THÈSE

En vue de l'obtention du

DOCTORAT DE L'UNIVERSITÉ DE TOULOUSE

Délivré par : *l'Université Toulouse 3 Paul Sabatier (UT3 Paul Sabatier)*

Présentée et soutenue le *Date de soutenance (07/10/2014)* **par :**

ANTHONY BIGOT

Analyse et catégorisation des systèmes de recherche d'information pour la sélection et l'adaption aux besoins en information

---

JURY

SÉBASTIEN DEJÉAN	Ingénieur de Recherche	Co-Directeur
JOSIANE MOTHE	Professeur des Universités	Co-Directrice
CÉLINE PAGANELLI	Maître de Conférence HDR	Rapporteur
ÉRIC GAUSSIER	Professeur des Universités	Rapporteur

École doctorale et spécialité :
MITT : Image, Information, Hypermedia
Unité de Recherche :
Institut de Recherche en Informatique de Toulouse UMR 5505
Directeur(s) de Thèse :
Josiane MOTHE et *Sébastien DÉJEAN*
Rapporteurs :
Céline PAGANELLI et *Éric GAUSSIER*

Table des matières

Liste des tableaux

Liste des abréviations

AP Average Precision
ACP Analyse en Composantes Principales
AFC Analyse Factorielle des Correspondances
CALMIP CALcul en MIdi-Pyrénées
CAH Classification Ascendante Hierarchique
L2C Learning to Choose
MAP Mean Average Precision
MeMo Mesure Moyenne
RaMo Rang Moyen
RI Recherche d'Information
SRI Système de Recherche d'Information
SVM Support Vector Machine
TREC Text REtrieval Conference

Introduction

Plusieurs études ont montré la variabilité des résultats des Système de Recherche d'Information (SRI) (Harman 2009). Ainsi, un système S_1 peut être très performant sur un besoin A mais échouer sur un besoin B et un système S_2 peut avoir des performances inverses.

Les travaux qui exploitent la variabilité des résultats sont divers ; les premiers datent des premières années de Text REtrieval Conference (TREC) en 1992.

La première exploitation de la variabilité est certainement celle de la fusion des résultats. Selon cette approche, plutôt que d'utiliser un seul moteur de recherche et donc une technique de recherche unique, il vaut mieux utiliser plusieurs moteurs et en fusionner les résultats. Fox Fox1994 a été le premier a développer cette idée dans le cadre de TREC. Plusieurs approches de fusion ont été développées, qui prennent en compte les rangs et/ou les scores obtenus par les documents via les moteurs à fusionner. L'idée générale est la suivante : les documents sur lesquels les moteurs convergent sont renvoyés d'abord (haut rang, score important).

Les méthodes de fusion de résultats initiales avaient tendance à favoriser les documents qui se ressemblent (selon la *Cluster hypothesis*). D'autres approches ont vu le jour favorisant au contraire la diversité. Dans ce cas, l'idée est plutôt que différents moteurs ou techniques peuvent être utilisées pour répondre à une requête, cela afin de favoriser la recherche par rapport à plusieurs aspects. Les travaux sont nombreux dans ce domaine. (Dudognon 2014) a par exemple proposé un modèle qui permet de fusionner les résultats issus de plusieurs méthodes de recherche pour diversifier les recommandations faites aux utilisateurs de l'outil Overblog (www.over-blog.com).

Selon ces approches, quelle que soit la requête, la même combinaison ou fusion est réalisée.

D'autres approches s'intéressent au contraire à adapter la fusion aux requêtes. L'idée ici est que toutes les requêtes ne doivent pas être traitées de la même façon. Des travaux récents ont ainsi étudié l'expansion sélective de requêtes : il s'agit de n'étendre que les requêtes qui en ont besoin, c'est à dire celles qui seront améliorées par l'expansion. (Amati 2004) est un des premiers à avoir expérimenté l'expansion sélective. (Chifu 2014) a également proposé une méthode sélective qui se base sur des prédicteurs de difficulté des requêtes afin de décider si l'expansion doit ou non être effectuée. Les prédicteurs retenus sont à la fois des prédicteurs linguistiques et statistiques. Le modèle de décision est appris par un Support Vector Machine (SVM). L'efficacité de la méthode a été montrée sur la collection TREC robust et la précision du système obtenu est améliorée d'environ 11% par rapport à un système non sélectif.

Les travaux présentés dans cette thèse s'incrivent dans le cadre du projet ANR-2010-CORD-001-01 CAAS (contextual analysis and adaptive search, http ://www.irit.fr/CAAS/) ainsi que dans le cadre du projet ACRIC (Analyse Canonique et Recherche d'Information Contextuelle) de la fédération CNRS FREMIT FR3424 (http ://www.irit.fr/FREMIT/). L'hypothèse sous-

jacente à ces deux projets était qu'il existe des contextes dans lesquels un SRIsera plus adapté qu'un autre mais que ces contextes doivent d'abord être détectés par une analyse poussée des différentes variables pouvant influer sur les résultats d'un SRI et par l'étude d'un grand nombre de cas. ACRIC était un projet exploratoire soutenu en 2010 par la fédération FREMIT. A l'issue de ce projet, le projet CAAS a pris le relais. L'hypothèse du projet CAAS est que la prise en compte de ce contexte pourrait améliorer les performances d'un SRI. Si dans le projet l'aspect contextuel portait sur plusieurs éléments contextuels tels que les besoins des utilisateurs et les requêtes, les documents et les composants du système, cette thèse s'est centrée sur le dernier point. Les questions de recherche portent donc sur :

- le contrôle de la variété des contextes : une question importante concerne la variété des traitements et leur adéquation aux différents contextes connus ;
- l'adaptation au contexte : le système doit pouvoir décider lui-même des technologies ou des méthodes de Recherche d'Information (RI) les plus adéquates en fonction d'un contexte donné, c'est-à-dire qu'il doit adapter les méthodes de RI au contexte ;
- la reconnaîssance d'un contexte : quand un contexte est rencontré, le système doit le détecter parmi les contextes connus ou appris pour pouvoir décider quelle méthode il doit appliquer.

Ce travail de thèse a été abordé selon deux axes complémentaires :

- l'étude des paramètres des systèmes et la définition de divers critères de qualité des systèmes en fonction des attentes supposées des utilisateurs ;
- la proposition d'une méthode adaptative, dépendant de la requête rencontrée.

Sur le premier axe, nous avons tout d'abord constitué un corpus de travail. Notre objectif était de pouvoir étudier des phénomènes sur un grand nombre de cas. Si les données de TREC sont très riches, elles se sont vites avérées insuffisantes pour traiter notre problème. Ainsi, à l'aide de la plateforme de RI Terrier, nous avons contruit un jeu de données en faisant varier, les requêtes et les différents paramètres des différents modules intervenants dans un processus de recherche. Il s'agit là d'une ressource dont la communauté ne disposait pas et qui a été utilisée par plusieurs autres membres de notre équipe avec des résultats prometteurs.

Dans ce premier axe, nous avons également proposé deux critères de choix de moteurs : les moteurs les plus performants, ce sont ceux qui permettent d'obtenir les meilleurs résultats, même si ce n'est que sur quelques requêtes et les moteurs les plus robustes, ceux qui traitent correctement toutes les requêtes. Il est certain que dans une approche sélective, les systèmes les plus performants ont un grand intér êt dès lors que l'on est capable de prédire quel moteur le plus performant doit être appliqué sur une requête donnée. Des prédicteurs commencent à voir le jour dans la littérature concernant le besoin d'expansion de requêtes, nous pensons que d'autres prédicteurs verront le jour.

Dans le second axe, nous avons proposé une méthode sélective qui permet de choisir le système à utiliser pour une requête donnée. Ce modèle est particulièrement bien adapté dans le cadre de requêtes répétées (qui sont fréquentes sur les moteurs en ligne). Selon cette méthode, le système apprend le système ou la configuration de systèmes à utiliser pour une requête sur la base d'un échantillon de documents. Nous montrons que cette méthode est très efficace (augmentation de 20% de la précision moyenne (Average Precision (AP)).

La suite de ce manuscrit est organisé comme suit :

Dans le chapitre 1, nous présentons l'existant : plus spécifiquement nous décrivons d'abord le cadre général de la RI, puis nous présentons les outils que nous avons utilisés tout au long de nos recherches : les outils mathématiques d'abord, mais également les outils de recherche d'information ; et enfin quelques travaux qui se sont intéressés comme nous à exploiter la variabilité des résultats des moteurs ainsi que des travaux qui visent à analyser les variantes de moteurs ou les paramètres de ces moteurs.

Le chapitre 2 présente les ressources que nous avons contruites pour mener à bien nos recherches. Elles sont de deux natures : la première ressource est construite à partir des exécutions transmises par les participants à TREC ; la seconde a été construite par nos soins à partir de la plateforme Terrier.

Dans le chapitre 3, nous présentons et analysons une méthode de sélection basée sur les rangs des systèmes ; cette méthode vise à sélectionner les systèmes selon deux critères : leur robustesse ou leur performance.

Le chapitre 4 présente l'étude préliminaire qui a motivé la définition d'un nouveau modèle de recherche d'information sélective. Ce modèle est également présenté dans ce chapitre.

Le chapitre 5 présente une évaluation détaillée du modèle appelé Learning to Choose (L2C).

Enfin, une conclusion et des perspectives à ce travail concluent ce rapport de thèse.

Chapitre 1

Contexte

Sommaire du chapitre

1.1 Contexte : la recherche d'information

La RI est le domaine scientifique qui vise à faciliter la recherche d'un besoin en information dans un fond documentaire. Un SRI, aussi communément appelé moteur de recherche, est une surcouche développée autour de la collection de documents pour effectuer cette recherche. Cette surcouche permet d'optimiser le stockage, l'organisation et l'accès à l'information (Amini 2013).

Bien que les SRI les plus connus traitent des collections constituées essentiellement de pages web (Google, Yahoo, Bing !, Exalead...), il existe une grande hétérogénéité au sein des collections et des documents traités (structurés, semi-structurés, non structurés, collections textuelles, d'images ou encore multimédias...). Par ailleurs, la recherche multimédia existe et est utilisée pour des applications à succès (comme Shazam ou SoundHound pour la recherche de musique par exemple), la recherche de contenus multimédias est très souvent basée sur une recherche textuelle ; seule la restitution des résultats diffère d'une recherche textuelle. Les travaux présentés dans cette thèse utilisent uniquement des documents textuels.

Malgré la variabilité induite au sein des SRI de par la nature des documents traités, un processus générique de RI existe (Baeza-Yates 1999) ; il est présenté ci-dessous.

1.1.1 Processus général de la recherche d'information

Le but de tout SRI est de parcourir une collection de documents afin de trouver ceux qui répondent le mieux au besoin exprimé par l'utilisateur et de restituer une liste ordonnée de ces documents potentiellement "pertinents".

Le processus générique de RI (voir 1.1 est constitué de quatre phases incontournables :
- l'indexation de la collection des documents ;
- l'expression du besoin en information (ou requête) ;
- l'appariement de la requête à la collection ;
- la restitution des résultats correspondant à la réponse du système à la requête.

Il est possible d'ajouter une ou plusieurs étapes optionnelles à ce processus dans le but d'améliorer les résultats retournés par le SRI. Pour n'en citer qu'une qui sera utilisée dans cette thèse, il est possible de reformuler automatiquement la requête à partir d'un sous ensemble des résultats. Mais il existe de nombreuses méthodes additionnelles qui peuvent intervenir à différentes étapes du processus de RI.

La suite de la subsection détaille les différentes étapes du processus de RI.

1.1.1.1 Indexation

Tandis qu'il est difficile d'évaluer la taille réelle du web et plus généralement des données produites dans le monde, une chose est certaine, cette masse de données est sans cesse croissante : la taille du web double tous les 5 ans ! (Zhang 2008). Même si cette tendance semble s'infléchir un peu, le potentiel de croissance reste encore important si l'on considère la population mondiale et celle qui est utilisatrice d'Internet.

Vouloir répondre à un besoin en information en cherchant directement dans ces données "brutes" est inconcevable.

L'indexation est une transformation des documents qui représente le contenu des documents afin de permettre un parcours efficace et rapide de la collection. La forme d'index la plus utilisée est l'index inversé. Sa construction est décomposée en plusieurs étapes :

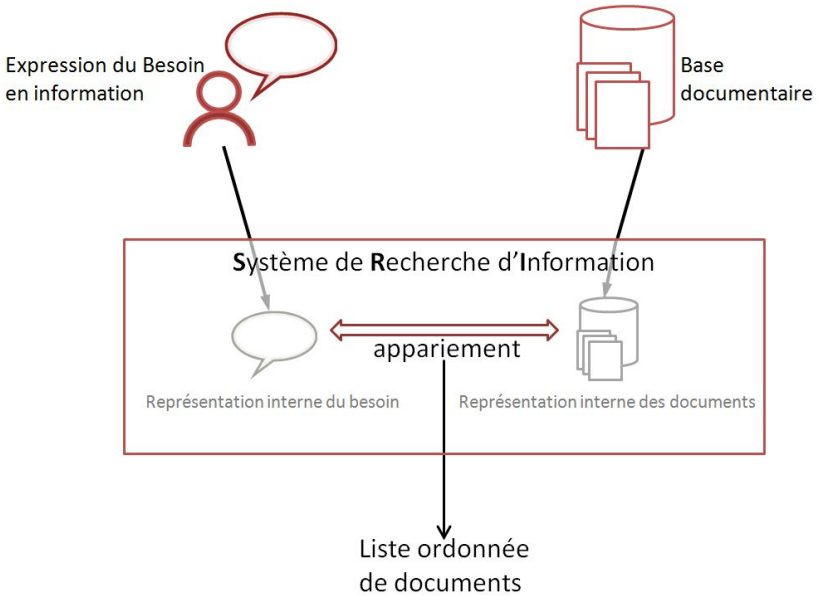

FIGURE 1.1 – Schéma simplifié de la RI.

1. suppression du format
2. extraction des mots utiles
3. normalisation
4. racinisation
5. pondération

La suite de la sous subsection détaille ces différentes étapes. Certaines étapes sont dépendantes de la langue utilisée, nous supposerons dans ce cas que le français est employé.

Suppression du format Une hétérogénéité latente existe au sein des documents. Ils peuvent être fournis en différents formats (texte, pdf, html...) et le texte peut être encodé de différentes manières (ISO-8859-1, UTF8, ASCII...). Une première étape de prétraitement consiste à supprimer tous ces formats pour transformer le document en une suite normalisée de caractères de manière à ce que seul le texte subsiste.

Extraction des mots utiles Les mots sont extraits par repérage des caractères séparateurs (comme les espaces ou la ponctuation). Les mots *vides* (car sans sémantique propre) sont retirés selon un anti-dictionnaire (ou *stoplist* en anglais) contenant l'ensemble des termes que l'on ne souhaite pas conserver.

Exemple 1 (Liste non exhaustive de mots vides français)**.**

à afin aussi assitôt aux ... c ça car ce ceci cela celle ... elle il ... la le les lui ...

Normalisation La normalisation regroupe différents mots que l'on souhaite analyser sous une forme unique car il n'est pas utile voire contreproductif de les considérer différemment pour la RI.

Exemple 2 (Normalisation)**.**

diode-électroluminescente, diode électroluminsescente, del, d.e.l., DEL, D.E.L. led, l.e.d., LED, L.E.D. peuvent être regroupés sous une forme unique.

Comme dans l'exemple précédent, il est parfois nécessaire de supprimmer les accents et/ou de modifier la casse. Cette approche pose certaines contraintes dans le cas de certains termes à majuscule comme les noms propres (par exemple "Boulanger" *vs* "boulanger") ou certaines marques.

Racinisation La racinisation et la lemmatisation consiste à décider d'une forme unique pour représenter les différentes variantes d'un terme. Les différentes formes d'un terme seront ramenées à une forme unique qui peut être un lemme si un algorithme de lemmatisation est utilisé ou un simple radical si un algorithme de racinisation est utilisé. Cette étape de l'indexation a pour but de réunir sous une seule forme des mots de même sens pour améliorer le processus de RI dans le cas où la variante du mot n'est pas la même dans la requête et dans le document.

Il existe plusieurs façons d'extraire le radical d'un mot. Une technique simple consiste à détecter les préfixes et suffixes et à les supprimer pour obtenir le radical ou à procéder à une troncature au $n^{ième}$ caractère ; dans ce dernier cas le radical correspond aux premiers caractères.

Cependant, les algorithmes les plus courament utilisés sont dérivés de celui de Porter (Porter 1980). Celui-ci est défini par une cinquantaine de règles en sept étapes permettant notament de détecter les pluriels et les différentes formes des noms, adjectifs et verbes.

Exemple 3 (Racinisation)**.**

Termes originaux : voiture voiturier covoiturage
Racinisations possibles : voitur ou voiture

La lemmatisation nécessite des traitements linguistiques plus poussés et plus coûteux en temps de calcul, ce qui peut expliquer que les algorithmes de type Porter leur soient préférés.

Pondération des termes Les termes obtenus à l'issue de l'étape précédente (radicaux ou lemmes) n'ont pas la même importance pour représenter les contenus des documents ; ils sont donc pondérés afin de refléter cette importance. Le poids d'un terme dans un document est calculé dans le but de modéliser le degré de représentativité du terme pour un document donné ou plus exactement pour représenter son pouvoir discriminant ; c'est à dire sa capacité à discriminer les documents pertinents des documents non pertinents pour une requête qui contiendrait ce terme.

La forme générique de pondération la plus utilisée est TF-IDF où TF représente la fréquence du terme t_i dans le document d et IDF la fréquence inverse, en lien avec le nombre de documents

qui contiennent ce terme. Différentes formules de TF-IDF existent et sont dérivées de celle proposée dans l'équation 1.1.

Équation (Pondération des termes TF-IDF).

$$Poids(t_{id}) = \underbrace{(1 + \log(tf_{id}))}_{TF} \times \underbrace{(\log(\frac{D}{d_i}))}_{IDF} \tag{1.1}$$

avec :
- tf_{id} la fréquence du terme i dans le document d
- D le nombre de documents dans la collection
- d_t le nombre de documents où le terme t apparait

Ainsi, un terme présent dans un document et peu présent dans les autres documents de la collection, est très représentatif du document pour la collection donnée.

L'index liste l'ensemble des termes sous leur forme racinisée, leur poids, les documents dans lesquels ils apparaissent et le plus souvent leur position dans le texte. Toutes ces informations sont essentielles pour calculer le score d'appariement de chaque document avec le besoin exprimé par l'utilisateur du SRI (*cf.* 1.1.1.2). Ce score est ensuite utilisé pour ordonner la liste des documents restituée à l'utilisateur (*cf.* 1.1.1.3).

1.1.1.2 Expression du besoin

L'expression du besoin en information formulée par un utilisateur est communément appelée *requête*. La requête est le plus souvent exprimée en langage naturel. Il serait plus juste de parler de formulation sans contrainte puisque les utilisateurs formulent généralement leurs requêtes sous la forme d'une juxtaposition de mots.

Pour être appariée aux documents, la requête de l'utilisateur subit le même processus d'indexation que les documents. Ainsi la forme de la requête convient à une comparaison avec l'index de la collection. On parle alors d'*appariement* de la requête.

Pour plus de détail sur la formulation de la requête, le lecteur est invité à lire (Tricot 2004) qui présente différentes définitions que l'on peut donner du besoin en information.

1.1.1.3 Modèle d'appariement

Depuis les débuts de la RI, de nombreux travaux se sont intéressés aux modèles de comparaison de la requête avec les documents puisque c'est le coeur du SRI.

Aujourd'hui, deux grandes familles de modèles se distinguent : les modèles de recherche vectoriels et les modèles de recherche probabilistes.

Modèles vectoriels Ces modèles représentent la requête et les documents sous forme d'un vecteur de termes (Salton 1968). Une fonction de score ou une combinaison de fonctions de score (Amini 2013) est alors utilisée pour calculer un score de pertinence supposée pour chaque document pour la requête considérée. La liste de documents restituée à l'utilisateur est ordonnée par ordre décroissant selon ce score de manière à ce que le document ayant obtenu le score le plus élevé soit premier de la liste.

Modèles probabilistes et de langue Ces modèles ne considérent pas la distance entre la requête et les documents mais ils s'appuient sur la théorie des probabilités conditionnelles. Ainsi, le modèle développé par S. Roberston calcule la probabilité qu'un document soit pertinent sachant la requête (Robertson 1977). Le principe d'ordonnancement probabiliste consiste à modéliser l'expérience aléatoire de tirer des documents (pertinents et non pertinents) selon des requêtes connues et à ensuite appliquer ce modèle pour trier les documents à restituer pour une nouvelle requête.

Sur ce principe, le modèle OKAPI BM25 est devenu une référence en matière de modèle de RI (Robertson 1996). Il utilise le fait que certains termes sont plus discriminants que d'autres pour déterminer la pertinence potentielle des documents en ce sens qu'ils sont très présents dans un petit nombre de documents et moins présents dans le reste de la collection.

Le modèle de langue développé par Ponte et B. Croft (Ponte 1998) s'appuie également sur la théorie des probabilités et calcule quant à lui la probabilité de générer un document sachant la requête. Le modèle de langue estime que les documents sont pertinents pour une requête s'il est possible de générer cette requête à partir de ces documents.

Par ailleurs, les modèles informationnels utilise le fait que la distribution des termes pertinents dans les documents est différent de la distribution des termes non pertinents (Harter 1975).

Le lecteur est invité à consulter (Amini 2013) pour plus de détails sur les différents modèles de RI.

La plateforme Terrier que nous avons utilisée dans le cadre de cette thèse implante nombre de modèles de la littérature, dont l'ensemble des modèles que nous avons présentés dans cette subsection.

1.1.1.4 Expansion de requête

Les modèles précédents supposent que la présence des termes de la requête dans les documents est un indicateur fort de la pertinence des documents. Cependant, il arrive bien souvent que l'expression d'un concept, d'une entité ou d'une notion soit différente dans la requête et dans les documents. La racinisation permet de s'affranchir de l'utilisation de différentes variantes, mais n'est souvent pas suffisante. L'expansion de requête est un traitement qui permet d'élargir la formulation de la requête et ainsi d'élargir le champ de recherche de cette requête. On parle également de reformulation automatique de la requête. Elle consiste à compléter la requête en y ajoutant des mots. Il existe différentes approches pour sélectionner les mots à ajouter. Généralement, les mots sont sélectionnés à partir de ceux issus des documents ayant obtenu le score le plus important suite à une première recherche.

Cette méthode est inspirée de la méthode de Rocchio (Rocchio 1971) qui choisissait les termes d'expansion à partir des documents retrouvés suite à une première recherche et jugés pertinents par l'utilisateur. Il s'est avéré que les utilisateurs ne sont pas prêts à juger les documents ; C. Burckley a donc proposé que les premiers documents retrouvés soient considérés comme pertinents (Buckley 1995). Ces méthodes sont appelées "retour de pseudo-pertinence" ou "rétro-pertinence aveugle" (respectivement "pseudo relevance feedback" et "blind relevance feeback" en anglais).

D'autres méthodes considèrent l'ensemble des documents de la collection et sélectionnent pour chaque terme ses plus proches voisins (Schütze 1998).

1.1.1.5 Bilan sur les paramètres de la recherche d'information

Le processus de RI est paramétré à chacune de ces étapes : indexation, pondération, appariement et expansion.

Ainsi, lorsqu'une nouvelle méthode est proposée, ses paramètres propres sont généralement étudiés sur différentes collections pour en connaitre l'influence.

Par ailleurs, des travaux se sont intéressés à l'analyse de ces paramètres de manière individuelle, c'est à dire sans en faire la combinaison. Ceux-ci seront détaillés dans le chapitre 2. Étant donné le grand nombre de combinaisons possibles des paramètres de RI, seuls quelques travaux s'intéressent à leurs combinaisons (*cf.* chapitre 2, subsection 2.2). Au contraire et comme nous le verrons plus loin, notre travail s'est centré sur l'analyse croisée de différents paramètres.

La subsection suivante présente différentes plateformes qui permettent de combiner ces paramètres ainsi que le cadre d'évaluation que nous avons utilisé pour les analyser.

1.1.2 Mise en œuvre et évaluation

1.1.2.1 Plateformes de recherche d'information

Différentes plateformes open sources sont utilisables pour travailler dans le contexte de la RI. Trois d'entres elles sont largement utilisées par la communauté et intègrent nativement certains paramètres du processus de RI : Indri (projet Lemur [1]), SolR/Lucene (projet Apache Lucene [2] [3]) et Terrier [4]. Il existe d'autres plateformes au sein de la communauté, comme la struture Unstructured Information Management application, qui ne sont pas présentées ici car utilisées pour des besoins différents de ceux de la thèse.

Indri a été développé dans la cadre du projet Lemur par l'université du Massachusetts et l'université Carnegie Mellon pour la recherche dans le domaine des modèles de langue et de la RI.

Lucene est une librairie pour la RI textuelle, développée par la fondation Apache. Elle est utilisée dans différents projets dont le projet **SolR** qui étend ses capacités et offre une facilité d'usage du langage de requête utilisé par Lucene.

Terrier est une plateforme développée par l'université de Glasgow permettant la construction de moteurs de recherche à partir de modules indépendants. Terrier implémente bon nombre de méthodes disponibles dans la littérature et permet de développer et d'ajuster ses propres méthodes.

Choix de la platefrome de RI Il serait trop long et fastidieux de détailler et de comparer tous les paramètres que les plateformes Indri, SolR/Lucene et Terrier intègrent ou permettent d'intégrer. Plusieurs études ont été menées pour comparer ces outils et tous trois possèdent des avantages et des inconvénients. Par exemple, Lucene permet une économie certaine sur la taille des index générés (Middleton 2008) tandis qu'Indri permet d'obtenir de meilleurs résultats (que SolR/Lucene) sur les collections que nous utilisons (Trotman 2012).

Notre objectif étant de générer un grand nombre de combinaisons de paramètres du processus de RI, trois points essentiels ont guidé notre choix :

- la liste des modèles déjà implémentés ;
- la documentation concernant ceux-ci ;

1. `www.lemurproject.org/indri.php`
2. `lucene.apache.org/`
3. `lucene.apache.org/solr/`
4. `terrier.org`

– la facilité à passer d'une configuration à une autre.

Il s'avère que Terrier possède une documentation très complète en ligne. Celle-ci concerne à la fois la liste des paramètres disponibles mais aussi leur implémentation via la javadoc de Terrier.

Cette documentation nous indique qu'un nombre suffisant de modèles et de paramètres existants sont supportés sans que nous ayons besoin d'en déployer. De plus, la configuration de Terrier est très facile ce qui permet une génération automatique pour pouvoir explorer l'ensemble des combinaisons de paramètres possibles.

1.1.2.2 Text REtrieval Conference

Les campagnes d'évaluation telles que TREC [5] ont permis de grandes avancées dans le monde de la RI. Ces campagnes sont basées sur le modéle d'évaluation de Cranfield (Cleverdon 1966). Selon ce modèle, un système à évaluer parcourt une collection figée de documents et recherche les documents à restituer à l'utilisateur pour un ensemble de besoins d'information.

Alors que Cleverdon proposait une évaluation exhaustive de la pertinence de tous les documents pour chacune des requêtes ; TREC propose un jugement partiel dans la mesure où le nombre de documents ne permet plus le jugement exhaustif. Ainsi, une fois que tous les participants à la campagne d'évaluation ont soumis leur exécution, appelée *run* en anglais, un échantillon de la collection basé sur leurs résultats est analysé à la main (méthode de vote ou *pooling* en anglais). Cette analyse vise à déterminer la liste des documents réellement pertinents pour chaque besoin.

Pour chaque SRI, la liste de documents restituée est comparée avec la liste de documents attendue pour un besoin donné. Cette comparaison permet de calculer les mesures de performances du système.

La collection de documents et de requêtes peut également servir à évaluer les nouveaux systèmes développés ou approches proposées. Ce cadre peut donc être utilisé pour comparer les SRI. Les mesures peuvent être calculées à l'aide de l'outil *trec_eval* fournit par TREC et présenté dans le chapitre 1.

1.1.2.3 Mesures de performance

Deux familles de mesures sont prépondérantes en RI : les mesures orientées précision et les mesures orientées rappel.

Mesures de précision Les mesures de précision déterminent si le SRI est capable de restituer d'avantage de documents pertinents que de documents non pertinents :

Équation (Calcul de la précision)**.**

$$Précision = \frac{nombre\ de\ documents\ pertinents\ restitués}{nombre\ de\ documents\ restitués} \tag{1.2}$$

Généralement, les utilisateurs des SRI s'intéressent seulement aux premiers documents restitués. C'est le cas par exemple lors de l'utilisation de moteurs de recherche en ligne. En calculant la

5. `trec.nist.gov/`

précision au rang r, c'est à dire après que r documents ont été restitués, il est possible de déterminer la capacité du SRI à restituer des documents pertinents. En fixant r à une valeur petite, par exemple 5 ou 10, on évalue la capacité du SRI à fournir des documents pertinents en tête de liste. Plusieurs valeurs de r sont généralement utilisées (5,10,20,30,50,100,200,500,1000). La R-précision correspond à r = le nombre de documents effectivement pertinents.

La précision moyenne (AP) quant à elle est calculée par :

Équation (Calcul de l'AP)**.**

$$AP = \frac{\sum_{r=1}^{n}(Précision(r) \times Pertinent(d_r)}{nombre\ de\ documents\ pertinents\ dans\ la\ collection} \tag{1.3}$$

avec :
- r le rang du document ;
- n le nombre de documents restitués ;
- d_r le document restitué au rang r ;
- $Précision(r)$ définie à l'équation (1.2) ;
- $Pertinent(d_r)$ qui vaut 1 si le document est pertinent, 0 sinon.

À chaque document pertinent restitué (appelé point de rappel), la précision est calculée. La somme de ces précisions est divisée par le nombe total de documents pertinents.

Ces mesures sont calculées pour une requête. Cependant l'évaluation d'un système s'appuie sur un ensemble de requêtes. La moyenne arithmétique est alors utilisée. Ainsi la MAP (Mean Average Percision) qui correspond à la moyenne des AP pour plusieurs requêtes est une des mesures les plus utilisées lorsque l'on souhaite comparer les résultats otenus par différents systèmes.

Équation (Calcul de la MAP)**.**

$$MAP = \frac{1}{R}\sum_{req=1}^{R} AP(req) \tag{1.4}$$

avec : R le nombre total de requêtes.

Mesures de rappel. Les mesures de rappel déterminent si le SRI est capable de restituer l'ensemble des documents pertinents de la collection. Elles mesurent donc sa capacité à être exhaustif.

Équation (Calcul du rappel)**.**

$$Rappel = \frac{nombre\ de\ documents\ pertinents\ restitués}{nombre\ de\ documents\ pertinents\ dans\ la\ collection} \tag{1.5}$$

Comme pour la précision, la mesure de rappel peut être calculée à différents rangs. Et de la même façon, les mesures sur un ensemble de requêtes sont obtenues en calculant la moyenne des valeurs obtenues pour chaque requête.

Autres mesures. Le problème des mesures de rappel et de précision est qu'elles varient en sens inverse. En effet, idéalement, les SRI obtiendrait à la fois de bons scores de rappel et de précision. Il s'avère que d'une manière générale, plus un SRI est précis, moins il est exhaustif, et inversement.

Pour permettre de comparer facilement des systèmes, d'autres mesures existent qui combinent le rappel et la précision. Certaines combinent directement les mesures de rappel et de précision, comme la F-mesure par exemple qui est la moyenne harmonique du rappel et de la précision :

Équation (Calcul de la F-mesure)**.**

$$F - mesure = \frac{2 \times Rappel \times Précision}{Rappel + Precision} \tag{1.6}$$

En réalité, les mesures d'évaluation sont nombreuses et on en dénombre plus d'une centaine dans l'outil trec_eval par exemple. Cet outil est présenté plus spécifiquement dans le chapitre 1.3.2.

1.2 Méthodes et outils statistiques

Cette section a pour objet de présenter succintement les méthodes statistiques qui sont intensivement utilisées dans les chapitres suivants. Elles sont regroupées ici en deux familles : les méthodes exploratoires et les tests statistiques. Dans chaque famille, nous rappelons l'objectif de l'outil utilisé, son principe de fonctionnement et nous en donnons une illustration sur un exemple.

1.2.1 Jeu de données illustratif

Dans cette section, certaines méthodes seront illustrées en utilisant le jeu de données présenté dans le tableau 1.1. Ce jeu de données contient la valeur de la MAP obtenue par 10 SRI sur 10 requêtes.

1.2.2 Analyses exploratoires

Les méthodes et outils de cette partie visent à proposer un point de vue sur les données sans a priori particulier. Nous nous plaçons ici dans le cadre d'une première prise de contact avec les données au sujet desquelles nous essayons d'avoir quelques premières informations.

TABLE 1.1 – Jeu de données illustratif contenant la MAP obtenue par 10 SRI sur 10 requêtes.

