiquement à l'aide d'un processus basé sur le langage naturel développé au sein de notre équipe par X. Tannier [Tannier *et al.*, 2005] ce qui donne par exemple pour le topique 375 :

Lucy `375 energy feasible hydrogen source status`

Conjonction proximité floue `375 energy & feasible & hydrogen & source & status`

Construction de la liste de réponses Nous rappelons que pour notre méthode de proximité floue, la liste des résultats est composée en premier des documents retournés par notre méthode et ensuite, pour éviter un rappel insuffisant, si le nombre de documents retournés par la proximité floue n'atteint pas 1000, nous complétons la liste avec les documents retournés par LUCY qui ne font pas déjà partie de la liste.

Pour la tâche Robuste, le fichier de réponses doit en plus contenir un classement des requêtes. Il s'agit de prédire le classement auquel on aboutirait si les requêtes étaient classées selon leur performance de la meilleure à la moins bonne. Nous avons associé à chaque requête, le score du premier document renvoyé, et nous avons classé les requêtes dans l'ordre décroissant en fonction des valeurs de ces scores.

5.3.1.2 Variations de la zone d'influence

Les résultats de toutes les expériences, illustrées dans la figure 5.6, montrent que la proximité floue obtient les meilleures performances jusqu'à 50% de rappel dans l'expérience **l200pf**

pour des requêtes manuelles avec une zone d'influence large ($k = 200$). Les deux dernières

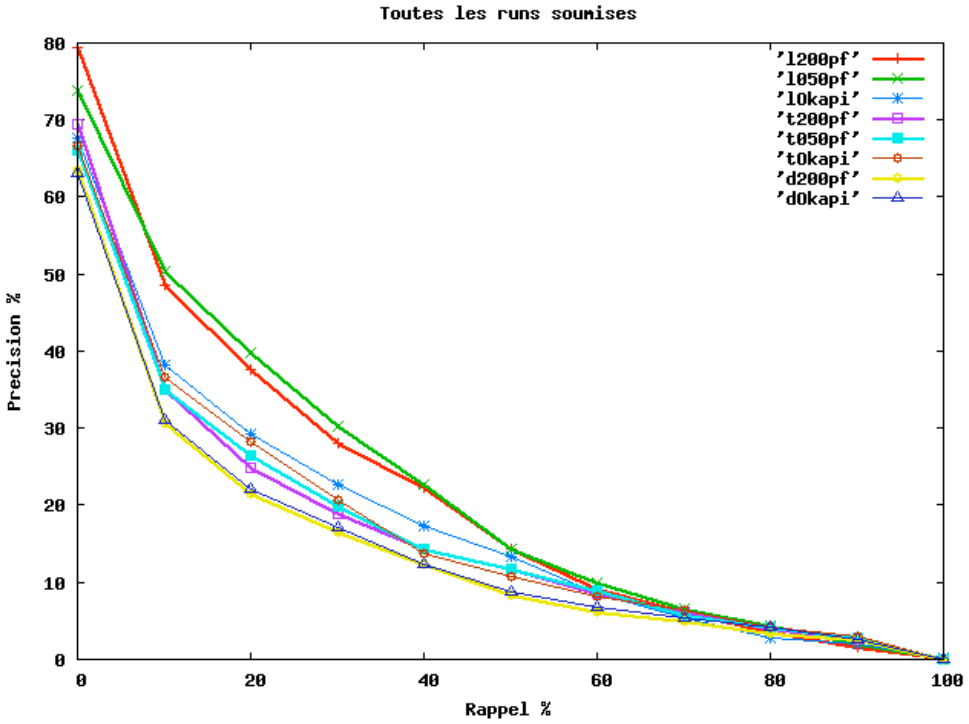


FIG. 5.6 – Aquaint 2005 - ensemble des *runs* soumises comparées à la méthode de référence de Okapi LUCY

courbes **d200pf** et **dOkapi** correspondent au jeu de requêtes construites automatiquement sur le contenu du ' champ « description » avec un processus basé sur le langage naturel. Ces résultats ne remettent pas en cause la méthode d'extraction de mots-clés utilisée car la faible performance n'est pas étonnante. La figure 5.7 montre ces deux courbes seules pour plus de lisibilité. Pour ces deux courbes, nous constatons que l'utilisation de la proximité dégrade les résultats de LUCY, nous pouvions dans une certaine mesure nous attendre à ce résultat car la proximité floue n'est pas encore adaptée aux requêtes comportant beaucoup de termes. Une solution pourrait être mise en place en décrivant des parties optionnelles dans les requêtes. Cette solution vise à trouver un compromis pour effectuer un rappel plus important de documents.

Les résultats utilisant le jeu de requêtes automatiquement construites sur le titre du topique sont illustrés dans la figure 5.8. La méthode de LUCY est meilleure que la notre avec la valeur de k égale à 50 et même 200 sauf au premier niveau de rappel. Au premier niveau de rappel, la précision de notre méthode est plus grande que celle de LUCY si nous prenons $k = 200$, en effet, avec une plus large zone d'influence la proximité floue donne de meilleurs résultats.

Le jeu de requêtes, construites manuellement, étant plus adapté à notre méthode, nous permet d'obtenir plus largement de meilleurs résultats (cf. figure 5.9). La différence entre la proximité floue et la méthode Okapi de LUCY est de l'ordre de 10% sur les trois premiers niveaux

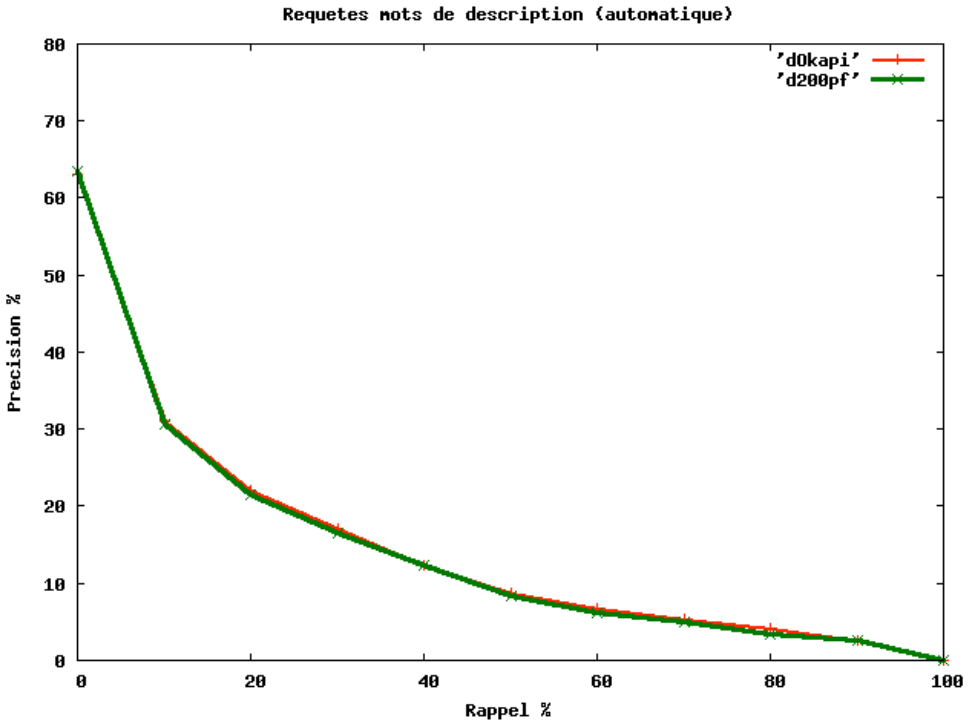


FIG. 5.7 – Aquaint 2005 - rappel précision - requêtes construites automatiquement à partir du champ description.

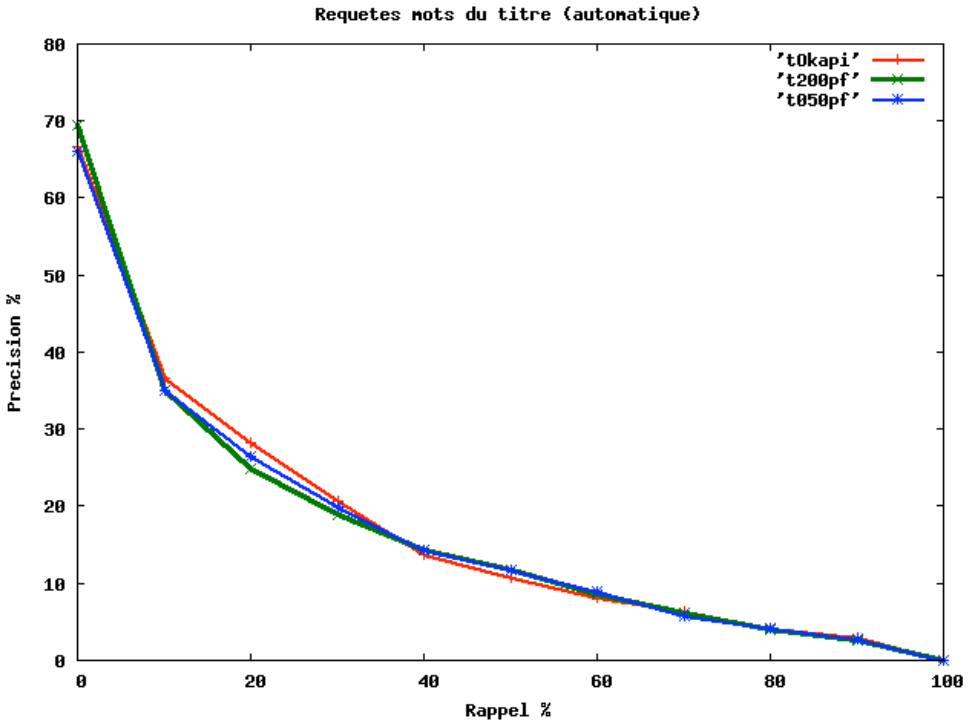


FIG. 5.8 – Aquaint 2005 - rappel précision - requêtes construites automatiquement à partir du champ titre.

de rappel puis diminue de moitié pour tendre vers 0 à partir de 60% de rappel. A partir de ce dernier niveau, les trois méthodes se rejoignent ce qui est tout à fait normal puisque nous avons complété notre liste de documents avec celle de LUCY quand la méthode de proximité floue n'obtient pas assez de réponses. Notre méthode est au-dessus de celle de LUCY à tous les niveaux de rappel, cependant les courbes pour $k = 50$ et $k = 200$ se croisent plusieurs fois. Contrairement aux requêtes automatiques construites à partir du titre, la précision de notre méthode de proximité floue en utilisant des requêtes manuelles, est donc plus grande que la précision de LUCY même avec $k = 50$. En effet, les requêtes écrites à la main permettent de compléter et préciser le besoin d'informations, le système sélectionne ainsi plus de documents avec notre méthode et peu de topiques sont complétés à l'aide de résultats de LUCY. Cette expérience, nous montre aussi que si la zone d'influence est plus grande, le nombre de documents sélectionnés par la proximité floue est plus important et la précision est meilleure au premier niveau de rappel.

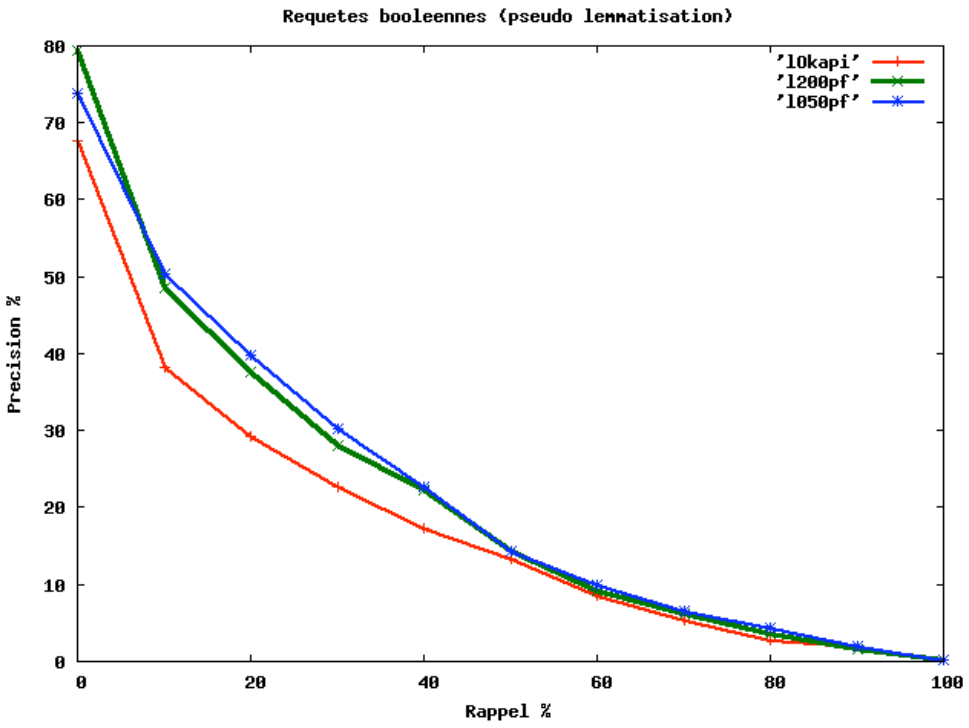


FIG. 5.9 – Aquaint 2005 - rappel précision - requêtes construites manuellement exploitant au mieux la proximité floue

Dans la section suivante, nous nous intéressons aussi à l'impact de la valeur de k sur les résultats mais nous nous concentrons plus particulièrement sur l'incidence de l'utilisation ou non de la lemmatisation (à l'indexation ou à l'interrogation). Le comportement de la proximité floue est étudié sur une seconde collection de test.

5.3.2 Tâche FR Adhoc CLEF 2005

Avant d'étudier l'impact de la lemmatisation, nous exposons les résultats que nous avons obtenus avec la méthode de la proximité floue pour la tâche « Adhoc Monolangue » en français à l'occasion de la campagne d'évaluation CLEF 2005[21].

Comme nous avons pris en exemple dans la section 5.1.3.1, la collection de test de CLEF 2005 pour décrire le principe général de la construction des requêtes, nous ne la détaillerons pas ici. Par ailleurs, la construction des listes de réponses est similaire à celle adoptée dans la tâche Robuste (collection AQUAINT) excepté le classement des topiques qui n'est pas demandé ici.

5.3.2.1 Résultats de la campagne CLEF 2005

Nous avons soumis différentes *runs* officielles basées sur des :

1. requêtes automatiques construites à partir des mots du titre avec $k = 20$ (run `t020pf`) et avec $k = 50$ (run `t050pf`);
2. requêtes manuelles construites à partir des mots des trois champs (titre, description et narration) avec $k = 50$ (run `l050pf`) et avec $k = 80$ (run `l080pf`).

Pour la campagne d'évaluation, aucune procédure de lemmatisation n'a été utilisée au moment de l'indexation (*runs* t$kpf ou tOkapi), par contre, nous considérons les requêtes manuelles comme pseudo-lemmatisées car des dérivés des mots-clés sont souvent utilisés dans des disjonctions de termes (*runs* l$kpf ou lOkapi). Un récapitulatif des *runs* est disponible dans le tableau 5.3.

	Requêtes titres automatiques	Requêtes tous champs manuelles
Okapi Lucy	tOkapi	lOkapi
Proximité floue $k = 50$	t50pf	l50pf
Proximité floue $k = 200$	t200pf	l200pf

TAB. 5.3 – Runs présentées à CLEF 2005

Pour les *runs* `tOkapi` et `lOkapi` utilisant la méthode Okapi, les requêtes sont plates (sac de termes). Les résultats sont produits avec le système LUCY original ce qui permet d'obtenir des *runs* de référence pour les comparer à notre méthode.

La table 5.4 présente la précision aux 11 points de rappel obtenue pour les expériences soumises : d'une part, les requêtes conjonctives (t$kpf) avec les mots du titre pour $k = 50$ et $k = 20$ et les requêtes manuelles (l$kpf) avec $k = 20$ et $k = 50$ et d'autre part, celles relatives à la méthode de référence de LUCY (tOkapi ou lOkapi).

