Génération de documents virtuels par intégration de relations entre documents structurés pour la recherche d'information

Delphine Verbyst

UNIVERSITÉ JOSEPH FOURIER – GRENOBLE 1

École Doctorale Mathématiques, Sciences et Technologies de l'Information, Informatique

Génération de documents virtuels par intégration de relations entre documents structurés pour la recherche d'information

THÈSE

présentée et soutenue publiquement le 14 octobre 2008

pour l'obtention du

Doctorat de l'Université Joseph Fourier – Grenoble 1

(spécialité informatique)

par

Delphine VERBYST

Directeur de thèse : Philippe MULHEM

Composition du jury

Président :	Christine VERDIER	LIG, Professeur à l'Université Joseph Fourier - Grenoble
Rapporteurs :	Patrick GALLINARI	LIP6, Professeur à l'UPMC - Paris
	Jacques LE MAITRE	LSIS, Professeur à l'Université du Sud Toulon-Var
Examinateur :	Edmond LASSALLE	Ingénieur expert Telecom, Orange Labs R&D - Lannion
Directeur de thèse :	Philippe MULHEM	LIG, Chargé de Recherche CNRS - Grenoble

Laboratoire d'Informatique de Grenoble

Remerciements

Ce travail n'aurait pu aboutir sans la participation et le soutien d'un certain nombre de personnes, que je tiens à remercier.

Je tiens tout d'abord à remercier Philippe Mulhem à la hauteur de son implication dans mon projet de recherche, pour sa patience et pour le partage de ses connaissances théoriques et pratiques. J'ai beaucoup aimé nos discussions, formelles et informelles, au laboratoire ou en mission, qui m'ont permis d'avancer scientifiquement au cours de mes trois ans de thèse. Ta grande disponibilité et tes capacités de travail m'ont toujours impressionnée et je suis très contente d'avoir toujours pu compter sur ton soutien tout au long de notre projet très structuré ! Finalement, passionné et profondément humain, tu as su me motiver dans les moments de doute probablement inévitables au cours d'une telle entreprise.

Je remercie également Edmond Lassalle, qui a suivi mon projet de thèse avec son regard de chercheur industriel. Nos discussions ont toujours été intéressantes et les objectifs fixés dans le cadre du projet France Telecom ont rythmé l'avancement de mes travaux.

Je remercie vivement les membres du jury qui ont accepté d'évaluer mes travaux. En particulier, merci à Patrick Gallinari, Professeur à l'Université Pierre et Marie Curie et directeur du LIP6, ainsi qu'à Jacques Le Maître, Professeur à l'Université de Toulon et du Var, d'avoir pris le temps de rapporter ma thèse et de l'intérêt qu'ils ont manifesté pour ce travail. Je remercie également Christine Verdier qui a accepté de présider ma soutenance.

J'ai eu le plaisir de faire des rencontres très enrichissantes pendant toutes ces années : tout d'abord, je remercie les chercheurs de l'équipe MRIM qui m'ont accueillie et ont partagé mon quotidien, Bernard qui m'a maintes fois sauvé la vie quand les serveurs me jouaient des tours et Valérie qui s'occupait des charges administratives. L'aventure humaine et scientifique s'intensifie en dehors du laboratoire, en conférences ou en écoles d'été, de Dublin à Fortaleza, en passant par des destinations moins exotiques et les idées fructuent lorsque les chercheurs sont concentrés, surtout sous le soleil :)

Je remercie mes amis, Maude, Caro et Roland, Carine et Loïc sans qui ma vie à Grenoble n'aurait pas été aussi agréable. Il nous reste bien des repas, des bières et des soirées à partager.

Je remercie mon amie Bérengère, dont le soutien est indéfectible. Ton amitié et ta confiance me touchent énormément.

Je remercie mon ami Robert, que j'ai toujours aimé retrouver à droite à gauche le temps d'une fin de semaine ; bientôt nous vivrons à moins de 500kms l'un de l'autre...

Je remercie avec toute mon affection ma famille, mes parents et mes soeurs que j'aime et admire, qui ne comprennent sans doute pas très bien pourquoi j'ai fait tout ça, mais qui m'ont soutenue et encouragée dans toute ma vie d'étudiante - oui oui, c'est bien fini maintenant :) -. Je remercie mes grands-parents qui sont si fiers de moi et ma marraine pour qui les systèmes de recherche d'information ont vraiment un sens !