	APL985L	APL985LC	APL985SC	AntHoc01	Brkly24
T351	0.23	0.23	0.17	0.29	0.30
T352	0.02	0.03	0.06	0.03	0.04
T353	0.33	0.31	0.29	0.21	0.37
T354	0.11	0.15	0.09	0.01	0.02
T355	0.10	0.07	0.03	0.14	0.10
T356	0.05	0.06	0.05	0.01	0.01
T357	0.14	0.18	0.14	0.10	0.33
T358	0.10	0.10	0.05	0.15	0.21
T359	0.04	0.03	0.01	0.02	0.03
T360	0.39	0.38	0.41	0.04	0.33
	Brkly25	Brkly26	CL98CLUS	CL98COMB	CL98RANK
T351	0.31	0.88	0.67	0.71	0.71
T352	0.01	0.31	0.44	0.51	0.48
T353	0.26	0.25	0.45	0.43	0.38
T354	0.11	0.16	0.15	0.17	0.09
T355	0.18	0.26	0.45	0.46	0.43
T356	0.05	0.03	0.26	0.26	0.17
T357	0.33	0.35	0.41	0.15	0.11
T358	0.39	0.44	0.71	0.72	0.68
T359	0.04	0.06	0.03	0.03	0.03
T360	0.04	0.33	0.49	0.50	0.30

1.2.2.1 Représentations graphiques

La plupart des résultats seront illustrés par des représentations graphiques. Nous utiliserons en particulier des représentations en boîtes à moustaches ou *boxplot*. Cette réprésentation vise à fournir un aperçu de la distribution d'un nombre important de données. Elle s'appuie sur le calcul des quartiles de la distribution des données : le premier (q1) et le troisième quartile (q3) donnent les limites de la boîte, le trait central matérialise la médiane des données. Les moustaches joignent les extrémités de la boîte (c'est à dire les quartiles) soit au minimum (ou au maximum) de la série de données soit à une valeur correspondant à $quartile +/- 1.5(q3 - q1)$. Toute valeur dépassant ces limites est alors représentée individuellement.

Comme le montre la figure 1.2, des boxplots représentées ici en parallèle pour illustrer les performances des 10 systèmes donne un aperçu immédiat sur d'éventuels décalages dans les distributions (entre les systèmes AntHoc01 et Brkly26 par exemple) ainsi que sur des dispersions différentes (entre les systèmes Brkly26 et CL98RANK).

Ce type de représentations sera abondamment utilisé au moment de comparer comme ici les performances obtenues par plusieurs SRI sur un ensemble de requêtes.

1.2.2.2 Analyses factorielles

De façon schématique, les analyses factorielles ont pour objectif de réduire la dimension d'un jeu de données en le projetant dans un espace de plus petite dimension permettant ainsi de le représenter de manière synthétique. Cette représentation est optimale dans le sens où

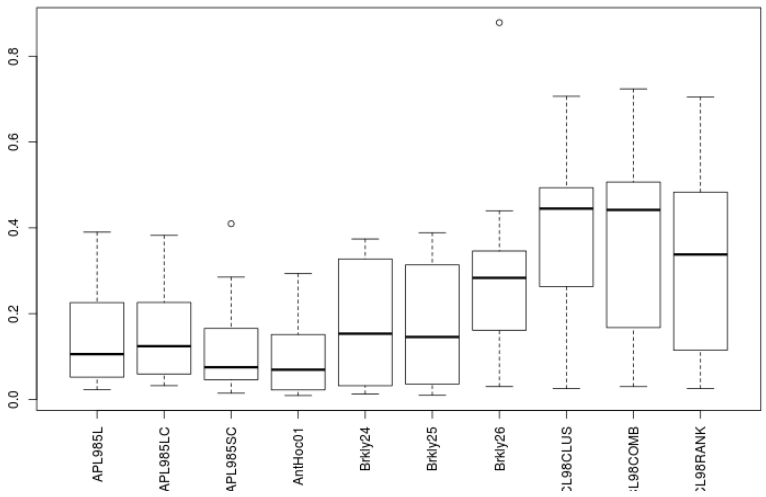

FIGURE 1.2 – Représentations graphiques sous forme de boxplot de 10 distributions des performances associées à chaque système.

elle conserve la plus grande part d'information (c'est à dire de variabilité) possible dans les dimensions de l'espace obtenu. Selon la nature des données à analyser, les méthodes utilisent des métriques différentes pour trouver la projection optimale.

Dans la suite des travaux, nous utiliserons l'Analyse en Composantes Principales (ACP) pour avoir un aperçu d'un jeu de données composés exclusivement de variables quantitatives. Plus ponctuellement, nous aurons recours à l'analyse des correspondances (simple ou multiple) pour traiter des données qualitatives (provenant éventuellement du codage en classes de variables quantitatives). Par exemple, une variable quantitative exprimant le score de plusieurs SRI pour une requête donnée pourra être recodée en classes "Bon", "Moyen", "Mauvais" exprimant la qualité d'un SRI.

Les résultats d'une ACP mise en œuvre sur les données du tableau 1.1 sont représentés dans la figure 1.3.

Sur une représentation de ce type, on peut ainsi identifier des systèmes au comportement relativement proches (APL985*) ainsi que des requêtes particulières. Par exemple, la position de la requête T359 à l'opposé des directions pointées par les flèches caractérisant chaque système indique une requête particulièrement difficile pour laquelle l'ensemble des systèmes ont obtenu des performances faibles. Une interprétation comme celle-ci doit s'accompagner d'une évaluation des parts de variance expliquée par les composantes principales. Ici, respectivement 69% et 17% sont portées par les première et deuxième composantes principales ce qui donne une confiance raisonnable dans l'interprétation de cette analyse.

1.2.2.3 Classifications

Les techniques de classification visent à rechercher sans a priori une structure dans les données. Parmi les nombreuses méthodes qui existent dans ce contexte, nous nous focaliserons

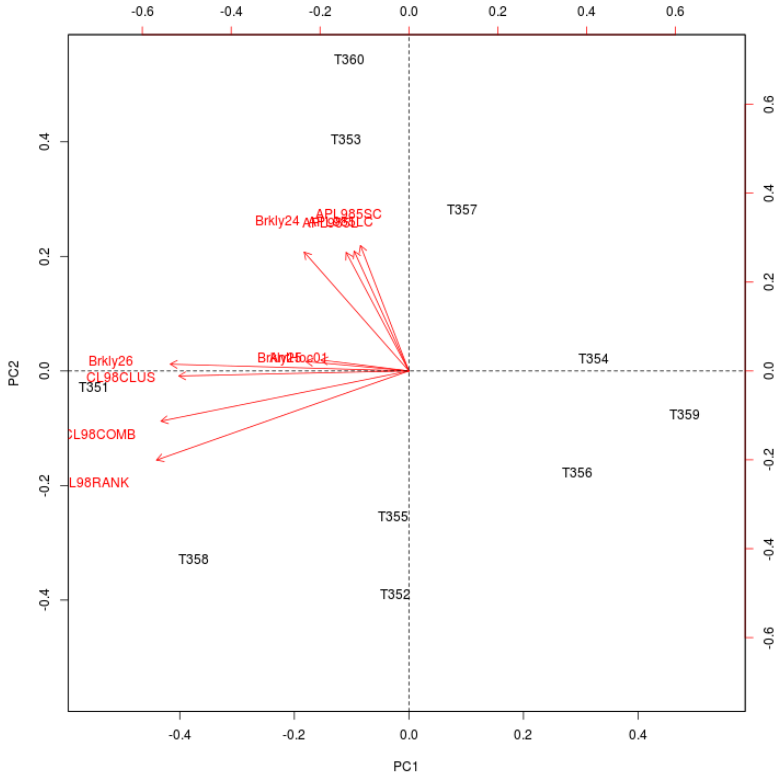

FIGURE 1.3 – Représentation simultanée des individus et des variables du tableau 1.1 après projection sur le premier plan principal expliquant environ 86% (PC1 : 69%, PC2 : 17%) de la variabilité totale des données.

sur l'utilisation (parfois combinée) de la Classification Ascendante Hierarchique (CAH) et de la classification par nuées dynamiques.

Classification ascendante hiérarchique Étant donnés une distance inter-individus (en général la distance euclidienne) et un critère d'agglomération (en général le critère de Ward), le principe de la CAH peut être décrit par la procédure itérative suivante :

- Début : chaque individu est un groupe
- Déroulement : regroupement des 2 objets les plus proches
- Fin : une classe regroupe tous les individus

Le résultat de la méthode est matérialisé par la construction d'un arbre de classification ou dendrogramme. Le fait de couper cet arbre à un certain niveau fournira alors un nombre de groupes selon lequel il est raisonnable de structurer les données.

Classification par nuées dynamiques Étant donné un nombre de groupes (k) fixé soit par une connaissance *a priori* du phénomène étudié, soit par une autre méthode (classification hiérarchique par exemple), un algorithme de classification par nuées dynamiques se déroule de

la façon suivante :
- Début : k centres (tirés aléatoirement ou imposés)
- Déroulement : tous les individus sont affectés au centre le plus proche ; les centres de chaque groupe sont recalculés
- Fin : les individus ne changent pas de groupe entre 2 étapes successives (ou le nombre maximal d'itérations est atteint en cas d'alternance systématique à chaque étape par exemple)

Le résultat de la méthode est une répartition des individus en k groupes.

Combinaison de méthodes de classification Comme cela est décrit dans (Lebart 2006), une approche hybride combinant les deux approches est parfois souhaitable pour mener à bien une démarche de classification. Dans ce cas, on s'appuie sur les points forts des deux approches : faibles temps de calculs pour la méthode des nuées dynanimques, absence d'information a priori pour la classification hiérarchique.

Face à un problème de classification d'un nombre important (plusieurs milliers) d'observations, il peut peut-être intéressant de procéder d'abord à une classification par nuées dynamiques avec un nombre relativement élévé de groupes (quelques dizaines voire quelques centaines). Les centres de ces groupes sont ensuite soumis à une classification hiérarchique qui donnera une indication pour déterminer le nombre de groupes K structurant l'ensemble des données. Une nouvelle application d'une méthode de nuées dynamiques à K groupes sur l'ensemble des données répond finalement au problème posé initialement.

Illustration Les résultats d'une méthode de classification hiérarchique sur les données du tableau 1.1 sont représentés sur la figure 1.4.

Le dendrogramme obtenu sur les requêtes suggère plusieurs niveaux de regroupement pertinents ; un niveau raisonnable étant caractérisé par un distance relativement grande entre les hauteurs de deux noeuds successifs entre lesquels on décide de couper. Ici le choix de deux groupes semble assez natuel, une classification plus fine en trois voire quatre groupes serait également pertinente. Concernant la classification des systèmes, le choix de deux groupes s'impose.

Dans les deux cas, la caractérisation des groupes peut être guidée par l'interprétation des résultats d'une analyse en composantes principales ou par des analyses univariées par groupe.

1.2.3 Tests statistiques

1.2.3.1 Principe

De manière globale, les tests statistiques ont pour objectif de prendre une décision concernant une population à partir des observations effectuées sur un échantillon de taille restreinte. Cette démarche est illustrée schématiquement sur la figure 1.5. En effet, sont représentés sur cette figure :
- deux courbes théoriques caractérisant la distribution de probabilités sur deux populations ;
- des marques sur l'axe des abscisses correspondant à des réalisations observées sur deux échantillons, un dans chaque population, de taille 10.

Un test statistique décidera, avec un certain risque d'erreur, si les observations permettent d'invalider une hypothèse posée sur les distributions théoriques.

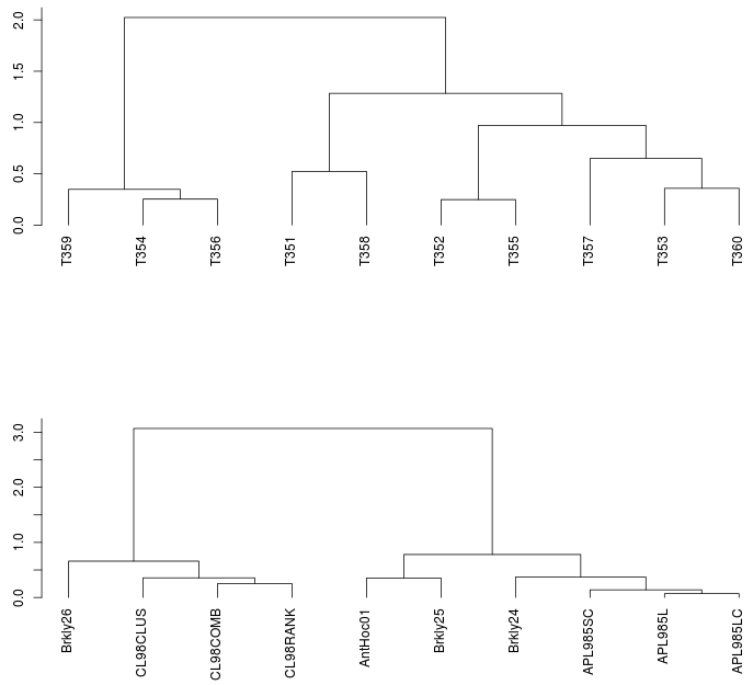

FIGURE 1.4 – Dendrogrammes issus de la classification ascendante hiérarchique mise en œuvre sur les requêtes (en haut) et sur les systèmes (en bas).

Dans nos études, nous aurons recours assez régulièrement à des tests de comparaison de moyennes afin de décider si, par exemple, les résultats d'un SRI sont significativement meilleurs que ceux d'un autre.

1.2.3.2 Comparaison de 2 moyennes

Un des tests les plus couramment utilisé est le test de Student qui vise à comparer les moyennes de deux séries d'observations. Dans sa version classique, il suppose égales les variances des 2 échantillons (ce qui peut-être vérifié par un test de comparaison de variances).

Nous illustrons ici sa mise en œuvre sur certaines données du tableau 1.1 en comparant les performances moyennes des systèmes APL985L et APL985LC d'une part et des systèmes AntHoc01 et CL98CLUS d'autre part.

Dans le premier cas, on obtient les moyennes 0.150 pour APL985L et 0.153 pour APL985LC. Le degré de significativité (ou p-value) du test, obtenue en se référant à une loi de Student à 18 degrés de liberté, est de 0.948 ce qui est très largement supérieur au risque d'erreur de 5% généralement fixé. L'hypothèse de base affirmant que les performances moyennes des 2 systèmes sont égales ne peut donc pas être invalidée par les observations effectuées sur les 10 requêtes.

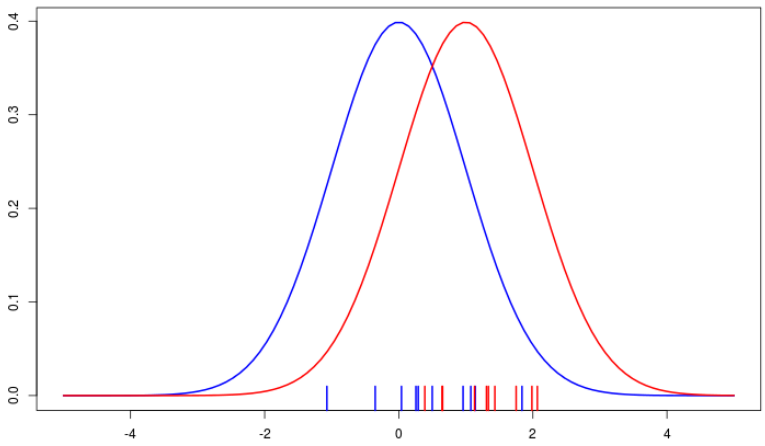

FIGURE 1.5 – Illustration du principe d'un test statistique par la représentation.

```
     Two Sample t-test
data:  dat[, 1] and dat[, 2]
t = -0.0667, df = 18, p-value = 0.9476
alternative hypothesis:
     true difference in means is not equal to 0
95 percent confidence interval:
  -0.1186414  0.1113414
sample estimates:
mean of x mean of y
   0.14999   0.15364
```

Dans l'autre cas, les performances moyennes pour AntHoc01 et CL98CLUS sont respectivement de 0.10 et 0.30. La p-value obtenue vaut 0.021 ce qui est inférieur au seuil de 5%. On peut dans ce cas affirmer, avec un risque de se tromper inférieur à 5%, que les performances des 2 systèmes sont significativement différentes. Un test unilatéral permettrait de préciser la conclusion en affirmant que la performance du système CL98CLUS est significativement supérieure à celle du système AntHoc01.

```
     Two Sample t-test
data:  dat[, 4] and dat[, 7]
t = -2.5338, df = 18, p-value = 0.02079
alternative hypothesis:
     true difference in means is not equal to 0
95 percent confidence interval:
  -0.37695541 -0.03520459
sample estimates:
mean of x mean of y
   0.10021   0.30629
```

1.2.3.3 Comparaison de k moyennes

Dans le cas de la comparaison de plusieurs moyennes (plus de deux), la méthode à utiliser est l'ANOVA. La mise en œuvre d'une ANOVA permet de tester l'hypothèse affirmant que l'ensemble des k moyennes étudiées sont égales. Une p-value supérieure à un seuil de 5% signifie que les mesures effectuées sur un échantillon ne permettent pas d'invalider l'hypothèse d'égalité des moyennes. Dans le cas contraire, la conclusion est qu'au moins une des conditions présente une moyenne différente des autres. Pour affiner cette conclusion, de nouveaux tests (parfois appelés tests post-hoc) peuvent être menés en comparant cette fois-ci les moyennes deux-à-deux. Des corrections tenant compte de la multiplicité des tests effectués sont généralement utilisées ; la méthode de Bonferronni qui consiste à diviser le risque d'erreur par le nombre de comparaisons à effectuer étant la plus classique et la plus conservative.

1.2.3.4 Comparaison de 2 variances

Le test F (ou test de Fisher-Snedecor) permet de comparer les variances de 2 séries d'observations. Nous utiliserons ce test afin de vérifier la robustesse des résultats fournis par nos approches. Nous associerons une plus grande robustesse à une approche fournissant des résultats moins variable qu'une méthode de référence.

Nous illustrons ici sa mise en œuvre sur certaines données du tableau 1.1. Dans le premier cas, nous testons l'égalité des variances des performances des systèmes APL985L et APL985LC. Visiblement les boxplots présentées précédemment indique une variabilité très similaire des 2 systèmes. C'est ce que confirme le test de comparaison de variances dont les sorties sont proposées ci-dessous. On constate effectivement un rapport des variances de 1.08 et un p-value associée de 0.91, ce qui exclut le fait de rejeter l'hypothèse d'égalité des variances.

```
     F test to compare two variances

data:  dat[, 1] and dat[, 2]
F = 1.0806, num df = 9, denom df = 9, p-value = 0.91
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
  0.2684068 4.3505065
sample estimates:
ratio of variances
          1.080604
```

Dans le second cas testé ici (comparaison des systèmes AntHoc01 et CL98RANK), en se fixant un risque d'erreur de 5%, on ne peut toujours pas rejeter l'hypothèse d'égalité des variances en faveur d'une hypothèse de différence des variances ; la p-value vaut 0.07. Cependant, le risque que l'on prendrait en rejetant cette hypothèse serait nettement moindre que dans le premier cas.

```
     F test to compare two variances

data:  dat[, 3] and dat[, 10]
F = 0.2791, num df = 9, denom df = 9, p-value = 0.07099
alternative hypothesis: true ratio of variances is not equal to 1
95 percent confidence interval:
  0.06932929 1.12373275
sample estimates:
ratio of variances
          0.2791193
```

Cela étant, en considérant une hypothèse alternative unilatérale, traduisant le fait que l'on souhaite savoir si l'une des variances est significativement inférieure à l'autre (et non plus différente), on obtient cette fois-ci un résultat significatif au seuil de 5%.

```
     F test to compare two variances

data:  dat[, 3] and dat[, 10]
F = 0.2791, num df = 9, denom df = 9, p-value = 0.03549
alternative hypothesis: true ratio of variances is less than 1
95 percent confidence interval:
  0.0000000 0.8872905
sample estimates:
ratio of variances
          0.2791193
```

1.3 Applications disponibles pour l'évaluation en recherche d'information

1.3.1 Moteur d'indexation et de recherche : Terrier

Terrier ou Terabyte Retriever est un projet initié en l'année 2000 à l'université de Glasgow au « Department of Computing Science». C'est un logiciel libre (open source) écrit en langage Java (Ounis 2006). Terrier est une plateforme qui intègre des modules d'indexation de documents ainsi que des modules de recherche robustes. Étant un projet open source, ses codes sont modifiables. Terrier peut être téléchargé sur le site http ://terrier.org. Terrier permet d'effectuer une tâche complète de RI. En effet, il permet d'effectuer l'indexation des documents ainsi que la recherche. Il intègre aussi l'outil d'évaluation de la recherche. La plateforme Terrier inclut de nombreux modèles de recherche.

Comme tout SRI, Terrier permet d'effectuer une indexation de documents. Il permet d'indexer diverses collections telles que celles de TREC et des collections Web. Terrier permet aussi d'indexer des documents de divers formats tels que word, excel, powerpoint, pdf, xml,... En général, le logiciel utilise les diverses étapes de l'indexation. Il permet d'éliminer les mots vides à partir d'une liste paramétrable. Il effectue ensuite la racinisation des termes en utilisant un algorithme paramétrable. Par défaut, il utilise l'algorithme de Porter pour la langue anglaise. Il possède d'autres modules d'algorithmes de Porter (voir aussi le projet SnowballStemmer http ://snowball.tartarus.org/) pour traiter d'autres langues comme le français, l'allemand, le

russe, l'italien, le danois, l'espagnol,... Terrier calcule les différentes fréquences qui seront utilisées telles que la fréquence des termes et le nombre de documents contenant les termes. Ainsi l'indexation est effectuée et les termes d'indexation avec les diverses pondérations sont stockés dans une structure de données. Le processus simplifié de l'indexation sur Terrier est présenté dans la figure 1.6.

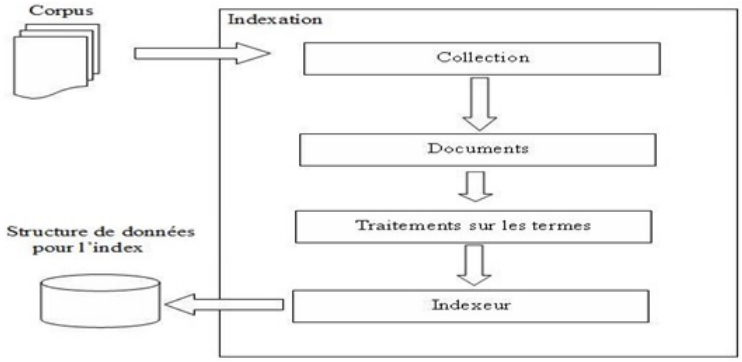

FIGURE 1.6 – Processus de l'indexation sur Terrier, traduit de (Ounis 2006)

Le logiciel Terrier permet aussi d'effectuer une recherche. Il offre plusieurs modèles avec divers paramètres. Les modèles de recherche par défaut de Terrier sont basés sur le modèle DFR (Divergence From Randomness) mais il intègre aussi d'autres modèles comme le modèle probabiliste BM25 aussi connu sous le nom du moteur associé Okapi, le modèle vectoriel tf*idf, le modèle de langue. Terrier permet d'effectuer une expansion de requêtes (pseudo relevance feedback) et une reformulation de requêtes basée sur la réinjection de pertinence (*relevance feedback*). Cette reformulation de requêtes n'est intégrée qu'à partir de la version 3.0. Le processus de recherche sans expansion de requêtes simplifié de Terrier est schématisé dans la figure 1.7.

1.3.2 Trec_Eval

Trec_eval est l'outil officiel fournit par TREC pour mesurer les performances des SRI partcipants aux campagnes d'évaluation. Cet outil prend deux documents en paramètre pour calculer les mesures : une exécution et un jugement de pertinence. Il permet de calculer 135 mesures dont les différentes précisions, MAP et rappel.

Deux types de mesures différents sont calculés. Les premières concernent les performances d'un système pour le traitement d'une requête donnée sur la collection de documents comme par exemple la mesure de précision moyenne. Les secondes concernent les performances d'un système pour l'ensemble des requêtes comme par exemple la Mean Average Precision (MAP).

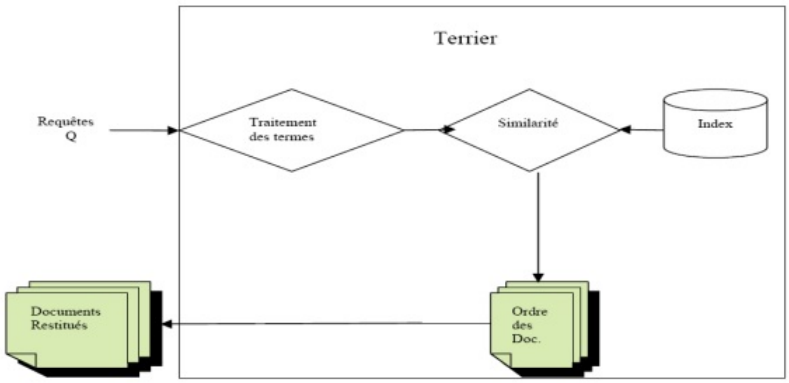

FIGURE 1.7 – Processus de la recherche sur Terrier, traduit de (Ounis 2006)

1.4 Etat de l'art

1.4.1 Combinaison de systèmes

Dans la littérature, différents travaux se sont intéressés à combiner les résultats provenant de différents systèmes. Lors de la campagne TREC-3 ont eu lieu les premières expérimentations sur la fusion des systèmes de recherche d'information (Fox 1994). Dans cet article, l'auteur analyse une série de méthodes de fusion des SRI, dont le célèbre CombMNZ, et montre une amélioration significative des résultats par rapport aux performances des systèmes initiaux. L'étude de ces méthodes de fusion montre que l'utilisation de rangs au lieu de la similarité produit de meilleurs résultats dans la combinaison des résultats lorsque les systèmes initiaux ont des rangs très différents (Lee 1997). De nombreux auteurs se sont employés à la mise au point de nouveaux algorithmes de fusion des SRI existants. Par exemple, Lillis (Lillis 2006) étudie les résultats de différents algorithmes de RI et propose un modèle de fusion basé sur les probabilités (probFuse). L'auteur compare ces résultats à ceux obtenus par CombMNZ et montre une amélioration de la MAP allant de 15% à 50% selon la taille de la collection utilisée durant la phase d'apprentissage.

Des analyses ont également porté sur les différents paramètres intervenant dans la combinaison comme le nombre initial de systèmes, leurs performances et leur diversité (Liu 2012). Cette étude considère les rangs obtenus par les systèmes et montre que la performance de la combinaison n'augmente pas forcément avec le nombre de systèmes utilisés. L'auteur montre également que les performances sont améliorées lorsque les systèmes initiaux sont relativement performants et qu'ils sont suffisamment différents selon le critère RSC ("Rank-score characteristics" en anglais)

Une seconde source de variabilité dans les performances des systèmes de RI provient de la variabilité des besoins en information et du contexte dans lequel ils sont formulés. La littérature montre de nombreuses analyses menées pour catégoriser les besoins en information. Mothe (Mothe 2005) analyse 16 traits linguistiques afin de prédire la difficulté d'un besoin en information. Les auteurs montrent que 3 de ces traits ont un impact significatif sur les mesures

de rappel et/ou de précision. Carmel (Carmel 2010) s'intéresse à la prédiction de la difficulté d'un besoin en information nouvellement formulé et donc non connu au préalable.

La variabilité des besoins en information et du contexte dans lesquels ils sont formulés a conduit la RI à évoluer vers une RI adaptative (Joho 2008). Les premiers travaux dans ce domaine s'intéressent aux modifications à apporter au processus de RI selon les informations tirées du contexte dans lequel il se déroule (Brown 2001). D'autres travaux se focalisent sur la formulation du contexte sous forme d'un besoin en information afin d'améliorer la pertinence des résultats restitués par les systèmes de RI (Menegon 2009). Cependant, la mise en place d'un cadre d'évaluation de tels algorithmes comportent de nombreuses difficultés (Mizzaro 2008).

En s'appuyant sur l'hypothèse d'une variabilité dans la manière de traiter les besoins en information, Kompaoré (Kompaoré 2008) étudie le meilleur choix possible de systèmes de RI selon des critères linguistiques du besoin en information.

Dans ce contexte, nous avons également proposé une méthode d'apprentissage du meilleur système de recherche d'information par classe de difficulté des besoins (Bigot 2011). Nous avons montré une amélioration significative de la mesure MAP de 10% pour les besoins classés faciles et difficiles sur la collection de documents test. Le gain en performance augmente à 24% sur les besoins classés moyennement difficiles à traiter. Ces résultats sont présentés en détail dans le chapitre 5 de cette thèse.

Par ailleurs, la variabilité des résultats de recherche d'information ont particulièrement été étudiés dans le workshop RAI (Reliable Information Access) (Harman 2004), (Harman 2009) et (**?**).

1.4.2 Analyse simultanée de différents moteurs de recherche ou de variantes de moteurs de recherche

Lors de la mise au point d'une nouvelle méthode intervenant dans le processus de RI, il est habituel d'analyser l'influence des paramètres de cette méthode sur une ou plusieurs collections. Il est également commun de comparer les résultats obtenus avec d'autres méthodes issues de la littérature. Dans cette section, nous ne présentons pas ce type d'analyse. Plutôt nous nous intéressons aux travaux qui ont eu pour objectif de réaliser des études globales sur différents moteurs ou sur des ensembles de paramètres.

En 2005, (Chrisment 2005) a étudié les résultats soumis par les 13 groupes participants à la tâche TREC Novely 2002, soit au total 43 systèmes. En pratique, un participant utilise généralement un seul outil pour lequel il teste différents paramètres. Partant du constat que les performances des systèmes sont calculées de façon globale, en calculant des mesures moyennes sur un ensemble de requêtes, les auteurs se sont intéressés aux résultats plus fins, basés sur les requêtes individuelles. Ils se sont plus particulièrement intéressés à l'étude des corrélations entre les résultats.

Les auteurs ont d'abord considéré uniquement deux mesures, prises de façon indépendante, le rappel et la précision. Chaque requête a été considérée comme un individu (une ligne de la matrice) et chaque système comme une variable (une colonne de la matrice) ; la matrice contenant soit le rappel, soit la précision obtenue. Les requêtes ont alors ensuite été classées en fonction des résultats obtenus par les différents systèmes. Il s'est donc agit de classer les requêtes par rapport à leurs profils similaires. Deux requêtes qui ont été traitées de façon similaire par les différents systèmes se retrouvent ainsi regroupées (par exemple deux requêtes avec une forte performance pour le système 1 et une faible performance pour le système 2).

En réalité, les résultats ont montré que cette classification permettait d'obtenir des groupes de requêtes de difficultés similaires : c'est à dire qu'un groupe contenait des requêtes faciles pour pratiquement tous les systèmes et qu'un autre groupe contenait les requêtes difficiles pour pratiquement tous les systèmes (voir le dendrogramme résultant de la Classification Ascendante Hierarchique (CAH) de la figure 1.8). Toutefois, les auteurs ont pu également montré que les systèmes qui correspondaient en fait à des variantes d'un même moteur obtenaient des résultats plus proches que deux systèmes différents. Pour cela ils ont utilisé une Analyse en Composantes Principales (ACP) sur les données précédentes (voir figure 1.9).

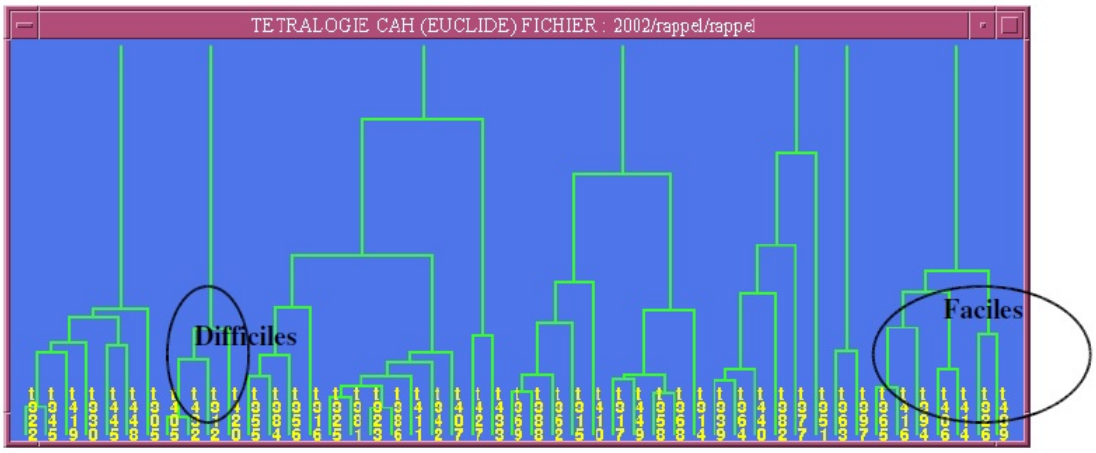

FIGURE 1.8 – Classification des requêtes de la tâche TREC 2002 Novelty basée sur la mesure Rappel (Chrisment 2005)

Une approche semblable a été utilisée dans (Dinçer 2007) qui a comparé les résultats obtenus par différents moteurs de recherche en utilisant une Analyse Factorielle des Correspondances (AFC).

Des travaux se sont également intéressés à étudier les paramètres les plus influents dans le processus de RI. Par exemple, (Compaoré 2011b) ont analysé les résultats de différentes variantes de moteurs de recherche (obtenues via la plateforme Terrier que nous présentons dans ce chapitre, section 1.3.1 selon la précision moyenne obtenue (mesure de performance AP - Average Precision). Les paramètres qu'ils ont étudiés sont présentés dans la figure 1.4. Durant l'analyse, les paramètres qe_md et qe_t ont été supprimés car redondants avec d'autres paramètres. Les auteurs ont analysé l'influence simultanée de chaque paramètre restant sur la performance. Pour cela, ils se sont appuyés sur CART (Classification And Regression Trees) (Breiman 1984). Les mesures de précision ont d'abord été traduites en valeurs qualitatives *Faible* (en dessous du premier quartile), *Moyenne* (entre le 2^{nd} et le $3^{ième}$ quartile) et *Forte* (au dessus du $3^{ième}$ quartile). Ce recodage a été réalisé d'une part pour les requêtes prises globalement et d'autre part pour les requêtes les plus faciles et pour les plus difficiles. La tableau 1.3 indique les valeurs de précision correspondantes.