[21] http://clef.isti.cnr.it/

	Requêtes automatiques (titre)			Requêtes manuelles (pseudo-lemmatisation)		
Rappel	tOkapi	t50pf	t20pf	lOkapi	l80pf	l50pf
0	**62**	*59*	57	68	**70**	*68*
10	**45**	*44*	44	**49**	*49*	48
20	**33**	32	*33*	39	**41**	*41*
30	**26**	25	*25*	31	*33*	**33**
40	21	*21*	**21**	25	*28*	**28**
50	19	*19*	**19**	21	**22**	*21*
60	**14**	14	*14*	17	**18**	*18*
70	*11*	11	**11**	13	**14**	*14*
80	7	**8**	*8*	8	**10**	*10*
90	4	4	4	5	**6**	*6*
100	1	1	1	1	*1*	**1**

TAB. 5.4 – Expériences officielles avec les requêtes construites manuellement ou de manière automatique. Les colonnes montrent la précision pour `tOkapi`, `t050pf`, `t020pf`, `lOkapi`, `l080pf`, et `l050pf`. En gras, le *meilleur* résultat, et, en italique le *second*.

Les valeurs de k ont été choisies en fonction des résultats que nous avions obtenus auparavant sur la collection WT10g. Finalement, les valeurs de k choisies pour nos soumissions utilisant les requêtes « automatiques » ne nous ont pas permis d'obtenir de meilleurs résultats pour notre méthode par rapport à l'implantation de la méthode BM-25 de LUCY. En revanche, les requêtes que nous qualifions de « manuelles » et qui permettent d'effectuer une sorte de lemmatisation à la main permettent d'exploiter au maximum notre méthode en atteignant pour l'ensemble des documents retournés un compromis entre rappel et précision. Le besoin d'informations peut être facilement détaillé dans le langage de requêtes booléen. De même qu'avec la collection AQUAINT, la colonne de droite de la table 5.4 montre que les performances de la proximité floue sont meilleures ou du moins égales à celles de LUCY avec des requêtes manuelles.

Bien que les jeux de requêtes « automatiques » ne favorisent pas notre méthode, nous nous efforçons de les présenter. En effet, la comparaison de résultats obtenus avec des requêtes différentes du point de vue de la structure (booléenne vs. plates) semble avantager les méthodes utilisant des langages complexes. De ce fait, les systèmes basés sur des requêtes plates peuvent être défavorisés car les requêtes ne sont pas aussi précises que dans une expression booléenne. Par conséquent, dans nos expérimentations, nous avons toujours présenté des résultats reprenant en simple conjonction ou disjonction les termes des requêtes plates construites à partir des mots du titre ou de la description.

Les campagnes d'évaluation restreignent le nombre d'expériences à soumettre. Cependant, nous avons utilisé la proximité floue avec de plus grandes valeurs de k (100 et 200) afin d'étudier l'impact de l'élargissement de la zone d'influence. La table 5.5 montre, comme nous l'avions constaté avec la tâche Robuste, que plus la zone d'influence d'un terme est large, meilleurs

	Requêtes automatiques (titre)			Requêtes manuelles (pseudo-lemmatisation)		
Rappel	tOkapi	t100pf	t200pf	lOkapi	l100pf	l200pf
0	**62**	60	*61*	68	**72**	*71*
10	**45**	*44*	43	49	*50*	**51**
20	**33**	33	*33*	39	*40*	**41**
30	**26**	*26*	26	31	*33*	**34**
40	21	*21*	**21**	25	*28*	**28**
50	**19**	*19*	19	21	**22**	*22*
60	**15**	*14*	14	17	*18*	**18**
70	**11**	*11*	11	13	**14**	*14*
80	7	**8**	*8*	8	*10*	**10**
90	**4**	*4*	4	5	**6**	*6*
100	*1*	**1**	1	1	**1**	*1*

TAB. 5.5 – *Runs* non officielles avec les requêtes construites manuellement ou de manière automatique. En gras, le *meilleur* résultat, et, en italique le *second*.

sont les résultats. Nous attribuons ce résultat positif à la sélection de plus de documents avec notre méthode. La proximité des termes dans les documents intervient beaucoup plus dans la constitution de la liste de réponses car le nombre de documents issus du modèle Okapi est réduit voire inexistant pour de nombreuses requêtes.

5.3.2.2 Utilisation de requêtes totalement disjonstives

Cette expérience illustre le point de la section 4.5.1 et simule le comportement de la méthode située entre le niveau de coordination et le modèle vectoriel (comptage de la fréquence des termes de la requête dans le document), nous avons utilisé le jeu de requêtes construites automatiquement avec le contenu du champ titre. Pour Okapi LUCY, les requêtes sont plates, pour la proximité floue les requêtes sont exclusivement des disjonctions de termes. La proximité floue fournit les meilleurs résultats quand la valeur de k est la plus petite comme illustré dans la figure 5.10. En employant des requêtes exclusivement disjonctives, les performances de notre méthode sont bien en dessous des méthodes à base d'intervalles et de Okapi Lucy, la meilleure valeur de précision est obtenue au premier niveau de rappel atteint seulement 30%. Ce comportement est tout à fait normal car en utilisant une disjonction de terme comme requête avec comme ici une zone d'influence très réduite un document possédant tous les termes proches n'est pas distingué d'un document ayant le même terme plusieurs fois dans le document. Pour $k = 1$, le classement est alors établi selon le nombre d'occurrences de termes trouvés dans les documents. Dans ce cas, il est donc compréhensible que la méthode Okapi, basé en partie sur le facteur df, ou les méthodes à intervalles prenant d'abord en compte les documents possèdant tous les termes de la requête puissent fournir de meilleurs résultats que la proximité floue. Notre « pseudo-méthode »

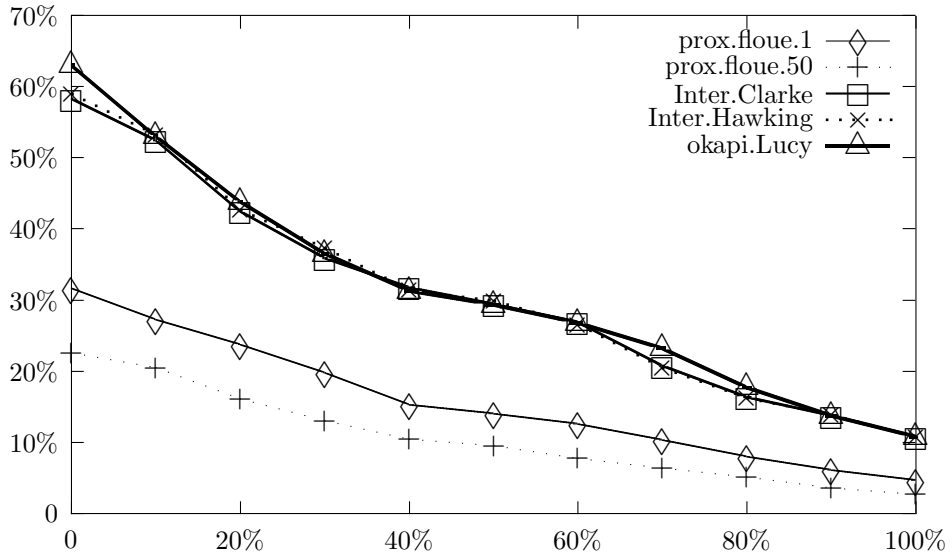


FIG. 5.10 – CLEF 2005 - rappel précision - requêtes disjonctives avec seulement les termes du titre

de coordination est moins bonne que la méthode Okapi comme il est classiquement admis.

5.3.3 Impact de l'utilisation de la lemmatisation à l'indexation

Bien que nous essayons de pallier l'absence de lemmatisation à l'aide de requêtes manuelles introduisant les dérivés des mots-clés à la construction des requêtes, le manque de lemmatisation dès l'indexation peut être vu comme une lacune. Ainsi, se pose la question suivante : la lemmatisation « manuelle » au moment de la construction de la requête est-elle une bonne solution ou faut-il quand même ajouter un pré-traitement à la collection pour indexer les lemmes plutôt que les mots sous leur forme originale ? Pour répondre à cette question, nous étudions dans ce qui suit différents cas de figure :

1. collection non lemmatisée requêtes conjonctives[22] vs. requêtes manuelles[23] ;
2. collection non lemmatisée requêtes manuelles vs. collection lemmatisée requêtes conjonctives ;
3. collection lemmatisée requêtes conjonctives vs. requêtes manuelles ;
4. collection non lemmatisée requêtes manuelles vs. collection lemmatisée requêtes manuelles.

[22] Conjonctions des mots non vides du champ titre.

[23] Disjonctions de conjonctions des mots des trois champs avec éventuellement des formes dérivées, des synonymes et des termes spécialisant ou généralisant le besoin d'informations.

C'est dans le dernier cas, qui peut sembler singulier, que la différence entre la lemmatisation à l'indexation et la pseudo-lemmatisation à l'interrogation peut être mise en relief. Un récapitulatif des *runs* est disponible dans le tableau 5.6

Indexation / Interrogation	Collection non lemmatisée	Collection lemmatisée
Titre	NoStemt (Okapi, 100pf, 200pf)	t (Okapi, 100pf, 200pf)
Pseudo-lemmatisation	NoSteml (Okapi, 100pf, 200pf)	t (Okapi, 100pf, 200pf)

TAB. 5.6 – Runs pour la comparaison entre la lemmatisation à l'indexation et la pseudo-lemmatisation à l'interrogation - CLEF 2005

Tournons nous, tout d'abord vers les résultats de la méthode de référence LUCY Okapi. Dans le cas de l'interrogation sur la collection avec lemmatisation à l'indexation, nous avons supprimé les lemmes obtenus en double pour n'en garder qu'un seul dans les requêtes créées automatiquement. Dans la suite, quand nous parlons de requêtes manuelles booléennes, nous considèrerons implicitement que la pseudo-lemmatisation est réalisée « à la main » au moment de l'interrogation. De plus, **NoStem** devant la dénomination d'une expérience, signifie qu'il n'y a pas de lemmatisation faite à l'indexation. La figure 5.11 montre que l'expérience :

1. **NoStemtOkapi**, interrogation avec les mots du titre sur la collection sans lemmatisation fournit les résultats les moins bons ;
2. **NoStemlOkapi**, méthode avec pseudo-lemmatisation est meilleure que **lOkapi** au premier niveau de rappel, croise ensuite la courbe **tOkapi**, utilisant les mots du titre sur la collection lemmatisée et reste largement en-dessous jusqu'à 100% de rappel ;
3. **tdOkapi** est la meilleure au premier niveau de rappel, passe seconde au deuxième niveau de rappel et devient meilleure ou équivalente à **lOkapi** à partir de 20% de rappel.

Nous constatons, grâce à cette première analyse, que lemmatiser la collection présente un avantage car notre pseudo-lemmatisation à l'interrogation ne fournit pas d'aussi bons résultats.

Nous pouvons ensuite étudier sur les mêmes types d'expériences l'impact de la lemmatisation sur la méthode de proximité floue. Etant donné que c'est avec les requêtes manuelles que nous obtenons les meilleurs résultats pour notre méthode, nous pouvons faire l'hypothèse que les écarts seront moins importants. Les chiffres des courbes précisions rappel sont illustrés dans les tables 5.15 et 5.16.

La figure 5.12 montre que l'expérience :

1. **NoStemt100pf** fournit les moins bons résultats ce qui rejoint les remarques faites pour la méthode de LUCY ;
2. **NoSteml100pf** est très proche de **l100pf** jusqu'à 20% de rappel, ensuite, l'expérience sans lemmatisation à l'indexation passe largement en-dessous à partir de 30% ;
3. **td100pf** possède la meilleure précision à partir de 30% de rappel ;

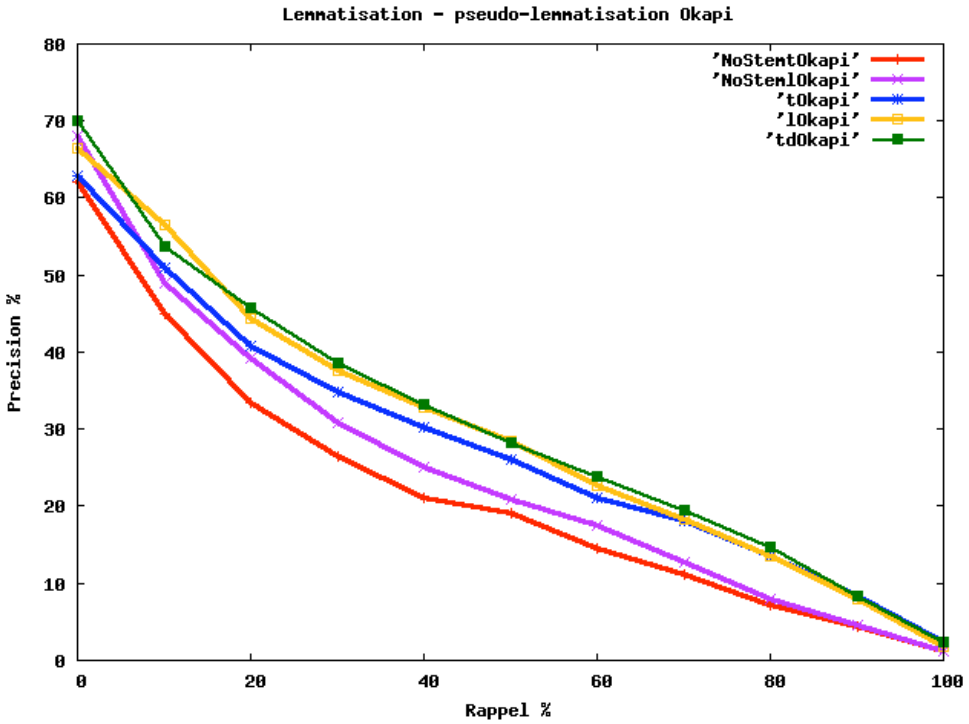


FIG. 5.11 – Okapi LUCY- Résultats obtenus pour la lemmatisation à l'indexation (tOkapi, lOkapi, tdOkapi) et sans lemmatisation (NoStemtOkapi, NoStemlOkapi).

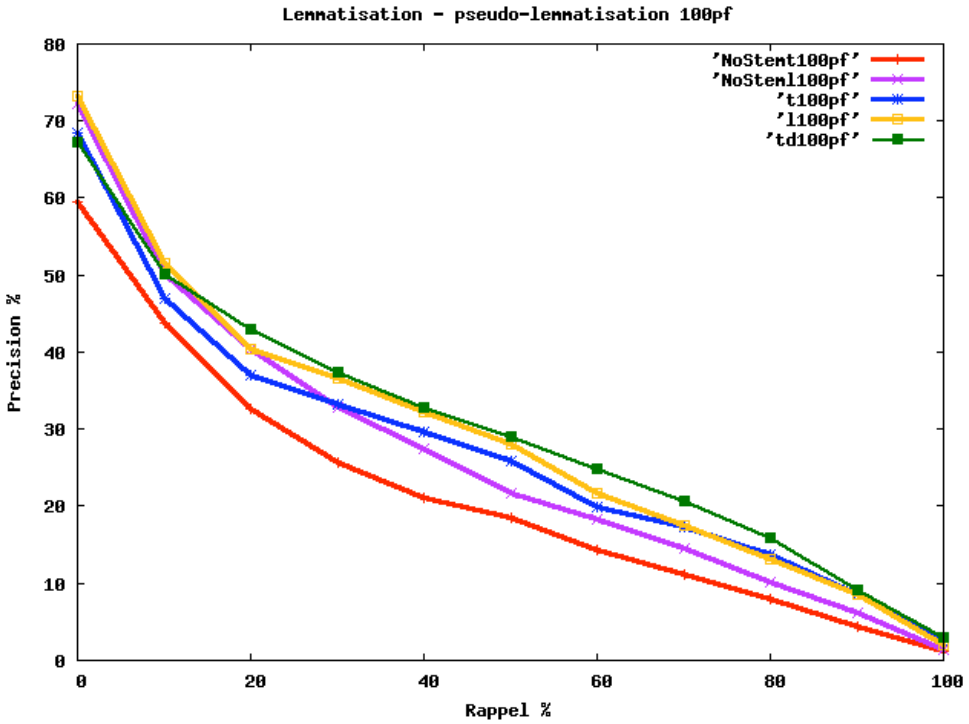


FIG. 5.12 – Proximité floue $k = 100$ - Lemmatisation à l'indexation (t100pf, l100pf, td100pf) ou non (NoStemt100pf, NoStemml100pf).