Je dédie cette thèse
à mes parents,
à mes soeurs,
Mylène et Charlotte.

Table des matières

Table des figures

Liste des tableaux

Introduction

Le 9 octobre 2007, une dépêche New York AFP titre "Google leader écrasant de la recherche sur Internet mondial"[1]. Partout dans le monde, des millions d'internautes effectuent chaque jour des recherches pour satisfaire leurs besoins d'information. 61 milliards de requêtes ont été traitées dans le monde au mois d'août 2007, sur différents moteurs de recherche, dont Google[2], Yahoo![3], Baidu[4], Live Search[5].

En France en particulier, Google représente 90% des parts de visite sur l'ensemble des moteurs de recherche utilisés comme le décrit l'article du Journal du Net[6]. Aux États-Unis, en juillet 2006, Google représentait 44,7% des parts de marché sur les moteurs de recherche américains. Le nombre de requêtes passées sur Google a augmenté de 55% en un an si on compare le second trimestre de 2006 au second trimestre de 2005. Les réponses à ces requêtes sont données grâce aux documents indexés dans la base de documents de Google qui croît constamment. Une idée de la taille de l'index Google est donnée en tapant une requête qui produit potentiellement de très nombreux résultats. Pour la requête "the", 12,96 milliards de résultats sont obtenus alors que l'index de Google comptait environ 1 milliard de pages début juillet 2000 et un peu plus de 8 milliards en septembre 2005.

Jusque dans les années 80, les documentalistes étaient les principaux usagers des systèmes de recherche d'information : ils/elles devaient exprimer leur besoin dans une syntaxe correcte, sans quoi leur demande ne pouvait être traitée. Cette contrainte très forte réduisait l'accessibilité des systèmes à un certain type d'usager. Par la suite, les concepteurs de systèmes de recherche d'information ont voulu faciliter l'utilisation des systèmes par les non experts, et ont donc davantage pris en considération les utilisateurs.

Considérons un utilisateur qui effectue une recherche sur l'encyclopédie en ligne Wikipédia[7] pour découvrir la façon de danser la salsa. Le document réponse à la requête "danser la salsa" présenté à cet utilisateur est le document complet intitulé Salsa[8], qui comporte une section "La danse" parmi beaucoup d'autres informations telles que "La musique" ou encore "Le mot salsa", qui n'intéressent *a priori* pas l'utilisateur. La réponse comporte donc du bruit, elle n'est pas précise.

[1] http ://afp.google.com/article/ALeqM5hmg2WxR4pLsvDGhzplle3bZWfcxg
[2] http ://www.google.com/
[3] http ://fr.yahoo.com/
[4] http ://www.baidu.com/
[5] http ://www.live.com/
[6] http ://www.journaldunet.com/cc/03_internetmonde/interfrance_moteurs.shtml
[7] Wikipédia - L'Encyclopédie libre - http ://fr.wikipedia.org
[8] Wikipédia - Article "Salsa" - http ://fr.wikipedia.org/wiki/Salsa

Intuitivement, le défi dans le domaine de la recherche d'information consiste à renvoyer très rapidement à un utilisateur l'information la plus pertinente pour ce dernier, information sélectionnée parmi des masses de données toujours plus importantes, le nombre de documents accessibles sur Internet augmentant en permanence. L'enjeu d'un système de recherche d'information est donc la satisfaction de l'utilisateur.

Depuis la moitié des années 90, le principe des documents complets, aussi dits insécables, retournés par un système de recherche d'information n'est plus suffisant pour répondre de manière ciblée au besoin d'un utilisateur. Aussi, l'avènement de HTML/XML permet de structurer les documents d'une base et des travaux se sont intéressés à proposer des approches de recherche d'information capables de ne fournir en réponse que des parties de documents afin de mieux répondre au besoin d'un utilisateur. Sur l'exemple proposé ci-dessus, on constate que la structure HTML du document "Salsa" n'est pas utilisée pour faciliter l'accès au résultat par une navigation rapide et efficace dans le document. Plusieurs questions s'imposent : la structure HTML a-t-elle été prise en considération au moment de la recherche d'information pour répondre au besoin de l'utilisateur ? Si oui, quelle structure et comment précisément ? Est-ce que la section "La danse" présentée seule aurait été une réponse satisfaisante pour l'utilisateur ?