La figure 1.10 montre les résultats pour l'ensemble des requêtes alors que la figure 1.11 (resp. 1.12) montre les résultats pour les requêtes les plus faciles (resp. les plus difficiles). Le

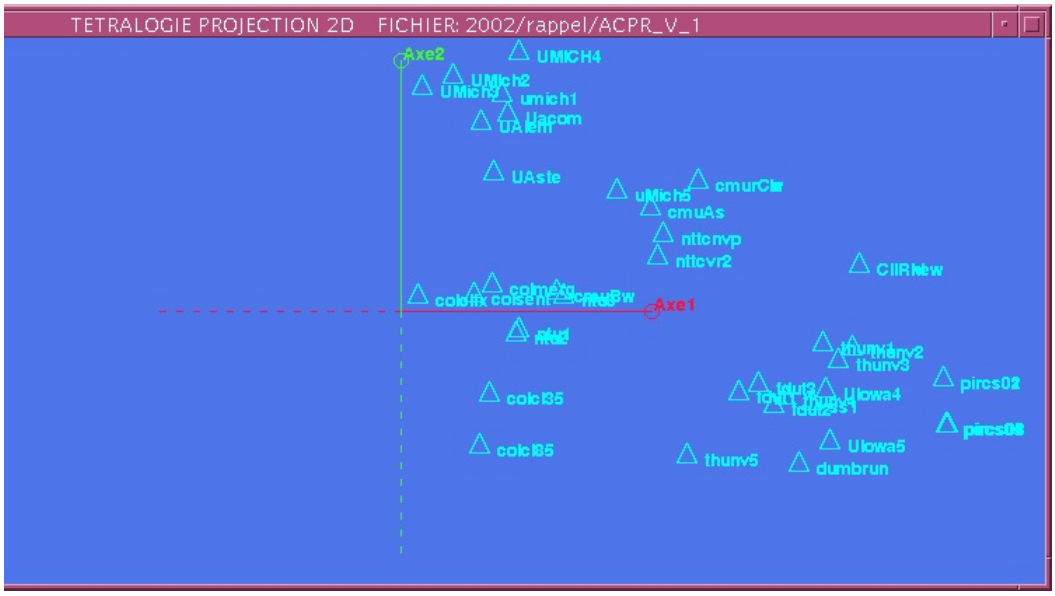

FIGURE 1.9 – Visualisation des variables (systèmes) après une ACP des résultats de la tâche TREC 2002 Novelty basée sur la mesure Rappel (Chrisment 2005)

paramètre qui est en haut de l'arbre est celui qui influence le plus les résultats (la performance) ; plus on descend dans l'arbre, moins le paramètre a d'influence dans les résultats.

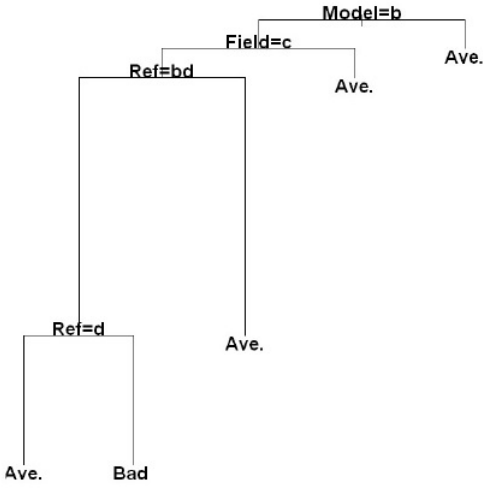

FIGURE 1.10 – Résultats de la méthode CART sur l'ensemble des requêtes (Compaoré 2011b)

Sans entrer dans les détails, un des résultats intéressants de cette étude est que la structure de l'arbre est différente lorsque l'on s'intéresse à l'ensemble des requêtes, aux requêtes les plus

Parameters	Meaning	Values
Top	Topic number	351, ..., 450
Field	Topic field	T; T+D; T+D+N
Bloc	Size of the indexing bloc	1, 5, 10
Idf	Inverse document frequency	FALSE, TRUE
Ref	Query reformulation	None, Bo1bfree, Bo2bfree, KLbfree
Model	Retrieval model	BB2c1, BM25b0.5, DFRBM25c1.0, IFB2c1.0, InexpB2c1.0, InexpC2c1.0, InL2c1.0, PL2c1.0, TFIDF
DocNb	Number of documents (reformulation)	0, 3, 10, 50, 100, 200
qe_md	Minimum number of documents in which the term should appear to be used in the query epension	0, 2
qe_t	Number of terms used in the query epension	0, 1

TABLE 1.2 – Paramètres et valeurs utilisés lors de la recherche (Compaoré 2011b)

Recodage	Global	Besoins faciles	Besoins difficiles
Faible	<0,057	<0,48	<0,005
Moyenne	>= 0,057 et <= 0,37	>= 0,48 et <= 0,69	>= 0,005 et <= 0,035
Forte	> 0,37	>0,69	> 0,035

TABLE 1.3 – Valeur de précision et recodage

Parameters	Meaning	Values
Top	Topic number	351, ..., 450
Field	Topic field	T; T+D; T+D+N
Bloc	Size of the indexing bloc	1, 5, 10
Idf	Inverse document frequency	FALSE, TRUE
Ref	Query reformulation	None, Bo1bfree, Bo2bfree, KLbfree
Model	Retrieval model	BB2c1, BM25b0.5, DFRBM25c1.0, IFB2c1.0, InexpB2c1.0, InexpC2c1.0, InL2c1.0, PL2c1.0, TFIDF
DocNb	Number of documents (reformulation)	0, 3, 10, 50, 100, 200
qe_md	Minimum number of documents in which the term should appear to be used in the query epension	0, 2
qe_t	Number of terms used in the query epension	0, 1

TABLE 1.4 – Paramètres et valeurs utilisés lors de la recherche (Compaoré 2011b)

faciles ou aux requêtes les plus difficiles.

(Ayter 2013) puis (Ayter 2014) ont complété les études précédentes en s'appuyant sur un jeu de données plus important. Ce jeu de données a été créé selon la méthodologie décrite dans le chapitre 3 de cette thèse. Ainsi, environ 80000 combinaisons différentes ont été ici étudiées

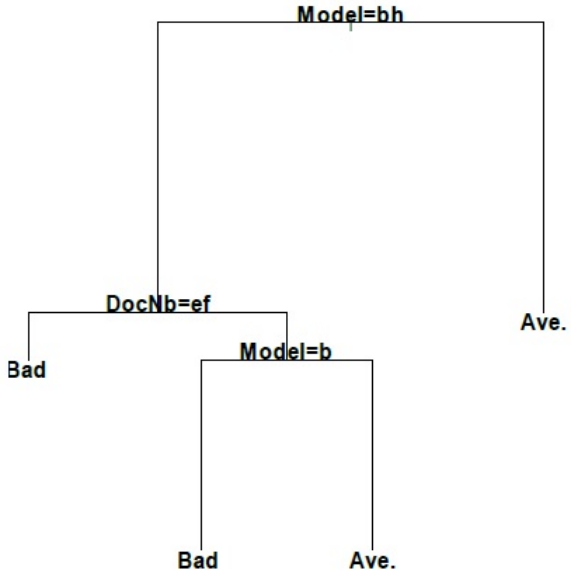

FIGURE 1.11 – Résultats de la méthode CART sur les requêtes faciles (Compaoré 2011b)

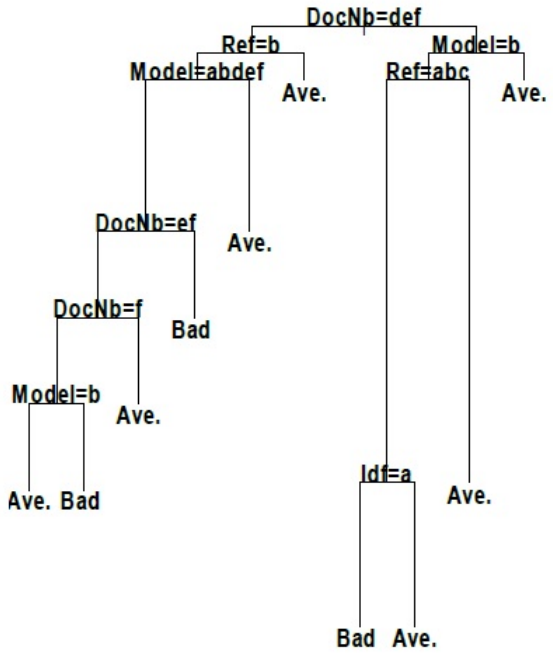

FIGURE 1.12 – Résultats de la méthode CART sur les requêtes difficiles (Compaoré 2011b)

correspondant aux 100 requêtes des collections TREC-7 et TREC-8. Les auteurs ont montré que pour les requêtes difficiles, les parties des besoins d'information utilisées (titre uniquement, ou titre et description par exemple) avaient une influence importante sur les résultats du moteur ; de même les algorithmes de reformulation utilisés avaient une grande importance.

Chapitre 2

Ressources pour l'analyse des paramétres des systémes de recherche d'information

Sommaire du chapitre

Dans ce chapitre nous présentons les ressources existantes ou que nous avons créées pour mener à bien nos recherches. Il s'agit d'une part de données qui ont été collectées directement des résultats obtenus par les participants à la tâche TREC *adhoc* (section 3.2) et d'autre part de données que nous avons nous même générer à partir de la plateforme Terrier sur cectte même tâche (section 3.3). Dans les deux cas, ces données s'appuient donc sur la tâche *adhoc* de TREC que nous présentons dans la section 3.1.

2.1 La tâche TREC Adhoc

Nous avons rapidement présenté les campagnes d'évaluation TREC dans le chapitre 1. Il s'agit en fait d'une conférence annuelle durant laquelle les participants présentent les résultats qu'ils ont obtenus en menant à bien des tâches de RI. De façon plus pratique, TREC met à disposition des participants des collections de test. Généralement, ces collections sont utilisées en suivant le guide de la tâche pour laquelle elles ont été conçues.

Une des premières tâches introduites à TREC est la tâche *adhoc* : dans cette tâche, les participants doivent fournir la liste des 1000 documents que leurs systèmes retrouvent pour 50 besoins d'information. Cette tâche a débuté en 1992 (TREC-1) et a continué sur une base annuelle sous la même forme jusqu'en 1999 (TREC-8).

Dans nos travaux, nous avons plus particulièrement travaillé avec les collections TREC-7 et TREC-8 que nous présentons ici.

Les collections de référence de TREC sont accessibles aux participants ; elles sont composées de trois éléments :

- un ensemble de documents ;
- un ensemble de besoins d'information ;
- les jugements de pertinence des documents pour chaque besoin d'information.

Le tableau 2.1 présente les caractéristiques des collections de référence TREC-7 et TREC-8.

Collection de référence	Documents	Besoins d'information
TREC-7 adhoc	Disks 4 and 5 excluding Congressional Record subcollection (env. 2 Go)	50 besoins (Topics 351-400)
TREC-8 adhoc	mêmes documents que TREC-7 adhoc	50 besoins (Topics 401-450)

TABLE 2.1 – Caractéristiques des collections TREC-7 et TREC-8

2.1.1 Ensembles de documents

Les ensembles de documents utilisés dans TREC-7 & TREC-8 sont des documents SGML, c'est à dire des documents dont la structure est marquée par des balises. En réalité, la structure est très légère et généralement omise dans les traitements ; seul le séparateur de documents est exploité.

Le même ensemble de documents est utilisé dans les collections TREC-7 et TREC-8.

2.1.2 Besoins d'information

Les besoins d'information de TREC (appelés *topics*) sont composés de plusieurs parties (voir figure 2.1). Outre son numéro, un besoin d'information contient :

– un titre : composé de quelques mots, il peut correspondre à ce que serait une requête soumise par un utilisateur à un moteur de recherche ;
– une description : composée d'une phrase ou deux, la description décrit le besoin d'information ;
– une narration : composée de quelques phrases, la narration précise des éléments permettant de savoir ce qui est attendu par l'utilisateur. Cette partie clarifie la pertinence ou la non pertinence des documents.

```
<num> 396
<title> sick building syndrome
<desc> Identify documents that discuss sick building syndrome
or building-related illnesses
<narr> A relevant document would contain any data that refers
to the sick building or building-related illnesses, including
illnesses caused by asbestos, air conditioning, pollution
controls. Work-related illnesses not caused by the building,
such as carpal tunnel syndrome, are not relevant
```

FIGURE 2.1 – Exemple de besoin d'information

Il y a au total 100 besoins d'information différents pour TREC-7 et TREC-8, 50 pour chacune des collections.

Un des paramètres de l'exécution d'une requête est la partie du besoin d'information utilisé. Lorsque seul le titre est utilisé, le besoin d'information simule une requête courte du type de celles qui peuvent être soumises à des moteurs de recherche réels. Dans la suite du document, lorsque nous parlerons de valeurs de paramètres, T fera référence à une requête générée à partir de la partie titre du besoin d'information, TD une requête générée à partir du titre et de la partie descriptive, et N sera utilisé pour référer à la partie narrative.

2.1.3 Jugements de pertinence

Les jugements de pertinence sont associés à une collection de documents pour les besoins d'information. Compte tenu du nombre de documents, il n'est pas possible de disposer de jugements exhaustifs de pertinence. Les jugements sont donc établis à partir des documents restitués par au moins un des systèmes participants de la campagne concernée.

Pour la tâche *adhoc*, ce sont les 1000 premiers documents des participants dont la pertinence est jugée (après suppression des doublons). Un extrait d'un fichier de jugements de pertinence (*qrels* en anglais pour *query relevance*) est présenté en figure 2.2.

Topic	it	Document	Pertinence
410	0	LA112490-0124	0
410	0	LA112989-0070	0
410	0	LA120490-0069	0
410	0	LA120490-0110	0
410	0	LA120589-0108	0
410	0	LA120690-0088	0
410	0	LA120889-0101	0
410	0	LA121589-0087	1
410	0	LA122689-0049	0
411	0	FBIS3-10248	0
411	0	FBIS3-10319	0
411	0	FBIS3-10334	0
411	0	FBIS3-1107	0
411	0	FBIS3-11453	0
411	0	FBIS3-11962	0
411	0	FBIS3-12674	0
411	0	FBIS3-12772	0
411	0	FBIS3-12871	0

FIGURE 2.2 – Extrait des jugements de pertinence

Les documents qui ne sont retrouvés par aucun un système participant ne sont pas évalués. Ils sont donc considérés comme non pertinents. Il en résulte que les SRI sont sous-évalués par les mesures de performance. Comme ce biais existe pour l'ensemble des SRI, il est peu dérangeant pour effectuer des comparaisons toutes choses égales par ailleurs (requêtes, documents et qrels).

Les collections TREC-7 et TREC-8 possèdent chacun leurs qrels.

2.2 Ressources directement issues des campagnes d'évaluation

2.2.1 Exécutions soumises par les participants

TREC met à disposition des participants les exécutions officielles des tâches passées. Dans le cadre de TREC *adhoc*, une exécution correspond à la liste des documents restitués par un système donné pour chacun des besoins d'information. Pour un besoin donné, le système ne doit restituer au maximum que 1000 documents (les documents suivants seront ignorés lors de l'évaluation) ; l'ordre des documents reflète leur pertinence supposée (pertinence décroissante). Une exécution est associée à une collection de reférence (ensemble de documents et de requêtes donnés).

La figure 2.3 donne un extrait d'une exécution (*run* en anglais).

```
401 Q0  FBIS4-18182 0   3.30433 ok8alx
401 Q0  FBIS3-18916 1   3.05985 ok8alx
401 Q0  FBIS3-18833 2   3.019   ok8alx
401 Q0  FT911-1796  3   2.90306 ok8alx
401 Q0  FBIS3-59042 4   2.89751 ok8alx
401 Q0  FBIS3-9104  5   2.81742 ok8alx
401 Q0  FBIS3-17077 6   2.75474 ok8alx
401 Q0  FBIS3-39117 7   2.74967 ok8alx
401 Q0  FBIS3-19881 8   2.70703 ok8alx
401 Q0  FBIS3-19290 9   2.7044  ok8alx
401 Q0  FBIS3-19459 10  2.68663 ok8alx
401 Q0  FBIS3-18684 11  2.68269 ok8alx
401 Q0  FT932-10861 12  2.66127 ok8alx
401 Q0  FT933-3792  13  2.66024 ok8alx
401 Q0  FBIS4-54799 14  2.64563 ok8alx
401 Q0  FBIS3-59212 15  2.63892 ok8alx
401 Q0  FBIS3-27159 16  2.61165 ok8alx
401 Q0  FBIS3-19645 17  2.6009  ok8alx
401 Q0  FBIS3-19893 18  2.5719  ok8alx
401 Q0  FBIS4-65928 19  2.56506 ok8alx
```

FIGURE 2.3 – Extrait de l'exécution officielle du système Okapi sur la tâche TREC-7

Le tableau 2.2 présente une synthèse des exécutions officielles de la tâche TREC-7 et TREC-8 *adhoc*.

Tâche et collection	Participants
TREC-7 adhoc	103
TREC-8 adhoc	129

TABLE 2.2 – Participants officiels aux campagnes TREC-7 et TREC-8 *adhoc*

2.2.2 Performance des systèmes participants

A partir des exécutions officielles, des qrels associés à la collection et du logiciel trec_eval (ce dernier a été présenté en section 1.3.2), il est possible de calculer pour chaque système les performances qu'il a obtenues en fonction de différentes mesures.

Ainsi, pour une collection de référence donnée, nous disposons pour chaque système participant et pour chaque besoin d'information de l'ensemble des mesures calculées par trec_eval pour les requêtes prises individuellement. Ces informations sont disponibles pour différentes collections de référence.

A partir de ces éléments, nous avons construit une matrice directement utilisable par les outils d'analyse exploratoire de données.

La figure 2.3 présente un extrait de la matrice obtenue à partir des exécutions officielles. La première colonne indique la campagne d'évaluation considérée, la seconde la tâche, la troisième

campagnes	taches	systemes	requetes	0.20R-prec	0.40R-prec	0.60R-prec
TREC1999	adhoc	1	401	0.016700	0.008300	0.011100
TREC1999	adhoc	1	402	0.000000	0.000000	0.000000
TREC1999	adhoc	1	403	0.000000	0.000000	0.000000
TREC1999	adhoc	1	404	0.000000	0.000000	0.000000
[...]						
TREC1999	adhoc	weaver2	447	0.750000	0.571400	0.600000
TREC1999	adhoc	weaver2	448	0.000000	0.000000	0.035700
TREC1999	adhoc	weaver2	449	0.000000	0.000000	0.000000
TREC1999	adhoc	weaver2	450	0.762700	0.686400	0.596600

TABLE 2.3 – Extrait de la matrice des performances des systèmes officiels participants à TREC

le nom du système participant officiellement, la quatrième le besoin d'information concerné, enfin, les autres colonnes présentent les valeurs obtenues sur les différentes mesures de performance calculées par trec_eval.

En termes statistiques, les individus correspondent à une ligne de cette matrice (un système, pour une collection et un besoin d'information donnés). Les valeurs correspondant à des mesures d'évaluation sont quant à elles les variables ; nous en dénombrons 130 au total. Dans ce travail de thèse, nous nous sommes appuyés sur les résultats officiels obtenus à TREC dans les chapitres 3 à 5.

D'autres travaux ont mis à profit ce type de données, nous en avons évoqué quelques uns au chapitre 1.

2.3 Ressources générées

Un inconvénient des exécutions fournies par TREC et donc par les participants est que ces exécutions sont peu documentées : il est ainsi difficile de connaitre avec précision comment elles ont été obtenues. Les articles de la conférence fournissent quelques éléments généraux, mais les valeurs de chaque paramètre sont généralement omis. Par ailleurs, même lorsqu'ils sont fournis, ces détails doivent être extraits des publications.

Il n'est donc pas possible de réaliser par exemple des études fines sur l'influence des différents paramètres à partir de ces exécutions.

Il nous a donc paru intéressant de générer différentes exécutions à partir de collections de référence et de différents modules de moteurs de RI. Un premier travail exploratoire avait été réalisé par J. Khalil (Khali 2013) sur la génération d'exécutions à l'aide de Terrier. Lors de notre première année de thèse, un des premiers travaux a consisté à systématiser cette génération.

2.3.1 Plateforme de calcul

Afin de générer un grand nombre de configurations du processus de RI, la plateforme CALcul en MIdi-Pyrénées (CALMIP) [1] a été utilisée. Généralement, cette infrastructure est

1. www.calmip.cict.fr/

utilisée pour effectuer du calcul parallèle. Comme notre problème était différent, CALMIP nous a fourni un outil de calcul à haute performance développé par Silicon Graphics (SGI).

Celui-ci permet de répliquer un script unique sur différents noeuds de calcul à raison d'un fichier d'entrée par noeud. Le script peut être ce que l'on veut (un script qui instancie Terrier par exemple) et le fichier d'entrée aussi (un fichier de configuration de Terrier dans notre cas). Quand le job d'un noeud est terminé, le script est relancé avec le fichier d'entrée suivant et ainsi de suite jusqu'au dernier.

Ainsi, il a été possible de générer les configurations Terrier. Le code développé pour créer toutes les combinaisons de paramètres est générique. Il a donc été utilisé pour d'autres expérimentations.

2.3.2 Paramètres d'exécution

Pour les deux collections de référence TREC-7 et TREC-8, nous avons ainsi fait varier des paramètres en nous appuyant sur la plateforme Terrier.

Les paramètres sont décrits les suivants :

- blocs_size : taille des blocs ;
- ignore_empty_documents : prise en compte des documents vides (FALSE/TRUE) ;
- Stemmer : raciniseur utilisé ;
- retrieving_model : modèle de pondération ;
- TrecQueryTags_process : champs à traiter ;
- ignore_low_idf_terms : prise en compte des termes qui ont un IDF bas (FALSE/TRUE) ;
- requete : numéro de la requête.

Une exécution est donc obtenue pour une combinaison de ces paramètres, appliquée à une collection de référence.

2.3.3 Exécutions obtenues

Concernant les collections de référence TREC-8, nous avons ainsi obtenu une variété d'exécutions. Le tableau 2.4 présente les différentes valeurs de paramètres (nombre de modalités et modalités) que nous avons utilisées. La dernière colonne du tableau indique le nombre d'individus par modalité.

Variable	Nombre de modalités	Modalités	Effectif par modalité
blocs_size	11	1, 2, 4, 6, 8, 10, 14, 22, 44, 100, 1000	205 800
ignore_empty_documents	2	FALSE, TRUE	1 131 900
stemmer	7	Crop, ESS, FSS, PS, TRPS, TRWPS, WPS	323 400
retrieving_model	21	BB2, DLH, InexpB2, PL2, BM25, DLH13, InexpC2, TFIDF, DFI0, DPH, InL2, XSqrAM, DFRBM25, HiemstraLM, JsKLs, DFRee, IFB2, LemurTFIDF, DirichletLM, InB2, LGD	107 800
TrecQueryTags_process	7	TITLE, DESC, NARR, TITLE/DESC, DESC/NARR, TITLE/NARR, TITLE/DESC/NARR	7323 400
ignore_low_idf_terms	2	FALSE, TRUE	1 131 900
requete	50	401, 402 ... 449, 450	45 276

TABLE 2.4 – Valeurs des paramètres utilisés pour générer les exécutions sur les collections TREC-7 et TREC-8

Une combinaison de paramètres permet d'obtenir une configuration de système.

2.3.4 Matrice de résultat

Comme dans le cas précédent, grâce aux qrels et au logiciel trec_eval, il est possible d'obtenir une matrice contenant pour chaque configuration de système et chaque requête les performances obtenues.

Le jeu de données ainsi obtenu à partir de TREC-8 comprend 2 263 800 lignes, c'est-à-dire 22 638 combinaisons de paramètres.

Ces données ont en particulier été utilisées dans les travaux de recherche que nous présentons au chapitre 5 de cette thèse.

Par ailleurs, ces travaux ont été exploitées pour d'autres travaux dans l'équipe. Par exemple, (Moulinou 2013) a réalisé une analyse exploratoire de ces données.

Elle a montré que la distribution des mesures de performance était disymétrique avec une sur-représentation des valeurs les plus faibles (voir figure 2.4).

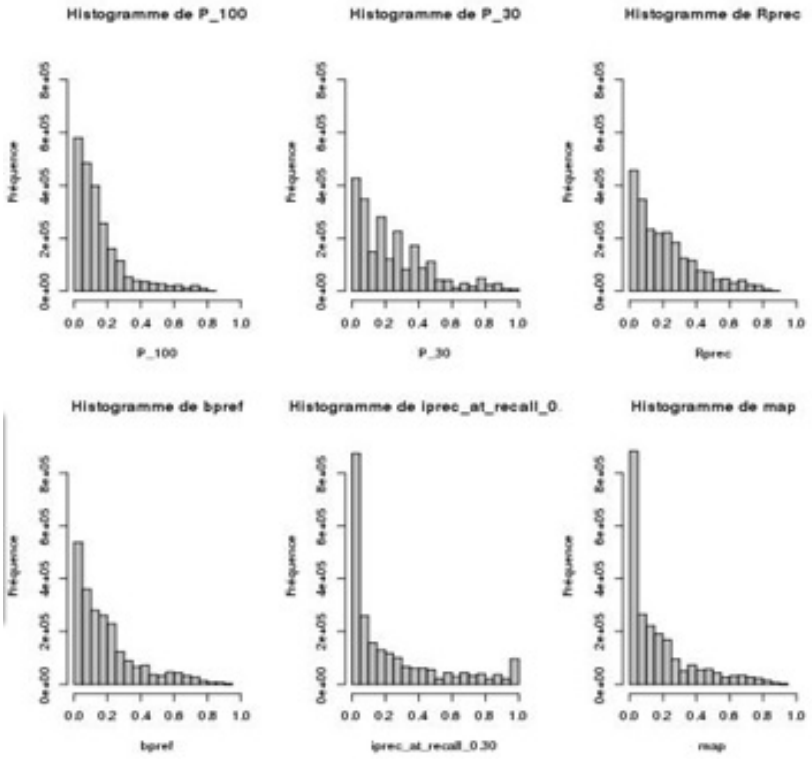

FIGURE 2.4 – Distribution de certaines mesures de performance relatives aux exécutions TREC-8

Ayter et Desclaux (Ayter 2013) puis Ayter *et al.* (Ayter 2014) ont également exploité les données que nous avons générées. Dans ce cas, l'ensemble des configurations générées a été étudié. La matrice correspondante comporte 8 159 800 lignes. L'analyse effectuée avait pour objectif d'analyser l'influence des différents paramètres dans les résultats. Les résultats principaux ont été présentés dans le chapitre 2 de cette thèse.

Chapitre 3

Meilleurs systèmes : définition et sélection

Sommaire du chapitre

3.1 Introduction

Les campagnes d'évaluation ont montré la variabilité des résultats obtenus par différents systèmes lorsqu'ils sont appliqués sur les mêmes collections d'évaluation. Cette variabilité a surtout été observée au niveau des requêtes : certains systèmes répondent bien pour une requête et mal pour une autre alors que d'autres systèmes ont un comportement inverse.

Cependant, à part le travail mené lors du workshop RIA (Harman 2004), (Harman 2009), peu de travaux ont étudié les résultats des différents participants qui constituent pourtant une source d'information que nous pensons importante. Ce chapitre s'appuie sur ces résultats afin d'étudier les performances des systèmes et surtout les contextes dans lesquels tel ou tel système devrait être utilisé.

Dans le travail présenté ici, nous faisons l'hypothèse que des besoins différents doivent être traités par des systèmes de RI différents, afin de tirer avantage de leur variabilité. Cela nécessite de définir ce qu'est un *meilleur* système. Nous faisons également l'hypothèse que les meilleurs systèmes pour traiter les besoins les plus faciles sont différents des systèmes plus aptes à traiter les besoins les plus difficiles. L'idée sous-jascente est qu'il est possible de sélectionner les meilleurs systèmes selon des classes de difficulté des besoins.

Ainsi, le travail présenté dans ce chapitre est double. Dans un premier temps, nous proposons de définir la notion de *meilleur* système à la fois en termes de performance et de robustesse, d'autre part nous montrons que la sélection d'un système particulier en fonction de groupes prédéfinis de requêtes basé sur les performances obtenus par les sysèmes peut permettre l'amélioration des performances globales des systèmes. Ce travail s'est appuyé sur les résultats obtenus par les systèmes participants à la tâche *adhoc* de TREC-8.

Le chapitre est organisé comme suit. Dans un premier temps, nous rappelons le contexte dans lequel les systèmes étudiés ont été évalués et nous expliquons la construction du jeu de données utilisé pour l'analyse (section 3.2). Dans la section 3.3, nous présentons deux méthodologies de classement des SRI, l'une basée sur la notion de mesure moyenne (3.3.3) et la seconde sur celle de rang moyen (3.3.4). Dans la section 3.4, nous présentons l'adaptation du choix du meilleur système au contexte de la requête. Ce contexte s'appuie sur la notion de difficulté des requêtes. Ainsi, la section 3.4.1 propose une méthode de groupement des besoins en information selon leur difficulté et la section 3.4.2 présente une méthodologie pour le sélectionner.

Enfin, la section 3.5 montre les résultats expérimentaux associés. Nous concluons par des pistes d'exploitation de ces résultats.

3.2 Contexte et données

La collection que nous avons utilisée est issue de TREC adhoc 8 qui est composée de 528 155 documents et 50 besoins en information. Chaque système participant a produit une liste ordonnée de 1000 documents pour chaque besoin en information. Ces listes ont été évaluées selon la méthodologie de vote expliquée au chapitre 1.1. Ainsi à chaque système participant est associé un ensemble de valeurs de mesures de performances calculées par *trec_eval*. Un total de 129 systèmes ont été utilisés par des participants cette année-là (nous utiliserons indifféremment le terme de *système* ou d'*exécution* dans la suite) ; dans certains cas, il s'agit de différentes versions d'un même moteur.

Parmi les 130 mesures de performances calculées par trec_eval pour chacun des 50 besoins

et chacune des exécutions, nous avons retenu uniquement les 118 mesures prenant leurs valeurs dans [0 ; 1]. Par ailleurs, pour mener les analyses, nous avons besoin d'un format des données adapté à l'utilisation de méthodes mathématiques. Nous avons donc structuré les données sous forme de matrice. Ainsi, la matrice des données initiales est un cube de données à trois dimensions.

La première dimension est constituée des 50 besoins en information ; la seconde dimension représente les 129 systèmes et la troisième dimension correspond aux 118 mesures de performances. Chaque cellule du cube contient la valeur de la mesure de performance pour l'exécution d'un système de RI sur un besoin en information.

3.3 Méthodologie de détection des meilleurs systèmes

Dans cette section, nous proposons deux définitions de la notion de meilleurs systèmes : le plus performant et le plus robuste. Nous proposons également la méthodologie pour les sélectionner parmi un ensemble d'exécutions. Ces méthodes sont générales et ne dépendent ni des besoins en information, ni des systèmes de RI considérés. Elles reposent toutes les deux sur le classement des systèmes dans leur traitement des requêtes qui peuvent être obtenus de différentes façons, en considérant les notions de rang et de mesure moyenne.

3.3.1 Vue générale de l'approche

La figure 3.1 donne une vue générale de l'approche permettant de définir le rang obtenu par un système parmi un ensemble de systèmes.

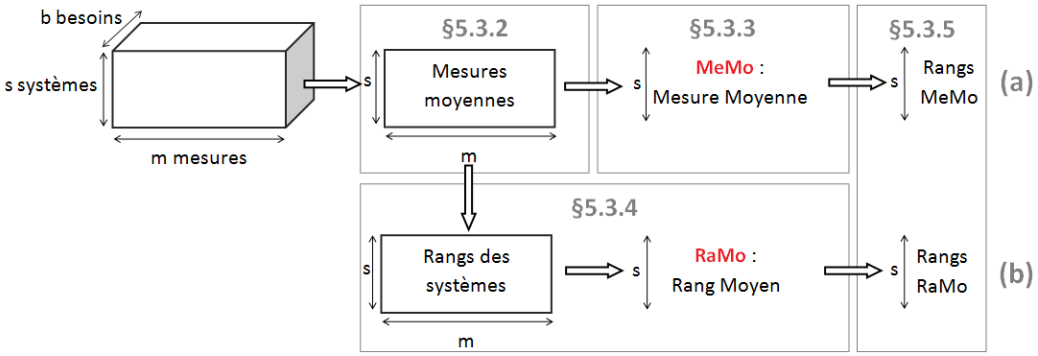

FIGURE 3.1 – Transformation des mesures en rangs de la mesure moyenne

La première étape consiste à calculer pour chaque système, sa performance moyenne sur les besoins en information pour chacune des mesures. Ensuite une mesure unique est calculée pour chaque système : soit la Mesure Moyenne (MeMo) directement depuis la matrice de mesures moyennes (figure 3.1 (a)), soit le Rang Moyen (RaMo) (figure 3.1 (b)). Enfin, les systèmes sont ordonnés selon la mesure choisie (MeMo ou RaMo).

3.3.2 Calcul des performances moyennes d'un système

Pour chacune des méthodes de sélection, nous commençons par créer la matrice des performances moyennes. Cette matrice est composée de s lignes (pour les systèmes) et de m colonnes (pour les mesures). Chaque cellule de la matrice est la performance moyenne $\bar{M}^{(S)}$ de chaque système S pour chaque mesure M donnée par :

Équation (Performance moyenne)**.**

$$\bar{M}^{(S)} = \frac{1}{b} \sum_{B=1}^{b} m_B^{(S)} \tag{3.1}$$

où $m_B^{(S)}$ est la valeur de la mesure attribuée pour la performance du système S sur le besoin B.

3.3.3 Mesure Moyenne (MeMo)

La mesure moyenne "MeMo" d'un système consiste à calculer la moyenne générale de chaque système sur la matrice des mesures moyennes (équation 3.2).