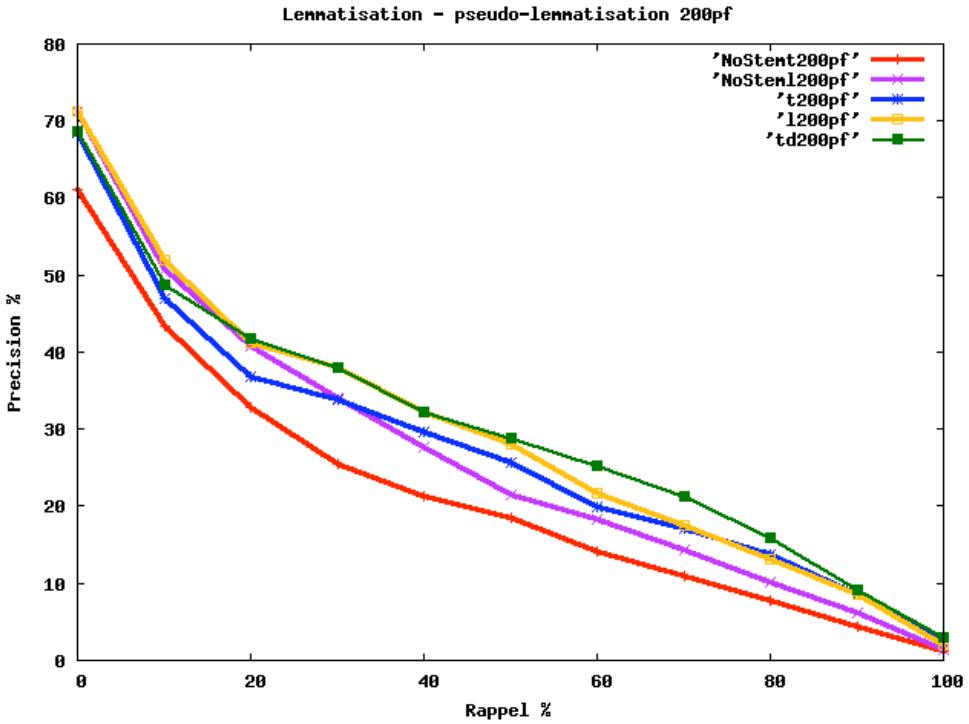


FIG. 5.13 – Proximité floue $k = 200$ - Lemmatisation à l'indexation (t200pf, l200pf, td200pf) ou non (NoStemt200pf, NoStemml200pf).

4. **t100pf**, interrogation avec les mots du titre sur la collection lemmatisée fournit des résultats moyens et passe au dessus de l'ensemble des expériences « NoStem » qu'à partir de 30% de rappel.

Les résultats pour la proximité floue avec $k = 200$ sont illustrés dans la figure 5.13. Nous remarquons que les courbes ont la même allure que celles que nous venons de décrire. De 0% à 30% de rappel, les requêtes manuelles sont les meilleures. Ensuite, la lemmatisation à l'indexation fait la différence puisque l'expérience sans lemmatisation passe $4^{\text{eme}}/5$ bien en dessous des trois autres courbes, même celle utilisant simplement les conjontions des termes du titre. Contrairement à notre première hypothèse, les différences sont plus marquées car l'utilisation de la lemmatisation à l'indexation permet d'augmenter de manière significative la précision aux faibles niveaux de rappel (**td200pf** de 50% à 90% et **l200pf** de 50% à 70%).

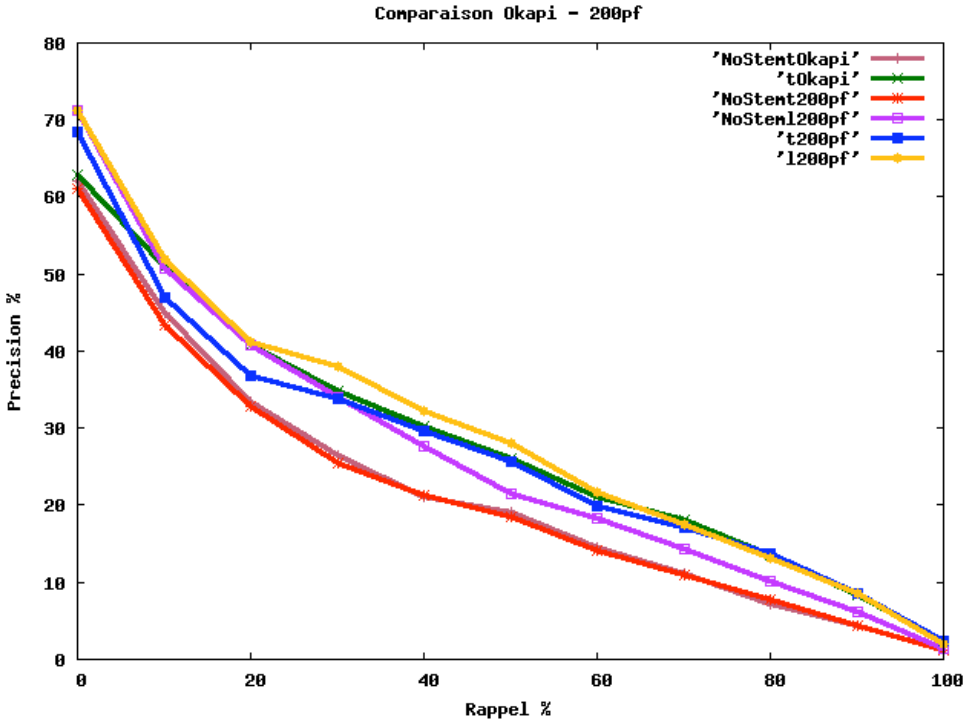


FIG. 5.14 – Méthode Okapi LUCY et proximité floue $k = 200$ - Lemmatisation à l'indexation (t200pf, l200pf) ou non (NoStemtOkapi, NoSteml200pf).

Pour terminer, nous comparons la méthode de proximité floue à celle de LUCY. Dans la figure 5.14, nous considérons **NoStemtOkapi** comme méthode de base car l'interrogation de la collection non lemmatisée avec les mots du titre y est réalisée. Par rapport à cette expérience de base, nous étudions quelle est la meilleure solution entre la lemmatisation à l'interrogation ou celle à l'indexation. Pour ces deux solutions, nous obtenons des améliorations pour la précision. Selon le type de lemmatisation, les performances sont meilleures soit aux trois premiers niveaux de rappel (**lNoStem**), soit aux autres niveaux (**t200pf**). Le gain maximum est obtenu grâce à des requêtes manuelles sur la collection lemmatisée : c'est encore mieux car la précision est

	NoStemt		t		l		
Rappel	100pf	200pf	100pf	200pf	100pf	200pf	Okapi
0	59.5	61.05	68.45	68.42	73.21	71.34	66.56
10	43.72	43.48	46.89	46.89	51.59	51.96	56.47
20	32.54	32.77	37.08	36.81	40.41	41.16	44.29
30	25.61	25.53	33.19	33.84	36.6	38.05	37.68
40	21.17	21.25	29.74	29.6	32.17	32.21	32.74
50	18.56	18.51	25.91	25.76	28.05	28.02	28.43
60	14.34	14.05	19.9	19.84	21.73	21.77	22.77
70	11.19	10.95	17.3	17.13	17.6	17.44	18.35
80	7.89	7.74	13.79	13.78	13.18	13.17	13.56
90	4.41	4.37	8.58	8.52	8.57	8.49	7.94
100	1.21	1.21	2.3	2.3	1.92	1.92	1.86

FIG. 5.15 – CLEF 2005 - Rappel Précision - Lemmatisation à l'indexation ou pseudo-lemmatisation à l'interrogation

	NoSteml			Okapi		td		
Rappel	100pf	200pf	Okapi	t	NoStemt	100pf	200pf	Okapi
0	72.24	71.27	68.08	62.9	62.1	67.33	68.75	70.04
10	50.14	50.79	49.04	51	44.96	50.24	48.79	53.69
20	40.43	40.71	39.25	40.87	33.41	43.03	41.86	45.85
30	32.84	34.07	30.82	34.78	26.38	37.35	37.92	38.7
40	27.56	27.71	25.17	30.28	21.1	32.88	32.3	33.17
50	21.6	21.57	20.8	26.06	19.14	28.96	28.81	28.34
60	18.27	18.29	17.46	21.11	14.62	24.89	25.35	23.82
70	14.43	14.41	12.75	18.03	11.23	20.79	21.23	19.48
80	10.19	10.23	8.04	13.61	7.26	15.93	15.91	14.81
90	6.26	6.16	4.63	8.29	4.42	9.2	9.15	8.42
100	1.3	1.3	1.23	2.31	1.21	2.91	2.91	2.33

FIG. 5.16 – CLEF 2005 - Rappel Précision - Lemmatisation à l'indexation ou pseudo lemmatisation à l'interrogation

améliorée à la fois aux premiers niveaux de rappel mais décrémente moins rapidement pour les suivants.

La synthèse de ces résultats est disponible dans la table 5.7. La conclusion principale est que nous constatons quand même des différences entre les expériences utilisant des requêtes manuelles avec ou sans lemmatisation à l'indexation. Ce résultat n'est pas étonnant car il existe une différence entre *stem* et lemme. Le *stem*, obtenu par le système, est différent du résultat qu'une personne obtient en appliquant la racinisation. Les algorithmes mis en œuvre ne sont pas parfaits et ne ramènent pas toujours les mots à leur entrée dans le dictionnaire. De plus, même si tel était le cas, des mots de la même famille n'aboutirait pas forcément à la même racine (forme verbale vs. forme nominale), ce qui n'est pas forcément un inconvénient car la lemmatisation conduit dans une certaine mesure à l'accentuation de phénomène de polysémie. A partir de cette expérience, nous pouvons déduire d'une part, que lemmatiser une collection de test à l'indexation permet d'augmenter le rappel et la précision moyenne, en grande partie car plus de documents sont retrouvés, et d'autre part, qu'utiliser la pseudo-lemmatisation à l'interrogation permet d'améliorer la précision et les performances en général grâce au retour de plus de documents puisque les différentes formes des mots de la requêtes permettent d'inclure dans les réponses des documents dont le sujet correspond au topique original.

	Collection sans lemmatisation		Collection avec lemmatisation	
	Titre auto	Manuelle	Titre auto	Manuelle
	- -	++		
		++ 1er niveau	++ sur le reste	
			- -	++
		- -		++

TAB. 5.7 – Synthèse sur la lemmatisation à l'interrogation ou à l'indexation

5.3.4 Tâche FR Adhoc CLEF 2006

Pour l'édition 2006 de la campagne d'évaluation CLEF nous avons donc lemmatisé les documents de la collection avant la phase d'indexation. Deux autres changements sont mis en œuvre. En premier, nous utilisons la fonction « adhoc » (cf. figure 4.3) pour représenter l'influence d'une occurrence de mot ; en second, les listes de réponses sont fusionnées de manière différentes pour tenir compte des documents retournés avec plusieurs valeurs de k. Pour ce faire, nous prenons d'abord, les documents retournés pour une valeur de $k = 200$ puis $k = 100$ puis 80, 50 20 et 5. Bien sûr, comme dans les autres expériences, dans le but d'avoir un rappel suffisant nous ajoutons, si besoin, des documents provenant de la liste de réponses retournée par la méthode okapi de LUCY. Nous avons soumis les expériences suivantes :

1. conjonction de termes issus du titre **RIMAM06TL** ;
2. conjonction de termes issus de la description **RIMAM06TDNL** et ;

3. requêtes manuelles construites à partir de tous les champs **RIMAM06TDML**.

Afin de pouvoir déterminer l'impact de notre fonction « adhoc », l'expérience **RIMAM06TDMLRef** utilise un fonction d'influence triangulaire. Nous ne pouvons pas commenter les résultats puisqu'ils ne nous ont pas encore été communiqués.

5.4 La proximité floue sur la collection TERABYTE

La collection de test « *terabyte* » est, en 2006, la collection la plus volumineuse disponible pour les expérimentations. Bien qu'elle s'appelle « *terabyte* », elle ne fait que $426Go$ de mémoire ce qui donne $80Go$ compressée. Cette collection n'est par conséquent pas distribué sur les CDs traditionnels mais sur un disque physique. Après nettoyage des documents de la collection, la masse de données textuelles que nous avons à indexer ne tient plus que sur $40Go$. Nous avons tout d'abord essayé d'indexer et d'interroger la collection avec l'outil LUCY mais la taille du vocabulaire présent dans l'index n'a pas pu être chargée en mémoire sur notre serveur. Afin de finaliser nos expérimentations nous avons adopté la solution suivante :

1. pour obtenir le score Okapi de référence, nous avons indexé la collection avec ZETTAIR en utilisant la procédure de lemmatisation basée sur l'algorithme de Porter. L'utilisation de la lemmatisation permet de réduire encore la taille du vocabulaire. Nous avons facilement réussi à interroger la collection avec des requêtes construites de manière automatique et manuelle.
2. pour obtenir le score issu de notre méthode de proximité floue, nous avons indexé la collection de manière controlée. Au moment de l'indexation, nous chargeons un dictionnaire contenant tous les mots-clés du fichier de requête et ne retenons dans l'index que les informations relatives aux mots de ce dictionnaire. Cette opération nous permet de réduire le vocabulaire indexé ainsi que la taille du dictionnaire chargé en mémoire à l'interrogation ce qui nous permet effectivement d'interroger la collection avec notre méthode.

Pour suivre les directives de la tâche TERABYTE, nous avons construit des requêtes automatiques et manuelles. Chaque participant devait faire fonctionner son système sur l'ensemble des « topiques » créés depuis 2004. Nous avons donc construit, de manière automatique pour les 150 « topiques », des requêtes basées sur les mots non vides du titre et, de manière manuelle (comme il était fortement conseillé) pour les 50 nouveaux « topiques » des requêtes basées sur tous les champs. Les résultats ne nous ont pas encore été communiqués. Comme la collection utilisée pour les campagnes d'évaluation 2004 et 2005 était la même et que nous avons construit les requêtes pour l'édition 2006, nous présentons dans la figure 5.17, les résultats des requêtes automatiques sur le titre avec $k = 50$ et $k = 200$. Comme pour les résultats obtenus sur les autres collections, nous remarquons que la précision interpolée au premier niveau de rappel est meilleure pour la plus grande valeur de k. Cependant, les courbes de rappel/précision restent proches pour chaque campagne. Notons que ces résultats ont été obtenus pour des requêtes constituées de

conjonctions des mots du titre. Les résultats des requêtes utilisant tout le potentiel de notre méthode ne nous étant pas encore parvenus, nous ne pouvons pas conclure sur l'efficacité de notre méthode.

5.5 Bilan

Tout d'abord, à partir de nos premières expériences sur les méthodes à intervalles et le passage à l'échelle, nous avons déterminé que l'utilisation de la proximité peut améliorer l'efficacité en termes de précision. Nous avons ensuite mis en œuvre le modèle que nous avons élaboré pour prendre en compte la proximité des termes dans la fonction de correspondance de notre système. Enfin, nous avons participé aux campagnes d'évaluation CLEF et TREC afin de situer les résultats de notre approche de proximité floue par rapport à la méthode de référence Okapi. Comme attendu, le système que nous avons construit est un système à haute précision et permet d'améliorer les résultats d'une recherche d'informations au plus haut de la liste des réponses. De plus, la visualisation possible des résultats reflétant la proximité des termes spécifiés dans les documents offre à notre méthode un avantage supplémentaire.