1 Problématique

Notre problématique traite de la recherche d'information parmi des documents structurés. Utiliser des documents structurés doit permettre de répondre de manière plus précise aux besoins des utilisateurs. En effet, le système de recherche d'information doit être capable de renvoyer des documents composés de parties de documents au lieu des documents complets en réponse à un besoin utilisateur. Nous qualifions ces documents de virtuels. Nous identifions trois difficultés principales dans ce cadre : la définition de la notion de document structuré, la description des documents virtuels à fournir en réponse à un utilisateur, et l'intégration de cette dimension de structure au système de recherche d'information classique pour fournir des documents virtuels en réponse.

1.1 Définition d'un document structuré

Selon Lalmas[9], tout document peut être considéré comme structuré selon un ou plusieurs types de structures :

- l'ordre des mots, des phrases, des paragraphes...
- la hiérarchie ou structure logique des chapitres de livre par exemple, des sections...
- les liens (hyperliens), les références croisées, les citations...

La structure est alors implicite. Elle peut également être explicite et formalisée dans ce cas par des standards de représentation de documents (langages de balises par exemple) appliqués à :

- la mise en page : LateX pour l'édition de documents, HTML pour l'édition de pages Web,
- la structure : SGML [ISO86], XML pour l'édition de pages Web,
- le contenu sémantique : RDF pour les ontologies.

[9]Structure/XML Retrieval, Mounia Lalmas - ESSIR 2005 -http ://www.dcs.qmul.ac.uk/ mounia/CV/ESSIR2005.pdf

Lorsqu'un document s'avère structuré, les différents éléments qui le constituent sont identifiés. L'exemple le plus courant de structure d'un document demeure sa structure logique, comportant des sections, sous-sections, etc. On peut retrouver des structures logiques dans des documents HTML (dans une certaine mesure) ou XML par exemple. Cette structure peut alors être exploitée pour renvoyer des portions de document en réponse à une requête. La figure 1 propose la représentation graphique de la structure logique d'un document : la structure logique du document D_1 contient le nœud racine D_1, père de 3 fils S_1, S_2 et ss_6. Les nœuds D_1, S_1 et S_2 sont non-atomiques et ss_6 est atomique. Les fils ss_1, ss_2 et ss_3 de S_1 sont atomiques et il en va de même pour les fils ss_4 et ss_5 de S_2. Au-delà des liens de composition structurelle qui apparaissent en figure 1, d'autres informations existent dans le document et peuvent enrichir l'arbre, comme des méta-informations et des informations relatives au contenu du document utilisables par la recherche d'information.

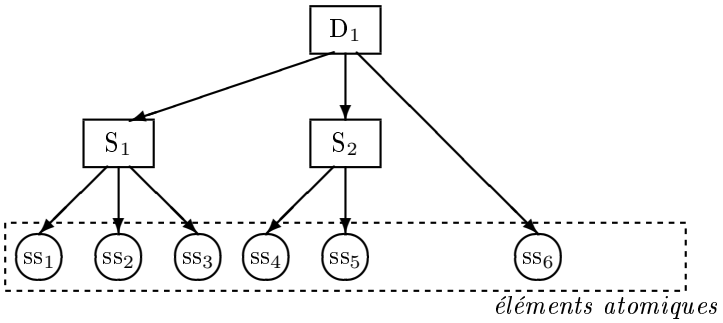

FIG. 1 – Exemple de représentation d'un document structuré (structuration explicite).

Lorsque la structure des documents n'apparaît pas de manière évidente, si l'objectif est de retourner à un utilisateur des parties de documents et non des documents atomiques, le système peut découper les documents atomiques en passages plus ou moins longs, et qui ne représentent absolument pas la structure logique des documents. Un exemple de document découpé en passages est proposé à la figure 2. Dans cette figure, le nœud du document D_2 est composé des nœuds passages p_1 à p_9. Les documents non-atomiques représentés par des passages peuvent être vus comme un cas particulier des documents structurés. De même, des méta-informations peuvent être ajoutées sur les éléments de documents, il faut cependant noter que ces informations doivent être produites par le système, alors qu'avec les documents structurés on peut utiliser le type du nœud par exemple. Une autre distinction avec les documents structurés vient du fait qu'il peut exister un recouvrement entre les passages correspondant à des parties de documents.