Équation (Mesure moyenne des systèmes)**.**

$$\text{s systèmes} \left\{ \overbrace{\begin{pmatrix} \bar{M}_1^1 & \cdots & \bar{M}_m^1 \\ \vdots & \ddots & \vdots \\ \bar{M}_1^s & \cdots & \bar{M}_m^s \end{pmatrix}}^{\text{m mesures}} \right. \Longrightarrow \overbrace{\begin{pmatrix} \bar{\bar{M}}^1 \\ \vdots \\ \bar{\bar{M}}^s \end{pmatrix}}^{\text{MeMo}} \Longrightarrow \overbrace{\begin{pmatrix} R^1 \\ \vdots \\ R^s \end{pmatrix}}^{\text{Rang MeMo}} \tag{3.2}$$

avec :
- $\bar{M}_j^i$ la moyenne de la mesure j pour le système i
- R_i le rang du système i

3.3.4 Rang Moyen (RaMo)

Nous considérons le calcul pour une mesure de performance fixée. Les systèmes de RI sont ordonnés par ordre décroissant selon la valeur obtenue pour la mesure considérée. Le système ayant la plus forte valeur se retrouve en tête de liste et le rang 1 lui est attribué. Le système suivant de la liste est le second meilleur système et le rang 2 lui est attribué et ainsi de suite jusqu'au dernier système qui obtient le rang maximum. Si plusieurs systèmes obtiennent la même valeur de mesure, ils sont ex-æquo et obtiennent un rang équivalent à la moyenne des rangs attribués s'il n'y avait pas d'ex-æquo. L'algorithme d'attribution des rangs reprend son cours normal après le dernier ex-æquo.

Équation (Rang moyen des systèmes)**.**

$$\text{s}\left\{\overbrace{\begin{pmatrix}\bar{M}_1^1 & \cdots & \bar{M}_m^1\\ \vdots & \ddots & \vdots\\ \bar{M}_1^s & \cdots & \bar{M}_m^s\end{pmatrix}}^{\text{m mesures}} \Longrightarrow \begin{pmatrix}R_1^1 & \cdots & R_m^1\\ \vdots & \ddots & \vdots\\ R_1^s & \cdots & R_m^s\end{pmatrix} \Longrightarrow \overbrace{\begin{pmatrix}\bar{R}_1\\ \vdots\\ \bar{R}_s\end{pmatrix}}^{\text{RaMo}} \Longrightarrow \overbrace{\begin{pmatrix}R_1\\ \vdots\\ R_s\end{pmatrix}}^{\text{Rang RaMo}} \right. \tag{3.3}$$

3.3.5 Ordre des systèmes

Calcul des rangs

Nous avons retenu deux façons d'ordonner les exécutions dans leur traitement d'un ensemble de requêtes : le rang de la mesure moyenne (MeMo) et le rang sur les rangs moyens (RaMo).

Classement sur le rang de la mesure moyenne (MeMo) Cette méthode consiste à calculer la moyenne générale de chaque système sur la matrice des mesures moyennes (équation 3.2). Ensuite, les systèmes de RI sont ordonnés selon cette valeur pour obtenir les rangs de la mesure moyenne "MeMo".

Classement sur le rang des rangs moyens (RaMo) Cette seconde méthode consiste à calculer les rangs des systèmes pour chacune des mesures à partir de la matrice des mesures moyennes (équation 3.3). Ainsi pour chaque variable de mesure, nous obtenons les systèmes ordonnés par rang. Ensuite, la moyenne des rangs obtenus à l'étape précédente permet d'ordonner les systèmes et de leur attribuer le rang de leur rang moyen "RaMo".

3.3.6 Réduction des données : comparaison des rangs RaMo et des rangs MeMo

Les rangs RaMo et MeMo ont le même objectif : celui de représenter l'ensemble des données de façon réduite, l'ensemble des mesures de performance étant ici ramené à une seule variable.

D'autres travaux se sont intéressés à réduire le nombre de mesures nécessaires à la comparaison de SRI. Ainsi, Baccini (Baccini 2011) montre que les 130 mesures de performances issues de trec_eval sont redondantes et les classe en 6 groupes. Les auteurs montrent également qu'un sous-ensemble composé d'une mesure représentative de chaque groupe est suffisant. Ainsi, les auteurs montrent que comparer des SRI soit à partir de l'ensemble des mesures de trec_eval, soit des 6 mesures retenues est équivalent. Les 6 mesures retenues peuvent par exemple être : la "mean average precision" (MAP) ; la précision à 30 (P30) ; la précision à 100 (P100) ; le rappel exact (exact_recall) ; la préférence binaire (bpref_retall) ; le rappel à 30 (R30). Baccini et al. (2011) ont montré que la corrélation entre les mesures moyennes des systèmes de RI en utilisant les données complètes et les mesures moyennes en utilisant les données réduites est supérieure à 0.99, ce qui confirme le choix de ces 6 mesures.

En partant de ce résultat, nous nous sommes intéressés à voir si les mesures MeMo et RaMo permettaient d'obtenir des résultats comparables lorsque l'ensemble des mesures de performance est utilisé et lorsque seulement 6 mesures sont utilisées.

Nous cherchons à comparer les deux méthodes de classement sur les données afin de déterminer si les deux nécessitent une étude plus approfondie. Pour chacune des méthodes MeMo et

RaMo, nous confrontons les rangs des systèmes obtenus sur les données réduites (6 mesures) avec ceux obtenus sur les données complètes (ensemble des mesures). Cela permet de déterminer si les méthodes sont sensibles aux mesures utilisées et à leur nombre. La figure 3.2 présente les résultats obtenus pour les deux méthodes de calcul de classement des systèmes. Nous observons peu d'écart entre les rangs obtenus avec les données complètes d'une part et ceux obtenus avec les données réduites d'autre part. Nous observons dans les deux cas que les résultats sont très corrélés. Toutefois, en utilisant la méthode MeMo (figure 3.2 (a)), nous obtenons un classement un peu plus homogène que par la méthode RaMo : les points sont plus proches de la droite tracée (figure 3.2 (b)). Nous supposons que la présence plus importante d'exæquo avec la méthode RaMo conduit à une instabilité dans les classements des systèmes. Cette méthode ne sera donc pas prise en compte dans la suite de l'analyse et la méthode de classement MeMo sera privilégiée dans la suite de cette étude.

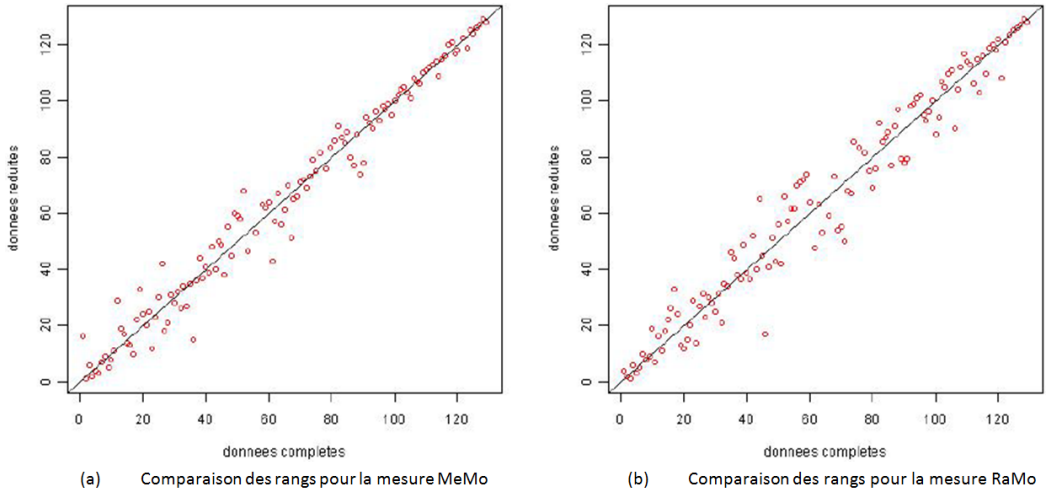

FIGURE 3.2 – Comparaison de la stabilité des méthodes de classement selon le nombre de mesures considérées.

Nous conserverons donc seulement six mesures pour la suite de l'analyse.

Les données réduites et la méthode de classement basée sur la mesure moyenne seront utilisées dans la suite de ce chapitre afin de déterminer quels sont les meilleurs systèmes selon la difficulté des besoins à traiter. Nous opposerons les systèmes obtenus par les rangs MeMo avec les systèmes obtenus en travaillant directement avec la MeMo.

3.4 Définition des meilleurs systèmes de recherche d'information par groupe de besoins en information

Les systèmes de RI ne traitent pas les besoins en information de la même manière : là où certains systèmes réussissent, d'autres échouent, et inversement (Harman 2009). Nous faisons l'hypothèse que les systèmes de RI sont plus ou moins performants selon la difficulté des besoins traités et cherchons à vérifier cette hypothèse.

3.4.1 Regroupement des besoins en information

Afin de déterminer différentes classes de besoins en information selon un critère de difficulté et afin de déterminer si elles nécessitent d'être traitées par des systèmes de RI différents, nous procédons à un regroupement des besoins en classes.

À partir des données présentées en 3.2, il est possible de calculer une mesure moyenne représentative de la performance du système selon l'ensemble de ces performances sur un besoin donné. Ensuite, pour chaque besoin, le rang MeMo de chaque système est calculé comme le montre la figure 3.3.

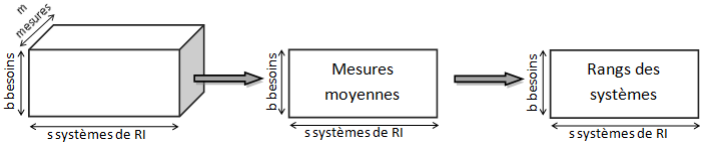

FIGURE 3.3 – Transformation des mesures en rangs de la mesure moyenne pour les besoins

Les requêtes sont ensuite classées selon la méthode de Classification Ascendante Hierarchique (CAH), *cf.* chapitre 1. La figure 3.4 montre le dendogramme qui en résulte. Le choix de 4 classes de besoin est raisonnable selon le critère usuel de rupture dans les éboulis de l'inertie (Lebart 2006).

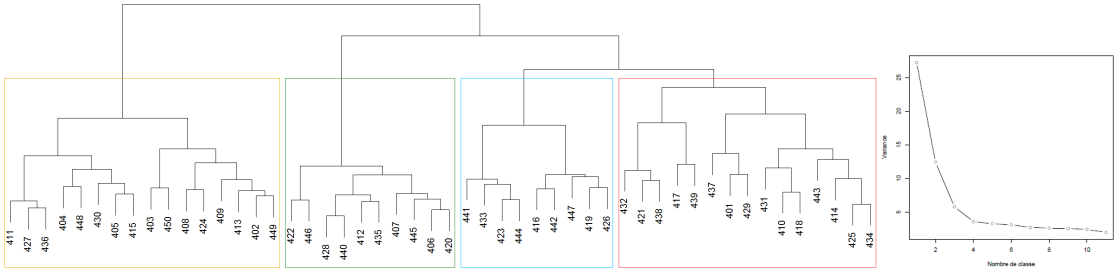

FIGURE 3.4 – Dendogramme et éboulis résultant de la classification ascendante hiérarchique

Ici, le nombre de classes obtenues par la CAH est dépendant d'un choix subjectif que nous avons fait. Afin de diminuer le biais introduit par ce choix, une méthode de ré-allocation dynamique non-supervisée est appliquée sur les classes obtenues. Les centres de gravité des classes ont été utilisés comme point de départ de l'algorithme des K-moyennes (Jain 1999). Cette méthode consiste à agréger chaque élément un à un au groupe dont la moyenne est la plus proche. La figure 3.5 donne une signification aux groupes de besoins en information créés.

Ainsi la figure 3.5 montre les boîtes à moustache de chaque requête des *cf.* MeMo (pour les explications des boites à moustache, voir 1).

Sur cette figure, chaque groupe de besoins est représenté par une couleur. Le groupe de besoins en information pour lesquels 75% des systèmes ont une MeMo supérieure à 0,5 est donc qualifié de groupe facile à traiter (en bleu). De la même manière, le groupe de besoins pour lesquels les systèmes obtiennent un premier quartile supérieur à 0,3 et un troisième quartile

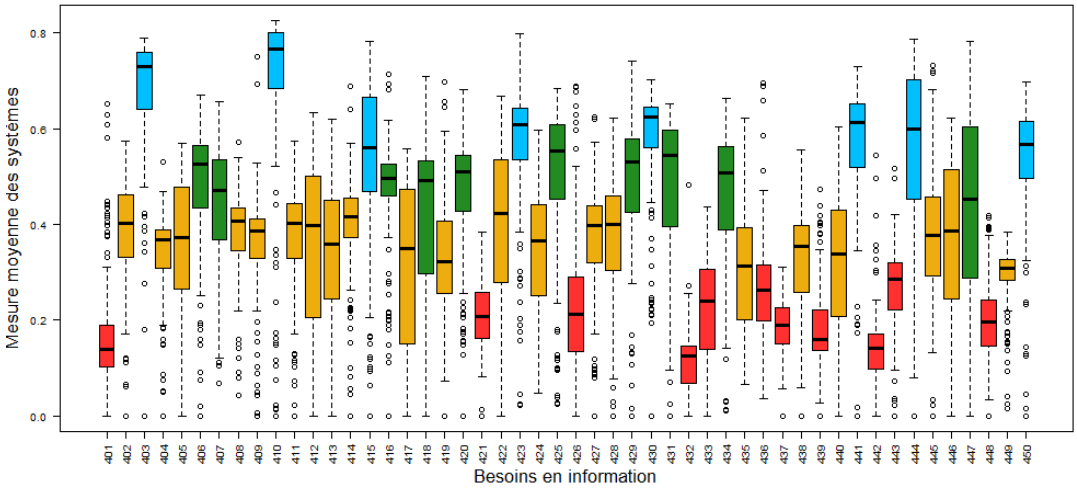

FIGURE 3.5 – Caractérisation des classes de besoins en information

inférieur à 0,6 est qualifié de groupe moyen (en vert) ; le troisième groupe est qualifié de groupe difficile à traiter (en jaune) avec une médiane des MeMo des systèmes comprise entre 0,3 et 0,4. Enfin, le groupe de besoins pour lesquels 75% des systèmes obtiennent une MeMo inférieure à 0,3 est qualifié de groupe très difficile (en rouge).

3.4.2 Performance et robustesse des systèmes de recherche d'information

La première question qui se pose est quelle définition donner au meilleur système pour un groupe de besoins donné. Nous avons retenu deux approches en fonction de l'objectif visé. Le meilleur système peut présenter le meilleur classement possible pour la grande majorité des besoins en information, et avoir un très mauvais classement pour quelques uns. Il s'agirait d'un système de RI qui donne les meilleurs résultats mais pas en toutes circonstances. Nous appelons ce système de RI le système le plus *performant*.

Alternativement, le meilleur système peut être un système avec un bon classement (pas nécessairement le meilleur) pour l'ensemble des besoins en information considérés. Il s'agit d'un système robuste en permanence mais ne donnant pas les meilleurs résultats possibles. Ce système de RI est qualifié de système le plus *robuste*. Idéalement, pour un groupe de besoins en information donné, le meilleur système de RI aura les deux propriétés susmentionnées : les meilleurs performances et cela pour l'ensemble des besoins en information. Dans les faits, un tel système n'existe pas forcément et les meilleurs systèmes étudiés dans nos travaux connaissent des failles pour au moins un besoin. C'est pourquoi dans la suite de l'analyse, nous essayerons de trouver le système le plus performant d'une part, et le système le plus robuste d'autre part ; et ce pour chacun des groupes de besoins introduits en section 3.4.1.

3.5 Analyse

3.5.1 Détection des meilleurs systèmes par groupe de besoins

Nous définissons deux règles qui correspondent aux données que nous traitons afin de déterminer les candidats aux deux types de *meilleurs* systèmes pour chacun des groupes de besoins considérés :

- candidats au système le plus *performant* : ces systèmes sont parmi les dix premiers pour au moins un besoin du groupe. Ces systèmes doivent également obtenir un rang inférieur à 20 pour au moins la moitié des besoins (i.e. la médiane de leurs rangs doit être inférieure à 20) ; ainsi nous sommes certains que les systèmes de RI candidats sont très performants pour au moins la moitié des besoins du groupe.
- candidats au système le plus *robuste* : ces systèmes ont un classement maximum inférieur à 80. De tels systèmes seront donc dans les deux premiers meilleurs tiers des systèmes de RI. La valeur 80 est fixée arbitrairement car suffisamment restrictive : en fixant ce seuil à un rang inférieur, aucun système de RI ne serait candidat au système le plus robuste pour certains groupes de besoins. Il est clair que les règles présentées ici peuvent mener à plusieurs systèmes candidats concourants pour être le plus performant et/ou le plus robuste. Nous verrons par la suite que le choix d'un système unique parmi les candidats est parfois possible et parfois non.

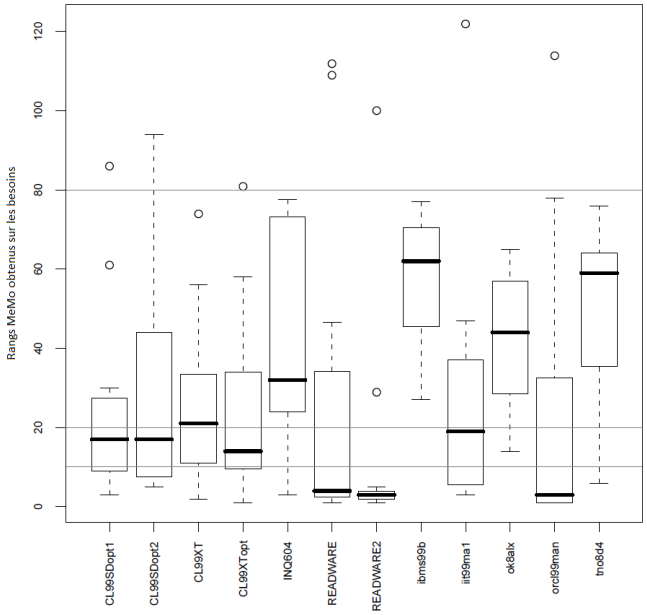

FIGURE 3.6 – Rangs MeMo : candidats aux meilleurs systèmes - groupe des requêtes très difficiles

La figure 3.6 présente les 12 systèmes candidats pour le groupe des besoins que nous avons définis comme très difficiles en section 3.4.1. Un système candidat est un système qui corres-

pond à au moins une des définitions présentées en début de cette section. Les différents seuils à respecter pour qu'un système soit candidat sont représentés par les lignes horizontales en pointillés. Le signe "+" (respectivement "o") sous le nom des systèmes indique le(s) système(s) finalement retenu(s) comme le(s) plus performant(s) (resp. robuste(s)).

D'après cette figure, sept systèmes de notre échantillon sont candidats au système de RI le plus performant. Cependant, le système READWARE2 (signe "+") est parmi les 10 meilleurs systèmes de RI pour tous les besoins classés très difficiles sauf deux. De plus, aucun autre candidat au système le plus performant pour ce groupe de besoins n'a de telles performances. READWARE2 est donc désigné comme le système le plus performant pour les besoins très difficiles.

Ici, cinq systèmes de RI peuvent correspondre au système le plus robuste car ils n'ont que des rangs inférieurs à 80 : CL99XT, INQ604, ibms99b, ok8alx et tno8d4. Cependant, CL99XT a 75% de ses rangs (Q3) inférieur à 40 ce qui signifie qu'il est classé 75% du temps dans le premier tiers des systèmes pour les besoins très difficiles. En conséquence, le système CL99XT est choisi comme système le plus robuste pour ce groupe de besoins.

Nous pouvons procéder à des analyses similaires pour les différents groupes de besoins obtenus en 3.4.1.

Ainsi, la figure 3.7 présente les 14 systèmes candidats pour les requêtes faciles. Le système le plus performant au sens de notre définition est le système MITSLStdn. Deux systèmes ont ici la médiane de leurs rangs inférieure à 20, mais l'autre système, fub99td obtient des rangs bien moins bons pour beaucoup de requêtes. Dans le mesure où il s'agit du groupe des requêtes faciles, il y a beaucoup de systèmes qui sont candidats pour être le plus robuste. MITSLStd est certainement le meilleur choix compte tenu du rang maximum qu'il obtient et de ses valeurs base de rang quelle que soit la requête.

Les figures 3.8 et 3.9 présentent les résultats obtenus pour les requêtes intermédiaires.

Le tableau 3.1 présente les résultats de ces analyses et la sélection des systèmes les plus performants et les plus robustes en fonction de la difficulté du besoin en information.

Groupes de besoins	**Système le plus**	
	perfomant	**robuste**
Faciles	MITSLStdn	MITSLStd
Moyens	CL99XT	READWARE2
Difficiles	READWARE2	ibms99a
Très Difficiles	READWARE2	CL99XT

TABLE 3.1 – Résultats de la sélection des rangs MeMo par groupe de besoins

Le tableau 3.1 montre que la méthode de sélection basée sur les rangs MeMo permet de choisir le meilleur système à utiliser selon deux critères. Le premier critère est la difficulté du besoin en information à traiter. Le second concerne les attentes de celui qui exprime ce besoin : attend-il les meilleurs résultats en acceptant quelques échecs ? ou préfère-t-il obtenir des résultats un peu moins bons mais qui le restent toujours ? Ainsi, en fonction de la difficulté des requêtes, les systèmes sélectionnés ne sont pas les mêmes ; de même, en fonction de l'objectif fixé (performance, robustesse), le système sélectionné n'est pas non plus toujours le même.

Cependant, pour un groupe de besoins fixé, le choix des "meilleurs" systèmes de RI n'est pas toujours trivial et nécessitent une étude plus approfondie sur les rangs des systèmes candidats.

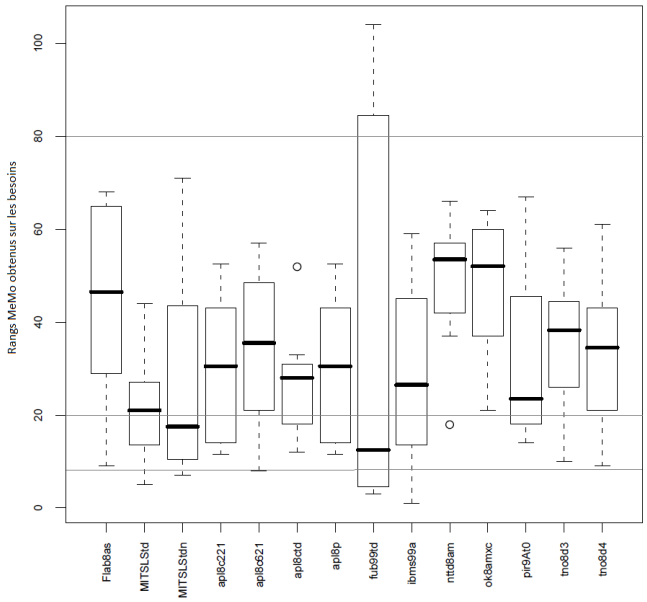

FIGURE 3.7 – Rangs MeMo : candidats aux meilleurs systèmes - groupe des requêtes faciles

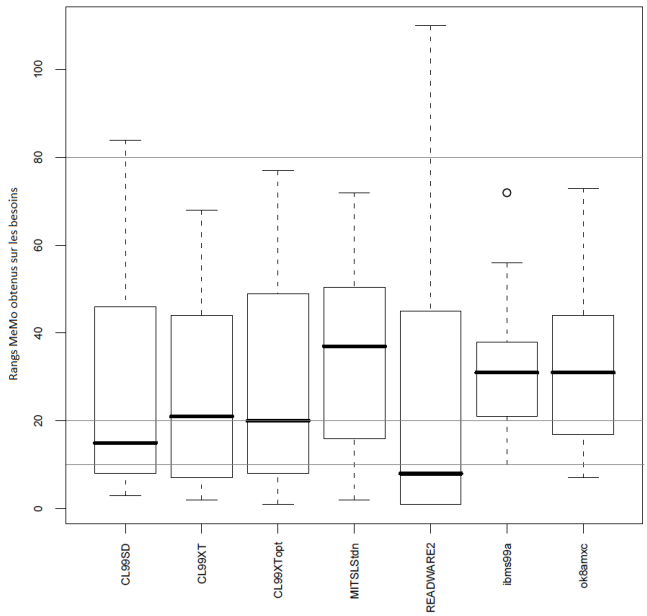

FIGURE 3.8 – Rangs MeMo : candidats aux meilleurs systèmes - groupe des requêtes difficiles

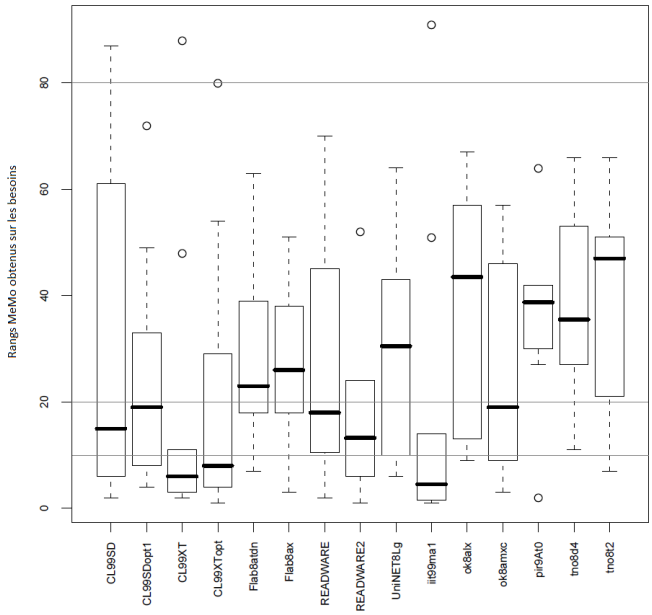

FIGURE 3.9 – Rangs MeMo : candidats aux meilleurs systèmes - groupe des requêtes moyennes

Nous supposons que cela est dû aux ex-æquos possibles lors de la transformation des MeMo en rangs. Nous procédons donc à une étude similaire des meilleurs systèmes de RI par groupe de besoins en information, cette fois basée directement sur les MeMo.

3.5.2 Détection des meilleurs systèmes par groupe de besoins par la mesure moyenne

Les critères de sélection du système le plus performant et du système le plus robuste sont analogues à celles présentées précédemment. L'étude est adaptée à la mesure moyenne. Dans l'étude précédente, une valeur de rang élevée était mauvaise puisque cela signifie que le système était classé parmi les moins bons. Désormais nous travaillons sur des mesures de performance : une valeur élevée signifie que le système fait partie des meilleurs.

La figure 3.10 présente les systèmes de RI candidats selon les MeMo pour le groupe de besoins très difficiles.

Les médianes respectives du système "READWARE2" et du système "orcl99man" sont nettement supérieures aux médianes des autres systèmes. La position de la médiane de "READWARE2" proche du troisième quartile montre une forte densité des MeMo obtenues dans l'intervalle [0,45 ; 0,50]. Cela nous pousse à le choisir comme unique système le plus performant pour ce groupe. Étant donné que les moins bonnes valeurs de MeMo de READWARE2 sont supérieures aux MeMo de tous les autres candidats, le système "READWARE2" est également choisi ici comme système le plus robuste. Le tableau 3.2 présente l'ensemble des systèmes retenus par groupe de besoins en information selon les MeMo.

Les figures suivantes montrent les résultats en fonction des groupes de système :

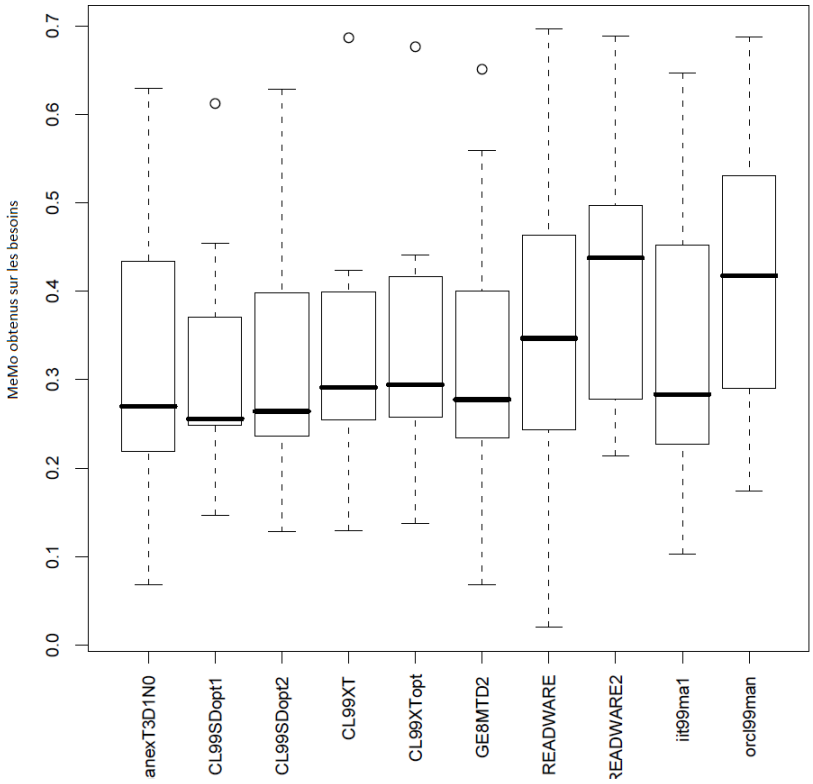

FIGURE 3.10 – MeMo : candidats aux meilleurs systèmes - groupe des requêtes très difficiles

– le groupe des requêtes faciles est présentés en figure 3.11 ;
– le groupe des requêtes moyennes est présentés en figure 3.12 ;
– le groupe des requêtes difficiles est présentés en figure 3.13.

Enfin, le tableau 3.2 présente le résultat de la sélection des systèmes les plus performants et les plus robustes en fonction de la difficulté des requêtes selon leur valeur de MeMo.

Groupes de besoins	**Système(s) retenu(s) comme le(s) plus performant(s)**	**Système(s) retenu(s) comme le(s) plus robuste(s)**
Faciles	—	—
Moyens	ii99mal	Flab8x, ok8amxc, CL99SDopt1
Difficiles	READWARE2	ibms99a
Très Difficiles	READWARE2	READWARE2

TABLE 3.2 – Résultats de la sélection des MeMo par groupe de besoins

Pour les groupes de besoins faciles et moyens, le choix d'un unique meilleur système est une tâche difficile voire impossible. En effet, ces groupes de besoins sont définis tels que tous les systèmes réussissent à bien les traiter : les performances obtenues par les systèmes sur le groupe facile sont très bonnes et donc très peu discriminantes pour choisir les "meilleurs" systèmes. Ceci n'est pas un problème, car comme tous les systèmes sont performants sur ces besoins, améliorer

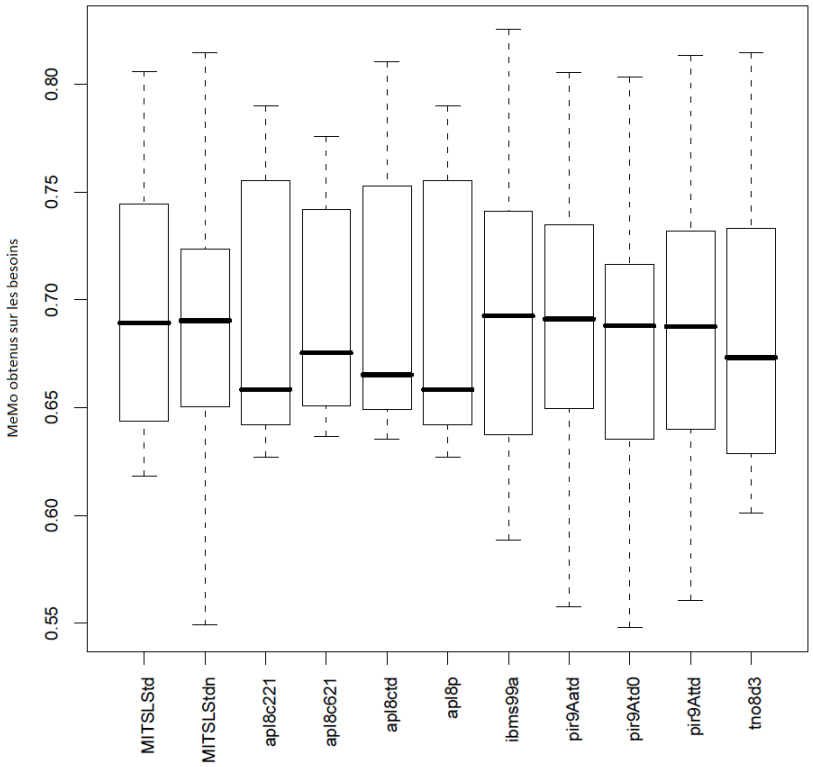

FIGURE 3.11 – MeMo : candidats aux meilleurs systèmes - groupe des requêtes faciles

les résultats est une tâche compliquée. De plus, améliorer un processus déjà très robuste et performant n'est pas nécessaire.