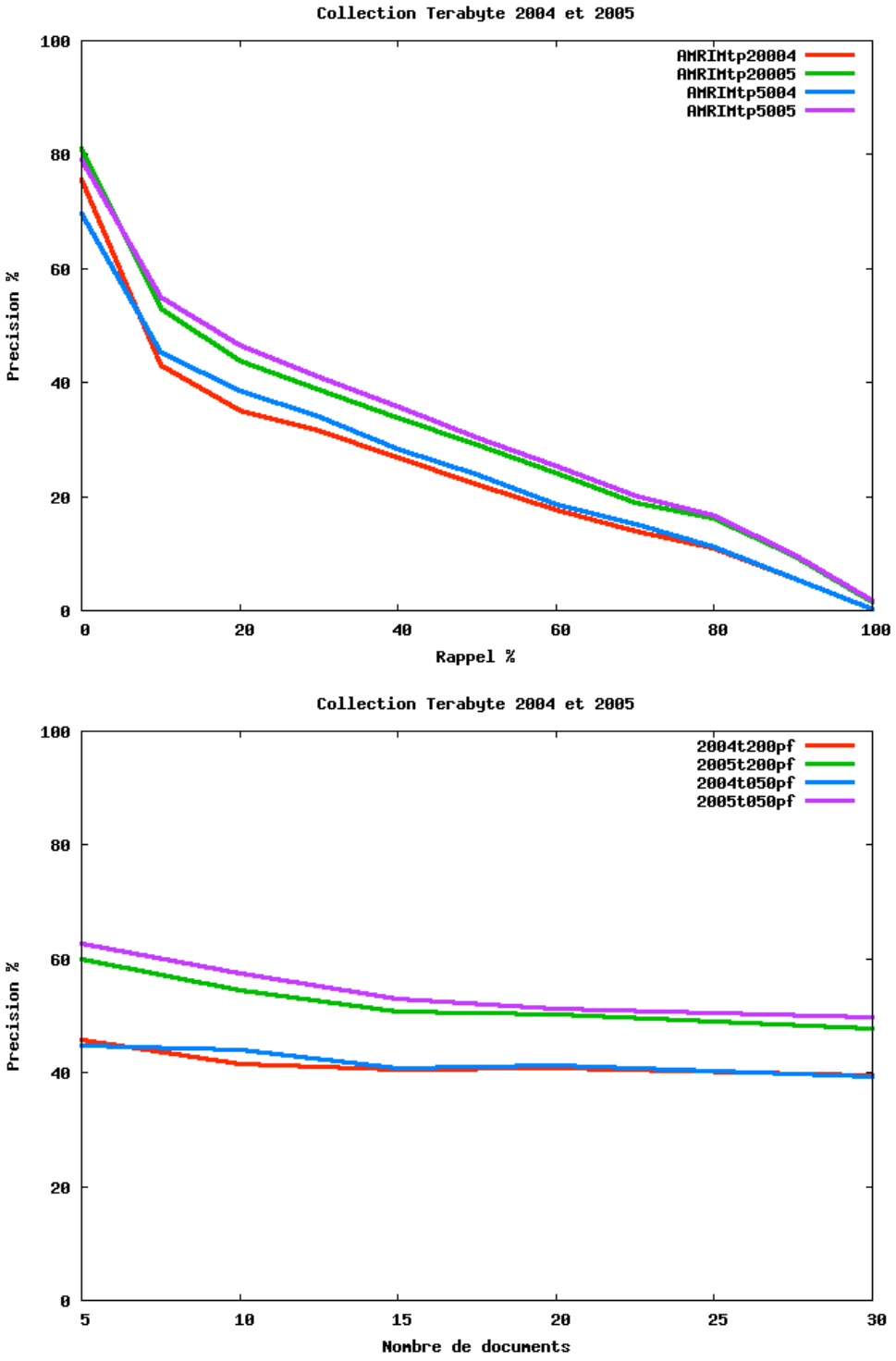


FIG. 5.17 – Résultats obtenus pour la tâche TERABYTE pour les requêtes (titre, automatique) des éditions 2004 et 2005

Chapitre 6

Conclusions et perspectives

Notre travail, situé dans le cadre de la recherche d'informations textuelles, a visé à étudier l'utilisation de la proximité des mots dans le texte pour l'amélioration des performances par rapport aux méthodes classiques. Dans notre approche, nous avons privilégié « la haute précision » ce qui correspond dans le domaine des moteurs de recherche sur la Toile, à présenter à l'utilisateur une partie des documents considérés comme ayant le plus de chance d'être pertinents plutôt qu'à présenter tous les documents pertinents sans forcément concentrer les meilleurs, en tête de la liste de réponses.

Nous avons commencé ce mémoire par la présentation des méthodes classiques (modèles booléen, à ensembles flous, vectoriel et probabiliste) et de méthodes moins classiques utilisant la proximité (méthodes à intervalles, recherche de passage, approches « signal »), ensuite nous avons défini la méthode de proximité floue ainsi que sa manière originale de visualiser les documents et enfin nous avons présenté les diverses expériences réalisées (passage à l'échelle des méthodes à intervalles, test de la proximité floue sur les collections WT10G, AQUAINT, CLEF et TERABYTE).

Il est temps maintenant de tirer les conclusions sur chaque point fondateur de notre approche, d'en établir les limites et de proposer des perspectives pour repousser ces limites.

6.1 Formuler les besoins d'informations

La plupart des systèmes utilisent des langages du type « sac de termes » et les résultats qu'ils produisent sont comparables. Cependant, pour notre approche, nous avons fait le choix d'un langage basé sur des expressions booléennes, il devient alors difficile de comparer nos résultats aux méthodes de références telles que Okapi ou le modèle vectoriel. Pour palier ce problème, nous avons toujours comparé notre approche avec des requêtes conjonctives prenant

simplement les mots vides du titre et les requêtes plates correspondantes pour les méthodes de référence. Néanmoins, pour exploiter au maximum notre méthode, nous avons également créé des requêtes booléennes plus complexes pour élargir les requêtes seulement conjonctives dans la plupart des cas et augmenter ainsi le rappel. Nous avons ainsi choisi le modèle booléen puisque d'une part, il est propice à la propagation de la proximité au moment de l'évaluation de l'arbre de requête et, d'autre part, il permet de formuler avec plus de précision le besoin d'informations par rapport aux requêtes plates. Néanmoins, nous nous sommes rendus compte au cours de campagnes d'évaluation de certaines limites de ce langage par rapport au modèle « sac de termes » :

- il manque la possibilité de spécifier des expressions, l'utilisateur précise entre guillemets une expression du vocabulaire que les documents doivent contenir comme par exemple, « pince à crabe » ;
- il est impossible de construire automatiquement des requêtes booléennes correspondant au besoin d'informations. Nous pouvons évidemment transformer les requêtes plates en conjonctions ou en disjonctions de termes mais ce genre de transformations ne convient pas. Les requêtes sont, soit trop contraignantes puisque tous les termes doivent être présents, soit imprécises puisque la présence d'au moins un terme de la requête rend le document pertinent.

En conclusion, nous pensons que la première limite peut être résolue en offrant la possibilité de spécifier des expressions. En revanche, pour la seconde, nous n'avons pas de solution pour passer automatiquement à des requêtes booléennes mais seulement quelques perspectives pour assister l'utilisateur dans leur construction.

Tout d'abord, une modification simple de notre langage consiste à y introduire les expressions pour rechercher des informations comme « Sinn Fein » et « Déclaration Anglo-Irlandaise » (topique 339 CLEF 2005). L'objectif est de trouver à l'aide de notre méthode les deux groupes de mots proches plutôt que chacun des mots individuellement comme c'est le cas pour l'instant [1].

Ensuite, pour assister l'utilisateur dans l'expression d'une requête booléenne et le conduire à construire des requêtes à l'image de celles que nous avons utilisées dans nos expériences, nous pensons utiliser des thésaurus ou des dictionnaires. Ces derniers permettraient de proposer de nouveaux mots (dérivés, synonymes, etc.) à insérer dans les clauses disjonctives des expressions booléennes.

Ces deux améliorations sont un point de départ pour repousser ces deux limites, cependant, il serait aussi intéressant de mettre en œuvre l'idée de relation de proximité proposée dans la méthode de Hawking et *al.*. Nous avons vu que les requêtes soumises aux systèmes basés sur la proximité devaient être constituées d'environ trois mots seulement. Avec notre méthode et l'utilisation du modèle de requêtes booléen nous pouvons repousser cela, néanmoins le choix

[1] Par contre, pour le système actuel, une conversion automatique des requêtes plates spécifiant des expressions pourrait également être envisagée et conduire à la création d'un nouveau mode d'interrogation. Nous transformerions implicitement les mots des expressions en conjonctions de termes et la requête finale serait une disjonction des conjonctions ainsi construites.

entre requêtes contraignantes et précises (conjonctives) ou, larges retrouvant de nombreux documents (disjonctives) se pose au moment de la formulation. Il nous semble que la solution serait d'exprimer une requête à l'aide de plusieurs sous-requêtes en donnant ou non un ordre préférentiel entre ces sous-requêtes. Ainsi si la requête entière ne retourne pas assez de documents, les sous-requêtes seraient évaluées au fur et à mesure pour retourner plus de documents. Cette solution se rapprocherait des ensembles de relations de proximité proposés dans le modèle de Hawking et *al.* et permettrait de conserver l'avantage de la formulation du besoin d'informations en expressions booléennes.

6.2 Utiliser différemment les intervalles de mots

Comme nous venons de l'expliquer, toutes les approches ne sont pas forcément comparables. Par exemple, dans notre première étude, nous avons pu comparer les méthodes à intervalles au modèle vectoriel car d'une part, nous avons indexé la collection WT10G avec le même système et d'autre part chacune des méthodes utilise des requêtes plates. En revanche, pour comparer les méthodes à intervalles par rapport à notre méthode basée sur la proximité floue, nous pensons que nous devons adapter ces dernières aux requêtes booléennes[2]. Prenons la requête `(A | B) & C`, nous avons deux solutions :

1. nous recherchons les intervalles qui satisfont la requête, puis nous appliquons le calcul des contributions et du score du document ;
2. nous effectuons une sélection plus large d'intervalles en fusionnant la liste des positions des termes reliés par l'opérateur OU. Pour simplifier, nous ne prenons en compte que le premier niveau, c'est-à-dire seules les listes de positions des termes sur les feuilles de l'arbre de requêtes seront fusionnées. Dans ce cas, nous préférons une forme normale conjonctive pour la requête ce qui signifie que si l'utilisateur propose `(A & C) | (B & C)` , sa requête n'est pas adaptée. Les intervalles sont sélectionnés par rapport aux nouvelles listes de positions puis le calcul des contributions et du score du document est effectué.

Une telle adaptation des méthodes à intervalles permettrait de les comparer à notre approche.

Après notre étude bibliographique, nous avons privilégié la piste de la densité des termes plutôt que celle des intervalles. L'une des raisons est que les premières méthodes à intervalles (Clarke et *al.*, Hawking et *al.*) utilisent deux clés de classement : le nombre de termes de la requête et le score basé sur les intervalles, ce qui peut poser problème pour des systèmes « fédérateurs » utilisant les scores pour classer des ensembles de documents provenant de différents systèmes. En revanche, ce problème ne se pose pas pour les méthodes suivantes (Monz, Rasolofo et *al.*, Song et *al.*) où le calcul basé sur la proximité est intégré à une fonction de similarité

[2] C'est pourquoi les expériences comparant les méthodes à intervalles (avec des requêtes plates) et notre méthode basée sur la proximité floue (requêtes booléennes) n'ont pas été rapportées dans ce mémoire.

globale. Une solution pour unifier ce score serait de prendre en compte tous les intervalles possibles d'un document c'est-à-dire les intervalles contenant k, $k-1$, $k-2$, etc. termes. Pour ce faire, nous pourrions construire l'ensemble des parties des termes de la requête et sélectionner les intervalles correspondant à chaque élément. Ainsi pour la requête, « championne 10000 metres feminin », un document contenant un intervalle avec les mots championne, 10000, metres et un autre avec 10000, feminin serait placé avant un document contenant un intervalle avec les mots championne, 10000, metres. Notre première perspective, nous semblant plus importante, sera l'objet de nos futures réflexions et implantations.

6.3 Prendre en compte la structure des documents

La recherche dans les documents structurés est un domaine en expansion de la recherche d'informations. Pour l'instant, autant les méthodes à intervalles que les méthodes à passages basées sur la densité des termes ne peuvent être directement appliquées aux documents structurés. Bien que les dernières définissent un passage autour de positions dans les documents, elles n'utilisent pas la structure du texte pour calculer la pertinence système. Pour les méthodes à intervalles, nous pouvons envisager deux stratégies utilisant de manière hiérarchique la structure des documents. En premier, la longueur des intervalles peut être redéfinie en fonction des éléments de structures auxquels les mots contenus dans l'intervalle appartiennent, par exemple, un mot dans un titre pourrait être distant de 1 avec un mot dans un titre de section, et distant de 2 avec un mot dans un titre de sous-section, etc. En second, certains intervalles pourraient être mis à l'écart ou être affectés d'un coefficient pour diminuer leur importance. Par exemple, les intervalles constitués de mots n'apparaissant pas dans le même élément de structure pourraient être supprimés ou bien le coefficient diminuant leur importance serait choisi en fonction du nombre d'éléments les séparant. De telles méthodes prenant en compte à la fois la proximité et la structure des documents pour le classement des documents retournés en réponse peuvent être validés à l'aide du corpus de test INEX. Notre méthode de proximité floue a d'ailleurs été étendue pour la campagne d'évaluation INEX 2006 par M. Beigbeder.

6.4 Étudier le passage à l'échelle sous un autre axe

Nous avons brièvement étudié la problématique du passage à l'échelle entre autres pour justifier nos recherches sur la proximité. Nous souhaitons compléter cette étude en se focalisant sur une autre caractéristique que la distribution des documents pertinents pour uniformiser la collection de départ. Le choix de cette caractéristique est lié à la construction de nos requêtes booléennes, en effet, nous souhaitons analyser l'impact du choix des termes de la requête. Un moyen envisageable est d'observer le comportement de notre méthode en fonction de la répartition des termes dans une collection de test. L'uniformisation consisterait à distribuer les documents en fonction des valeurs de tf voire de idf. Nous envisageons deux possibilités pour

le choix des termes à répartir dans les sous-collections, nous pouvons, soit nous restreindre au vocabulaire des requêtes d'une campagne d'évaluation, soit à un ensemble de mots issus d'un thésaurus reflétant par exemple le vocabulaire d'une langue. Cette étude pourrait être réalisée avec la collection TERABYTE qui est pour l'instant la collection de test la plus volumineuse disponible.

6.5 Améliorer notre outil

Tout d'abord, nous pouvons intégrer différents paramétrages à notre système. Nous avons déjà la possibilité de choisir la largeur de la zone d'influence d'un terme, nous souhaitons élargir le choix pour préciser le type de fonction d'influence. Nous pensons à plusieurs types de fonctions d'influence. Les fonctions triangulaires que nous avons utilisées ont la pente de la fonction linéaire $f(x) = x$. Nous pouvons augmenter la pente pour vraiment restreindre la zone d'influence mais il serait intéressant de construire d'autres fonctions, comme celle utilisée pour CLEF 2006, afin d'augmenter fortement la valeur de proximité floue au plus près de l'occurrence et de diminuer cette valeur aux extrémités de la zone d'influence. Par ailleurs, plutôt que de préciser seulement la constante k, des fonctions rectangles pourraient être utiles pour spécifier si les termes doivent se retrouver très proches comme dans une expression, un peu plus éloignés au niveau de la phrase ou encore plus au niveau d'un passage. Ensuite, comme plusieurs fonctions sont disponibles en logique floue pour effectuer l'évaluation des opérateurs aux nœuds de l'arbre de requête, nous pensons ajouter un paramètre pour choisir le type d'évaluation.