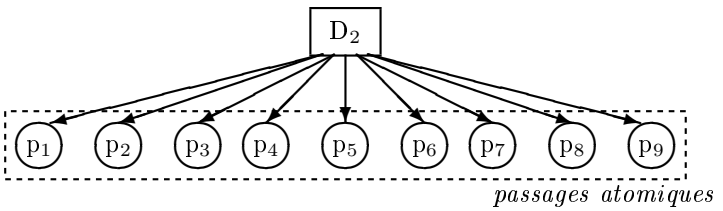

FIG. 2 – Exemple de représentation d'un document en passages (structuration implicite).

Nous proposons la définition suivante d'un document structuré :

Définition 1. *Un document structuré est un document qui peut être découpé et représenté sous forme d'arbre de profondeur variable. Les éléments de document ainsi créés peuvent se recouvrir. Un document structuré s'oppose par définition à un document insécable, dit atomique.*

Nous définissons également ce que nous appelons dans la suite de ce manuscrit un doxel :

Définition 2. *Un doxel est un élément de document.*

Finalement, qu'est-ce qu'une portion de document ? Intuitivement, il semblerait intéressant de retourner la (les) section(s) du document qui répond(ent) le mieux à la requête ; mais, qui dit "section" sous-entend que le document est pré-découpé en parties, sections, sous-sections, ... ce qui n'est malheureusement pas toujours le cas. Ainsi, la première véritable interrogation est la suivante :

Quel découpage adopter pour le corpus de documents servant de base à l'interrogation ?

Nous verrons que plusieurs possibilités s'offrent à nous : un découpage selon la structure, selon le contenu, les thèmes abordés, un découpage en fenêtres, fonction d'un nombre de mots fixé... Les approches traditionnelles ont expérimenté différentes descriptions de la structure d'un document pour la recherche d'information, en profitant des avancées technologiques dans la définition des standards de représentation de documents.

1.2 Description d'un document virtuel

Une idée pour présenter des résultats de recherche consiste à générer des documents virtuels dans lesquels l'utilisateur peut naviguer, ces documents comportant l'information pertinente retournée par le système de recherche d'information. La production de tels documents n'est pas simple, en particulier parce que l'assemblage de parties de documents nécessite de comprendre le sens de ces parties, mais aussi parce que dans le contexte de la recherche d'information, ce document doit être produit instantanément et que sa durée de vie est très courte. Afin de bien comprendre la problématique de la génération de documents virtuels, nous commençons par définir le terme *document virtuel* dans notre cadre.

Définition 3. *Un document virtuel est un document répondant à une requête utilisateur en lui présentant de façon organisée un ensemble de doxels pertinents pour son besoin d'information. Ces doxels sont issus du corpus ou générés à partir des éléments du corpus. Le document produit a alors une certaine structure, à déterminer.*

Pour Martin [MA96], "un document virtuel se compose de copies d'éléments de documents". Tazi et Altawki [TA99] donnent une définition qui prend en compte la dimension utilisateur, en ne considérant pas seulement les aspects systèmes : "Un document virtuel est généré à partir d'une composition de fragments de contenus (texte, image ou son) en utilisant des scripts ou des programmes et en définissant des liens vers d'autres documents. [...] Il répond à un besoin d'interactivité, et est généralement éphémère". Roisin [Roi99] abonde dans le même sens : pour elle, un document virtuel "est conçu par un auteur dans le but d'être reçu par un ou plusieurs lecteurs". De même, Gruber [GVR97] définit les documents virtuels comme "des documents hypermédia générés à la demande, en réponse à des données utilisateur".

Toutes ces définitions insistent sur les aspects composition et génération de documents virtuels, en gardant à l'esprit que les documents sont créés pour un utilisateur. Nous formulons la deuxième difficulté comme suit :

Quel document virtuel produire pour répondre à une requête ?