3.5.3 Comparaison des systèmes retenus par les deux approches

Le tableau 3.3 montre que les deux méthodes de sélection produisent des résultats similaires pour les besoins en information les plus difficiles (groupes difficiles et très difficiles) et de plus larges variations dans les choix pour les besoins moins difficiles (classés moyens et faciles). Nous remarquons également que la méthode de sélection basée sur les rangs a tendance à gommer les faibles écarts entre différents systèmes. Cela présente l'avantage de proposer un choix unique de "meilleur" système lorsque plusieurs obtiennent des performances proches les unes des autres. Les besoins les plus difficiles sont aussi les plus discriminants pour les systèmes, c'est à dire qu'ils permettent de repérer plus facilement les meilleurs systèmes de RI et donc de les sélectionner. Nous avons vu que la méthode de sélection des systèmes basée sur le rang MeMo permet toujours de faire le choix d'un unique système au contraire de la sélection directement basée sur les MeMo. Finalement, la méthode de sélection basée sur les rangs MeMo donnent des résultats similaires à la sélection basée sur les MeMo pour les besoins les plus difficiles. Elle permet aussi d'opérer une sélection unique pour les besoins les moins difficiles donc la sélection sur les rangs MeMo devrait être préférée. Afin de valider ces observations, nous procédons à

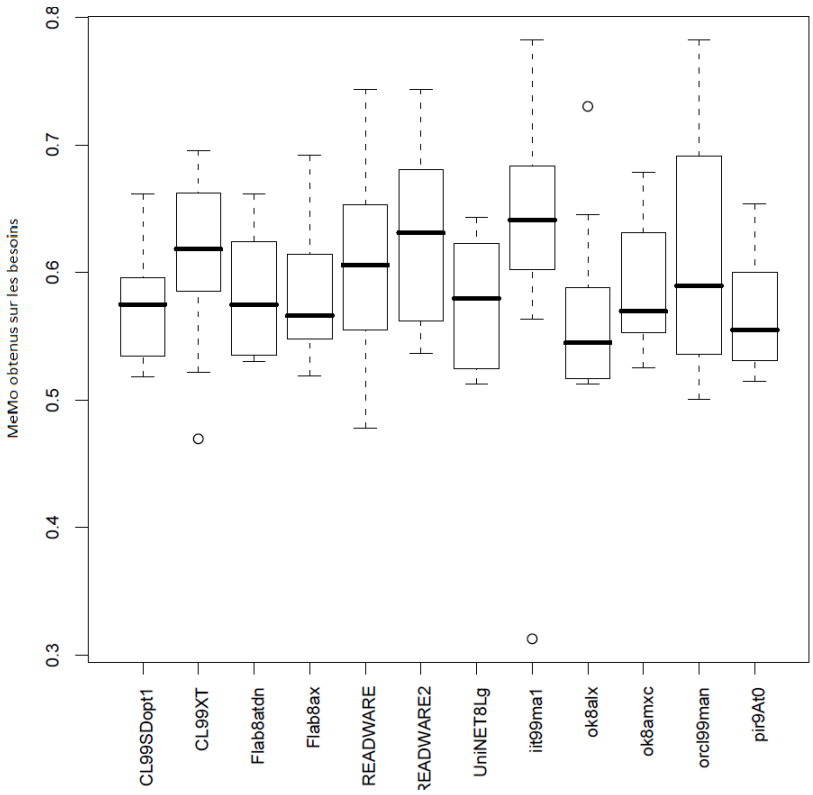

FIGURE 3.12 – MeMo : candidats aux meilleurs systèmes - groupe des requêtes moyennes

Groupes de besoins	**Système le plus performant**		**Système le plus robuste**	
	Rangs MeMo	MeMo	Rangs MeMo	MeMo
Faciles	MITSLStd	—	MITSLStd	—
Moyens	CL99XT	ii99mal	READWARE2	Flab8x ok8amxc CL99SDopt1
Difficiles	READWARE2	READWARE2	ibms99a	ibms99a
Très Difficiles	READWARE2	READWARE2	CL99XT	READWARE2

TABLE 3.3 – Comparaison des méthodes de sélection par groupe de besoins

l'évaluation des deux méthodes de sélection.

3.5.4 Évaluation des méthodes de sélection

Ici, nous cherchons à déterminer si les méthodes de sélection sont plus robustes (resp. performante) que les systèmes considérés initialement. En d'autres termes, nous cherchons à vérifier si les méthodes de sélection peuvent apporter une plus-value à un ensemble de systèmes de RI existants.

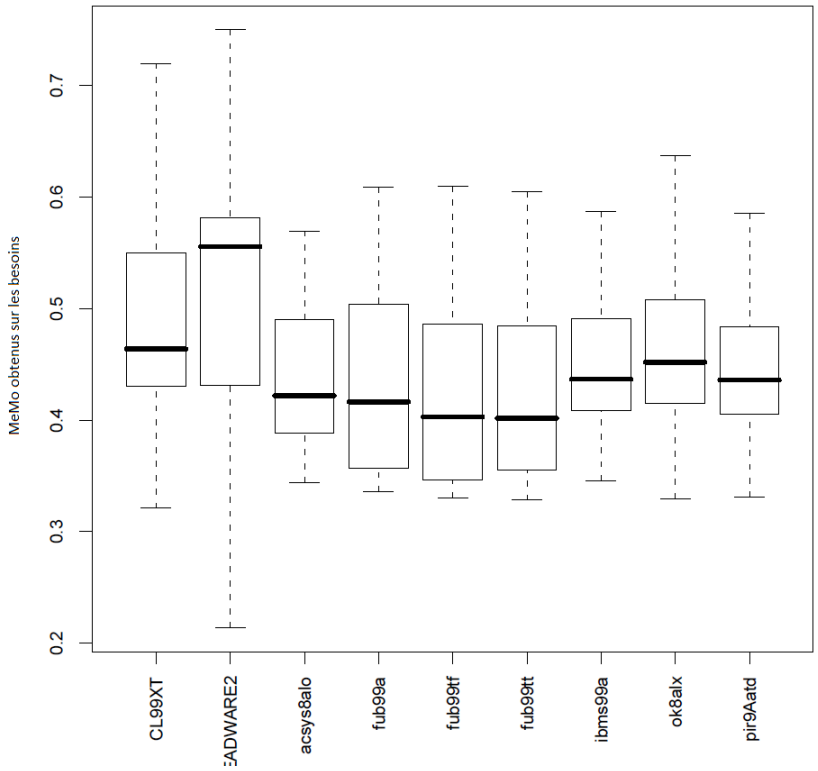

FIGURE 3.13 – MeMo : candidats aux meilleurs systèmes - groupe des requêtes difficiles

Sélection des systèmes de référence. Un système de référence doit être sélectionné pour tester les méthodes selon l'approche désirée : la performance ou la robustesse. Pour tester la performance, le système qui a obtenu la meilleure mesure moyenne sur l'ensemble des besoins est sélectionné. Ce système est "orcl99man" avec une mesure moyenne générale de 0.5220. Cette moyenne générale est utilisée comme référence pour évaluer la performance des méthodes de sélection sur l'ensemble des besoins. Pour affiner l'évaluation, nous procédons également à des tests sur chacun des groupes de besoins. La référence alors utilisée est la moyenne des mesures moyennes obtenues par "orcl99man" sur les besoins du groupe étudié.
Pour tester la robustesse, nous pourrions nous contenter d'utiliser le système ayant obtenu la variance la plus faible. Le risque est d'obtenir une référence certes très stable mais avec des performances faibles. Pour palier à cela, nous sélectionnons les deux meilleurs tiers des systèmes sur l'ensemble des besoins considérés ; parmi les systèmes restants, celui à plus faible variance est choisi comme référence.

Création des méta-exécutions à partir des sélections. La procédure d'évaluation est identique pour les deux méthodes. Pour chaque groupe de besoins, le système sélectionné pour ce groupe est utilisé. La moyenne des mesures de performance du système est calculée pour chaque besoin du groupe. Si pour un groupe de besoins, plusieurs systèmes sont retenus, la moyenne de leurs mesures moyennes est utilisée. En concaténant les résultats obtenus pour chacun des groupes, nous obtenons les mesures moyennes d'une méta-exécution de tous les besoins. Une

méta-exécution est utilisée pour les tests de performance et une seconde est utilisée pour les tests de robustesse.

Tests statistiques. La procédure de test est identique pour chaque groupe de besoins et pour l'ensemble de besoins. Pour tester si la différence des performances entre le système de référence et la méta-exécution est significative, nous calculons le t-test (ou test de Student) au seuil de 5% car celui-ci permet de comparer des moyennes (*cf.* 1.2). Pour cela, nous comparons les mesures moyennes obtenues par le système de référence avec les mesures moyennes de la méta-exécution sur les besoins considérés. Le t-test unilatéral est employé afin de déterminer si la performance de la méta-exécution est significativement supérieure (resp. significativement inférieure) au point de comparaison lorsque celle-ci est supérieure (resp. inférieure) à la référence. Pour tester si la robustesse de la méta-exécution est significativement différente de la robustesse de la référence, nous procédons au test de Fisher qui permet de comparer des variances et donc de vérifier quels systèmes ont les résultats les plus robustes (*cf.* 1.2). Ce test nous permet d'établir si la variance des mesures moyennes obtenues par la méta-exécution est significativement plus petite ou plus grande que la variance des mesures moyennes du système de référence. Un t-test est également réalisé sur les moyennes pour comparer les performances de ces exécutions.

Résultats de l'évaluation. Les tableaux suivants présentent les résultats obtenus par les méthodes de sélection pour le système le plus performant (tableau 3.4) et pour le système le plus robuste (tableau 3.5). Dans ces tableaux, "(++)" (resp. "(- -)") indique que la sélection est significativement supérieure (resp. inférieure) à la référence selon le test employé ; "(+)" et "(-)" indiquent un écart non significatif ; "(=)" indique qu'en valeur absolue l'écart est inférieur à 1%.

Groupes de besoins	**MeMo**		**Rangs MeMo**	
	Référence	Sélection	Référence	Sélection
Faciles	—	—	0.6792	(+) 0.6974
Moyens	0.6110	(+) 0.6253	0.6110	(=) 0.6082
Difficiles	0.4723	(+) 0.5195	0.4723	(+) 0.5195
Très Difficiles	0.4215	(=) 0.4207	0.4215	(=) 0.4207
Tous	0.4920	(+) 0.5188	0.5220	(+) 0.5440

TABLE 3.4 – Évaluation de la sélection du système le plus performant - test sur la moyenne

Dans le tableau 3.4, nous observons que hormis pour les besoins les plus difficiles, la sélection basée sur la MeMo obtient des performances supérieures à celles de la référence. La sélection sur les rangs MeMo est également meilleure pour les besoins faciles et difficiles et obtient des résultats très proches de la référence pour les deux autres groupes de besoins. Le meilleur gain est obtenu pour le groupe de besoins difficiles avec une amélioration de 10,0% par rapport à la référence.

Dans le tableau 3.5, pour les besoins de difficulté moyenne, nous observons que la variance de la sélection MeMo est significativement inférieure à celle de la référence ; on en déduit que la sélection est significativement plus robuste que la référence pour ces besoins avec un gain de 68% en robustesse avec des performances 21% plus élevées que celles de la référence. Au contraire, sur les besoins très difficiles, la référence est significativement plus robuste. Cependant, la sélection MeMo obtient des performances significativement plus élevées quel que soit le groupe de besoin considéré.

Groupes de besoins	MeMo				Rangs MeMo			
	Référence		Sélection		Référence		Sélection	
	Moyenne	Variance	Moyenne	Variance	Moyenne	Variance	Moyenne	Variance
Faciles	—	—	—	—	0,6066	0,00703	(++) 0,6974	(-) 0,00439
Moyens	0,4805	0,00431	(++) 0,5831	(- -) 0,00138	0,4805	0,00431	(++) 0,6298	(+) 0,00498
Difficiles	0,3992	0,00553	(++) 0,4513	(-) 0,00368	0,3992	0,00553	(++) 0,4513	(-) 0,00368
Très Difficiles	0,2306	0,00470	(++) 0,4207	(++) 0,02749	0,2306	0,00470	(++) 0,3321	(++) 0,02134
Tous	0,3744	0,01341	(++) 0,4752	(-) 0,01273	0,4116	0,01962	(++) 0,4789	(+) 0,02987

TABLE 3.5 – Évaluation de la sélection du système le plus robuste - tests sur la variance et la moyenne

La sélection sur les rangs MeMo permet d'obtenir une meilleure robustesse lorsque les besoins faciles et difficiles sont traités. Là encore, la robustesse est significativement moins bonne que celle de la référence pour les besoins très difficiles malgré des performances supérieures.

Lorsque tous les besoins sont considérés, les deux méthodes de sélection des systèmes selon la robustesse permettent d'obtenir des meilleurs performances que la référence. La sélection MeMo améliore la robustesse de 5% en conservant des performances de 27% supérieures à la référence. La sélection sur les rangs MeMo quant à elle dégrade la robustesse de moitié en conservant des performances 16% meilleures.

3.6 Conclusion et perspectives

Nous avons vu deux méthodes de sélection des systèmes. Une méthode consiste à calculer une unique mesure pour quantifier la performance moyenne de chaque système. La seconde méthode introduit les rangs des systèmes selon leur performance moyenne. Nous avons montré qu'avec ces méthodes il est possible de choisir le meilleur système selon un critère de robustesse ou de performance pour différents groupes de difficulté des besoins en information. La sélection des meilleurs systèmes est compliquée pour les besoins classés faciles ou moyens : ceux-ci sont très peu discriminants pour les systèmes de RI car, par définition, les systèmes réussissent tous à traiter ces besoins avec succès. Pour les groupes de besoins difficiles et très difficiles, le choix d'un unique meilleur système basé sur la performance ou sur la robustesse est plus aisé. L'évaluation montre que, à partir d'un ensemble de systèmes de RI donné, les méthodes de sélection analysées sont capables d'améliorer les résultats obtenus par les systèmes de manière individuelle. L'évaluation montre que l'on peut atteindre une amélioration de 10% des performances si celle-ci est privilégiée. Il est également montré que la robustesse peut être améliorée significativement jusqu'à 68% de gain tout en conservant de bonnes performances.

Pour les deux méthodes, un choix final des systèmes est opéré "à la main" et introduit un biais subjectif dans la sélection. Une extension de ce travail pourrait être de proposer une façon d'automatiser la détection des seuils fixés arbitrairement ici et d'analyser les différences de performances selon les valeurs de seuil retenues. L'emploi de méthodes statistiques telle que les skylines, les forêts aléatoires ou les analyses factorielles pourraient être étudiées pour détecter les valeurs des seuils à employer.

Le travail qui a été présenté dans ce chapitre a été publié et présenté à la conférence INFORSID (Bigot 2013), article retenu pour une extension dans la revue (Bigot 2014a). Par ailleurs, deux stagiaires ont contribué à ce travail (Poirier 2010).

Dans ce travail, nous avons considéré des classes de besoins en information définies selon

la difficulté que les SRI ont à traiter. Cependant, lorsqu'une nouvelle requête arrive, il est nécessaire au préalable de déterminer quel système doit la traiter. Il existe des travaux qui prédisent la difficulté, par exemple à partir de traits linguistiques (Mothe 2005), statistiques (Carmel 2010) ou une combinaison des deux (Chifu 2013).

Un prolongement de ces travaux a été de proposer une méthode d'apprentissage en commençant par entrainer les méthodes de sélection sur un ensemble connu de besoins en information. Ensuite, à partir de la difficulté prédite d'un besoin en information nouveau, au sens où il n'a pas été utilisé lors de l'apprentissage, le système sélectionné pour le groupe de difficulté concerné sera utilisé. Avec un tel procédé, l'efficacité et les gains apportés par les méthodes de sélections pourront être vérifiés.

Chapitre 4

Sélection des systèmes de recherche d'information selon les besoins en information

Sommaire du chapitre

4.1 Introduction

Les campagnes d'évaluation ont montré une grande variabilité des résultats obtenus par les SRI lorsqu'ils traitent les mêmes collections d'évaluation. La variabilité des résultats est issue de la variabilité qui existe au sein des systèmes de RI d'une part, et de celle qui existe parmi les besoins en information d'autre part.

Ainsi, un système S_1 obtiendra de bons résultats pour un besoin A et de moins bons résultats pour un besoin B tandis que le système S_2 obtiendra des résultats inverses. Nous cherchons donc à proposer une méthode qui permet de privilégier le système S_1 pour traiter le besoin A et le système S_2 pour traiter le besoin B.

Le but de ce chapitre est de tirer profit de la variabilité des systèmes et de celle des besoins en information. Pour cela, une étude approfondie de cette variabilité est nécessaire. Le travail présenté ici est donc double. Dans un premier temps nous proposons une analyse approfondie des besoins et des performances des systèmes de la collection d'évaluation. À partir de ces observations, une méthode de sélection des systèmes basées sur leur performance en fonction des besoins est proposée. Les performances puis les paramètres de cette méthode sont analysés.

Le chapitre est donc organisé comme suit. La section 4.2 analyse en profondeur la variabilité des besoins en information de la collection ansi que les performances des systèmes. La section 4.3 présente la méthode de sélection "Learning to Choose" décomposée en trois sous-méthodes qui tirent profit des variabilités observées en section 4.2.

Cette étude se place dans le contexte des campagnes d'évaluation TREC présentées en section 1.1.2.2 et les mtrices de données utilisées sont équivalentes à celles présentées en 1.2.1.

4.2 Analyse des besoins en information et des performances des systèmes

Dans cette section et dans les suivantes, nous nous plaçons dans le contexte des campagnes d'évaluation TREC décrites en section 2.1. La section 1.2 détaille davantage les méthodes de visualisation de données et les méthodes de classification utilisées dans cette étude.

Il est également difficile de donner un sens aux axes factoriels lorsque l'on fait usage d'analyses factorielles.

Dans la suite, nous donnerons un sens aux classes qui sont faites chaque foi que cela est possible ; quand ce n'est pas le cas, nous garderons à l'esprit que les SRI d'un même groupe traitent les requêtes de la même manière et que les requêtes d'une même classe ont le même niveau de difficulté, relatif ou absolu, pour chaque système.

4.2.1 Classification des besoins de TREC-7 ad hoc : précision moyenne

Une premiere analyse consiste à regrouper les besoins en information à l'aide la CAH. Dans cette analyse, les besoins en information jouent le rôle des individus et les systèmes celui des variables. La mesure utilisée est la précision moyenne que chaque SRI a obtenu pour chacun des besoins en information (*cf.* table 4.1). Le package cluster présent dans R est utilisé (http ://cran.r-project.org/web/packages/cluster/index.html).

Le dendogramme présenté en figure 4.1 est obtenu en appliquant la CAH aux données. Sur cette figure, la hauteur des branches représente la distance entre les groupes. Partant de la

	APL985L	APL985LC	APL985SC	AntHoc01	Brkly24	Brkly25
T351	0.2257	0.2261	0.1655	0.2933	0.2987	0.3137
T352	0.0229	0.0321	0.0594	0.0277	0.0379	0.0097
T353	0.3271	0.3052	0.2852	0.2091	0.374	0.264
T354	0.1119	0.1496	0.0908	0.0139	0.0192	0.1084
T355	0.0973	0.0688	0.0327	0.1365	0.0987	0.183
T356	0.052	0.0593	0.0462	0.0091	0.0128	0.0452
T357	0.1358	0.1803	0.1391	0.0984	0.3284	0.3277
T358	0.0994	0.0988	0.0489	0.1514	0.2078	0.3887
T359	0.0378	0.0337	0.0146	0.0223	0.0319	0.0357
T360	0.39	0.3825	0.4096	0.0404	0.3275	0.036

TABLE 4.1 – Exttrait de la matrice des précisions moyennes - TREC-7 ad hoc

matrice que nous analysons, deux besoins en information sont proches l'un de l'autre et donc regroupés si chaque système obtient une précision moyenne proche pour les deux besoins en information considérés. Quelques changements de groupes surviennent lorsque les K-moyennes sont appliquées à cette première classification (le détail des groupes finaux est décrit en table 4.2).

Couper le dendogramme à un niveau donné permet de définir une partition, chaque groupe est alors défini par la liste des éléments qu'il contient. Le niveau de coupe est déterminé selon la distance existant entre les noeuds. Pour être pertinent, un niveau de coupe devrait être choisi de manière à maximiser la distance (hauteur) entre deux noeuds du dendogramme. La figure 4.1 montre qu'une partition en 2 groupes est optimale ; le niveau suivant à 5 groupes l'est également (niveau indiqué sur la figure 4.1).

La classification en elle-même ne caractérise pas les groupes. Dans le cas spécifique de notre analyse, cette caractérisation peut être faite en regardant les valeurs dans la matrice initiale. En considérant le partitionnement en cinq groupes et en réordonnant la figure 4.1 pour des raisons de lisibilité, nous obtenons sur la gauche les besoins en information très faciles pour les systèmes : le groupe de besoins pour lesquels la moyenne des précisions moyennes des systèmes est la plus élevée (ce groupe est constitué d'une unique besoin : c'est le plus facile de la collection). Le second groupe en partant de la gauche est le groupe qui obtient la plus forte précision moyenne (après le besoin T365) moyennée sur les systèmes (de 0.3370 à 0.4628 ; 0.3908 en moyenne). Le groupe le plus à droite de la figure 4.1 correspond aux besoins les plus difficiles. Les 11 besoins de ce groupe obtiennent les précisions moyennes les plus basses (en moyenne sur les systèmes), variant de 0.0242 à 0.0981, 0.0493 en moyenne pour les besoins de ce groupe. La classification des besoins en information conduit à classer ensemble les besoins selon leur niveau de difficulté en moyenne.

Les groupes obtenus ici (et les couleurs mentionnées) seront utilisés par la suite et combinés avec l'analyse des systèmes.

4.2.2 Similarité des systèmes basée sur la précision moyenne pour TREC-7 ad hoc

De la même façon que pour les besoins en information, il est possible de regrouper les systèmes. Considérant les mêmes données que dans la section 4.2.1, nous regroupons les systèmes en utilisant la CAH. Le dendrogramme résultant est présenté en figure 4.2. Compte tenu de cette

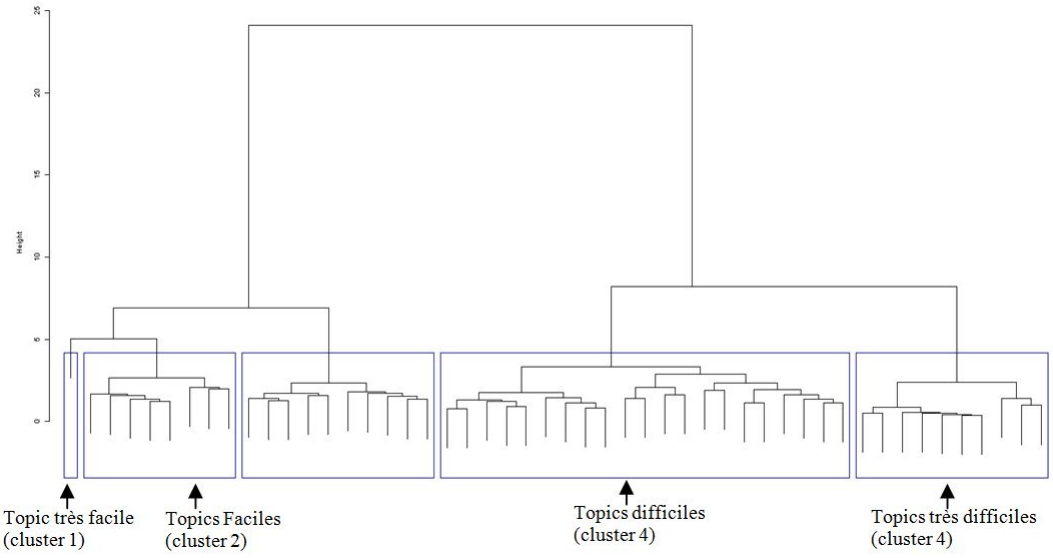

FIGURE 4.1 – Dendogramme résultat de la classification des besoins en information en utilisant la précision moyenne de TREC-7 ad hoc

analyse, les systèmes sont considérés comme similaires s'ils obtiennent des résultats similaires pour chaque besoin. De cette façon, deux systèmes sont proches l'un de l'autre, s'ils ont tous deux échoués ou réussis sur les mêmes besoins en information. Les K-moyennes ont également été appliquées ; le détail du contenu des groupes est listé ci-dessous.

La figure 4.2 et le contenu des groupes montrent que les versions d'un même système ont tendance à être très proches. Par exemple, les trois versions du système CLARIT98 sont regroupés dans le même groupe (premier groupe sur la gauche de la figure 4.2). Notez que les exécutions de CLARIT98 sont parmi les meilleures exécutions et pour ces exécutions, la MAP est comprise entre 0,3351 et 0,3702. La même chose se produit avec les versions Okapi qui sont regroupées dans le groupe 2. Le fait que les différentes versions du même système ont tendance à produire des performances similaires n'est pas surprenant étant donné que généralement, différentes versions d'un système correspondent au paramètre de réglage. Cela montre que l'on ne peut pas attendre de grande amélioration de la performance du système de RI simplement en apportant des modifications mineures. Ceci, dans un sens, soutient la conclusion de (Croft 2000) considére la fusion de données : les systèmes fusionnés doivent être indépendants les uns des autres. C'est aussi l'un des résultats qui motive la mise en place de techniques de combinaison que nous développons dans la section 4.3. Une analyse plus approfondie des résultats présentés dans la figure 4.2 révèle quelques autres informations intéressantes :

– la lecture de la publication associée aux exécutions soumises par CLARITECH (trec.nist.gov) montre que CLARIT98COMB est une combinaison de plusieurs autres exécutions soumises par CLARITECH. Les résultats montrent que ce n'est pas une combinaison très efficace (il n'y a pas de variation importante des résultats). Une conclusion pourrait être que, pour faire une fusion efficace de différents systèmes, ils doivent être indépendants les

Besoin Très Facile groupe 1 Bleu	Besoins Faciles groupe 2 Vert	Besoins Moyens groupe 3 Jaune	Besoins Difficiles groupe 4 Violet	Besoins Très Difficiles groupe 5 Brun
MAP 0.628	MAP 0.358	MAP 0.231	MAP 0.153	MAP 0.046
1-T365	4-T351	T353	T352	44-T356
	6-T361	T357	T354	45-T359
	3-T364	T358	T355	41-T363
	2-T368	T360	T362	47-T371
	5-T382	T366	T367	48-T378
	8-T396	T369	T370	46-T381
	7-T400	T373	T372	42-T383
		T374	T376	50-T386
		T375	T379	49-T389
		T377	T380	43-T394
		T385	T384	40-T397
		T392	T387	
			T388	
			T390	
			T391	
			T393	
			T395	
			T398	
			T399	

TABLE 4.2 – Précision moyenne obtenus sur les besoins en information de TREC-7 ad hoc par groupe de besoins en information. Les noms des besoins les plus faciles et les plus difficiles sont précédés de leur rang en terme de difficulté (1 étant le plus facile et 50 le plus difficile).

uns des autres.

– le laboratoire de l'IRIT a envoyé trois exécutions à l'aide du système Mercure ; deux appartiennent au même groupe (troisième groupe à partir de la gauche), mais le troisième est classé dans le dernier groupe. Ceci suggère que les versions du système conduisent à des modifications importantes, ce qui en fait était le cas. Les tests statistiques appliqués aux résultats le montrent aussi.

Compte tenu de la façon dont la classification est effectuée, mathématiquement, les groupes ne reflètent pas nécessairement la performance moyenne sur les besoins en information. Plutôt, les groupes reflètent l'homogénéité sur les performances obtenues individuellement sur les besoins en information. Cependant, une analyse plus approfondie de la matrice de données initiale montre que :

– le premier groupe sur la gauche (figure 4.2) contient le meilleur système à l'égard de la MAP (CLARITY98COMB, 0,4548). En moyenne, les systèmes de ce groupe obtiennent 0,345 pour la MAP, contre 0,199 lors de l'examen de tous les systèmes. Ce groupe comprend 7 systèmes. Les données montrent également que ce groupe contient les systèmes les mieux classés (les rangs 1 à 6 et 8 lorsque l'on regarde la MAP). Le système classé $7^{ième}$ est ok7ax qui appartient à un autre groupe ;

– les systèmes qui ont la plus faible performance sont regroupés dans le 4è*me* groupe à partir la gauche sur figure la 4.2. Ces 11 systèmes obtiennent la MAP la moins élevée qui est, en moyenne de 0,045. Ils correspondent aux systèmes aux rangs 93 à 103 en considérant

Orange	Rose	Violet	Bleu foncé	Vert	Bleu clair
7 systèmes	21 systèmes	23 systèmes	11 systèmes	15 systèmes	26 systèmes
CLARIT98CLUS	Cor7A1clt	Brkly25	KD70000	AntHoc01	APL985L
CLARIT98COMB	LNaTit7	Brkly26	KD71010q	fub98a	APL985LC
CLARIT98RANK	LNaTitDesc7	Cor7A2rrd	KD71010s	fub98b	APL985SC
iit98ma1	LNmanual7	Cor7A3rrf	ScaiTREC 7	gersh1	Brkly24
t7miti1	acsys7al	FLab7ad	dsir07a01	Ic98san3	ETHAB0
uwmt7a1	att98atc	INQ501	dsir07a02	Ic98san4	ETHAC0
uwmt7a2	att98atdc	INQ502	jalbse013	jalbse011	ETHAR0
	att98atde	INQ503	kslsV1	jalbse012	FLab7at
	bbn1	MerAdRbtnd	lanl981	nectitech	FLab7atE
	mds98t	MerTetAdtnd	nthu3	nsasgrp3	LIAClass
	mds98t2	acsys7mi	umd98a2	nsasgrp4	LIArel2
	mds98td	fsclt7m		nthu1	LIAshort2
	ok7am	harris1		umd98a1	MerAdRbtd
	ok7as	ibms98a		unc7aal1	acsys7as
	ok7ax	ibms98b		unc7aal2	fsclt7a
	pirc8Aa2	ibms98c			gersh2
	pirc8Ad	iit98au1			gersh3
	pirc8At	iowacuhk1			ibmg98a
	tno7exp1	iowacuhk2			ibmg98b
	uoftimgr	Nectitechall			ibmg98c
	uoftimgu	nectitechdes			iit98au2
		tno7cbm25			nthu2
		tno7tw4			nttdata7Al0
					nttdata7Al2
					nttdata7At1
					uwmt7a0
$M\bar{A}P$ 0.345	$M\bar{A}P$ 0.262	$M\bar{A}P$ 0.238	$M\bar{A}P$ 0.045	$M\bar{A}P$ 0.144	$M\bar{A}P$ 0.170

TABLE 4.3 – Précision moyenne obtenus par les systèmes de TREC-7 ad hoc par groupe de systèmes. $M\bar{A}P$ est la moyenne des MAP obtenues par les systèmes du groupe.

la MAP.

Ces deux éléments montrent que les systèmes les plus efficaces ont tendance à se comporter de la même manière, comme le font les systèmes qui permettent d'atteindre des résultats médiocres. Notez que "se comporter" signifie obtenir des résultats relativement mieux / pire / égaux sur les mêmes besoins en information. Concernant les autres groupes, la MAP n'est pas uniformément distribuée et ne peut donc pas être utilisée pour caractériser correctement les groupes. La figure suivante montre les systèmes ordonnés par valeur croissante de la MAP qu'ils ont obtenues (en moyenne sur les besoins en information) ; les couleurs correspondent aux groupes obtenus lors de la classification (*cf.* table 4.3 et leurs noms ont été remplacés par le numéro du groupe auxquels ils appartiennent :

Aux extrèmes (groupes 1 et 4), la performance des groupes et des systèmes individuels est fortement corrélée ; pour les autres groupes, ce lien est plus faible bien que les systèmes du groupe 2 ont tendance à être moins bons que ceux du 1 ; ces derniers ont également tendance à être moins performants que ceux du groupe 3 qui eux mêmes de manière globale obtiennent de moins bons résultats que ceux du groupe 5.

Ces observations vont en faveur du développement des techniques qui seraient dépendantes

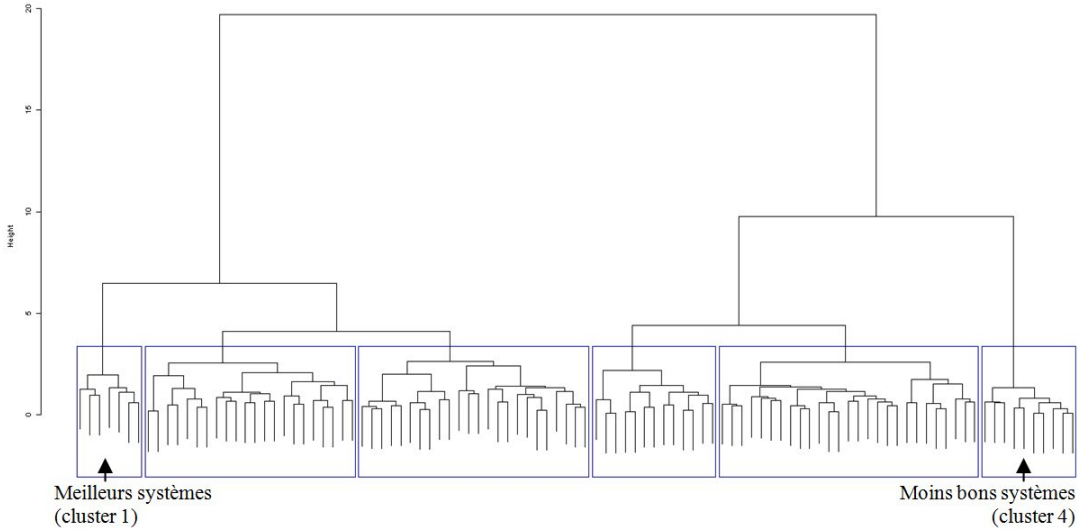

FIGURE 4.2 – Dendogramme résultat de la classification des systèmes en utilisant la précision moyenne de Trec 7 ad hoc. Le détail des groupes est fourni dans le tableau 4.3

FIGURE 4.3 – Systémes organisés par groupes et triés par performance croissante.

des besoins et qui considéreraient des types de systèmes. En effet, nous avons constaté que certains systèmes (que nous avons regroupés) ont tendance à se comporter de la même manière pour les mêmes besoins en information (soit avec de la réussite, soit en échec) mais que d'autres groupes de systèmes ne se comportent pas de la même manière sur les mêmes besoins. Ces observations sont le point de départ de la méthode développée dans la section 4.3.