Enfin, nous souhaiterions développer une interface pour assister l'utilisateur dans le choix des termes tel que nous l'avons expliqué dans la section 6.1. Également, pour assister l'utilisateur mais au niveau de dépouillement des résultats, la visualisation des documents montrant le détail du score pour toutes les occurrences de mots pourrait être présentée en même temps que la réponse. De plus, notre méthode ne dépendant pas de constantes de collection, les requêtes pourraient être envoyées en parallèle pour plus d'efficacité. La proximité floue pourrait aussi être utilisée dans le cadre de la recherche d'informations distribuée afin d'affiner les résultats fournis par divers serveurs.

6.6 Conclusion générale

Après un bref rappel du processus et des méthodes classiques de recherche d'informations, nous avons montré que la notion de proximité avait déjà été utilisée sous diverses formes : opérateur NEAR, méthodes à intervalles ou méthodes à passage utilisant la densité de termes. Bien que notre approche reprenne certains aspects de ces dernières, son originalité réside dans l'alliance de l'expression des requêtes en langage booléen et du classement avec la proximité.

Nous avons défini un modèle basé sur la proximité des termes qui est une extension du modèle booléen. Ce dernier offrait déjà la possibilité d'utiliser la proximité grâce à l'opérateur NEAR mais ne permettait pas de donner un score aux documents et donc un classement de ces derniers en fonction de la proximité des occurrences de termes de la requête retrouvés dans les documents. Pour tester l'hypothèse : *les documents ayant des occurrences de termes de la requête proches doivent être classés en premier*, nous avons introduit la notion de proximité floue tout utilisant des requêtes booléennes. Cette approche évite le problème d'inconsistance de l'opérateur NEAR lorsqu'il est appliqué à des locutions. Nous avons montré que les modèles classiques de recherche d'informations tels que le modèle vectoriel et le modèle booléen constituent des cas particuliers de notre approche en contrôlant la largeur de la zone d'influence d'un terme. Pour cette dernière, une valeur de l'ordre de 5 permet de spécifier une proximité de l'ordre de la locution, une valeur de 15 à 30 la situe au niveau de la phrase, enfin une valeur d'environ 200 correspondrait à la taille d'un passage.

Nous avons ensuite présenté l'implantation de notre modèle ainsi que les tests conduit lors des diverses campagnes d'évaluation CLEF, TREC et INEX. Nous avons constaté qu'en exploitant au maximum notre méthode c'est-à-dire en construisant des requêtes booléennes de manière manuelle, notre méthode dépasse la méthode Okapi implantée dans LUCY. Nous avons également présenté nos tentatives d'amélioration du système introduisant une phase de lemmatisation, un changement de la fonction d'influence des termes et une modification dans la construction de la liste de résultats. Nous avons aussi montré les résultats de notre participation à TREC 2006 pour la tâche TERABYTE.

Bien évidemment, notre système est encore perfectible et nous souhaitons poursuivre nos recherches et nos expérimentations afin de concrétiser les perspectives que nous envisageons notamment celles pour la formulation des requêtes qui nous paraît la plus importante.

Annexe A

Schémas de pondérations du modèle vectoriel

A.1 Fonctions tf appliquées à la fréquence des termes

TAB. A.1 – Les différentes fonctions tf

SMART[a]	fonction C[b]	définition		image
`b**`	`tfwt_binary`	$tfb(d,t)$	$=$ positif$(f(d,t))$	$[0,1]$
`n**`		$tfn(d,t)$	$= f(d,t)$	$[0,+\infty]$
`m**`	`tfwt_max`[c]	$tfm(d,t)$	$= \frac{f(d,t)}{\max_{t'} f(d,t')}$	$[0,1]$
`a**`	`tfwt_aug`[c]	$tfa(d,t)$	$= \frac{1}{2} + \frac{1}{2}\frac{f(d,t)}{\max_{t'} f_{d,t'}}$	$[\frac{1}{2},1]$
`s**`	`tfwt_square`	$tfs(d,t)$	$= f(d,t)^2$	$[0,+\infty]$
`l**`	`tfwt_log`	$tfl(d,t)$	$= 1 + \log f(d,t)$	$[1,+\infty]$

[a] La lettre de cette colonne correspond à la configuration de SMART.
[b] La fonction C dans le source du programme SMART.
[c] La fonction de SMART ne divise pas par le $\max$, mais par $\max + 0.00001$.

A.2 Fonctions idf appliquées à la fréquence documentaires

TAB. A.2 – Les différentes fonctions idf

SMART[a]	fonction C[b]		définition	image
`*n*`		$idfn(t)$	$= 1$	$[1, 1]$
`*i*`	`idfwt_idf`	$idfi(t)$	$= \log \frac{\|D\|}{df(t)}$	$[0, \log(\|D\|)]$
MG[c]		$idfI(t)$	$= \log(1 + \frac{\|D\|}{df(t)})$	$[\log(2), \log(\|D\| + 1)]$
`*f*`	`idfwt_freq`	$idff(t)$	$= \frac{1}{df(t)}$	$[\frac{1}{\|D\|}, 1]$
`*p*`	`idfwt_prob`	$idfp(t)$	$= \log \frac{\|D\| - df(t)}{df(t)}$	$[-\infty, \log(\frac{\|D\|-1}{1})]$
MG[c,d]		$idfP(t)$	$= \log(1 + \frac{\max_{d,t'} f_{d,t'}}{df(t)})$	$[\log(2), \log(1 + Cte)]$
`*s*`	`idfwt_s_idf`	$idfs(t)$	$= (\log \frac{\|D\|}{df(t)})^2$	$[0, (\log(\|D\|))^2]$

[a] La lettre de cette colonne correspond à la configuration de SMART.
[b] La fonction C dans le source du programme SMART.
[c] Fonction non disponible dans SMART, mais citée dans [Witten *et al.*, 1999].
[d] `*P*` dans SMART concerne la pondération des expressions (*phrases* en anglais).

A.3 Calcul des poids des documents $w_{t,d}$

TAB. A.3 – Les différentes facteurs de normalisation

SMART	`compute()`		définition	image
`**n **x`		$nn(d)$	$= 1$	
`**c`	`weight_cos normwt_cos`	$nc(d)$	$= \sqrt{\sum_t w(d,t)^2}$	
`**s`	`normwt_sum`	$ns(d)$	$= \sum_t w(d,t)$	
`**f`[a]		$nf(d)$	$= \sum_t w(d,t)^4$	
`**m`[a]		$nm(d)$	$= \max_t w(d,t)$	
`JS`		$nu(d)$	$= (1-c)\text{mean}(nt) + c \cdot nt_d$	

[a] Cette fonction apparaît dans la documentation mais pas dans le code.

Annexe B

Exemple de fichiers : Collection *Adhoc fr* CLEF 2006

Exemple de document

```
<DOC>
<DOCNO>LEMONDE94-000003-19940101</DOCNO>
Le nouveau droit de la nationalit
A parti du 1 janvi, la manifest de la volont de deveni
francai sera exig de certain enfant d’imigr etrang.
Prevu par la loi du 22 juilet reformant le droit de la
nationalit, les cond de cete manifest sont detail
dan un decret publ au Journal oficiel du 31 decembr 1993.
  Page 7
</DOC>

<DOC>
<DOCNO>LEMONDE94-000004-19940101</DOCNO>
Chomag : haus de 0,1 % en novembr
En novembr, le chomag a augment de 0,1 %, en done corig.
La Franc comptait 3 285 700 demandeu d’emploi, soit 3 200 de plu
qu’en octobr. En done brut, la bais est de 0,2 %. Cete
acalm est brouil par une oper de " destockag " : environ
16 000 chomeu age de plu de cinquant-cinq ans ont ete dispens
de la recherch d’un emploi et ne figurent donc plu dan les
statist.
  Page 10
</DOC>
```

Exemple de « topique »

```
<top>
<num> C301 </num>
<FR-title> Les Produits Nestlé </FR-title>
<FR-desc> Quels produits sont commercialisés par
Nestlé dans le monde? </FR-desc>
<FR-narr> Les articles pertinents indiqueront le nom de
produits distribués mondialement par Nestlé ou par des
entreprises qui appartiennent au groupe Nestlé. Dans un
second temps, le document doit faire clairement référence
à la société mère. </FR-narr>
</top>
```

Exemple de jugement de pertinence

```
251 0 ATS.940113.0133 0
251 0 ATS.940116.0073 0
251 0 ATS.940117.0049 1
251 0 ATS.940127.0137 0
251 0 ATS.940127.0159 0
251 0 ATS.940202.0117 0
251 0 ATS.940209.0057 0
```

Exemple de requêtes

```
mots du titre pf       301 produit \& nestl
mots du titre \okapi   301 produit nestl

titre et description pf
         301 (produit nestl ) | (produit \&  nestl \&
             (comercialis | mond | mondial))
titre et description \okapi
         301 produit nestl comercialis mond mondial

tous les champs pf
         301 ((produit | articl)  \& (nestl)) |
             ((produit | articl)  \& (nestl) \&
(group | mond | comercialis | distribu | vent))
tous les champs \okapi
```

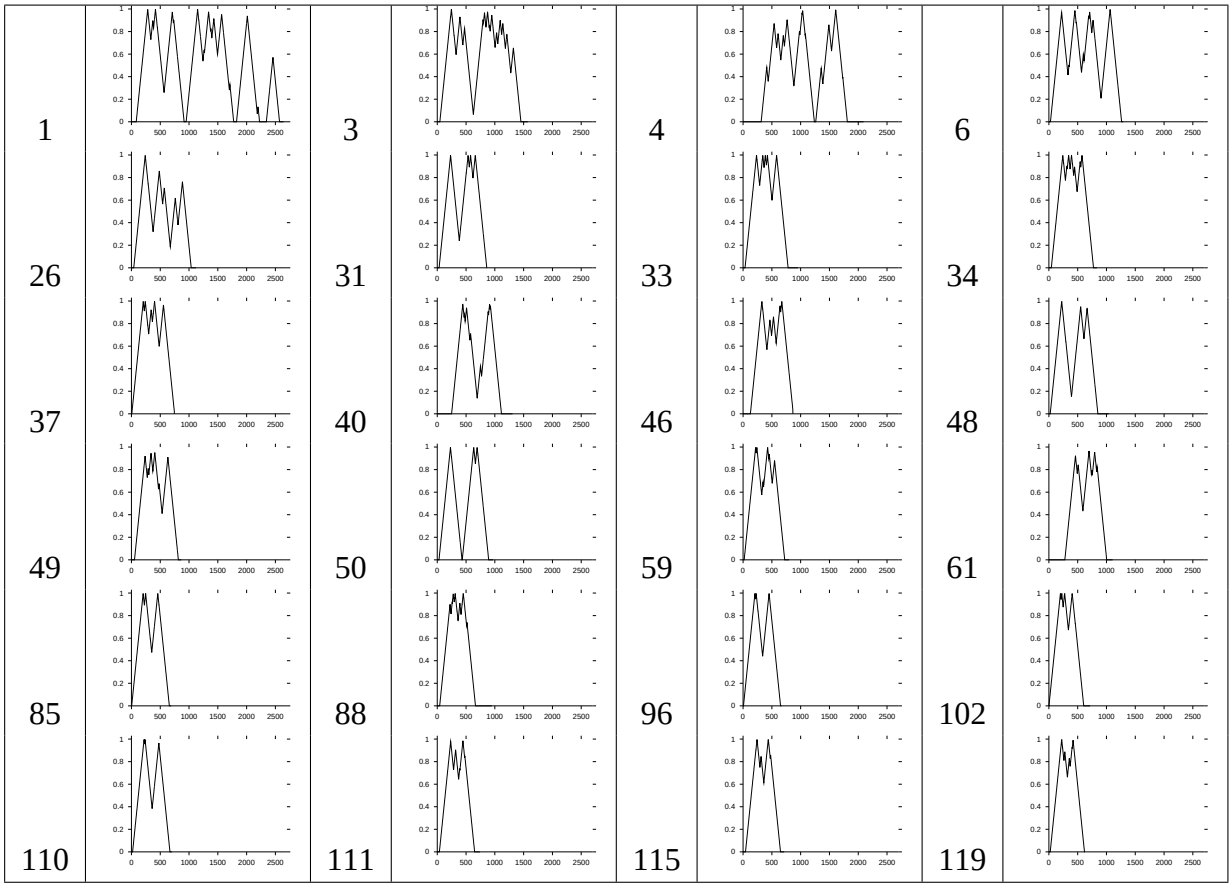
```
         301 produit articl nestl group mond
            comercialis distribu vent
```

Exemple de sortie trec_eval

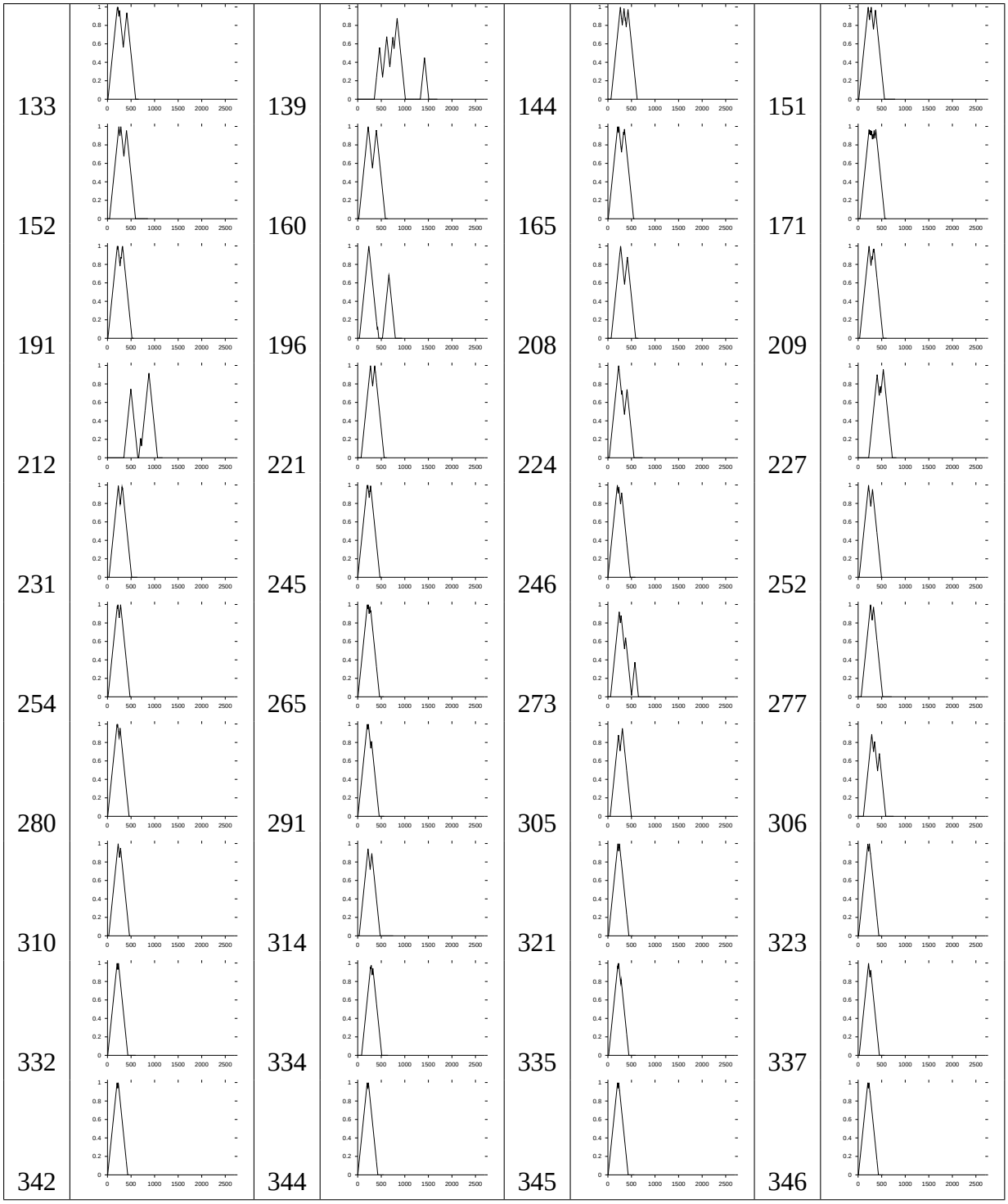
```
Queryid (Num):      267
Total number of documents over all queries
    Retrieved:     1000
    Relevant:        25
    Rel_ret:         25
Interpolated Recall - Precision Averages:
    at 0.00       1.0000
    at 0.10       0.8000
    at 0.20       0.7143
    at 0.30       0.4643
    at 0.40       0.4643
    at 0.50       0.4643
    at 0.60       0.4103
    at 0.70       0.3750
    at 0.80       0.3284
    at 0.90       0.2875
    at 1.00       0.1000
Average precision (non-interpolated) for all rel
                     docs(averaged over queries)
                  0.4588
Precision:
  At    5 docs:   0.8000
  At   10 docs:   0.5000
  At   15 docs:   0.4000
  At   20 docs:   0.4000
  At   30 docs:   0.4333
  At  100 docs:   0.2300
  At  200 docs:   0.1150
  At  500 docs:   0.0500
  At 1000 docs:   0.0250
R-Precision (precision after R (= num_rel for a query)
                                        docs retrieved):
    Exact:        0.4000
```