La liaison composant/composé est à prendre en compte : si deux sous-sections d'une section sont pertinentes pour une requête, renvoie-t-on les deux sous-sections indépendamment, ou bien la section englobante ? Est-ce uniquement un problème de présentation ? De plus, le fait de considérer des parties de documents comme élément de réponse peut donner l'idée de structurer les réponses de manière intelligente (i.e. plus seulement sous forme d'une simple liste ordonnée par valeur de pertinence décroissante) : nous nous retrouvons alors face à l'alternative de créer un (ou des) document(s) virtuel(s) comme réponse à une requête.

1.3 Système de recherche d'information et structure

"La recherche d'information s'intéresse à la structure, l'analyse, l'organisation, le stockage, la recherche et la sélection d'informations"[10] [Sal68]. Ces tâches concourent toutes à retrouver l'information qui répond le mieux à un besoin utilisateur parmi un ensemble de documents.

Système de recherche d'information classique. Un système de recherche d'information est l'outil qui effectue la recherche d'informations. Il établit le lien entre des requêtes d'utilisateurs et des documents stockés dans une base de documents appelée corpus.

Une décomposition fonctionnelle d'un système de recherche d'information (cf. figure 3) comporte essentiellement deux parties :

- une partie "indexation", qui correspond à un traitement des documents accessibles pour les insérer dans la base de données d'interrogation, celle sur laquelle porteront les requêtes des utilisateurs : les documents sont analysés pour en extraire le contenu, cette étape génère une représentation du contenu du document, le modèle de document.
- une partie "interrogation", qui correspond à la partie gérant l'interaction avec l'utilisateur, incluant essentiellement le traitement de la requête et la présentation des résultats : l'utilisateur exprime son besoin sous forme d'une requête, qui est également interprétée pour en extraire le contenu, afin d'obtenir le modèle de requête. La fonction de correspondance permet de comparer les deux représentations de contenu. À l'issue de cette étape, le système de recherche d'information renvoie les documents pertinents classés par ordre de pertinence.

Indexation L'indexation en recherche d'information a pour objectif de représenter le contenu des documents avec suffisamment d'acuité pour les retrouver si ce document est pertinent pour un besoin d'information, tout en permettant une bonne distinction entre les documents lors de recherches. La représentation du contenu d'un document textuel (ou index) est basée sur la notion de terme d'indexation.

Une question cruciale pour obtenir une bonne indexation est donc de trouver de bons termes d'indexation. Un bon terme d'indexation est un terme qui discrimine bien les documents et qui représente bien le contenu des documents.

L'un des principes de la recherche d'information est que la notion de pertinence d'un document pour une requête n'est pas binaire. Cette correspondance non binaire se base en particulier

[10] "Information retrieval is a field concerned with the structure, analysis, organization, storage, searching, and retrieval of information"