4.2.3 Analyse de la corrélation entre systèmes et besoins

Une autre façon d'analyser le comportement des systèmes (et des besoins en information), les uns par rapport aux autres, est d'envisager l'AFC. Les visualisations associées à l'AFC affichent les distances entre les besoins et entre les systèmes (voir section 1.2). Nous avons utilisé le package ade4 de R (http ://pbil.univ-lyon1.fr/ade4/).

La figure 4.4 montre les deux premiers facteurs principaux qui correspondent à 31,3% de l'inertie totale. Le facteur 1 correspond à 20,5% de l'inertie totale : cela signifie que les systèmes sont d'abord distingués par rapport à ce facteur. Les deux premiers facteurs expliquent environ 1/3 de l'information. Il y a 25 axes factoriels (facteurs), nombre fixé mathématiquement, et chaque barre verticale représente la quantité d'information portée par le facteur correspondant (voir Proportion% sur le côté gauche, figure 4.5). La courbe en pointillés exprime la proportion cumulée de l'inertie expliquée par les axes factoriels et montre que, par exemple, l'utilisation des cinq premiers facteurs exprimerait la moitié de l'information totale (voir Proportion cumulée

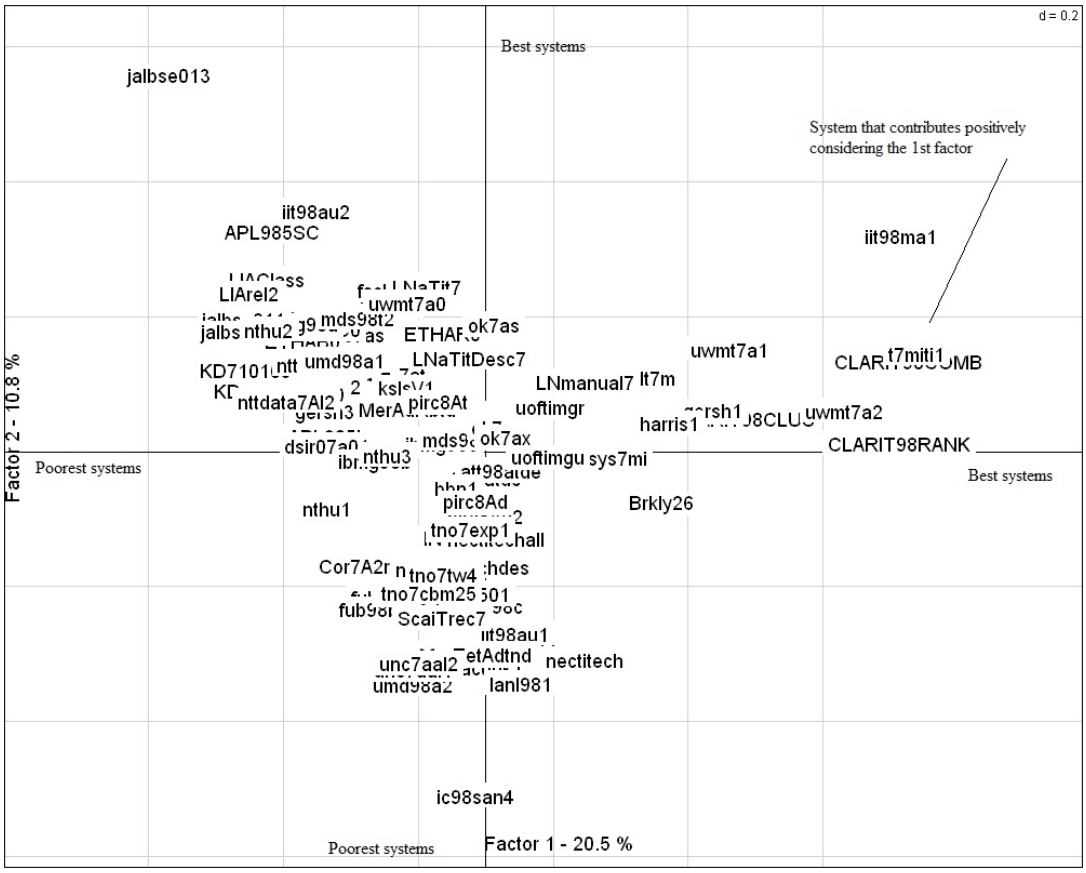

FIGURE 4.4 – Deux premiers axes factoriels de l'AFC à partir de la matrice des précisions moyennes de TREC-7 ad hoc. Les systèmes sont représentés dans l'espace des deux premiers facteurs qui correspondent à 20.5% and 10.8% de l'inertie totale. Certains éléments sont cachés par d'autres car ils sont trop proches les uns des autres.

sur le côté droit).

Dans la figure 4.4, la disposition des individus colonnes (les systèmes) selon les deux facteurs peuvent se lire comme suit : plus leur valeur est élevée, plus leur contribution au facteur l'est également (soit positivement, soit négativement). Lors de l'analyse d'un facteur, il faut considérer les coordonnées (la contribution) du point au facteur. Dans le cas de la figure 4.4, les points affichés correspondent aux systèmes. Les systèmes sur la droite du graphique contribuent positivement au facteur 1.

Par exemple CLARIT98XX, itt98ma1, t7miti1, et umwt7aXX sont les systèmes qui contribuent le plus positivement au facteur 1. Ces systèmes sont parmi les meilleurs lorsque l'on considère la MAP. Au contraire, les systèmes qui se trouvent sur la gauche du facteur 1 sont des systèmes qui y contribuent négativement (par exemple les systèmes jalbse013 et KD7XX

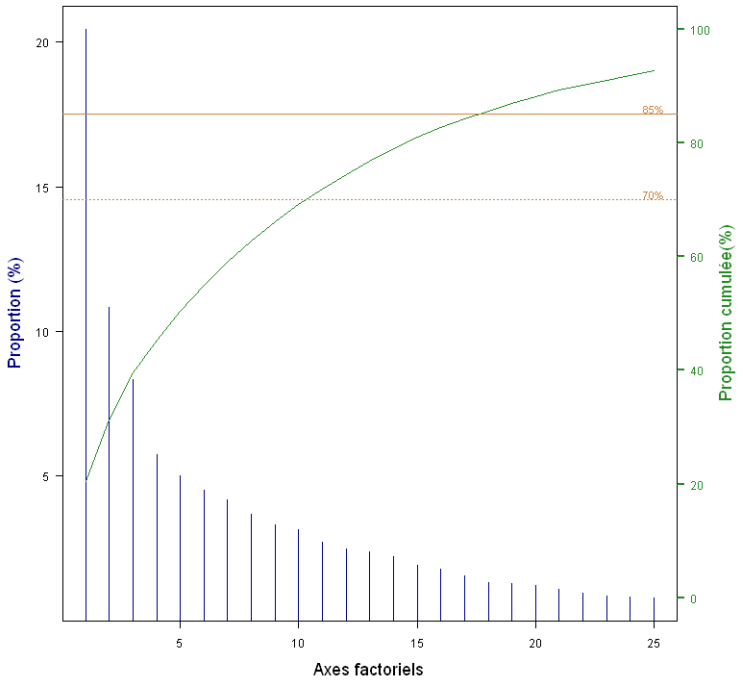

FIGURE 4.5 – Inertie relative et interie relative cumulée exprimée par les axes factoriels

systèmes). Ils sont à l'opposé des systèmes précédents compte tenu de ce facteur. En effet, pour revenir à la matrice initiale, les systèmes sont parmi ceux qui obtiennent la plus faible MAP (classe 99, 100 et 102). Les systèmes sur le côté gauche du facteur 1 peuvent être étiquetés comme *les moins bons* en termes de performance et *les meilleurs systèmes* sur le côté droit.

Il est important de noter que sans surprise le contenu de la figure 4.4 et de la figure 4.2 donnent des informations qui se rejoignent. Par exemple, les trois versions des systèmes de CLARIT98XX sont proches les unes des autres (sur la droite du facteur 1). Ils sont également à proximité de t7miti1 et uwmt7a2. CLARIT98XX, itt98ma1, t7miti1, et umwt7aXX appartiennent au même groupe (voir figure 4.2 ou table 4.3). De plus, lors de l'application de l'ACP sur les mêmes données (les données sont d'abord centrées sur les observations), les résultats restent cohérents. Par exemple, les systèmes CLARIT98XX et umwt7aXX sont proches les uns des autres et les systèmes de KD7XX sont dans une région différente de l'espace. Cependant, l'ACP ne permet pas d'expliquer, d'une manière simple, les relations entre variables et observations. Pour cette raison, nous présentons uniquement les résultats de l'AFC.

Il est intéressant de noter l'agencement spécifique des ok7ax (près du centre de la figure 4.4). Ce système obtient le rang 7 (sur 103) en considérant la MAP ; ce qui signifie qu'il est parmi les meilleurs systèmes en termes de performance. La CAH ne l'a pourtant pas regroupé avec les autres meilleurs systèmes (comme les systèmes CLARIT98XX par exemple). Les résultats que nous obtenons avec la CAH sont cohérents puisque ok7ax et les systèmes CLARIT98XX sont lointains si l'on considère le facteur 1. Puisque les deux sont de bons systèmes (compte tenu de leur performance en terme de MAP), on peut conclure qu'ils ne sont pas bons pour les mêmes

raisons (pas les mêmes besoins en information).

Ces deux observations (ok7ax et -faisons le choix de- CLARIT98COMBare sont parmi les meilleurs systèmes en ce qui concerne la MAP et les deux systèmes ne se comportent pas de la même manière car ils ne sont pas groupés ensemble) donne à penser que ces deux systèmes peuvent être combinés d'une manière quelconque, par exemple sur une base par besoin. Ceci est une autre motivation de la méthode que nous proposons en section 4.3.

Considérant le facteur 2, les systèmes qui contribuent le plus dans le sens positif sont : jalbse013, iit98au2, APL9858C, LIAXX, LNaTit7 ... D'autre part, ic98san4, Ianl981, umd98a2, iit98au1 sont parmi les systèmes qui contribuent le plus à l'opposé. Ic98san4 (hors du champs de la figure 4.4) est sur la partie négative du facteur 2 ; il est placé entre les systèmes les plus faibles selon la MAP (86e rang). Le facteur 2 caractérise les systèmes faibles sur la partie inférieure et de bons systèmes sur la partie supérieure. La CAH et l'AFC montrent tous deux que les deux groupes de systèmes échouent en raison de besoins différents. Ils n'appartiennent pas aux mêmes groupes de la classification et ne contribuent pas de la même façon à la construction des principaux facteurs. Par exemple, KD7XX contribuent négativement à tenir le facteur 1 alors que Ianl981 contribuent négativement au facteur 2.

Lors de l'analyse des autres facteurs, d'autres systèmes apparaissent comme ayant un comportement spécifique. Par exemple, le facteur 3 (voir figure 4.9) montre clairement le groupe qui contient les systèmes ok7XX (groupe numéro 2 dans 4.3 et le deuxième groupe en partant de la gauche sur la figure 4.2).

Une caractéristique intéressante de l'AFC est que nous pouvons visualiser simultanément les individus colonnes et les individus lignes (systèmes et besoins dans notre cas). Comme les projections des lignes et des colonnes sont liées par une relation barycentrique (Murtagh 2005), il est possible de définir des distances dans l'espace euclidien et d'afficher les observations et les variables à l'aide d'une unique représentation graphique. Mais comme il y a de nombreux points, il est préférable de montrer deux graphiques. Tout d'abord, la figure 4.6 présente seulement les besoins en information. Il est important de comprendre que les figures 4.4 et 4.6 utilisent le même plan factoriel (les mêmes 2 premiers axes). Ensuite, la figure 4.7 affiche le centre de gravité de chaque groupe de systèmes (les 6 groupes qui sont affichés sur la figure 4.4), au lieu d'afficher les 103 systèmes, en plus des 50 besoins.

La figure 4.6 présente les résultats de l'AFC sur la base de la matrice des précisions moyennes, en affichant les individus lignes / besoins seulement. Les besoins en information sont projetés sur les facteurs précédents 1 et 2 ; n'oublions pas qu'ils correspondent à plus de 30% de l'inertie totale et ont été caractérisés selon la MAP obtenue par les systèmes. Les besoins qui contribuent le plus à la construction du facteur principal 1 sont ceux qui obtiennent les plus hautes coordonnées sur ce facteur. En considérant les figures 4.6 et 4.8, ensemble ou compte tenu de la figure 4.7, nous pouvons expliquer le fait que les systèmes dans le coin en haut à droite du plan traitent spécifiquement bien les besoins qui sont dans la même direction.

L'AFC montre que le besoin T389 (voir figure 4.6, partie très à droite et figure 4.7 aussi), T383, T397, T356 et à un niveau inférieur, T376, T393, T372 et T394 par exemples sont typiques des systèmes qui contribuent positivement au facteur 1. T379, T383 et T397 à un niveau inférieur, T393, T386 sont associés à des systèmes qui contribuent positivement au facteur 2. Étant donné que ces systèmes correspondent à des systèmes qui fonctionnent bien, ces besoins en information sont dits *faciles* pour les systèmes qui sont dans le coin supérieur droit du premier plan factoriel. En effet, pour revenir à la matrice des précisions moyennes, iit98ma1 est le meilleur pour T389 (même direction en considérant les figures 4.4 et 4.7.) ; c'est

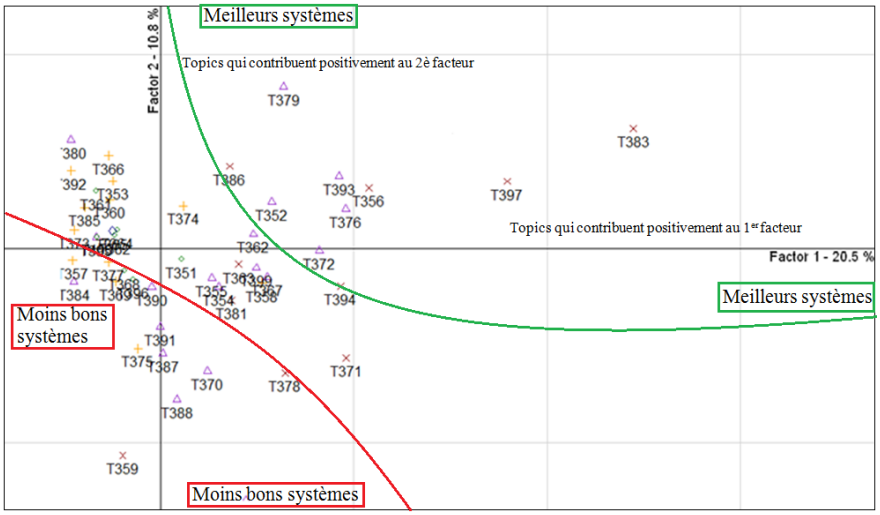

FIGURE 4.6 – Premiers facteurs de l'AFC pour les besoins en information uniquement - matrice des précisions moyennes de TREC-7 ad hoc

un besoin en information difficile : la moyenne obtenue par les systèmes est 0,0250 et 0,4906 par iit98ma1.

Si on imagine un vecteur depuis l'origine des facteurs 1 et 2 allant jusqu'à iit98ma1, on voit que T389 est dans la même direction. T397 est également un besoin difficile (les systèmes obtiennent une moyenne des précisions moyennes de 0,0980), et CLARIT08RANK effectue le meilleur résultat sur ce besoin (0,6089). Il s'agit donc d'un besoin atypique puisque les systèmes ont en moyenne des résultats médiocres dessus, mais quelques systèmes réussissent. T383 est aussi un besoin difficile (moyenne de 0,0593) ; les meilleurs systèmes sont également performants sur ce besoin.

Au contraire, T376 est un besoin facile (moyenne de 0,5179) ; LNaTitDesc7 est le système qui a obtenu le meilleur score sur ce besoin, suivi par la plupart des meilleurs systèmes. La même chose se produit pour le besoin T372, les systèmes qui se classent le mieux pour ce besoin sont parmi les meilleurs systèmes. Néanmoins, compte tenu du coté gauche du facteur 1 (coordonnée négative), T357 par exemple, est à l'opposé des systèmes qui fonctionnent bien. Ce besoin n'est pas si difficile (0,1468), mais le premier système du meilleur groupe de systèmes apparaît au rang 15 et celui de la deuxième classe 41$^{è\ 1}$! Les meilleurs systèmes ne parviennent donc pas à traiter correctement ce besoin en information.

Les besoins en information qui sont proches de l'origine de la représentation des facteurs sont ceux qui sont typiques (par opposition aux atypiques) ce qui signifie qu'ils sont proches du profil "moyen" et se comportent comme les autres. Ces besoins ne permettent pas de distinguer

1. lorsque l'on classe les systèmes du meilleur (rang 1 pour la plus forte valeur de précision moyenne) au moins bon.

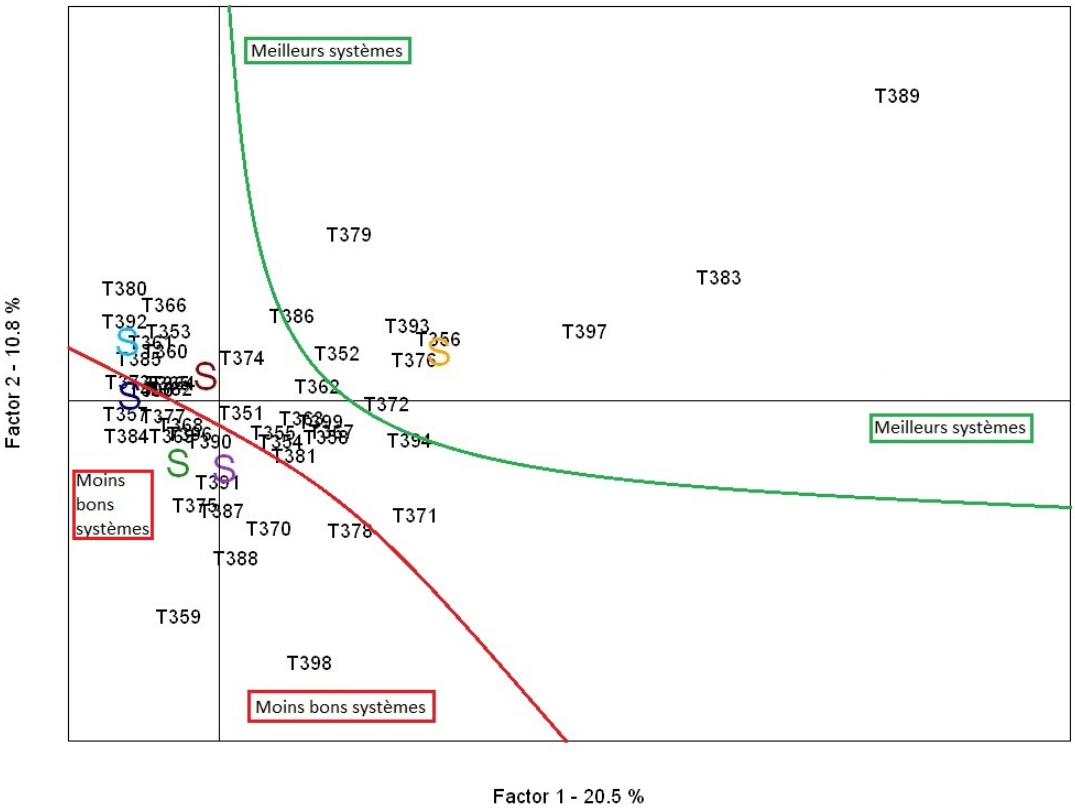

FIGURE 4.7 – Premiers facteurs de l'AFC pour les besoins en information et les centroïdes des groupes de systèmes - matrice des précisions moyennes de TREC-7 ad hoc.

les systèmes sur le côté droit du facteur 1 (qui se trouvent être de bons systèmes) des systèmes qui se trouvent sur le côté gauche du facteur 1 (systèmes moins bons). Ces besoins sont par exemple T351, T390, T374. Pour ces besoins, de bons systèmes échouent tandis que certains mauvais systèmes réussissent. Pour T390 par exemple (moyenne des précisions moyennes des systèmes de 0,0959), le système qui a effectué le meilleur score est CLARIT98RANK (rang 1 pour ce besoin), un système qui appartient au meilleur groupe de systèmes (groupe 1 sur la figure 4.1). Le prochain système qui appartient à ce groupe se trouve rang 13 pour ce besoin. Nous pouvons conclure que les systèmes du meilleur groupe de systèmes manque d'un comportement spécifique pour T390 : ce besoin ne permet pas de distinguer ce groupe des autres.

Le facteur 3 (voir figure 4.9) permet de distinguer le groupe 2 des systèmes. Quand on regarde les besoins en information en même temps, il est possible de voir que T352, T390, T385, T376, et T357 sont les besoins qui contribuent le plus dans la même direction que les systèmes du groupe 2 ; par opposition à T363 qui contribue à l'opposé.

La figure 4.6 affiche la combinaison des résultats de l'AFC et de la CAH (voir section 1.2 pour les raisons de la combinaison des deux). Dans la figure 4.6, une couleur (pour la version

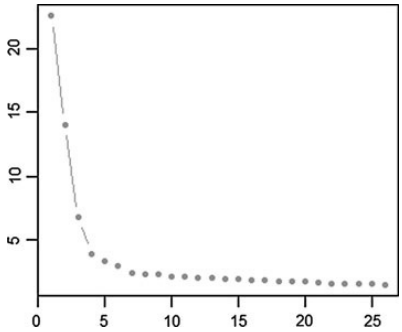

FIGURE 4.8 – Exemple de distance entre les groupes de besoins en information.

numérique) et un symbole spécifique ont été associés à chaque groupe de besoins détectés dans la première phase. Par exemple, les besoins appartenant au groupe qui apparaît en dernier sur le dendogramme de la 4.2 sont tracés en croix brunes et (×) dans la figure 4.6 (besoins *difficiles*). En jaune et '+' sont représentés les besoins *faciles* (troisième groupe). Cela facilite l'observation puisque les besoins difficiles sont principalement sur le côté droit de facteur 1, tandis que les besoins faciles sont situés en majorité sur le côté gauche.

4.2.4 Discussion

De l'analyse fournie dans les sections précédentes, nous pouvons conclure que certains systèmes sont préférables pour traiter certains groupes de besoins, mais nous ne pouvons pas conclure qu'ils sont meilleurs pour le traitement de tous les besoins en information d'un groupe. On remarque que les besoins difficiles sont sur le côté droit du facteur 1 de l'AFC, comme le sont les meilleurs systèmes. Cette observation montre que les meilleurs systèmes sont meilleurs pour le traitement des besoins difficiles (ce qui pourrait être une raison pour laquelle ils sont considérés comme les meilleurs systèmes mais pas la seule, car les résultats sont moyennés sur un ensemble de besoins en information). Les besoins faciles quant à eux sont à l'opposé sur le facteur 1. Les meilleurs systèmes ne sont donc pas nécessairement les meilleurs systèmes pour traiter ces besoins.

En analysant les deux figures 4.6 et 4.8 simultanément, nous serions enclins à conclure que certains besoins spécifiques doivent être traités par un système sélectionné : T389 par iit98ma par exemple. Allant plus loin, nous pouvons développer une nouvelle technique de combinaison de systèmes qui apprend le système ou le type de systèmes à utiliser selon un besoin donné. Cela est utile lorsque les requêtes sont répétées dans les systèmes réels : apprendre une fois pour toutes les futures occurrences de la même requête. Il pourrait être quelque peu coûteux de mettre en œuvre cette méthode, mais cela vaut la peine si les demandes répétées sont assez fréquentes. Cela pourrait être trop coûteux si cela doit être fait sur chaque requête qui se produit. Par conséquent, il serait utile de prévoir des requêtes qui ont la plus haute probabilité de se répéter et de concentrer l'effort sur ces requêtes.

Une motivation pour définir une méthode qui apprend le système à utiliser pour une requête donnée est basée sur le fait que, dans les systèmes réels, les requêtes sont répétées dans le temps. Diverses études valident cette hypothèse et montrent une proportion importante de demandes répétées : 15% (Smyth 2004), 17% (Tyler 2010), 33% (Teevan 2007) et un peu plus de 50% (Sanderson 2007). En outre, (Zhang 2009) proposent des traits des requêtes pour faciliter la

détection des requêtes les plus susceptibles d'être répétées.

De ces études, il semble raisonnable de promouvoir une approche qui apprend le procédé ou système ou groupe de systèmes qu'il est préférable d'utiliser pour traiter une requête donnée tout en gardant à l'esprit que l'effort (ou coût) serait nécessaire pour un faible nombre de requêtes seulement, celles pour lesquelles une répétition est prédite. Notre méthode vise à proposer un tel procédé qui apprend quel est le meilleur système. Dans la section suivante, nous présentons une nouvelle méthode pour combiner des systèmes. Cette méthode s'appuie sur l'analyse faite dans la section 4.2 et est basée sur la sélection du système via une approche par besoin.

4.3 Méthodes de sélection des systèmes pour tirer profit de la variabilité

L'analyse que nous avons menée dans la section précédente nous conduit à proposer un nouveau procédé de combinaison. Nous avons considéré l'hypothèse suivante : les systèmes doivent être sélectionnés sur la base de leur non-corrélation. Plus précisément, notre hypothèse est que, tenir compte de systèmes complémentaires, en termes de dépendance telles que définies par le regroupement, doit être plus efficace que la combinaison des systèmes analogues (systèmes qui appartiennent à la même classe). Dans notre approche, nous considérons que deux systèmes sont complémentaires s'ils ne sont pas efficaces pour les mêmes besoins, c'est à dire s'ils n'appartiennent pas aux mêmes groupes détectés par l'analyse. Cette perspective diffère des autres travaux connexes qui se concentrent plutôt sur le chevauchement des listes de documents récupérés (Beitzel 2003) (Croft 2000).

Nous proposons plusieurs variantes de la même approche. Pour chacune d'elle, l'idée est de combiner les systèmes non par agrégation des listes de documents récupérés, mais en choisissant l'une des listes de documents restitués.

La première variante est la plus naturelle : chaque besoin est associé un système unique. Nous appelons cette méthode OneT2OneS (one topic to one system). La seconde variante OneT2ClusterS (one topic to one system cluster) associe un groupe de systèmes à un besoin en information. La dernière variante est la ClusterT2ClusterS (one topic cluster to one system cluster). Celle-ci vise à étudier le comportement de l'approche lorsque les groupes de besoins en information sont envisagés.

4.3.1 OneT2OneS

L'objectif principal de la méthode OneT2OneS est d'associer à chaque besoin, le système qui devrait être utilisé pour le traiter (un besoin, un système). Notre méthode diffère de l'apprentissage de méthodes de classement (Liu 2010), dans lequel la meilleure méthode de classement ne dépend pas du besoin. Au contraire, nous proposons une approche qui sélectionne le système selon le besoin en information.

Plus précisément, pour chaque besoin, nous sélectionons le système qui maximise la performance pour la mesure que nous considérons (par exemple la précision moyenne). Pour évaluer la méthode, nous considérons l'ensemble des besoins. Nous apprenons le meilleur système à utiliser pour chaque besoin sur un ensemble de documents d'apprentissage et évaluons les résultats sur les mêmes besoins, mais sur un ensemble de documents test, ensemble qui diffère de l'ensemble d'apprentissage.

Cette méthode est censée générer les meilleurs résultats en termes d'efficacité car elle permet d'optimiser la mesure de la performance de chaque besoin de manière individuelle.

Cependant, l'utilisation de ce procédé a un effet indésirable qui est qu'un système qui réussit par hasard sur un besoin peut être sélectionné ; il serait dangereux de proposer un métasystème qui peut exploiter cela dans un cas général.

Pour éliminer cet effet, nous considérons OneT2ClusterS (one topic cluster to one system cluster).

4.3.2 OneT2ClusterS

Dans cette variante, pour chaque besoin, la méthode associe le groupe de systèmes qui doit le traiter. Pour ce groupe, un système parangon est choisi. Cette méthode devrait avoir de moins bonnes performances que OneT2OneS mais devrait être plus rovuste. Une autre raison pour introduire cette variante est qu'elle permet de réduire le nombre de systèmes à utiliser.

En effet, plutôt que de considérer le meilleur système pour chaque besoin, nous calculons le meilleur groupe de systèmes pour chaque besoin en information. Pour cela, nous regroupons d'abord les systèmes (voir 1.2, classification hiérarchique en utilisant le critère de Ward et K-moyennes). Nous définissons le système représentatif pour chaque groupe en tant que système qui obtient la meilleure valeur de la mesure pour ce groupe (par exemple $max(précision\_moyenne)$). Lorsque deux groupes 'gagnent', le vainqueur final est le système qui obtient la meilleure valeur de la mesure sur les besoins en information. Par rapport à OneT2OneS, ce procédé vise à éliminer certains systèmes qui peuvent avoir un comportement très inhabituel, à savoir, être très mauvais sur tous les besoins, sauf un ou quelques-uns.

Encore une fois, afin d'évaluer le procédé, la méthode apprend d'abord sur la collection de documents d'apprentissage et est évaluée sur l'ensemble test, tous les besoins sont considérés à la fois pour l'apprentissage et le test.

4.3.3 ClusterT2ClusterS

Dans la méthode ClusterT2ClusterS, nous analysons si les résultats sont dépendants des groupes de besoins en information. Par exemple, nous voulons savoir si la méthode fonctionne mieux sur des besoins difficiles ou mieux sur les faciles, ou s'il n'existe aucune indication que la difficulté du besoin a un impact sur l'efficacité du procédé.

4.4 Conclusion

L'objectif à long terme de notre travail est de construire un système qui serait basé sur plusieurs techniques de RI dans les différentes étapes du processus de et choisir celles appropriées en fonction de la requête à traiter. Les systèmes de question-réponse ont utilisé cette idée : selon le type de requête (qui, quoi, etc) que le système reçoit ; la méthode utilisée pour extraire la réponse (ou le type d'objets que le système restitue) diffère. Cependant, la généralisation de ce principe à tout type de requêtes n'est pas triviale. Ce travail examine de nouvelles orientations à cette fin. Pour ce faire, notre premier objectif était d'analyser les résultats du système sur les mêmes besoins utilisateurs mais avec de nouvelles approches pour que nous puissions montrer que de nouvelles voies pourraient être explorées dans le cadre de l'adaptation des systèmes et / ou de la combinaison de systèmes.

Le chapitre 5 présente une évaluation de ces méthodes et de leurs performances comparativement aux performances des systèmes initiaux.

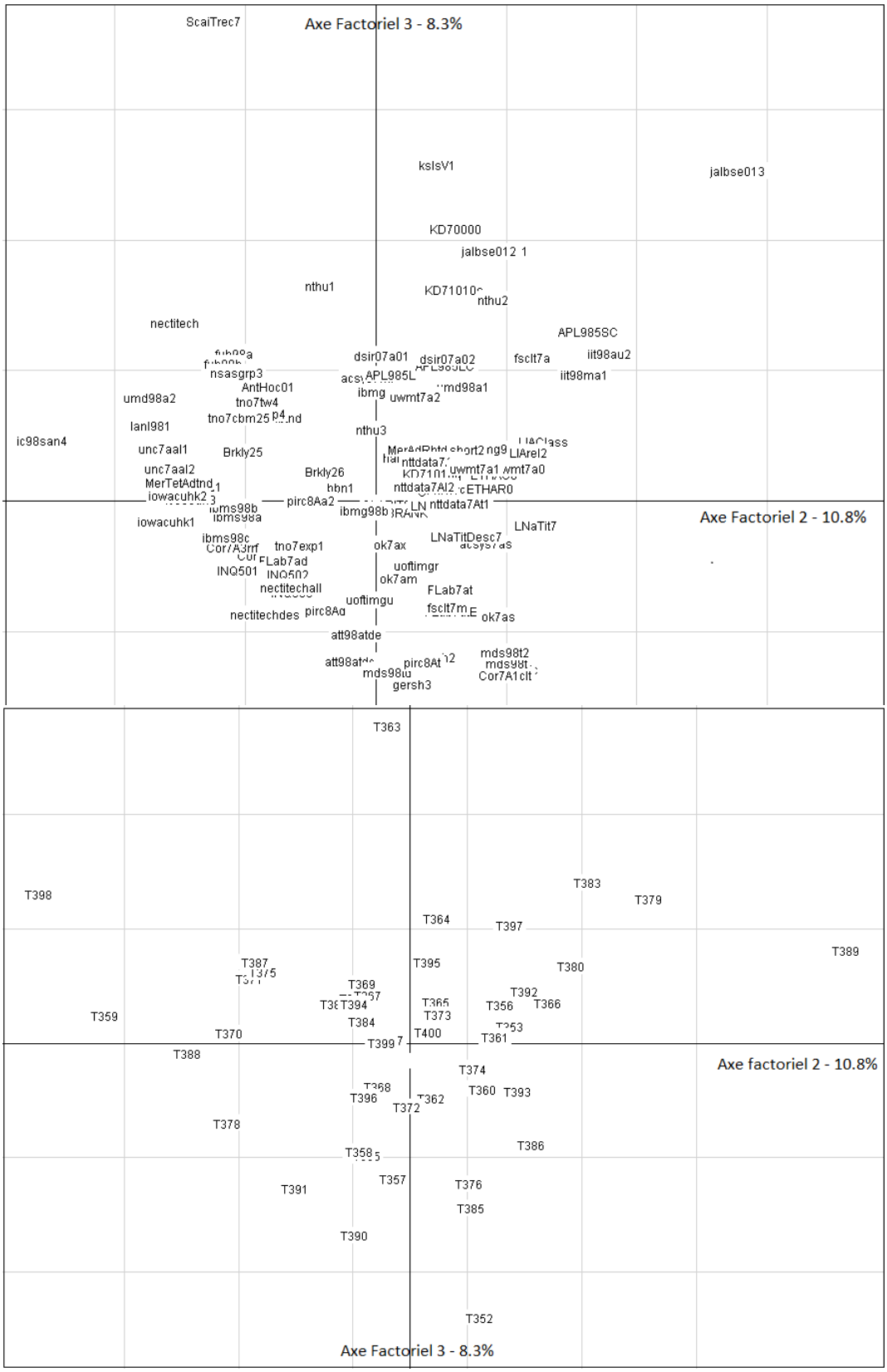

FIGURE 4.9 – Facteurs 2 et 3 de l'AFC sur les précisions moyennes de TREC-7 ad hoc. Les deux facteurs repérsentent 10.8% and 8.3% de l'inertie totale. Les systèmes sont représentés sur le graphique du haut et les requêtes sur celui du bas.