Annexe C

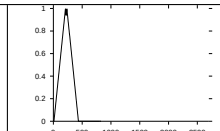
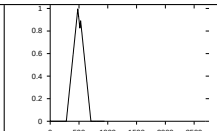
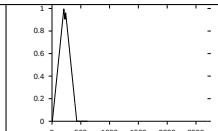
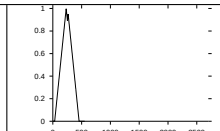
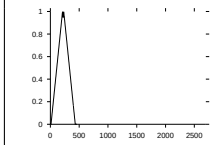
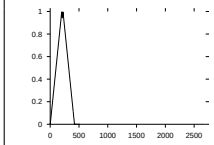
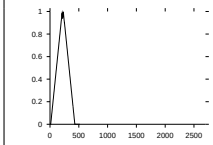
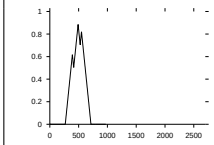
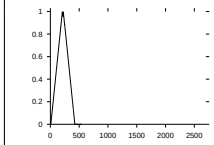
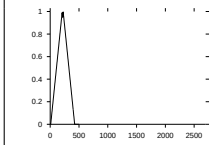
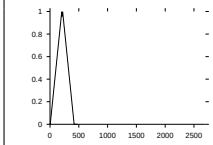
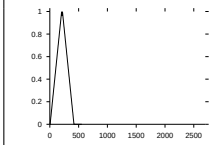
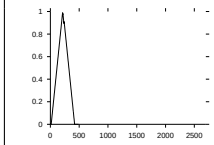
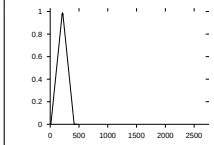
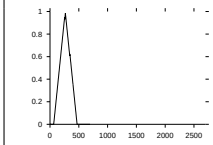
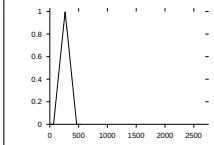
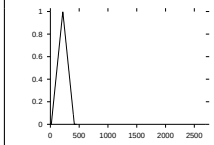
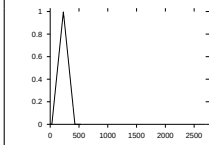
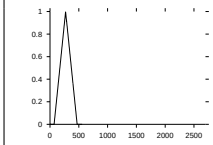
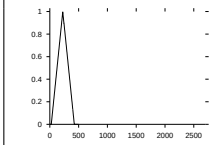
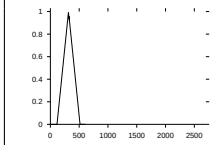
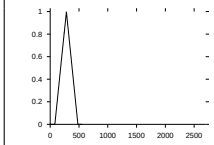
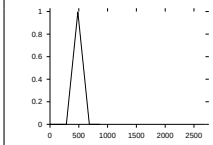
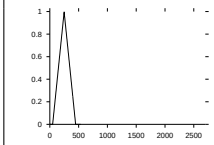
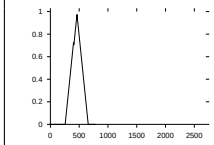
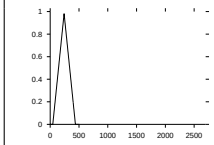
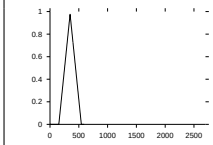
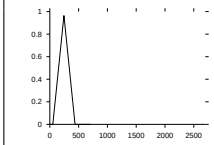
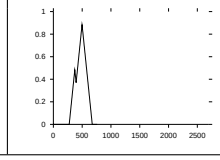
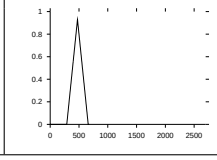
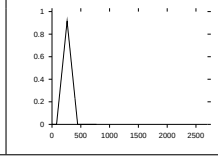
Visualisation de la proximité floue



TAB. C.1 – Visualisation de la proximité floue à la requête 298



TAB. C.2 – Visualisation de la proximité floue à la requête 298 (suite)

347		351		352		356	
362		364		376		378	
385		388		394		404	
431		435		436		450	
459		477		513		525	
539		562		733		737	
856		881		893		916	
941		948		956			

TAB. C.3 – Visualisation de la proximité floue à la requête 298 (fin)

Annexe D

Liste des travaux et articles

Revues

[1] Annabelle Mercier et Michel Beigbeder. *Calcul de pertinence basée sur la proximité pour la recherche d'informations*, Volume 9, Numéro 1, pages 43–60. **Document numérique**, Février **2006**.

Communications aux conférences d'audience internationale avec sélection

[2] Annabelle Mercier and Michel Beigbeder. Sphere of influence model in information retrieval. In *proceedings of **FUZZ-IEEE 2005***, pages 120–125, Reno, Nevada, USA, 22-25 may 2005.

[3] Michel Beigbeder and Annabelle Mercier. An information retrieval model using the fuzzy proximity degree of term occurences. In *proceedings of **SAC 2005**, the 20th Annual ACM Symposium on Applied Computing Information Access and Retrieval (IAR) track*, pages 1018–1022, Santa Fe, New Mexico, USA, march 2005.

[4] Michel Beigbeder and Annabelle Mercier. Experiments with different proximity based information retrieval models. In *proceedings of **AISTA 2004**, International Conference on Advances in Intelligent Systems - Theory and Applications* in cooperation with the IEEE Computer Society (in CD ISBN 2-9599776-8-8), Luxembourg, Luxembourg, november 2004.

Campagnes d'évaluation internationales

[5] Annabelle Mercier, Michel Beigbeder. Fuzzy term proximity with boolean queries at 2006 TREC Terabyte task, In proceedings of The 15th Text REtrieval Conference (**TREC 2006**), Gaithersburg, Maryland, USA, Novembre 2006.

[6] Annabelle Mercier, Michel Beigbeder. ENSM-SE at CLEF 2006 : AdHoc Uses of Fuzzy Proximity Matching Function, In CLEF 2006 Workshop, 20-22 September, Alicante, Spain.

[7] Annabelle Mercier, Amelie Imafouo, Michel Beigbeder. *ENSM-SE at CLEF 2005 : Uses of Fuzzy Proximity Matching Function*, Adhoc french track, Accessing Multilingual Information Repositories : 6th Workshop of the Cross-Language Evaluation Forum, **CLEF 2005**, Vienna, Austria, Revised Selected Papers. Carol Peters, Fredric C. Gey, Julio Gonzalo, Gareth J.F.Jones, Michael Kluck, Bernardo Magnini, Henning Muller, Maarten de Rijke (Eds.). Series : **Lecture Notes in Computer Science**, Vol. 4022.

[8] Annabelle Mercier et Michel Beigbeder. Fuzzy Proximity Ranking with Boolean Queries, In proceedings of *The Fourteenth Text REtrieval Conference (**TREC 2005**)* Gaithersburg, Maryland, USA, 15-18 Novembre 2005.

Communications aux conférences d'audience nationale ou francophone avec sélection

[9] A. Mercier, A. Imafouo et M. Beigbeder. Modèles de proximité : conception et comparaison à une méthode de recherche de passages. *Actes de **CORIA'05**, 2ème Conférence en Recherche d'Information et Applications* sous la direction de Claude Chrisment, pages 279–291. Grenoble, mars 2005.

[10] Annabelle Mercier et Michel Beigbeder. Détection de pertinence dirigée par la proximité des termes de la requête retrouvés dans les documents. *Actes de **SETIT 2005**, Sciences Electroniques Technologies de l'information et des Télécommunications*, page 146, Actes électroniques, Sousse, Tunisie, mars 2005.

[11] Annabelle Mercier et Michel Beigbeder. *Extraction de la localisation des termes pour le classement des documents*, volume 1, pages 275–280. Actes de la conférence Extraction et gestion des connaissances (**EGC 2005**), sous la direction de Nicole VINCENT et Suzanne PINSON. Numéro spécial de la *Revue des nouvelles technologies de l'information - **RNTI-E3*** Cépaduès, janvier 2005.

[12] Michel Beigbeder et Annabelle Mercier. Application de la logique floue à un modèle de recherche d'information basé sur la proximité. *Actes de **LFA 2004**, 12es rencontres francophones sur la Logique Floue et ses Applications*, pages 231–237, Cépaduès, novembre 2004.

[13] Annabelle Mercier. Etude comparative de trois approches utilisant la proximité entre les termes de la requête pour le calcul des scores des documents. *Actes de **INFORSID 2004**, 22e congrès informatique des organisations et des systèmes d'information et de décision* sous la direction de Danielle Boulanger, pp. 95–106. Biarritz, mai 2004.

Communications aux ateliers d'audience nationale avec sélection

[14] Michel Beigbeder et Annabelle Mercier. Etude des distributions de tf et de idf sur une collection de 5 millions de pages html. ***Atelier*** *Recherche d'information : un nouveau passage à l'échelle, associé à* ***INFORSID 2003***. Nancy, juin 2003.

Posters et démonstrations

[15] Annabelle Mercier et Michel Beigbeder. Système de recherche d'informations basé sur la proximité des termes de la requête. *Actes de la conférence Ingénierie des connaissances* ***(IC'2005)*** sous la direction de Marie-Christine Jaulent. Nice, juin 2005.

[16] Annabelle Mercier et Michel Beigbeder. Utiliser la proximité en recherche d'informations. Journée de la recherche de l'École Doctorale de l'Université Jean Monnet et de l'École Nat. Sup. des Mines de St-Etienne, mars 2004.

Participation aux séminaires

[17] Annabelle Mercier. Modèle de recherche d'information basé sur la proximité et évaluation des systèmes. Journée du GT 3.5 Indexation et Recherche d'Informations - **GDR I3/GDR ISIS**, Liris - Insa de Lyon, 24 Septembre **2004**.

[18] Annabelle Mercier. Utilisation de la proximité lexicale en recherche d'informations. Exposé dans le cadre des **séminaires** de l'équipe MRIM du laboratoire **CLIPS-IMAG**, Grenoble, 18 Juillet **2003**.

[19] Annabelle Mercier. Les notions de cooccurence et de proximité en recherche d'informations. Exposé dans le cadre du séminaire de l'**ISDN**, Yennes, 10 Juin **2003**.

[10] Annabelle Mercier. Présentation du sujet de thèse : la proximité en recherche d'information. Exposé dans le cadre du séminaire de l'ISDN, Yennes, 3 Février **2003**.

Rapports

[21] Annabelle Mercier. Recherche d'informations et proximité. Rapport d'activité de 1ère année de thèse sous la direction de Michel Beigbeder. Centre Simmo, Département RIM. École Nat. Sup. des Mines de St Etienne, Sept. 2003.

[22] Annabelle Mercier. Etude bibliographique d'un modèle général de contractualisation pour les composants logiciels. Rapport de DEA Informatique sous la direction de Philippe Collet. Laboratoire I3S, Equipe Objets et Composants Logiciels. Université de Nice Sophia Antipolis, Juin 2002.

Index

Bibliographie

[Attar et Fraenkel, 1977] R. Attar et A. S. Fraenkel, 1977. Local feedback in full-text retrieval systems. *Journal of the ACM*, v. 24(3), pages 397–417. 18

[Baeza-Yates et Ribeiro-Neto, 1999] R. Baeza-Yates et B. Ribeiro-Neto, 1999. *Modern Information Retrieval*. ACM Press / Addison-Wesley. ISBN 0-201-39829-X, 513 pages. 13, 24, 42

[Beigbeder et Mercier, 2003] M. Beigbeder et A. Mercier, 2003. Etude des distributions de tf et de idf sur une collection de 5 millions de pages html. *Atelier Recherche d'information : un nouveau passage à l'échelle, associé à Inforsid 2003*. Nancy. 108, 109

[Belew, 2000] R. Belew, 2000. *Finding Out About : A Cognitive Perspective on Search Engine Technology and the WWW*. Cambridge University Press. ISBN 0-521-63028-2. 41

[Belkin *et al.*, 1993] N. J. Belkin, C. Cool, W. B. Croft, et J. P. Callan, 1993. The effect of multiple query representations on information retrieval performance. *Proceedings of the 16th Annual International ACM SIGIR Conference*, pages 339–346. 13

[Bellet *et al.*, 1998] M. Bellet, T. Kirat, et C. Largeron, 1998. *Approches multiformes de la proximité*. Collection Interdisciplinarité et nouveaux outils. Lavoisier, Paris, 343 pages. 50

[Blair et Maron, 1985] D. Blair et M. Maron, 1985. An evaluation of retrieval effectiveness for a full-text document-retrieval system. *Communications of the ACM*, v. 28(3), pages 289–299. 4

[Blair et Maron, 1990] D. C. Blair et M. E. Maron, 1990. Full-text information retrieval : Further analysis and clarification. *Information Processing and Management*, v. 26(3), pages 437–447. 25

[Bookstein, 1980] A. Bookstein, 1980. Fuzzy requests : An approach to weighted boolean searchs. *Journal of the American Society for Information Science*, v. 31(4), pages 240–247. 35

[Bordogna et Pasi, 1995a] G. Bordogna et G. Pasi, 1995a. Fuzzy indexing and querying in information retrieval. R. De Caluwe, rédacteur, *Third series in lectures on fuzziness and data bases*. University of Ghent, Computer Science Dept. 35

[Bordogna et Pasi, 1995b] G. Bordogna et G. Pasi, 1995b. Handling vagueness in information retrieval systems. *Second New Zealand International Two-Stream Conference on Neural Networks and Expert Systems*, pages 110–116. Dunedin, New Zealand. 35

[Bordogna et Pasi, 1998] G. Bordogna et G. Pasi, 1998. Introduction to the special issue on management of imprecision and uncertainty in databases and information retrieval systems. *Journal of the American Society of Information Science*, v. 49(3). 35

[Bordogna *et al.*, 1991] G. Bordogna, P. Carrara, et G. Pasi, 1991. Query term weights as constraints in fuzzy information retrieval. *Information Processing and Management*, v. 27(1), pages 15–26. 35

[Borgman, 1986] C. Borgman, 1986. Why are online catalogs hard to use ? lessons learned from information-retrieval studies. *Journal of the American Society for Information Science*, v. 37(6), pages 387–400. 27

[Brin et Page, 1998] S. Brin et L. Page, 1998. The anatomy of a large-scale hypertextual web search engine. *The Seventh International World Wide Web Conference*. URL `http://www7.scu.edu.au/programme/fullpapers/1921/com1921.htm`. 18, 50

[Buckley *et al.*, 1995] C. Buckley, A. Singhal, M. Mitra, et G. Salton, 1995. New retrieval approaches using smart : TREC 4. [Harman, 1995]. 40, 51, 60

[Buell, 1982] D. A. Buell, 1982. An analysis of some fuzzy subset applications to information retrieval systems. *Fuzzy Sets and Systems*, v. 7(1), pages 35–42. 35

[Buell et Kraft, 1981] D. A. Buell et D. Kraft, 1981. A model for a weighted retrieval system. *Journal of the American Society for Information Science*, v. 32(3), pages 21l–216. 35