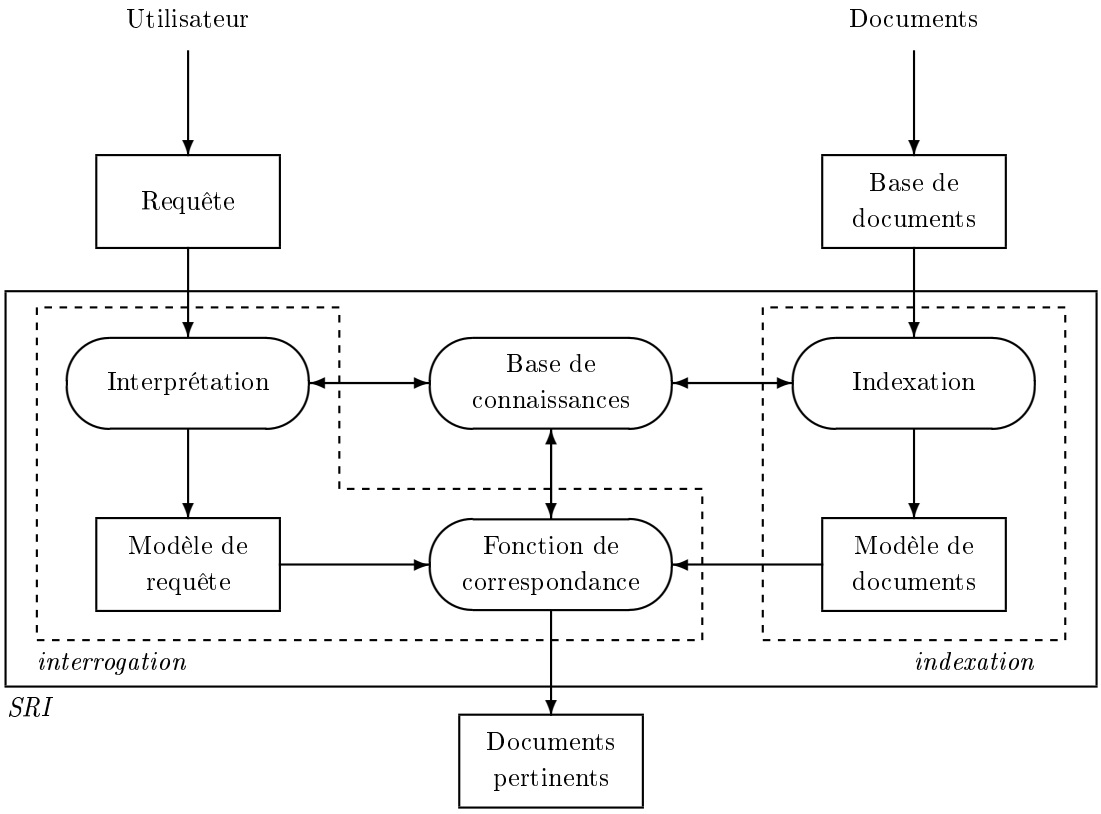

FIG. 3 – Architecture d'un système de recherche d'information.

sur le fait que la représentation du contenu d'un document indique dans quelle mesure un terme le décrit bien. On parle de poids (ou pondération) d'un terme pour un document. La pondération la plus courante d'un terme pour un document se définit ainsi : soit tf (*term frequency*) l'importance d'un terme dans un document d et soit idf (*inverse document frequency*) la discrimination d'un terme dans le corpus, alors le poids d'un terme t dans un document d, noté $w_{t,d}$, est exprimé par $w_{t,d} = tf_{t,d} * idf_t$. En effet, l'expérience prouve que l'utilité d'un terme pour représenter un contenu augmente avec la fréquence du terme dans un document mais diminue avec le nombre de documents auxquels le terme appartient [Sal68].

Interrogation Les requêtes des utilisateurs suivent un traitement semblable à celui appliqué à un document afin d'en extraire les termes pertinents pour la recherche d'information : découpage de la requête en mots, génération de termes, exclusion des termes non pertinents. Dès lors, une fonction de correspondance est utilisée pour déterminer la valeur de pertinence des documents. Suivant les modèles utilisés, les fonctions de correspondance vont de l'utilisation de correspondances floues (modèle booléen pondéré) à l'utilisation de probabilités (modèles de langues), en passant par des calculs de cosinus (dans le modèle vectoriel).

Évaluation des systèmes de recherche d'information Un problème de base pour la recherche d'information consiste à permettre la comparaison des systèmes de recherche d'information. Les comparaisons les plus courantes se basent sur une approche *boîte noire* des systèmes, vue l'hétérogénéité des modèles sous-jacents. Ces approches utilisent des collections de tests. Une collection de tests se définit par un corpus et un ensemble de requêtes résolues (c'est-à-dire des requêtes pour lesquelles on connaît a priori l'ensemble des documents pertinents du corpus). À partir d'une collection de tests, les deux valeurs de rappel (*recall*) et de précision (*precision*) sont calculées :

- le rappel évalue la capacité du système à fournir en réponse tous les documents pertinents ;
- la précision mesure la capacité du système à ne fournir que des documents pertinents en réponse.

À partir de ces valeurs, les courbes de rappel/précision permettent de comparer des systèmes de recherche d'information entre eux. Notons que ces valeurs de rappel et de précision se basent sur des documents atomiques, et non sur des parties de documents.

Système de recherche d'information classique et structure. Le fait d'envisager de répondre à une requête par des portions de documents présentées sous forme de documents virtuels soulève un certain nombre de problèmes non-triviaux qui remettent en cause le système de recherche d'information classique. Au niveau de l'indexation, dans le cas où l'élément unitaire n'est plus le document mais une partie de document, nous ne savons plus quel poids assigner aux termes par exemple. Quant à la fonction de correspondance, elle doit être adaptée pour ne plus retourner des documents complets mais des parties de documents à l'issue de la phase d'interrogation. Finalement, l'évaluation de la qualité des réponses d'un système de recherche d'information qui manipule des documents structurés doit aussi être remise en cause.