Chapitre 5

Évaluation des méthodes de sélections

Sommaire du chapitre

Dans ce chapitre nous développons l'évaluation des méthodes proposées au chapitre précédent. Pour cela, nous proposons une application de ces méthodes au cas des requêtes répétées. Cette application vise à étudier plus en profondeur les paramètres de la méthode proposée en 4.3 pour étudier leur impact sur les performances finales dans l'environnement d'évaluation proposé par TREC.

La section 5.1 présente une étude préliminaire afin de déterminer les gains en performances des méthodes de sélection à la fois sur l'ensemble des besoins en information et par catégories de besoins. Ces résultats ont été publié dans (Bigot 2011). Dans la section 5.2, nous considérons que les méthodes proposées au chapitre 4 sont différentes version d'une même méthode que nous nommons Learning to Choose (L2C). Aussi, nous essayons de proposer un contexte d'évaluation proche de cas réels d'utilisation où il y a souvent peu de systèmes différents disponibles pour traiter les besoins. Ce travail sera prochainement publié dans (Bigot 2014b). Enfin, la section 5.3 conclue ces travaux.

5.1 Évaluation préliminaire

Comme indiqué dans le chapitre précédent, il faut une phase d'apprentissage pour évaluer notre méthode qui est dépendante des besoins. Pour cette raison, les mêmes besoins doivent être utilisés pendant la phase d'apprentissage et la phase de test. D'autre part, les documents sur lesquels les besoins sont traités devraient être différents pour les deux phases. De cette façon, nous démontrons qu'il est possible d'apprendre notre procédé sur un sous-ensemble de documents et que la fonction apprise peut être appliquée avec succès sur d'autres ensembles de documents. La collection de documents est donc divisée en deux partitions : un ensemble de documents d'apprentissage et un ensemble de documents de test.

Pour évaluer notre méthode, nous avons appliqué dix validations croisées. Pour cela, nous considérons la collection TREC ad hoc et faisons une partition : tous les besoins en information sont utilisés dans la phase de formation, mais les documents qui sont impliqués sont divisées en une partie d'apprentissage (2/3 du total) et une partie de test (1/3). Plutôt que de considérer chaque document comme indépendant, nous les regroupons selon le début de leur identifiant (par exemple, tous les documents qui commencent par FR940105 seront soit partie de collection d'apprentissage soit de l'ensemble de test). Dans TREC-7, 2/3 de ces groupes de documents font partie de l'apprentissage, le 1/3 restant de la série de tests. Les éxecutions (*runs*) et qrels sont traités afin de prendre en compte cette division. Plus précisément, chaque éxecution est divisée en apprentissage et test de sorte que pour l'exécution en cours :

- chaque document de la collection d'apprentissage va dans la partie appentissage de l'exécution ;
- les documents restants vont dans la partie test de l'exécution.

Les fichiers qrels sont traités de la même façon. Un partitionnement unique des données de cette façon, cependant, ne suffit pas à rendre de solides conclusions, car il existe encore un élément de hasard. Pour cette raison, nous avons effectué 10 partitions différentes de la collection de documents. Pour chaque partition, les documents tombent dans l'ensemble d'apprentissage ou dans l'ensemble de test suivant une fonction aléatoire uniforme. Nous apprenons le meilleur système ou le meilleur groupe de systèmes à utiliser sur chaque document de la collection d'apprentissage. L'apprentissage consiste à maximiser la MAP (compte tenu de la liste de documents restituée et du qrel associé). Après la phase d'apprentissage, nous utilisons le système

ou le groupe de systèmes appris et les appliquons sur les documents tests correspondants. Par conséquent, il y a 10 expériences marquées exp1, ..., exp10 et les résultats correspondants. Nous avons ensuite fait la moyenne des résultats obtenus au cours des 10 expériences.

Dans les tableaux suivants, la valeur de MAP obtenue par notre méthode, résultat du *métasystème* issu du procédé de combinaison, est suivi par une '*' lorsque la différence par rapport au meilleur système est statistiquement significative. Suite aux recommandations de (Smucker 2007) et (Hull 1993), la significativité statistique de la différence de moyenne entre deux approches est testée en utilisant le t-test apparié de Student. La différence est calculée entre les valeurs des systèmes appariés pour tous les besoins et la différence entre les échantillons tests est dite statistiquement significative lorsque $p < 0,05$. Bien que ce test théorique exige une distribution normale des données, il est relativement assez robuste aux violations de cette condition (Hull 1993).

5.1.1 OneT2OneS : apprentissage et test sur la MAP

Dans le tableau 5.1, nous présentons les résultats détaillés à l'aide des collections d'apprentissage décuplées. Le niveau de référence correspond au meilleur système sans tenir compte de

Expérience	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	
Meilleur système	0.381	0.369	0.381	0.392	0.378	
OneT2OneS	0.544*	0.533*	0.557*	0.565*	0.540*	
Expérience	Exp 6	Exp 7	Exp 8	Exp 9	Exp 10	Moyenne (sur les 10)
Meilleur système	0.381	0.383	0.375	0.387	0.384	0.381
OneT2OneS	0.544*	0.555*	0.540*	0.541*	0.597*	0.552 (+44%)

* indique une différence significative (p<0.05 selon le t-test apparié de Student)

TABLE 5.1 – MAP de l'apprentissage de OneT2OneS sur TREC-7 ad hoc

l'apprentissage ; il s'agit du meilleur système qui participe officiellement à TREC-7 (deuxième ligne). Il n'est pas surprenant que OneT2OneS (troisième ligne) surpasse le meilleur système sur les données d'apprentissage, puisque le principe de l'apprentissage est de choisir le meilleur système pour chaque besoin (le meilleur système en moyenne n'est pas nécessairement le meilleur pour tous les besoins).

Lorsque l'on considère les collections de tests (tableau 5.2), OneT2OneS surpasse encore de loin le meilleur système.

Apprendre le meilleur système à utiliser pour un besoin améliore la MAP d'environ 21% en moyenne.

Expérience	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	
Meilleur système	0.387	0.416	0.391	0.393	0.417	
OneT2OneS	0.468*	0.521*	0.464	0.443	0.505*	
Expérience	Exp 6	Exp 7	Exp 8	Exp 9	Exp 10	Moyenne (sur les 10)
Meilleur système	0.397	0.379	0.405	0.414	0.384	0.398
OneT2OneS	0.495*	0.474*	0.489*	0.504*	0.445	0.481 (+21%)

* indique une différence significative (p<0.05 selon le t-test apparié de Student)

TABLE 5.2 – MAP du test de OneT2OneS sur TREC-7 ad hoc

Expérience	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	
Meilleur système	0.387	0.416	0.391	0.393	0.417	
OneT2ClusterS_30	0.460	0.513	0.474	0.445	0.469	
OneT2ClusterS_*	0.454	0.503*	0.463	0.436	0.479	
Nombre de groupes	11	14	10	11	9	
Expérience	Exp 6	Exp 7	Exp 8	Exp 9	Exp 10	Moyenne (sur les 10)
Meilleur système	0.397	0.379	0.405	0.414	0.384	0.398
OneT2ClusterS_30	0.491*	0.486*	0.498*	0.501*	0.448	0.478 (+20%)
OneT2ClusterS_*	0.458	0.455*	0.470	0.446	0.445	0.461 (+15%)
Nombre de groupes	11	12	12	13	15	

* indique une différence significative ($p<0.05$ selon le t-test apparié de Student)

TABLE 5.3 – MAP du test de OneT2ClusterS sur TREC-7 ad hoc

5.1.2 OneT2ClusterS : apprentissage et test sur la MAP

Dans le tableau 5.3, nous présentons les résultats détaillés que nous avons obtenus en utilisant les collections d'essais décuplées (l'apprentissage n'est pas présenté ici). Dans la deuxième ligne, nous indiquons la MAP du meilleur système participant à TREC-7. La troisième ligne (OneT2ClusterS_30) correspond à la combinaison obtenue pour notre système lorsque le nombre de groupes de systèmes est fixé à 30 quel que soit la sous-collection (1/3 du nombre total de systèmes). Dans la quatrième ligne (OneT2ClusterS_*), le nombre de classes de systèmes n'est pas défini à l'avance. Nous considérons plutôt la distance entre les groupes de systèmes dans la phase d'apprentissage et réduisons le dendogramme au niveau où une diminution importante de cette distance est observée. Le nombre de groupes de systèmes résultant de ce processus est compris entre 9 et 15, en fonction des collections. La cinquième ligne indique ce nombre de groupes de systèmes qui ont été définis lors de l'apprentissage suivant le processus que nous venons d'expliquer. En moyenne, OneT2ClusterS améliore la MAP, soit en considérant l'examen d'un relativement grand nombre de groupes ou soit en envisageant un plus petit nombre de groupes qui dépend d'avantage de la structure de la classification hiérarchique. Dans le premier cas, la MAP est améliorée de 20% par rapport au meilleur système, et de 15% dans le dernier cas. En dépit de l'importance relative de l'amélioration, la différence n'est pas statistiquement significative ($p > 0,05$ en appliquant le t-test apparié de Student) pour toutes les sous collections. Cependant, il est important de noter que, dans tous les cas, la méthode améliore la MAP. Nous considérons ensuite la méthode ClusterT2ClusterS afin d'avoir un examen plus approfondi des groupes de besoins.

5.1.3 ClusterT2ClusterS : apprentissage et test sur la MAP

Dans cette section, nous reprenons les résultats de la méthode OneT2ClusterS et les exprimons en ce qui concerne les groupes de besoins en information. Comme regroupement, nous considérons dans cette section 3 groupes de besoins qui peuvent être considérés comme des besoins faciles, moyens et difficiles. Nous avons fait ce choix parce que le nombre idéal de groupes de besoins peut différer d'une collection à l'autre. Par exemple, étant donné la distance entre le groupes de besoins comme présentée à la figure 4.8, un premier niveau de coupe pertinent conduirait à 3 groupes, et la prochaine coupe pertinente serait à 4 groupes ; la distance devient ensuite de plus en plus petite jusqu'à une coupe qui conduirait à 7 groupes.

Topics difficiles

Expérience	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	
Meilleur système	0.349	0.378	0.297	0.315	0.324	
ClusterT2ClusterS	0.364	0.429*	0.364*	0.329	0.349	
Expérience	Exp 6	Exp 7	Exp 8	Exp 9	Exp 10	Moyenne (sur les 10)
Meilleur système	0.333	0.303	0.348	0.337	0.320	0.330
ClusterT2ClusterS	0.312	0.386*	0.390	0.353	0.323	0.360 (+9%)

Topics faciles

Expérience	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	
Meilleur système	0.592	0.578	0.618	0.511	0.740	
ClusterT2ClusterS	0.686	0.668	0.679	0.664*	0.844	
Expérience	Exp 6	Exp 7	Exp 8	Exp 9	Exp 10	Moyenne (sur les 10)
Meilleur système	0.560	0.631	0.779	0.697	0.571	0.628 (+11%)
ClusterT2ClusterS	0.644	0.599	0.800	0.712	0.676*	0.695

Topics moyens

Expérience	Exp 1	Exp 2	Exp 3	Exp 4	Exp 5	
Meilleur système	0.370	0.403	0.470	0.469	0.445	
ClusterT2ClusterS	0.487*	0.526*	0.551*	0.522	0.540*	
Expérience	Exp 6	Exp 7	Exp 8	Exp 9	Exp 10	Moyenne (sur les 10)
Meilleur système	0.397	0.400	0.428	0.439	0.376	0.420
ClusterT2ClusterS	0.524*	0.501*	0.559*	0.497	0.479*	0.519 (+24%)

* indique une différence significative ($p<0.05$ selon le t-test apparié de Student)
L'amélioration (%) est calculée par comparaison au meilleur système sur les topics du groupe considéré, par exemple 24% est l'amélioration de 0.420 à 0.519.

TABLE 5.4 – MAP du test de ClusterT2ClusterS sur TREC-7 ad hoc

Choisir 3 groupes de besoins a du sens, de même que 4. En effet, pour chacune des 10 collections, 3 ou 4 groupes a du sens en ce qui concerne la distance entre les groupes. Le tableau 5.4 indique les résultats, avec chaque groupe de besoins présenté séparément. Pour chaque groupe, deux lignes sont proposées. La première correspond au meilleur système qui participe à TREC-7, considérant les besoins du groupe étudié. La seconde ligne correspond aux résultats que nous avons obtenus en utilisant notre méthode sur les mêmes besoins. Les meilleures améliorations ne sont pas obtenues sur des besoins difficiles ou faciles. La MAP améliore plus sur les besoins entre les deux (+24%, statistiquement significative).

5.1.4 Discussion

Méthode	MAP
Meilleur système	0.398
OneT2OneS (test)	0.481 (+21%)
OneT2ClusterS (test)—30 groupes	0.478 (+20%)
OneT2ClusterS (test)—12 groupes (average)	0.461 (+15%)

TABLE 5.5 – Résumé des résultats obtenus par les trois méthodes en phase de test

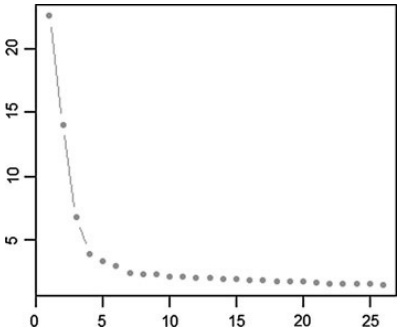

FIGURE 4.8 – Rappel : exemple de distance entre les groupes de besoins en information.

Nos résultats sont difficiles à comparer aux résultats publiés notamment dans les domaines de fusion des données et des méthodes de réordonnancement automatique comme *learning to rank*.

- Les fonctions comme CombMNZ considèrent deux listes de documents différentes qu'elles fusionnent avant d'évaluer la liste fusionnée résultante. Ces fonctions (*CombMNZ-like*) diffèrent essentiellement dans la façon dont les documents sont fusionnés (examen des rangs des documents ou des scores des documents, donnant plus de poids aux documents qui sont récupérés dans les deux listes, etc.) Les techniques de fusion peuvent être appliquées à plus de deux, ou même à l'ensemble des listes classées de documents qui sont disponibles. Nous pourrions donc comparer notre métasystème à l'application de CombMNZ aux systèmes. Cependant, une grande différence avec notre approche est l'absence de phase d'apprentissage pour CombMNZ. Une phase d'apprentissage pourrait être utile si un des objectifs était d'apprendre les systèmes à fusionner. À notre connaissance, il n'existe pas de publication qui rapporte de telles méthodes et résultats. Cependant, pour donner une idée, si nous fusionnons deux exécutions de TREC-7 en utilisant CombMNZ, et en ne gardant que la meilleure paire fusionnée, la MAP est de 0,49 (meilleur système seul : 0,37). Notez que cela peut être considéré comme une phase d'apprentissage utilisant l'ensemble des documents puisque cela implique de connaître les systèmes à fusionner pour obtenir le meilleur résultat. Le métasystème présenté dans nos travaux obtient 0,55 durant l'apprentissage (et 0,48 dans la phase de test), mais sur un sous ensemble de documents ;
- Dans *learning to rank* (Cao 2007), comme dans notre approche, il y a une phase d'apprentissage et une phase de test. Cependant, dans *learning to rank*, la tâche est d'apprendre la fonction de classement à partir de détails donnés sur les besoins et sur l'ordonnancement idéal des listes de documents. La fonction de classement est conçue pour être la même quel que soit le besoin. Notre hypothèse est différente puisque nous concevons une méthode qui est dépendante des besoins.

Dans la suite de ce chapitre, nous nous intéressons à l'application des méthodes présentées en 4.3 et nous étudions l'impact du nombre de systèmes sur l'amélioration des résultats.

5.2 Learning to Choose pour traiter les requêtes répétées

Dans cette analyse nous considérons que les méthodes présentées dans le chapitre 4 sont une unique méthode avec différents paramètres. Nous appelons cette méthode L2C.

Aussi, plutôt que de n'utiliser que les données officielles issues des campagnes d'évaluation TREC, nous considérons également des exécutions personnalisées (ou configurations) créées à l'aide de l'outil Terrier comme expliqué en section 2.3.

Utiliser un grand nombre de systèmes (ou de configurations) différents n'est pas un concept acceptable pour des cas d'utilisation réels. En effet, nous avons rarement à disposition plus de quelques systèmes différents pour traiter les requêtes et potentiellement obtenir des résultats diversifiés. Il est donc nécessaire de choisir un ensemble restreint de configurations de systèmes.

La section 5.2.1 détaille la méthode de sélection d'un sous-ensemble de systèmes mise au point pour cette étude.

5.2.1 Définir le jeu de configuration de systèmes

Principe. L'ensemble des configurations de systèmes doit être composé de configurations qui fonctionnent bien en moyenne sur les requêtes. Cependant, comme nous l'avons dit précédemment, la variabilité des systèmes peut être importante. Pour cette raison, nous ajoutons une contrainte supplémentaire : pour être conservée une configuration devrait obtenir de meilleurs résultats que la moyenne sur au moins un sous-ensemble de requêtes. En outre, il n'est pas obligatoire qu'une requête donnée soit bien traitée par plusieurs configurations sélectionnées ; une seule étant suffisante pour garantir la satisfaction des utilisateurs. En fait, il n'est pas nécessaire d'envisager toutes les configurations possibles ; ce qui est nécessaire est de faire en sorte que chaque requête soit bien exécutée par au moins une configuration ou un système disponible. Compte tenu de l'exemple donné dans le tableau 5.6, nous tenons à sélectionner la configuration du système S_1 ou S_2 pour assurer que la requête Q_1 est bien traitée par au moins une configuration. Il en est de même pour la configuration S_3 ou S_4 pour la requête Q_2.

TABLE 5.6 – Illustration - Performances de 4 systèmes sur 2 requêtes.

	S_1	S_2	S_3	S_4
Q_1	0.8	0.9	0.2	0.1
Q_2	0.2	0.1	0.9	0.8

Sélection des configurations de SRI. Pour commencer, nous disposons de différentes configurations de SRI et de leur performance sur l'ensemble des requêtes traitées sur l'ensemble des documents. Comme nous voulions effectuer une analyse assez large, dans nos expériences, nous considérons 100 configurations systèmes initiales. Cependant, l'idée est de décider quelles configurations sont les plus importantes. Pour définir les configurations à garder, nous sélectionnons de manière itérative la configuration la plus intéressante de la façon suivante :

1. Pour chaque requête, nous calculons la valeur de la mesure de performance correspondant au plus haut centile ; la configuration du système qui obtient cette valeur devient un candidat. Par exemple, si l'on considère la précision moyenne (MAP) comme mesure de la performance, nous calculons la valeur MAP map_{001} dont 1% des valeurs de MAP obtenues par les systèmes sera plus élevé que map_{001} et donc 99% des valeurs de MAP seront inférieures à map_{001} pour cette requête. Considérons que la requête traitée est facile et que nous disposons de 100 configurations de systèmes, le meilleur système S_{best} a une MAP de $0,85$, map_{001} pour cette requête est défini sur $0,85$; S_{best} devient un candidat ;

2. Les configurations sélectionnées à l'étape 1 constituent l'ensemble des candidats à cette première itération ;
3. Pour chaque candidat, nous calculons le nombre de fois où il a été sélectionné ; c'est-à-dire le nombre de requêtes qui ont sélectionné chaque configuration candidate. Nous conservons la configuration qui est la plus fréquente. De plus, pour garder la configuration, il est nécessaire que sa fréquence soit supérieure à un seuil que l'on se fixe. Par exemple, S_{best} a été sélectionné à l'étape 1 par 10 requêtes et est la configuration la plus sélectionnée, étant donné un seuil de 6 requêtes minimum, S_{best} est définitivement maintenue ;
4. Les requêtes qui ont sélectionné cette configuration sont retirées de l'ensemble des requêtes. Compte tenu de notre exemple, les 10 requêtes qui choisissent la configuration S_{best} sont supprimées.
5. Nous retournons à l'étape 1 en envisagent toutes les configurations et les requêtes restantes (dans notre exemple en utilisant $Nombre\,de\,requêtes - 10$).

S'il n'y a pas de configuration sélectionnée, le processus est réitéré avec le deuxième centile (la valeur de la performance pour laquelle 98% des valeurs sont moins élevées que lui), puis le troisième centile, ... Si moins de T requêtes restent non apariée à une configuration, elles doivent toutes être sélectionnées en une seule fois (le citère du centile est relâché). Cette règle est utilisée pour éviter de sélectionner les requêtes les plus difficiles une par une et donc de faire du cas par cas tandis qu'inévitablement le centile considéré continue de diminuer.

5.2.2 Contexte d'évaluation

Collection d'évaluation. L'évaluation de notre méthode dans un contexte réel implique d'avoir accès à des données réelles (requêtes répétées, documents, jugement de pertinence) ; une telle collection libre d'accès n'existe pas à notre connaissance. Pour cette raison, nous avons simulé l'environnement par l'évaluation de notre méthode sur les données de TREC-7 et TREC-8 ad hoc (*cf.* chapitre 2).

Nous avons créé différentes exécutions avec la plateforme Terrier (Ounis 2006) en faisant varier différents paramètres (*cf.* section 2.3). Pour être plus réaliste en matière d'utilisabilité, nous avons considéré une seule indexation (qui est la partie du processus de RI la plus exigeante en terme d'utilisations des ressources) ; et donc les paramètres d'indexations sont fixés une fois. Ces exécutions correspondent à des configurations de système.

Mesure d'évaluation. Pour les mêmes raisons qu'au chapitre 4 nous ne considérons que la mesure MAP.

Apprentissage et validation croisée. Nous reprenons la structure d'évaluation utilisée en 5.1, soit 10 partitionnements de la collection de documents avec pour chacune d'elles 2/3 des documents utilisés pour l'apprentissage et 1/3 pour le test. Les résultats sont moyennés sur les 10 collections.

Références. La référence est la meilleure configuration sur l'ensemble des requêtes. La moyenne des 10 références forme la référence globale ; cela est nécessaire puisque le partitionnement des documents varie et donc la MAP varie ainsi en fonction du tirage au sort.

5.2.3 Sélection des configurations de SRI

5.2.3.1 Étude préliminaire pour sélectionner les configurations.

Terrier permet de choisir différents modules pour les étapes du processus de RI (voir 2.3 pour le détail complet) :

- Indexation (algorithme de radicalisation, taille des blocs, documents vides ignorés ou non) ;
- Appariement (modèle de stemming, champs du besoin à utiliser, termes à bas idf ignorés ou non) ;
- expansion de requêtes (modèle d'expansion, nombre de termes à ajouter à la requête initiale, nombre de documents de haut rang à considérer pour l'expansion, le nombre de documents de haut rang dans lequel un terme doit apparaître pour être utilisé pendant l'expansion).

L'indexation est coûteuse en termes de temps de calcul, d'espace disque et de mises à jour lors de la création d'un flux de RI. Pour cette raison, nous considérons un seul index et les seuls paramètres que nous faisons variés sont ceux d'appariements et d'expansion de requête. La phase d'indexation utilise la liste de mots vides par défaut de Terrier, l'algorithme de racinisation de Porter, une taille de bloc de 1 et ne tient pas compte des documents vides.

En fait, utiliser des listes de mots vides différentes n'a pas un impact significatif sur les résultats, mais en utiliser une (plutôt qu'aucune) améliore significativement les résultats (Dolamic 2010). Fuller et al. (Fuller 1998) a montré que l'algorithme de Porter est la plus précise en matière de racinisation. Aussi Compaoré et al. (Compaoré 2011a) ont montré que la taille des blocs différents de 1 et la prise en compte des documents vides ne changent pas les résultats de manière significative.

Concrètement, nous avons pris en compte de nombreuses configurations, mais en avons gardé seulement 100 qui obtiennent une MAP de plus de 0,2 pour au moins 50% des besoins. Ce seuil de 0,2 de MAP est la moitié du meilleur système utilisée dans (Bigot 2011) et est fixé de manière à ce que les performances des configurations de systèmes sélectionnés soient assez bonnes.

5.2.3.2 Impact du nombre de configurations sur les performances.

En conditions réelles, utiliser 100 configurations différentes ou 100 systèmes peut être coûteux. Pour cette raison, nous étudions l'impact du nombre de systèmes à prendre en compte. Nous considérons différents seuils de sélection (10, 20, 30 ... 100 configurations ou systèmes). Une solution pourrait être de les choisir au hasard. Cependant, cette solution ne profiterait pas de la variabilité des systèmes. Pour cette raison, nous préférons choisir des configurations de système qui se comportent différemment sur les besoins en information. Pour ce faire, une classification hybride (classification hiérarchique combinée à K-moyenne) est appliquée de manière à ce que les configurations du système de chaque groupe soient proches en termes de performances (MAP) pour le traitement des besoins sur l'ensemble des documents d'apprentissage. Pour chaque groupes, la meilleure configuration est sélectionnée pour être utilisée dans le processus de L2C. Ce procédé assure que les configurations choisies sont suffisamment différentes. Nous considérons alors plusieurs seuils : nous analysons les résultats lorsque 100% des configurations sont utilisées (pas de regroupement), quand 90% des configurations intiales sont utilisées (ce qui signifie que nous regroupons les configurations en 90 groupes), 80 %. .. jusqu'à

10 % (ce qui signifie que seulement 10 configurations sont utilisées, correspondant à 10 groupes différents ou 10 profils de configuration).

La figure 5.1 montre les résultats de cette expérience. Les lignes grises correspondent à la phase d'entraînement, et les noires à la phase de test. Les lignes en tirets longs sont les références, et les lignes pleines avec des points correspondent à la moyenne sur les dix sous-collections, comme expliqué dans 5.2.2. Enfin, les lignes en pointillé représentent le résultat d'une sélection aléatoire. Notez qu'une sélection aléatoire conduit à l'espérance de la MAP et donc à la MAP moyenne obtenue par les systèmes.

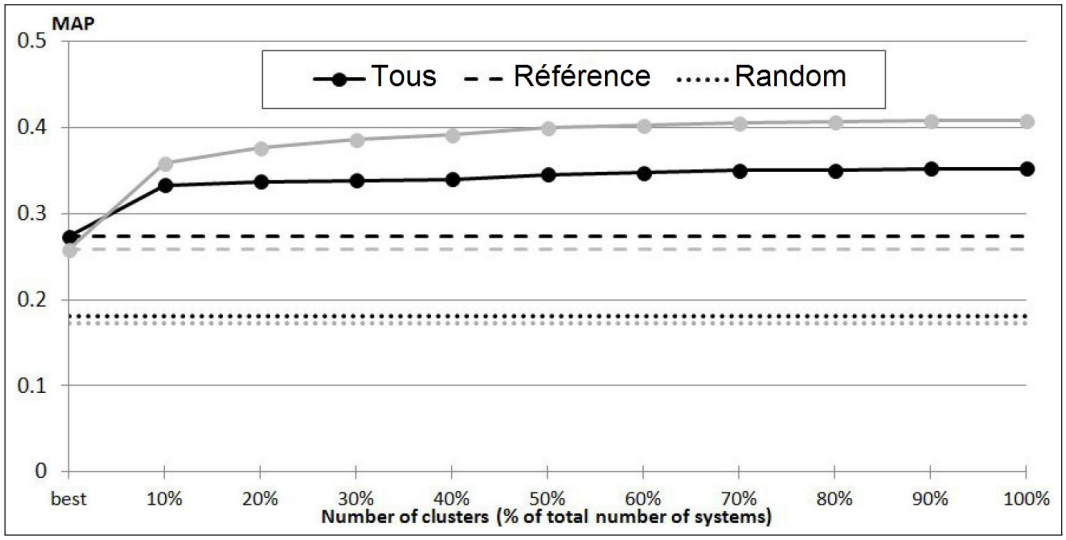

FIGURE 5.1 – Analyse du nombre de configurations différentes à considérer pour L2C.

Les résultats présentés dans la figure 5.1 montrent que l'utilisation de seulement 10 configurations différentes de système améliore les résultats de 22%. L'utilisation de plus de 10 configurations candidates améliore très peu les résultats.

De cette étude préliminaire, nous concluons que 10 configurations sont suffisantes pour améliorer de façon significative la performance mais une valeur inférieure peut également suffire.

Dans ce qui suit, nous sélectionnons un ensemble minimal de configurations de système (inférieure à 10) afin de faire une analyse plus poussée du comportement de la technique de sélection L2C sur un petit échantillon de configurations.

5.2.3.3 Ensemble minimal de configuration de systèmes.

Pour sélectionner un ensemble minimal de configurations, nous appliquons la méthode présentée dans 5.2.1 pour la série précédente de 100 systèmes.

Dans la figure 5.2, la ligne noire représente le nombre de configurations sélectionnées et la ligne grise est la MAP moyenne ; la MAP moyenne considère la moyenne de la MAP des systèmes sélectionnés seulement pour les besoins appariés lors de la sélection de l'ensemble minimal (section 5.2.3). La figure 5.2 montre que plus le nombre de besoins (T) est petit, plus spécifique est la sélection, et meilleurs sont les résultats. Cependant, nous ne voulons pas être

trop précis, car cela conduit à la sélection de trop nombreuses configurations ; pour cette raison, nous considérons 10 configurations ou moins.

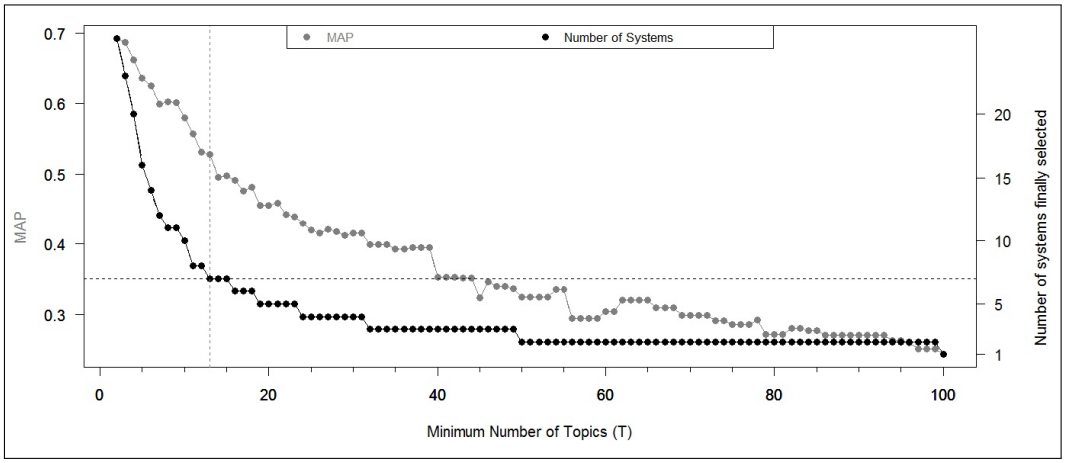

FIGURE 5.2 – Nombre de configurations sélectionnées (ordonnées à droite) et performance (axe de gauche) selon T.

La figure 5.3 affiche la moyenne de la MAP en fonction du nombre de configurations de système ; 10 étant le nombre maximum de configurations de systèmes que nous permettons d'être sélectionnés par l'algorithme. Pour faire un compromis entre efficacité et nombre de configurations sélectionnées, nous choisissons 7 configurations possibles.

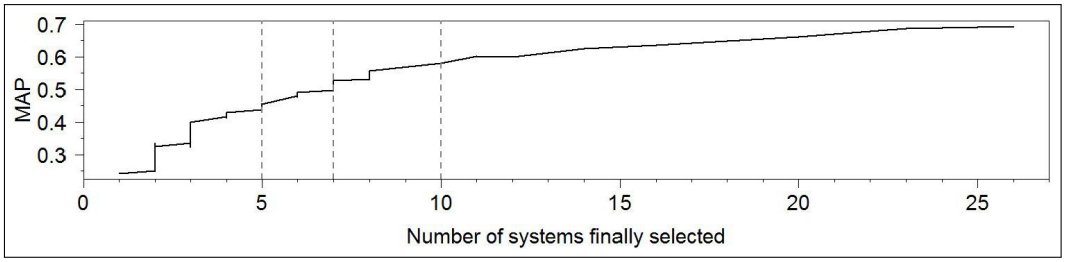

FIGURE 5.3 – Performances moyennes selon le nombre de configurations sélectionnées.

Cela correspond à 3 possibilités différentes de T (voir les points traversés par la ligne horizontale pointillée sur la figure 5.2) et la meilleure MAP (0.528) est obtenue pour $T = 13$ (ligne pointillée verticale). L'ensemble de ces 7 configurations est utilisé pour évaluer la méthode L2C dans la section 5.2.4.