[Callan, 1994] J. P. Callan, 1994. Passage-level evidence in document retrieval. [Croft et van Rijsbergen, 1994], pages 302–310. URL `http://www.cs.cmu.edu/~callan/callan794.ps.gz`. 65, 67

[Callan *et al.*, 1992] J. P. Callan, W. B. Croft, et S. M. Harding, 1992. The inquery retrieval system. *The third International Conference on Database and expert System Applications*, pages 78–83. URL `http://www.cs.cmu.edu/~callan/Papers/callancroftdexa92.ps.gz`. 54

[Cater et Kraft, 1987] S. Cater et D. Kraft, 1987. TIRS : a topological information retrieval system satisfying the requirements of the waller-kraft wish list. *10th annual international ACM SIGIR conference on Research and development in information retrieval*, pages 171–180. New Orleans, Louisiana, United States. 35

[Chevallet et Haddad, 2001] J.-P. Chevallet et H. Haddad, 2001. Proposition d'un modèle relationnel d'indexation syntagmatique : mise en oeuvre dans le système Iota. *INFORSID*, pages 465–483. 21

[Church et Hanks, 1990] K. Church et P. Hanks, 1990. Word association norms, mutual information and lexicography. *Computational Linguistics*, v. 16(1), pages 22–29. 40

[Clarke et Cormack, 1996] C. Clarke et G. Cormack, 1996. Interactive substring retrieval : Multitext experiments for trec-5. [Harman, 1996]. URL http://trec.nist.gov/pubs/trec5/papers/waterloo.ps.gz. 55

[Clarke *et al.*, 1995] C. L. A. Clarke, G. V. Cormack, et F. J. Burkowski, 1995. Shortest Substring Ranking (MultiText Experiments for TREC-4). [Harman, 1995]. URL http://trec.nist.gov/pubs/trec4/papers/uwaterloo.ps.gz. 55

[Clarke *et al.*, 2000] C. L. A. Clarke, G. V. Cormack, et E. A. Tudhope, 2000. Relevance ranking for one to three term queries. *Information Processing and Management*, v. 36(2), pages 291–311. 55, 108

[Cleverdon, 1967] C. Cleverdon, 1967. The cranfield tests on index language devices. *Aslib Proceedings*, v. 19(6), pages 173–193. 3

[Cleverdon, 1984] C. Cleverdon, 1984. Optimizing convenient online access to bibliographic databases. *Informantion Service and Use*, v. 4(1-2), pages 37–47. ISSN 0167-5265. 29

[Cooper, 1988] W. S. Cooper, 1988. Getting beyond boole. *Information Processing and Management*, v. 24(3), pages 243–248. 27

[Crestani et Pasi, 1999] F. Crestani et G. Pasi, 1999. Soft information retrieval : Applications of fuzzy set theory and neural networks. N. Kasabov et R. Kozma, rédacteurs, *Neuro-Fuzzy Techniques for Intelligent Information Systems*, pages 287–315. Physica Verlag (Springer Verlag), Heidelberg, Germany. 35

[Croft, 2000] W. B. Croft, 2000. *Advances in Information Retrieval*, chap. Combining Approaches to Information Retrieval. Kluwer Academic Publishers. 65

[Croft et van Rijsbergen, 1994] W. B. Croft et C. J. van Rijsbergen, rédacteurs, 1994. URL http://www.acm.org/pubs/contents/proceedings/ir/188490/. 160, 166

[Cross, 1994] V. Cross, 1994. Fuzzy information retrieval. *Intelligent Information Systems*, v. 3(1), pages 29–56. 35

[de Kretser et Moffat, 1999a] O. de Kretser et A. Moffat, 1999a. Effective document presentation with a locality-based similarity heuristic. M. Hearst et R. Tong, rédacteurs, *SIGIR '99 : Proceedings of the 22nd Annual International ACM SIGIR Conference on Research and Development in Information Retrieval*, pages 113–120. ACM. 69

[de Kretser et Moffat, 1999b] O. de Kretser et A. Moffat, 1999b. Locality-based information retrieval. *10th Australasian Database Conference*, pages 177–188. 70

[de Kretser *et al.*, 1998] O. de Kretser, A. Moffat, et J. Zobel, 1998. Teraphim : an engine for distributed information retrieval. *SIGIR '98, Proceedings of the 21st annual international ACM SIGIR conference on Research and development in information retrieval*, page 384. ISBN 1-58113-015-5. URL http://www.acm.org/pubs/articles/proceedings/ir/290941/p384-de_kretser/p384-de_kretser.pdf. 69

[Deerwester *et al.*, 1990] S. C. Deerwester, S. T. Dumais, T. K. Landauer, G. W. Furnas, et R. A. Harshman, 1990. Indexing by latent semantic analysis. *jasis*, v. 41(6), pages 391–407. 40

[Diderot et D'Alembert, 1751] Diderot et D'Alembert, 1751. *L'Encyclopédie, ou Dictionnaire Raisonné des Sciences, des Arts et des Métiers*. 18000 pages. 3

[Dubois et Prade, 1985] D. Dubois et H. Prade, 1985. A review of fuzzy sets aggregation connectives. *Information Sciences*, v. 3, pages 85–121. 35

[Fenichel, 1981] C. Fenichel, 1981. Online searching : Measures that discriminate among users with different types of experiences. *Journal of the American Society for Information Science*, v. 32(1), pages 23–32. 28

[Gaussier et Stéfanini, 2003] E. Gaussier et M.-H. Stéfanini, 2003. *Assistance intelligente à la recherche d'informations*. Traité des sciences et techniques de l'information, 314 pages. 2

[Greiff, 2000] W. R. Greiff, 2000. The use of exploratory data analysis in information retrieval research. W. B. Croft, rédacteur, *Advances in Informational Retrieval : Recent Research from the Center for Intelligent Information Retrieval*. Kluwer Academic Publishers, Norwell, MA, USA. ISBN 0792378121. 38

[Gupta et Padmini, 1987] D. Gupta et Padmini, 1987. Boolean interpretation of conjunctions for document retrieval. *Journal of the American Society for Information Science*, v. 38(4), pages 245–254. 27

[Harman, 1992] D. Harman, rédacteur, 1992. *Overview of the First Text REtrieval Conference*. National Institute of Standards and Technology special publication. 4

[Harman, 1995] D. K. Harman, rédacteur, 1995. 500-236. Department of Commerce, National Institute of Standards and Technology. 160, 161, 162

[Harman, 1996] D. K. Harman, rédacteur, 1996. 500-238. Department of Commerce, National Institute of Standards and Technology. 161, 168

[Hawking et Thistlewaite, 1995] D. Hawking et P. Thistlewaite, 1995. Proximity operators - so near and yet so far. [Harman, 1995]. URL `http://trec.nist.gov/pubs/trec4/papers/anu.ps.gz`. 55, 57

[Hawking et Thistlewaite, 1996] D. Hawking et P. Thistlewaite, 1996. Relevance weighting using distance between term occurrences. Rapport technique TR-CS-96-08, Australian National University. URL `http://cs.anu.edu.au/techreports/1996/TR-CS-96-08.ps.gz`. 55, 57

[Hearst, 1993] M. A. Hearst, 1993. Texttiling : A quantitative approach to discourse segmentation. Rapport technique S2K-93-24, University of California. URL `citeseer.nj.nec.com/hearst93texttiling.html`. 65

[Hearst, 1997] M. A. Hearst, 1997. Texttiling : segmenting text into multi-paragraph subtopic passages. *Computational Linguistics*, v. 23(1), pages 33–64. ISSN 0891-2017. URL `http://portal.acm.org/citation.cfm?id=972687`. 65

[Hearst et Plaunt, 1993] M. A. Hearst et C. Plaunt, 1993. Subtopic structuring for full-length document access. [Korfhage *et al.*, 1993], pages 59–68. URL `http://www.acm.org/pubs/articles/proceedings/ir/160688/p59-hearst/p59-hearst.pdf`. 65

[Hersh et Over, 2001] W. R. Hersh et P. Over, 2001. Interactivity at the text retrieval conference (trec). *Information Processing Management*, v. 37(3), pages 365–367. 28

[Howard, 1982] H. Howard, 1982. Measures that discriminate among online searchers with different training and experience. *Online Review*, v. 6(4), pages 315–327. 28

[Imafouo et Beigbeder, 2005] A. Imafouo et M. Beigbeder, 2005. Passage à l'échelle : une méthodologie pour l'étude de l'influence du volume de collection sur les modèles de ri. *Actes de la 2ème conférence Francophone en Recherche d'Information-CORIA'05*. 110

[Karbasi et Tamine, 2005] S. Karbasi et L. L. Tamine, 2005. Analyse expérimentale de la structure des index documentaires et leur impact sur l'efficacité de la recherche : cas de collections volumineuses. C. Chrisment, rédacteur, *CORIA'05, 2ème Conférence en Recherche d'Information et Applications*, pages 373–388. Grenoble. 108

[Kaszkiel et Zobel, 1997] M. Kaszkiel et J. Zobel, 1997. Passage retrieval revisited. [Voorhees, 1997], pages 178–185. URL `http://www.acm.org/pubs/articles/proceedings/ir/258525/p178-kaszkiel/p178-kaszkiel.pdf`. 65

[Keen, 1991a] E. M. Keen, 1991a. The effectiveness of term position and frequency for output ranking. *In Proceedings of the British Computer Society 13th Information Retrieval Colloquium*, pages 22–37. 53

[Keen, 1991b] E. M. Keen, 1991b. The use of term position devices in ranked output experiments. *Journal of Documentation*, v. 47, pages 1–22. 53

[Keen, 1992a] E. M. Keen, 1992a. Some aspects of proximity searching in text retrieval systems. *Journal of Information Science*, v. 18, pages 89–98. 54

[Keen, 1992b] E. M. Keen, 1992b. Term position ranking : some new test results. pages 66–76. 54

[Kent *et al.*, 1955] A. Kent, M. Berry, F. Luehrs, et J. Perry, 1955. Machine literature searching : VIII. Operational criteria for designing information retrieval systems. *American Documentation*, v. 6(2), pages 83–101. 45

[Kerre *et al.*, 1986] E. E. Kerre, R. B. R. C. Zenner, et R. M. M. De Caluwe, 1986. The use of fuzzy set theory in information retrieval : a survey. *Journal of the American Society for Information Science*, v. 37(5), pages 341–345. 35

[Kise *et al.*, 2001] K. Kise, M. Junker, et A. Dengel, 2001. Experimental evaluation of passage-based document retrieval. *Sixth International Conference on Document Analysis and Recognition (ICDAR '01)*, pages 592–596. 72

[Kise *et al.*, 2004] K. Kise, M. Junker, A. Dengel, et K. Matsumoto, 2004. Passage retrieval based on density distributions of terms and its applications to document retrieval and question answering. *Lecture Notes in Computer Science*, v. 2956, pages 306–327. Springer. ISBN 3-540-21904-8. No electronic version. 72

[Korfhage *et al.*, 1993] R. Korfhage, E. M. Rasmussen, et P. Willett, rédacteurs, 1993. URL `http://www.acm.org/pubs/contents/proceedings/ir/160688/`. 162, 166, 167

[Korfhage, 1997] R. R. Korfhage, 1997. *Information Storage and Retrieval*. Wiley. ISBN 0-471-14338-3, 349 pages. 79

[Kraft *et al.*, 1999] D. Kraft, G. Bordogna, et G. Pasi, 1999. *Fuzzy Sets in Approximate Reasoning and Information Systems*, chap. Fuzzy Set Techniques in Information Retrieval, pages 469–510. The Handbooks of Fuzzy Sets. Kluwer Academic Publishers. 35

[Kraft et Buell, 1983] D. H. Kraft et D. A. Buell, 1983. Fuzzy sets and generalized boolean retrieval systems. *International Journal of Man-Machine Studies*, v. 19(1), pages 45–46. 35

[Lebart et Salem, 1994] L. Lebart et A. Salem, 1994. *Statistique textuelle*. 21

[Lesk et Salton, 1969] M. Lesk et G. Salton, 1969. Relevance assessments and retrieval system evaluation. *Information Storage and Retrieval*, v. 4, pages 343–359. 46

[Losee, 2001] R. M. Losee, 2001. Term dependance : A basis for luhn and zipf models. *Journal of American Society for Information Science and Technology*, v. 52(12), pages 1019–1025. 40

[Lovins, 1968] J. Lovins, 1968. Development of a Stemming Algorithm. *Mechanical Translation and Computational Linguistics*, v. 11(1), pages 22–31. 23

[Lucarella et Morara, 1991] D. Lucarella et R. Morara, 1991. First : fuzzy information retrieval system. *Information Science*, v. 17(2), pages 81–91. 35

[Luhn, 1958] H. P. Luhn, 1958. The automatic creation of literature abstracts. *IBM Journal of Research and Development*, v. 2, pages 159–168. 50, 78

[Manning et Schütze, 1999] C. D. Manning et H. Schütze, 1999. *Foundations of statistical natural language processing*. The MIT Press, Cambridge, Massachussets, USA. URL http://nlp.stanford.edu/fsnlp/. 21

[Maron et Kuhns, 1960] M. Maron et J. Kuhns, 1960. On relevance, probabilistic indexing and information retrieval. *Journal of the ACM*, v. 7(3), pages 216–244. URL http://portal.acm.org/citation.cfm?id=321035\&coll=Portal\&dl=GUIDE\&CFID=56353916\&CFTOKEN=55069558. Repr. in : Sparck Jones, Karen and Willett, Peter (eds.) Readings in information retrieval (1997, p.39-46). 41

[Mercier, 2004] A. Mercier, 2004. Etude comparative de trois approches utilisant la proximité entre les termes de la requête pour le calcul des scores des documents. D. Boulanger, rédacteur, *INFORSID 2004, 22e congrès informatique des organisations et des systèmes d'information et de décision*, pages 95–106. Biarritz. 106

[Mercier et Beigbeder, 2005] A. Mercier et M. Beigbeder, 2005. Sphere of influence model in information retrieval. *FUZZ-IEEE 2005*, pages 120–125. Reno, Nevada, USA. 93

[Mercier et Beigbeder, 2006] A. Mercier et M. Beigbeder, 2006. Calcul de pertinence basée sur la proximité pour la recherche d'informations. *Document numérique*, v. 9(1), pages 43–60. 82

[Mercier *et al.*, 2005] A. Mercier, A. Imafouo, et M. Beigbeder, 2005. Modèles de proximité : conception et comparaison à une méthode de recherche de passages. C. Chrisment, rédacteur, *CORIA'05, 2ème Conférence en Recherche d'Information et Applications*, pages 279–291. Grenoble. 66, 110

[Mitchell, 1973] P. C. Mitchell, 1973. A note about the proximity operators in information retrieval. *Proceedings of the 1973 meeting on Programming languages and information retrieval*, pages 177–180. ACM Press. 52

[Miyamoto, 1990a] S. Miyamoto, 1990a. *Fuzzy sets in Information Retrieval and Cluster Analysts*. Kluwer Academic. 36

[Miyamoto, 1990b] S. Miyamoto, 1990b. Information retrieval based on fuzzy associations. *Fuzzy Sets and Systems*, v. 38(2), pages 191–205. 36

[Miyamoto, 2003] S. Miyamoto, 2003. Proximity measures for terms based on fuzzy neighborhoods in document sets. *International Journal of Approximate Reasoning*, v. 34(2), pages 181–199. 54

[Monz, 2003] C. Monz, 2003. *From Document Retrieval to Question Answering*. Thèse de doctorat. URL `http://www.illc.uva.nl/Publications/Dissertations/DS-2003-04.text.pdf`. 59

[Monz, 2004] C. Monz, 2004. Minimal span weighting retrieval for question answering. R. Gaizauskas, M. Greenwood, et M. Hepple, rédacteurs, *SIGIR Workshop on Information Retrieval for Question Answering*, pages 23–30. URL `http://www.dcs.qmul.ac.uk/~christof/`. 59