5.2.4 Evalution de L2C dans un contexte proche d'un cas réel

Dans cette section, nous évaluons L2C sur un petit ensemble de configurations de systèmes. Les besoins sont considérés comme nouveaux au cours de la phase d'apprentissage et ils sont

considérés comme des requêtes répétées au cours de la phase de test. Pour éviter l'effet de hasard qui pourrait conduire à une expérience chanceuse ou malchanceuse, nous procédons à une validation croisée sur dix divisions différentes de la collection de documents comme expliqué dans la sec tion5.2.2.

5.2.4.1 Difficulté des besoins

Comme nous menons 10 expériences, le nombre de groupes de besoins peut changer d'une expérience à l'autre. Pour être en mesure de fusionner les résultats, nous devons garder le même nombre de groupes issus de la classification pour les 10 expériences. Chaque groupe est caractérisé par une étiquette de difficulté selon la MAP obtenues par les systèmes. Parmi les 10 expériences, nous avons observé que 4 à 6 niveaux de difficulté pour les besoins seraient optimaux pour la méthode L2C. Pour fixer le nombre de niveaux de difficulté à envisager, nous procédons à une CAH sur les données calculées avec la collection complète de documents pour conclure que 5 niveaux de difficulté est un choix optimal comme le montre la figure 5.4.

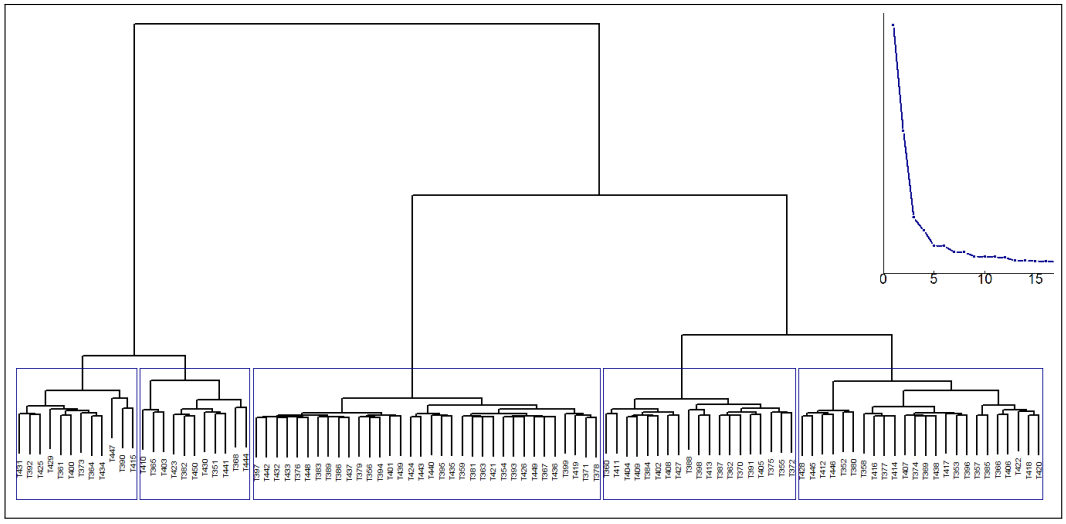

FIGURE 5.4 – CAH des besoins pour les 7 configurations retenues

5.2.4.2 Résultats de la sélection.

Comme le nombre de besoins dans un niveau de difficulté varie au cours des 10 expériences, la MAP moyenne globale par groupe est pondérée par le nombre de besoins inclus dans le niveau considéré. Ceci est réalisé à la fois pour le métasystème L2C et la référence. La figure 5.5 affiche la moyenne de la MAP par niveau de difficulté de besoins au cours de la phase de test. Les besoins sur lesquels les configurations ont obtenu de mauvaises performances sont qualifiés de "difficiles", ceux sur lesquels ils sont moins mauvais sont appelés "moyennement difficile" et ainsi de suite pour le niveau «moyennement facile", niveau "facile" et enfin le niveau " très facile" pour lesquels les systèmes obtiennent de très bons résultats.

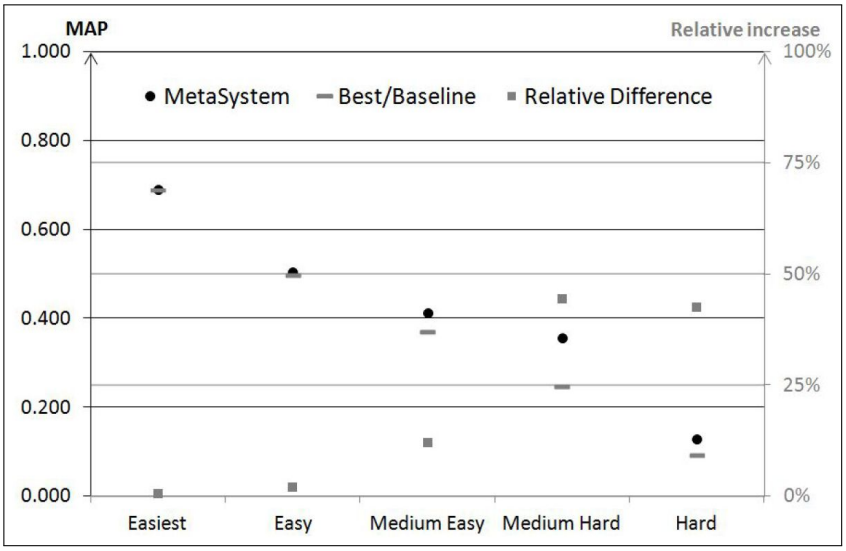

FIGURE 5.5 – Résultats de la sélection selon la difficulté du besoin.

Les points noirs correspondent à la MAP du métasystème et les tirets gris à la MAP de référence et sont lus sur l'axe de gauche (avec les lignes directrices noirs). Les carrés gris montrent la différence relative entre le métasystème et la référence et sont lus sur l'axe de droite (avec les lignes directrices grises).

Nous montrons que les performances sont peu améliorées sur les besoins faciles ; ce qui peut s'expliquer par le fait qu'il est difficile d'améliorer les performances du système quand elles sont déjà très élevées. Au moins, les performances ne sont pas dégradées. La performance moyenne sur les besoins faciles est améliorée de 12% ; selon le t-test, l'augmentation est significative pour 6 expériences sur 10.

La meilleure amélioration est observée pour les besoins moyennement difficiles avec une augmentation de 44 % (augmentation qui est significative pour 9 expériences). Enfin, les besoins les plus difficiles sont améliorés de 43%. Cependant l'augmentation absolue est petite (0,04) mais reste siginificative pour 8 expériences.

5.3 Conclusion

Dans ce travail, nous avons d'abord montrer qu'il est possible d'utiliser à bon escient la variabilité qui existe au sein des systèmes et la variabilité intrinsèque des besoins en information.

Les méthodes de sélection proposées réussissent à améliorer signficativement les performances sur l'ensemble des besoins (20% d'amélioration de la MAP). Nous avons également montré que cette amélioration varie selon les groupes de besoins considérés et qu'il n'est pas nécessaire de considérer les systèmes individuellements mais que des profils peuvent être créés afin d'éviter les systèmes qui réussiraient par chance à être performant sur un besoin.

Ensuite, nous avons analysé le comportement de la méthode L2C dans un cadre d'application réel : peu de systèmes sont utilisables et les requêtes sont répétées. Nous constatons qu'un

ensemble constitué de peu de systèmes est nécessaire pour observer une amélioration des résultats. Une méthode de sélection d'un sous-ensemble de système a donc été proposée et utilisée. Les analyses montrent qu'avec peu de systèmes, il est possible d'améliorer significativement les performances à partir du moment où il existe une certaine variabilité au sein de l'ensemble de systèmes de départ, surtout sur les requêtes dîtes difficiles avec un gain pouvant aller jusqu'à 44%.

Les méthodes de sélections proposées au chapitre 4 ainsi que la première évaluation menée dans ce chapitre en section 5.1 ont été publiées dans le journal Information Retrieval Journal (Bigot 2011). La seconde évaluation présentée en section 5.2, quant à elle, a reçu le prix du meilleur papier étudiant à la conférence Spanish Conference on Information Retrieval et aura prochainement une version étendue dans le journal associé.

Conclusion et perspectives

Une première contribution de ces travaux de thèse a été la création des données nécessaires aux expérimentations. Ces exécutions personnalisées créées à l'aide de Terrier ont été mises à disposition et ont permi de mener de nouvelles expérimentations pour d'autres publications. Aussi, les scripts de génération des fichiers de configuration de Terrier ont été mis à disposition pour lancer d'autres tâches sur les serveurs de calculs. Cela a notament permi de mener une étude sur les modèles de clic dans les moteurs géoréférencés (Laporte 2012).

La seconde contribution scientifique de ces travaux a porté sur la sélection de Système de Recherche d'Information basée sur leurs rangs plutôt que sur les mesures de performances. Nous avons montré que connaître les rangs des systèmes plutôt que l'ensemble de leur performance sur les requêtes est parfois sufffisant. Nous avons étudié deux aspects pour la sélection des systèmes : améliorer la performance des résultats et améliorer leur robustesse. L'évaluation montre qu'il est possible d'améliorer les performances lorsque cet aspect est privilégié. Nous montrons également que la robustesse peut être facilement améliorée (jusqu'à 68% de gain) tout en conservant de bonnes performances. Ces travaux sont publiés dans (Bigot 2014a).

Enfin la troisième contribution scientifique, contribution majeure de ces travaux, porte sur une analyse poussées des SRI et des besoins en information. Cette étude nous permet de proposer différentes variantes d'une méthode de sélection que nous apppellerons par la suite Learning to Choose (L2C). La première partie de ces travaux est publiée dans (Bigot 2011) et la seconde partie dans (Bigot 2014b).

Lorsque nous analysons les systèmes, nous montrons que les systèmes peuvent être regroupés en fonction de la façon dont ils se comportent sur les besoins, et les besoins peuvent être regroupés en fonction de la façon dont les systèmes les exécutent : les systèmes qui réussissent de la même manière sur les mêmes besoins seront groupés ensemble et vice versa. Nous avons observé que les meilleurs systèmes se regroupent et que les systèmes moins bons se regroupent aussi. La même chose se produit pour les besoins : les faciles se regroupent ainsi que les plus difficiles. Les autres groupes de systèmes sont moins liés à leur efficacité alors que les autres groupes de besoins le sont.

Nous confirmons l'hypothèse que les systèmes les plus étroitement corrélés sont différentes versions d'un même système. La variabilité de leurs performances en fonction des besoins est faible par rapport à la variabilité entre différents systèmes. En effet, généralement parlant, les participants TREC présentent plusieurs exécutions (même tâche, même année), et celles-ci ne diffèrent pas fortement. Ceci pourrait suggérer que, plutôt que d'essayer de régler certains paramètres d'un unique système, nous devrions envisager des techniques radicalement différentes pour traiter certains types de besoins.

En utilisant la CAH, nous montrons que certains besoins sont fortement corrélés avec les meilleurs systèmes ce qui signifie que les systèmes se comportent différemment pour ces besoins contrairement aux autres systèmes. Encore une fois en tenant compte des principaux facteurs, nous montrons que certains systèmes qui réussissent sont orthogonaux d'une certaine façon : ils ne réussissent pas pour la même raison ; c'est-à-dire sur des besoins différents. Sur la base de ces résultats, nous proposons une nouvelle façon de combiner les systèmes, dans approche besoin dépendante des requêtes ou des besoins d'information.

D'après la littérature dans le domaine de la fusion de données (Croft 2000), (Wu 2006), nous avons fait l'hypothèse que les systèmes à combiner doivent être indépendants. Cependant, contrairement aux techniques de fusion habituelles, l'indépendance n'est pas basée sur les listes classées de documents, ni sur le chevauchement des documents pertinents et non pertinents, mais sur le fait qu'ils se comportent différemment sur un sous-ensemble de besoins (système A est bon pour le besoin 1 et mauvais pour le besoin 2 alors que le système B obtient des résultats opposés). Nous favorisons un moyen de combiner des systèmes complémentaires, en fonction de leurs profils et en fonction de leur efficacité à traiter des besoins sur un ensemble de documents d'apprentissage.

Pour chaque besoin, nous apprenons quel est le meilleur groupe de systèmes est donc le système qui sera utilisé pour traiter le besoin en information. À leur tour, les besoins sont regroupés (CAH et K-moyennes) ; nous pouvons alors voir comment la méthode améliore les résultats de chaque groupe de besoins. Nous présentons trois versions de la même méthode. Dans la méthode OneT2OneS, nous considérons autant de groupes de systèmes qu'il y a de systèmes (un système par groupe) et un besoin unique par groupe. Dans OneT2ClusterS, il y a moins de groupes de systèmes que de systèmes et également un seul besoin par groupe. Enfin, dans le procédé de ClusterT2ClusterS, il y a moins de groupes de systèmes que de systèmes (comme dans les OneT2ClusterS) mais plus d'un besoin par groupe. L'évaluation montre que :

- la MAP peut être améliorée de 20% après une phase d'apprentissage ;
- prendre en compte un nombre restreint de groupes de systèmes ne diminue pas de façon importante les performances ;
- l'amélioration est meilleure sur des besoins de difficulté moyenne, pas sur les plus difficiles (comme on pourrait le penser intuitivement), ni sur les plus faciles.

Bien que cette méthode ne s'applique pour le moment qu'aux requêtes qui sont connues à l'avance, son utilisation sur les requêtes répétées peut s'avérer pertinent.

Plusieurs études (Smyth 2004), (Sanderson 2007), (Teevan 2007) et (Tyler 2010) démontrent que la répétition se produit fréquemment dans des applications réelles. En outre, certaines études ont montré que la répétition dans certains cas est prévisible (Zhang 2009). Pour ces raisons, le fait que les requêtes soient nécessaires à l'apprentissage de la méthode à appliquer n'est pas une limitation de ce travail. En outre, dans notre cadre expérimental, nous avons considéré le jugement de pertinence dans la phase d'apprentissage ; dans les applications réelles, il est possible de considérer les actions des utilisateurs pour induire un ensemble de documents pertinents.

Aussi nous avons montré que les variantes d'un même système conduisent à une efficacité comparable et que la méthode doit prendre en compte différents systèmes mais pas nécessairement un grand nombre.

Les trois variantes de la méthode peuvent être considérées comme une seule méthode que nous appelons Learning to Choose (L2C) avec des paramètres différents. Pour chaque groupe de systèmes, le système parangon qui traitera les besoins qui entrent dans son champ d'application

est calculé.

De plus utiliser plus de dix systèmes différents pour une technique de fusion n'est pas réaliste. Nous avons donc mis en place une méthode facile à reproduire pour sélectionner un sous-ensemble optimal de systèmes. Les systèmes sont choisis sur la base leurs performances, s'ils appartiennent ou non au plus haut centile pour un nombre défini de besoins. Les besoins appariés sont successivement supprimés de la sélection jusqu'à ce qu'aucun besoin ne reste. L'analyse de cette méthode de sélection conduit à sélectionner sept systèmes afin d'équilibrer entre les performances et les conditions réelles. Même si les derniers besoins appariés sont mal traités par les systèmes choisis, en raison de la méthode de sélection elle-même, l'évaluation n'est pas dégradée.

Learning to Choose est ensuite évaluée sur les sept systèmes et des groupes de besoins sont définis selon les performances des systèmes : plus les performances des systèmes sont bonnes, plus les besoins sont faciles. L'évaluation montre que, sur la base d'un ensemble donné de systèmes, une méthode de fusion est capable d'améliorer les performances de restitution de 12% sur les besoins de difficulté moyenne jusqu'à 44 % sur des besoins difficiles. Comme les résultats du processus de recherche d'informations ne sont pas stockés, la méthode est capable de traiter les collections dynamiques.

Dans cette étude, des groupes de requêtes sont définis après que les requêtes ont été traitées au moins une fois. Ceci est possible dans le cadre de requêtes répétées. Cependant, un tel procédé n'est pas en mesure de traiter les demandes soumises pour la première fois. En fait, les travaux futurs devraient se concentrer sur l'application des techniques de prévision de la difficulté. Ces techniques peuvent être basées sur des prédicteurs linguistiques (Mothe 2007) ou sur différents prédicteurs tels que l'ambiguïté de la requête (Chifu 2013).

À l'avenir, nous devrions d'avantage tenir compte des caractéristiques du système (méthode d'indexation utilisée, choix de la reformulation de la requête, fonction de classement utilisée, paramètres du modèle, etc. Ensuite, nous devrions analyser en profondeur les paramètres de recherche des systèmes qui sont spécifiques aux documents afin de savoir si nous pouvons extraire des motifs dans la façon dont les systèmes possèdent et corréler cela avec les groupes de systèmes et les méthodes ou les modèles utilisés dans les systèmes. En complément, nous allons enquêter sur la formulation de la requête en profondeur. Considérant TREC par exemple, la plupart des participants tiennent compte de deux parties de la description des besoins en information (le titre et la description du besoin en information) afin de créer les requêtes qui seront envoyés à leur système. Ces requêtes ont différentes caractéristiques ; dans quelle mesure les caractéristiques des besoins sont correlées avec les performances du système est l'une des nouvelles perspectives que nous pourrions étudier.

Bibliographie

Giambattista Amati, Claudio Carpineto et Giovanni Romano. *Query Difficulty, Robustness, and Selective Application of Query Expansion.* Advances in Information Retrieval, pages 127–137, 2004. (Cité en page 1.)

Massih-Reza Amini et Éric Gaussier. Recherche d'information - applications, modèles et algorithmes. Eyrolles, 2013. (Cité en pages 6, 9 et 10.)

Julie Ayter et Cecile Desclaux. Performance analysis of information retrieval systems. Rapport de projet long, INSA Toulouse. IRIT, 2013. (Cité en pages 28 et 38.)

Julie Ayter, Cecile Desclaux, Adrian Chifu, Sébastien Déjean et Josiane Mothe. *Performance Analysis of Information Retrieval Systems (regular paper).* In and, editeur, Spanish Conference on Information Retrieval, Coruna, 19/06/2014-20/06/2014, page (electronic medium), http ://www.springerlink.com/, 2014. Springer-Verlag. (Cité en pages 28 et 38.)

A. Baccini, Sébastien Djean, Josiane Mothe et Laetitia Lafage. *How many performance measures to evaluate Information Retrieval Systems ?* Knowledge and Information Systems, vol. 30, no. 3, pages 693–713, 2011. (Cité en page 43.)

Ricardo Baeza-Yates et Berthier Ribeiro-Neto. Modern information retrieval. Addison-Wesley, 1999. (Cité en page 6.)

Steven M Beitzel, Ophir Frieder, Eric C Jensen, David Grossman, Abdur Chowdhury et Nazli Goharian. *Disproving the fusion hypothesis : an analysis of data fusion via effective information retrieval strategies.* In Proceedings of the 2003 ACM symposium on Applied computing, pages 823–827. ACM, 2003. (Cité en page 72.)

Anthony Bigot, Claude Chrisment, Taoufiq Dkaki, Gilles Hubert et Josiane Mothe. *Fusing different information retrieval systems according to query-topics : a study based on correlation in information retrieval systems and TREC topics.* IR Journal, vol. 14, no. 6, pages 617–648, 2011. (Cité en pages 25, 78, 85, 90 et 91.)

Anthony Bigot. *Adapter les moteurs de recherche aux besoins en information - Prise en compte de la difficulté du besoin (regular paper).* In INFormatique des Organisations et Systemes d'Information et de Decision (INFORSID), Paris, 29/05/13-31/05/13, pages 59–74, http ://www.univ-paris1.fr/, mai 2013. Universit¿ Paris 1. (Cité en page 56.)

Anthony Bigot, Sébastien Déjean et Josiane Mothe. *Choisir la meilleure configuration d'un système de recherche d'information.* Document numérique, Document Numérique 1/2014, vol. Hors-série, 2014. (Cité en pages 56 et 91.)

Anthony Bigot, Sébastien Déjean et Josiane Mothe. *Learning to Choose - Automatic Selection of the Information Retrieval Parameters (regular paper).* In Spanish Conference on Informa-

tion Retrieval, Coruña, 18/06/2014-20/06/2014, page (on line), http ://www.springerlink.com, avril 2014. Springer. (distinction décernée : Best student paper). (Cité en pages 78 et 91.)

L. Breiman, J. Friedman, R. Olshen et C. Stone. Classification and Regression Trees. Wadsworth and Brooks, Monterey, CA, 1984. new edition (**?**) ? (Cité en page 26.)

Peter J. Brown et Gareth J. F. Jones. *Context-aware Retrieval : Exploring a New Environment for Information Retrieval and Information Filtering.* Personal and Ubiquitous Computing, vol. 5, no. 4, pages 253–263, 2001. (Cité en page 25.)

Chris Buckley, Amit Singhal, Mandar Mitra et Gerard Salton. *New retrieval approaches using SMART : TREC 4.* In Proceedings of the Fourth Text REtrieval Conference (TREC-4), pages 25–48, 1995. (Cité en page 10.)

Zhe Cao, Tao Qin, Tie-Yan Liu, Ming-Feng Tsai et Hang Li. *Learning to rank : from pairwise approach to listwise approach.* In Machine Learning, Proceedings of the 24th ICML, Vol. 227 of ACM Int. Conf. Proc. Series, pages 129–136, 2007. (Cité en page 82.)

David Carmel et Elad Yom-Tov. Estimating the query difficulty for information retrieval. Synthesis Lectures on Information Concepts, Retrieval, and Services. Morgan & Claypool (ed.), 2010. (Cité en pages 25 et 57.)

Adrian Chifu. *Prédire la difficulté des requêtes : la combinaison de mesures statistiques et sémantiques.* In CORIA, pages 191–200, 2013. (Cité en pages 57 et 93.)

Adrian Chifu et Josiane Mothe. *Expansion sélective de requêtes par apprentissage (regular paper).* In and, editeur, Conférence francophone en Recherche d'Information et Applications (CORIA), Nancy, France, 19/03/14-21/03/14, pages 231–246, http ://www.loria.fr, mars 2014. LORIA. (Cité en page 1.)

Claude Chrisment, Taoufiq Dkaki, Josiane Mothe, Sandra Poulain et Ludovic Tanguy. *Recherche d information - analyse des résultats de différents systèmes réalisant la même tâche.* Revue des Sciences et Technologies de l'Information, vol. 10, no. 1, pages 31–55, 2005. (Cité en pages 25, 26 et 27.)

Cyril Cleverdon, Jack Mills et Michael Keen. Factors determining the performance of indexing systems, volume 1. ASLIB Cranfield Research Project, 1966. (Cité en page 12.)

Jonathan Compaoré, Sébastien Déjean, Adji Mairam Gueye, Josiane Mothe et Joelson Randriamparany. *Mining Information Retrieval Results : Significant IR parameters.* IARIA, 2011. (Cité en page 85.)

Jonathan Compaoré, Sébastien Déjean, Adji Maïram Gueye, Josiane Mothe et Joelson Randriamparany. *Mining Information Retrieval Results : Significant IR parameters (regular paper).* In Advances in Information Mining and Management, Barcelone, 23/10/2011-28/10/2011, page (electronic medium), http ://www.iaria.org/, octobre 2011. IARIA. (Cité en pages vii, 26, 27, 28 et 29.)

W. Bruce Croft. Combining approaches to information retrieval, chapitre 1, pages 1–36. Kluwer Academic Publishers, 2000. (Cité en pages 62, 72 et 92.)

Bekir Taner Dinçer. *Statistical Principal Components Analysis for Retrieval Experiments : Research Articles.* Journal of the Association for Information Science and Technology, vol. 58, no. 4, pages 560–574, 2007. (Cité en page 26.)

Ljiljana Dolamic et Jacques Savoy. *When stopword lists make the difference.* JASIST, vol. 61, no. 1, pages 200–203, 2010. (Cité en page 85.)

Damien Dudognon. *Diversité et système de recommandation : application à une plateforme de blogs à fort trafic*. Thèse de doctorat, Université Paul Sabatier, Toulouse, France, avril 2014. (Cité en page 1.)

Edward A. Fox et J. A. Shaw. *Combination of multiple searches*. In Proc. TREC-2, 1994. (Cité en page 24.)

Michael Fuller et Justin Zobel. *Conflation-based Comparison of Stemming Algorithms*. In Proceedings of the 3rd Australian Document Computing Symposium, pages 8–13, 1998. (Cité en page 85.)

Donna Harman et Chris Buckley. *SIGIR 2004 workshop : RIA and "where can IR go from here ?"*. SIGIR Forum, vol. 38, no. 2, pages 45–49, 2004. (Cité en pages 25 et 40.)

Donna Harman et Chris Buckley. *Overview of the Reliable Information Access Workshop*. Inf. Retr, vol. 12, no. 6, pages 615–641, 2009. (Cité en pages 1, 25, 40 et 44.)

S. P. Harter. *A probabilistic approach to automatic keyword indexing. Part II : An algorithm for probabilistic indexing.*, 1975. (Cité en page 10.)

David Hull. *Using statistical testing in the evaluation of retrieval experiments*. In Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval, pages 329–338. ACM, 1993. (Cité en page 79.)

A. K. Jain, M. N. Murty et P. J. Flynn. *Data Clustering : A Review*. CSURV : Computing Surveys, vol. 31, no. 3, pages 264–323, 1999. (Cité en page 45.)

Hideo Joho, Jana Urban, Robert Villa, Joemon M. Jose et C. J. van Rijsbergen. *AIR 2006 : First International Workshop on Adaptive Information Retrieval*. SIGIR Forum : Special Interest Group on Information Retrieval Forum, vol. 42, no. 1, pages 63–66, 2008. (Cité en page 25.)

J Khali. Rapport de stage, SID Toulouse. IRIT, 2013. (Cité en page 36.)

Désiré Kompaoré, Josiane Mothe et Ludovic Tanguy. *Combining Indexing Methods and Query Sizes in Information Retrieval in French*. In José Cordeiro et Joaquim Filipe, editeurs, ICEIS 2008 - Proceedings of the Tenth International Conference on Enterprise Information Systems, Volume AIDSS, Barcelona, Spain, June 12-16, 2008, pages 149–154, 2008. (Cité en page 25.)

Léa Laporte, Laurent Candillier, Sébastien Déjean et Josiane Mothe. *Evaluation de la pertinence dans les moteurs de recherche géoréférencés (regular paper)*. In INFormatique des Organisations et Systemes d'Information et de Decision (INFORSID), Montpellier, 29/05/2012-31/05/2012, pages 281–297, http ://inforsid.irit.fr, mai 2012. Association INFORSID. (Cité en page 91.)

Ludovic Lebart, M Piron et A Morineau. Statistique exploratoire multidimensionnelle : visualisations et inférences en fouille de données. Dunod, 2006. (Cité en pages 18 et 45.)

Joon Ho Lee. *Analyses of multiple evidence combination*. In SIGIR '97 : Proceedings of the 20th annual international ACM SIGIR conference on Research and development in Inf. Ret., pages 267–276. ACM Press, 1997. (Cité en page 24.)

David Lillis, Fergus Toolan, Angel Mur, Liu Peng, Rem W. Collier et John Dunnion. *Probability-based fusion of information retrieval result sets*. Artificial Intelligence Review, vol. 25, no. 1-2, pages 179–191, 2006. (Cité en page 24.)

Tie-Yan Liu, Thorsten Joachims, Hang Li et Chengxiang Zhai. *Introduction to special issue on learning to rank for information retrieval.* Inf. Retr, vol. 13, no. 3, pages 197–200, 2010. (Cité en page 72.)

Hongzhi Liu, Zhonghai Wu et D. Frank Hsu. *Combination of Multiple Retrieval Systems Using Rank-Score Function and Cognitive Diversity.* In AINA : Advanced Information Networking and Applications, IEEE 26th International Conference on Advanced Information Networking and Applications, AINA, 2012, Fukuoka, Japan, March 26-29, 2012, pages 167–174. IEEE, 2012. (Cité en page 24.)

Davide Menegon, Stefano Mizzaro, Elena Nazzi et Luca Vassena. *Evaluating Mobile Proactive Context-Aware Retrieval : An Incremental Benchmark.* In Advances in Information Retrieval Theory, Second International Conference on the Theory of Information Retrieval, ICTIR 2009, Cambridge, UK, September 10-12, 2009, Proceedings, volume 5766 of *Lecture Notes in Computer Science*, pages 362–365. Springer, 2009. (Cité en page 25.)

Christian Middleton et Ricardo Baeza-yates. *A Comparison of Open Source Search Engines*, 2008. (Cité en page 11.)

Stefano Mizzaro, Elena Nazzi et Luca Vassena. *Retrieval of context-aware applications on mobile devices : how to evaluate ?* In Proceedings of the 2nd International Conference on Information Interaction in Context, IIiX 2008, London, UK, October 14-17, 2008, volume 348 of *ACM International Conference Proceeding Series*, pages 65–71. ACM, 2008. (Cité en page 25.)

Josiane Mothe et Ludovic Tanguy. *Linguistic features to predict query difficulty - a case study on previous TREC campaigns.* In ACM Conference on research and Development in Information Retrieval, SIGIR : Special Interest Group on Information Retrieval, Predicting query difficulty - methods and applications workshop, pages 7–10, 2005. (Cité en pages 24 et 57.)

Josiane Mothe et Ludovic Tanguy. *Linguistic Analysis of Users' Queries : towards an adaptive Information Retrieval System.* In SITIS 07 Third International IEEE Conference on Signal-Image Technologies and Internet-Based System. HAL - CCSD, 2007. (Cité en page 93.)

Marion Moulinou. Analyse des performances des systèmes de recherche d'information. Rapport de stage, SID Toulouse. IRIT, 2013. (Cité en page 38.)

Fionn Murtagh. Correspondence analysis and data coding with java and r. CRC Press, 2005. (Cité en page 68.)

I. Ounis, G. Amati, V. Plachouras, B. He, C. Macdonald et C. Lioma. *Terrier : A High Performance and Scalable Information Retrieval Platform.* In Proceedings of ACM SIGIR'06 Workshop on Open Source Information Retrieval (OSIR 2006), 2006. (Cité en pages 22, 23, 24 et 84.)

Julia Poirier et Benoît Sansas. Evaluation des systèmes de recherche d'information. Rapport de projet long, INSA Toulouse. IRIT, 2010. (Cité en page 56.)

Jay M. Ponte et W. Bruce Croft. *A Language Modeling Approach to Information Retrieval.* In Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, SIGIR '98, pages 275–281, New York, NY, USA, 1998. ACM. (Cité en page 10.)

Martin F Porter. *An algorithm for suffix stripping.* Program : electronic library and information systems, vol. 14, no. 3, pages 130–137, 1980. (Cité en page 8.)

S. E. Robertson. *The Probability Ranking Principle in IR.* Journal of Documentation, vol. 33, no. 4, pages 294–304, 1977. (Cité en page 10.)

S. E. Robertson, S. Walker, S. Jones, M. M. Hancock-beaulieu et M. Gatford. *Okapi at TREC-3*, 1996. (Cité en page 10.)

J. J. Rocchio. *Relevance feedback in information retrieval.* In The SMART Retrieval System, pages 313–323, 1971. (Cité en page 10.)

Gerard Salton. *Search and retrieval experiments in real-time information retrieval.* In IFIP Congress (2), pages 1082–1093, 1968. (Cité en page 9.)

Mark Sanderson et Susan T. Dumais. *Examining Repetition in User Search Behavior.* In Advances in Information Retrieval, 29th European Conference on IR Research, ECIR 2007, volume 4425 of *Lecture Notes in Computer Science*, pages 597–604, 2007. (Cité en pages 71 et 92.)

Hinrich Schütze. *Automatic Word Sense Discrimination.* Comput. Linguist., vol. 24, no. 1, pages 97–123, Mars 1998. (Cité en page 10.)

Mark D Smucker, James Allan et Ben Carterette. *A comparison of statistical significance tests for information retrieval evaluation.* In Proceedings of the sixteenth ACM conference on Conference on information and knowledge management, pages 623–632. ACM, 2007. (Cité en page 79.)

Barry Smyth, Evelyn Balfe, Jill Freyne, Peter Briggs, Maurice Coyle et Oisin Boydell. *Exploiting query repetition and regularity in an adaptive community-based web search engine.* User Modeling and User-Adapted Interaction, vol. 14, no. 5, pages 383–423, 2004. (Cité en pages 71 et 92.)

Jaime Teevan, Eytan Adar, Rosie Jones et Michael A. S. Potts. *Information re-retrieval : repeat queries in Yahoo's logs.* In SIGIR 2007 : Proceedings of the 30th Annual International ACM SIGIR, pages 151–158, 2007. (Cité en pages 71 et 92.)

André Tricot. *La prise de conscience du besoin d'information : une compétence documentaire fantôme.* publié en ligne sur Docs pour Docs, 2004. (Cité en page 9.)

Andrew Trotman, Charles LA Clarke, Iadh Ounis, Shane Culpepper, Marc-Allen Cartright et Shlomo Geva. *Open source information petrieval : a report on the SIGIR 2012 workshop.* In ACM SIGIR Forum, volume 46, pages 95–101. ACM, 2012. (Cité en page 11.)

Sarah K. Tyler et Jaime Teevan. *Large Scale Query Log Analysis of Re-Finding.* In 3rd Web Search & Data Mining, pages 191–200, 2010. (Cité en pages 71 et 92.)

Shengli Wu et Sally McClean. *Performance prediction of data fusion for information retrieval.* Information Processing & Management, vol. 42, no. 4, pages 899–915, 2006. (Cité en page 92.)

Guo-Qing Zhang, Guo-Qiang Zhang, Qing-Feng Yang, Su-Qi Cheng et Tao Zhou. *Evolution of the Internet and its cores.* New Journal of Physics, vol. 10, no. 12, page 123027, 2008. (Cité en page 6.)

Dell Zhang et J. Lu. *What queries are likely to recur in web search ?* In Proceedings of the 32nd international ACM SIGIR conference on Research and development in information retrieval - SIGIR '09, pages 827–828. ACM Press, 2009. (Cité en pages 71 et 92.)