[Nie et Brisebois, 1996] J. Y. Nie et M. Brisebois, 1996. An inferential approach to information retrieval and its implemantation using a manual thesaurus. *Artificial Intelligence Review*, v. 38(8). 40

[Oldroyd et Schroder, 1982] B. K. Oldroyd et J. Schroder, 1982. Study of strategies used in online searching : 2. positional logic–an example of the importance of selecting the right boolean operator. *Online Review*, v. 6(2), pages 127–133. 28

[Park *et al.*, 2001] L. A. F. Park, M. Palaniswami, et K. Ramamohanarao, 2001. Internet document filtering using fourier domain scoring. *Principles of Data Mining and Knowledge Discovery, 5th European Conference, PKDD 2001*. Freiburg, Germany. 75

[Pasi, 1999] G. Pasi, 1999. *Application de la théorie des ensembles flous pour la définition de systèmes flexibles de recherche d'information*. Thèse de doctorat, Université de Rennes 1. 36

[Pasi et Marques Pereira, 1999] G. Pasi et R. A. Marques Pereira, 1999. A decision making approach to relevance feedback in information retrieval : a model based on a soft consensus dynamics. *International Journal of Intelligent Systems*, v. 14(1), pages 1–18. 36

[Peat et Willett, 1991] H. J. Peat et P. Willett, 1991. The limitations of term co-occurrence data for query expansion in document retrieval systems. *Journal of the American Society of Information Science*, v. 42(5), pages 378–383. 54

[Pechoin, 1991] D. Pechoin, 1991. *Thésaurus Larousse*. Paris. 40

[Porter, 1980] M. Porter, 1980. An Algorithm for Suffix Stripping. *Program*, v. 14(3), pages 130–137. 23

[Qui et Frei, 1993] Y. Qui et H. Frei, 1993. Concept based query expension. [Korfhage *et al.*, 1993]. URL `http://www.acm.org/pubs/contents/proceedings/ir/160688/`. 40

[Radecki, 1976] T. Radecki, 1976. Mathematical model of information retrieval system based on the concept of fuzzy thesaurus. *Information Processing and Management*, v. 12(5), pages 313–318. 35, 36

[Radecki, 1979] T. Radecki, 1979. Fuzzy set theoretical approach to document retrieval. *Information Processing and Management*, v. 15(5), pages 247–260. 35

[Radecki, 1981] T. Radecki, 1981. Outline of a fuzzy logic approach to document retrieval. *International Journal of Man-Machine Studies*, v. 14(2), pages 169–178. 35

[Rasolofo et Savoy, 2003] Y. Rasolofo et J. Savoy, 2003. Term proximity scoring for keyword-based retrieval systems. *25th European Conference on IR Research, ECIR 2003*, nº 2633 LNCS, pages 207–218. Springer. 42, 55, 58

[Robertson, 1977] S. Robertson, 1977. The probability ranking principle in IR. *Journal of Documentation*, v. 33(4), pages 294–304. Repr. in : Sparck Jones, Karen and Willett, Peter (eds.) Readings in information retrieval (1997, p.281-286). 41

[Robertson et Sparck Jones, 1976] S. Robertson et K. Sparck Jones, 1976. Relevance weighting of search terms. *Journal of the American Society for Information Science*, v. 27, pages 129–146. Pas en electronique. 41

[Robertson, 1990] S. E. Robertson, 1990. On term selection for query expansion. *Journal of Documentation*, v. 46(12). 40

[Robertson et Walker, 1994] S. E. Robertson et S. Walker, 1994. Some simple effective approximations to the 2-poisson model for probabilistic weighted retrieval. [Croft et van Rijsbergen, 1994], pages 232–241. URL `http://www.acm.org/pubs/contents/proceedings/ir/188490/`. 42

[Robertson et Walker, 1997] S. E. Robertson et S. Walker, 1997. On relevance weight with little relevance information. [Voorhees, 1997]. URL `http://www.acm.org/pubs/contents/proceedings/ir/258525/`. 18, 40

[Robertson *et al.*, 1994] S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-Beaulieu, et M. Gatford, 1994. Okapi at trec-3. D. K. Harman, rédacteur, *Overview of the Third Text REtrieval Conference (TREC-3)*, PB95-216883, pages 109–. Department of Commerce, National Institute of Standards and Technology. URL `http://trec.nist.gov/pubs/trec3/papers/city.ps.gz`. 42, 101

[Roussey *et al.*, 1999] C. Roussey, S. Calabretto, et J.-M. Pinon, 1999. Etat de l'art en indexation et recherche d'information. *Document Numérique, numéro spécial Gestion des documents et gestion des connaissances*, v. 3(3-4), pages 121–150. 24

[Salton, 1971a] G. Salton, 1971a. *Relevance feedback in information retrieval*. [Salton, 1971b]. 18, 40

[Salton, 1971b] G. Salton, 1971b. *The SMART retrieval system : experiments in automatic document processing*. Automatic Computation. Prentice-Hall, Englewood Cliffs, New Jersey. 4, 7, 36, 167

[Salton, 1980] G. Salton, 1980. Automatic term class construction using relevance–a summary of work in automatic pseudoclassification. *Information Processing and Management*, v. 16(1), pages 1–15. 40

[Salton et Buckley, 1988] G. Salton et C. Buckley, 1988. Term-weighting approaches in automatic text retrieval. *Information Processing and Management*, v. 24(5), pages 513–523. Repr. in : Sparck Jones, Karen and Willett, Peter (eds.) Readings in information retrieval (1997, p.323-328). 39

[Salton et Buckley, 1990] G. Salton et C. Buckley, 1990. Improving retrieval performance by relevance feedback. *Journal of American Society for Information Science*, v. 41(4), pages 288–297. 40

[Salton et McGill., 1983] G. Salton et M. J. McGill., 1983. *Introduction to Modern Information Retrieval*. McGraw-Hill Book Company. 4, 7, 53

[Salton *et al.*, 1983] G. Salton, E. A. Fox, et H. Wu, 1983. Extended Boolean information retrieval. *Communications of the ACM*, v. 26(11), pages 1022–1036. 35

[Salton *et al.*, 1993] G. Salton, J. Allan, et C. Buckley, 1993. Approaches to passage retrieval in full text information systems. [Korfhage *et al.*, 1993], pages 49–58. URL `http://www.acm.org/pubs/articles/proceedings/ir/160688/p49-salton/p49-salton.pdf`. 61, 64, 65

[Salton *et al.*, 1996] G. Salton, A. Singhal, C. Buckley, et M. Mitra, 1996. Automatic text decomposition using text segments and text themes. *Conference on Hypertext*, pages 53–65. URL `citeseer.ist.psu.edu/80864.html`. 66

[Sanchez, 1989] E. Sanchez, 1989. Importance in knowledge systems. *Information Systems*, v. 14(6), pages 55–464. 35

[Savoy, 1997] J. Savoy, 1997. Ranking schemes in hybrid boolean systems : A new approach. *Journal of American Society for Information Science*, v. 33(4), pages 495–512. 36

[Savoy, 1999] J. Savoy, 1999. A stemming procedure and stopword list for general French corpora. *Journal of American Society for Information Science*, v. 50(10), pages 944–952. 23

[Smadia, 1993] F. Smadia, 1993. Retrieving collocations from text : Xtract. *Computational Linguistics*, v. 19(1), pages 143–177. 40

[Small, 1973] H. Small, 1973. Co-Citation in the scientific literature : a new measure of the relationship between documents. *Journal of the American Society for Information Science*, v. 42(4), pages 265–269. 40

[Song *et al.*, 2005] R. Song, J.-R. Wen, et W.-Y. Ma, 2005. Viewing term proximity from a different perspective. Rapport technique. URL ftp://ftp.research.microsoft.com/pub/tr/TR-2005-69.doc. 61

[Sparck Jones et van Rijsbergen, 1975] K. Sparck Jones et C. van Rijsbergen, 1975. Report on the need for and provision of an "ideal" information retrieval test collection. Rapport technique, Computer Laboratory, University of Cambridge. British Library Research and Development Report 5266. 45

[Spark Jones, 1971] K. Spark Jones, 1971. *Automatic keyword classification for information retrieval*. Londre. 40

[Spink *et al.*, 2001] A. Spink, D. Wolfram, B. Jansen, et T. Saracevic, 2001. Searching the web : The public and their queries. *Journal of the American Society of Information Science and Technology*. URL http://jimjansen.tripod.com/academic/pubs/jasist2001/jasist2001.pdf. 50

[Strzalkowski et Carballo, 1996] T. Strzalkowski et J. P. Carballo, 1996. Natural Language Information Retrieval : TREC-4 Report. [Harman, 1996], pages 245–258. URL http://trec.nist.gov/pubs/trec4/t4_proceedings.html. 21

[Su, 1992] L. T. Su, 1992. Evaluation measures for interactive information retrieval. *Information Processing and Management*, v. 28(4), pages 503–516. 79

[Tahani, 1976] V. Tahani, 1976. A fuzzy model of document retrieval systems. *Information Processing and Management*, v. 12(3), pages 177–187. 35

[Tajima *et al.*, 1999] K. Tajima, K. Hatano, T. Matsukura, R. Sano, et K. Tanaka, 1999. Discovery and retrieval of logical information units in web. R. Wilensky, K. Tanaka, et Y. Hara, rédacteurs, *1999 ACM Digital Library Workshop on Organizing Web Space (WOWS)*, pages 13–23. URL http://www.db.cs.kobe-u.ac.jp/~tajima/papers/wows99www.ps.gz. 73

[Tannier *et al.*, 2005] X. Tannier, J.-J. Girardot, et M. Mathieu, 2005. Analysing Natural Language Queries at INEX 2004. *Lecture Notes in Computer Science*, v. 3493, pages 395–409. Springer-Verlag, Schloss Dagstuhl, Germany. 117

[Tong Tong, 1995] J. R. Tong Tong, 1995. *La logique floue*. Hermès. 30

[Van Rijsbergen, 1977] C. J. Van Rijsbergen, 1977. A theoretical basis for the use of co-occurrence data in information retrieval. *Journal of Documentation*, v. 33(2), pages 106–119. 40, 54

[van Rijsbergen, 1979] C. J. van Rijsbergen, 1979. *Information Retrieval*. Butterworth (London). 41

[Voorhees, 1997] E. Voorhees, rédacteur, 1997. URL http://www.acm.org/pubs/contents/proceedings/ir/258525/. 163, 166

[Voorhees et Harman, 2000] E. M. Voorhees et D. Harman, 2000. Overview of the sixth text retrieval conference (trec-6). *Information Processing and Management*, v. 36(1), pages 3–35. 38, 45

[Wilkinson, 1994] R. Wilkinson, 1994. Effective retrieval of structured documents. *SIGIR 94 proceedings*, pages 311–317. Springer-Verlag New York. ISBN 0-387-19889-X. 65, 66

[Witten *et al.*, 1999] I. H. Witten, A. Moffat, et T. C. Bell, 1999. *Managing Gigabytes : Compressing and Indexing Documents and Images*. Morgan Kaufmann. ISBN 1-55860-570-3. 100, 144

[Wong *et al.*, 1987] S. K. M. Wong, W. Ziarko, V. Raghavan, et C. Wong, 1987. On modelling of information retrieval concepts in vector spaces. *ACM Transactions on Database Systems*, v. 12(2), pages 299–321. 40

[Yager, 1987] R. R. Yager, 1987. A note on weighted queries in information retrieval systems. *Journal of the American Society for Information Science*, v. 38(1), pages 23–24. 35

[Zadeh, 1973] L. Zadeh, 1973. Outline of a new approach to the analysis of complex systems and decisions processes. *IEEE transactions*. 30

[Zadeh, 1975] L. A. Zadeh, 1975. The concept of linguistic variable and its application to approximate reasoning. *Information Science, part I, II*, v. 8, pages 199–249, 301–357. 30

[Zadeh, 1983] L. A. Zadeh, 1983. A computational approach to fuzzy quantifiers in natural languages. *Computing and Mathematics with Applications*, v. 8, pages 149–184. 35

[Zadeh, 1988] L. A. Zadeh, 1988. Fuzzy logic. *IEEE Computer*, v. 21(4), pages 83–93. 35

[Zobel *et al.*, 1995] J. Zobel, A. Moffat, R. Wilkinson, et R. Sacks-Davis, 1995. Efficient retrieval of partial documents. *Information Processing and Management*, v. 31(1), pages 361–377. 65

Ecole Nationale Supérieure des Mines
de Saint-Etienne

N° d'ordre : 417 I

Annabelle MERCIER

Fuzzy term proximity information retrieval model and system

Computer Science

Information retrieval, term proximity, fuzzy logic, scalability

Abstract The huge size of digital data accentuates the scientific challenge of information retrieval (IR) consisting in finding a compromise between recall and precision. We propose an IR model based on fuzzy proximity (FP) of the query terms which is aimed to high precision. It combines the expressivity of the Boolean query model and the ranking of the documents thanks to the use of proximity. Each keyword defines an influence zone at the query evaluation time. The fuzzy operations associated to the traditional Boolean operators propagate the proximity to the root of the query tree. The FP model was largely validated on the traditional test collections and at the 2005 and 2006 editions of the international IR evaluation campaigns (TREC, CLEF and INEX 2006). The results obtained with the automatically built queries are equivalent to the baselines (Okapi/Lucy and vector/MG). Moreover, with manual queries adapted to FP, the results are better than the baselines.

Ecole Nationale Supérieure des Mines
de Saint-Etienne

N° d'ordre : 417 I

Annabelle MERCIER

Modélisation et prototypage d'un système de recherche d'informations basé sur la proximité des occurrences des termes de la requête dans les documents

Informatique

Recherche d'informations, proximité des termes, logique floue, passage à l'échelle, recherche de passages

Résumé L'objectif de la recherche d'information (RI) est de mettre en place des modèles et des systèmes pour sélectionner les documents les plus proches du besoin d'information de l'utilisateur dans une base documentaire. De nos jours, l'importance de la croissance des données numériques accentue le principal verrou scientifique de la recherche d'information qui consiste à trouver un compromis entre exhaustivité et précision des résultats retournés. Bien que les modèles exploitant les liens hypertextes (solution basée sur la popularité des pages) aient porté leurs fruits dans une application industrielle connue du grand public, l'utilisation du contenu lui-même n'a pas encore livré tous ses secrets. En utilisant la proximité des mots-clés dans les documents comme indicateur de pertinence, nous visons ainsi une approche à haute précision. Auparavant, l'usage de la proximité était présent dans certains systèmes de recherches bibliographiques au travers de l'opérateur NEAR. Ce dernier, ajouté au langage de requête booléen présente l'avantage de retrouver des syntagmes nominaux. Mais, cette solution reste limitée sur deux plans. D'une part, le système est contraint à donner une réponse binaire empêchant ainsi le classement des documents et, d'autre part, l'opérateur NEAR ne peut être considéré au même titre que les autres opérateurs du langage booléen, car celui-ci s'applique seulement à deux feuilles de l'arbre de requête. Pour étendre la proximité à plus de deux termes, d'autres méthodes, basées sur la sélection d'intervalles contenant les mots-clés dans les documents ont été développées mais perdent la précision apportée par l'utilisation des requêtes booléennes. Le modèle de proximité floue que nous proposons dans notre thèse permet d'allier expressivité des requêtes booléennes et utilisation de la proximité. Chaque mot-clé dans le document possède une zone d'influence utilisée dans l'évaluation de la requête. Les opérations floues associées aux opérateurs booléens classiques sont utilisées pour cette évaluation : la proximité est ainsi propagée jusqu'à la racine de l'arbre. Le modèle proposé dans notre thèse a été largement validé sur les collections de test classiques et pour les éditions 2005 et 2006 des campagnes d'évaluations internationales de recherche d'informations (TREC, CLEF et INEX 2006). Les résultats obtenus avec des requêtes construites automatiquement sont, dans la plupart des cas, équivalents à ceux des méthodes de référence (Okapi implanté dans le système Lucy, le modèle vectoriel pour le système MG). Par contre, avec l'utilisation de requêtes manuelles adaptées au modèle de proximité floue, les résultats sont très largement supérieurs aux modèles classiques.