La difficulté la plus importante est de répondre à l'interrogation suivante :

Comment intégrer la dimension de structure au système de recherche d'information classique ?

La recherche de documents structurés ouvre un nouveau champ d'opportunités pour proposer en réponse à des requêtes de "meilleurs" résultats que ceux offerts par les documents complets. Nous proposons un modèle de recherche de documents structurés répondant à ces questions.

2 Objectifs

Nos travaux se focalisent sur les tâches d'indexation et d'interrogation du système de recherche d'information sur des documents structurés pour renvoyer des documents virtuels en réponse à un utilisateur. Notre objectif consiste à proposer un modèle de recherche d'information répondant aux difficultés posées ci-dessus. Nous souhaitons, grâce à ce modèle, être en mesure d'effectuer une recherche d'information sur n'importe quel corpus de documents structurés et naviguer parmi les documents retrouvés. Le système doit être capable d'intégrer et de tirer parti de différents types de structures et de l'environnement des doxels.

Un point important de ce travail concerne l'aspect évaluation des contributions. Nous souhaitons valider le modèle proposé. Nous ne proposons pas de mesure d'évaluation, mais un objectif majeur de cette thèse est de participer aux campagnes d'évaluation INEX[11]. Cela induit que le système doit être capable de traiter de grands corpus de données dans des délais contraints imposés par les organisateurs des campagnes. D'autre part, ce travail a été réalisé dans le cadre d'un contrat France Telecom dans lequel l'aspect implémentation s'avère très important.

3 Approche

Notre approche consiste à proposer une extension du modèle de recherche d'information classique en un modèle de recherche d'information pour des documents virtuels afin de favoriser la navigation de l'utilisateur dans l'espace des résultats, d'un doxel vers d'autres doxels pertinents pour la recherche. Cette idée a déjà été explorée par Bates [Bat86] pour être en adéquation avec le fait qu'un utilisateur est capable de reconnaître des informations qu'il/elle recherche alors qu'il/elle ne savait pas les décrire au préalable, et qu'il peut également parcourir des données et percevoir des informations intéressantes pour son besoin d'information en un coup d'œil. Ainsi elle ne fait qu'utiliser les capacités humaines et surtout elle ne les occulte pas. Afin de favoriser cette navigation, nous choisissons de nous appuyer sur l'environnement d'un doxel pour permettre cette navigation. Cette nouvelle modélisation nécessite des ajustements dans les différentes parties du système de recherche d'information.

Dans nos travaux, un utilisateur a un besoin d'information à satisfaire en accédant à des documents structurés. Les documents structurés sont indexés de façon à obtenir un modèle de ces derniers à l'étape d'indexation. De la même façon, un modèle de requête est produit durant la phase d'interrogation. La fonction de correspondance permet alors de comparer les deux représentations afin de renvoyer les documents pertinents organisés. Les documents virtuels produits par le système sont reliés entre eux et permettent à l'utilisateur de naviguer parmi les résultats. La figure 4 représente un système de Recherche d'Information pour des documents virtuels.

Tout comme dans la figure 3, cette décomposition fonctionnelle comporte essentiellement deux parties :

- une partie "indexation", qui correspond à un traitement des documents structurés accessibles pour les insérer dans la base de données d'interrogation, celle sur laquelle porteront les requêtes des utilisateurs ;
- une partie "interrogation", qui correspond à la phase gérant l'interaction avec l'utilisateur, incluant essentiellement le traitement de la requête et la présentation des résultats ; la phase d'organisation correspond à un arrangement des résultats à présenter sous forme de document virtuel.

Dans l'indexation des documents structurés, afin de favoriser l'organisation des résultats, nous choisissons d'exploiter l'environnement des doxels. Dans cette optique, nous utilisons non seulement la structure des documents, mais également des liens non-structurels et des relations non-compositionnelles, telles que des relations de voisinage. Nous définissions formellement l'environnement des doxels dans notre modèle.

[11] INitiative for the Evaluation of XML Retrieval - http ://inex.is.informatik.uni-duisburg.de/

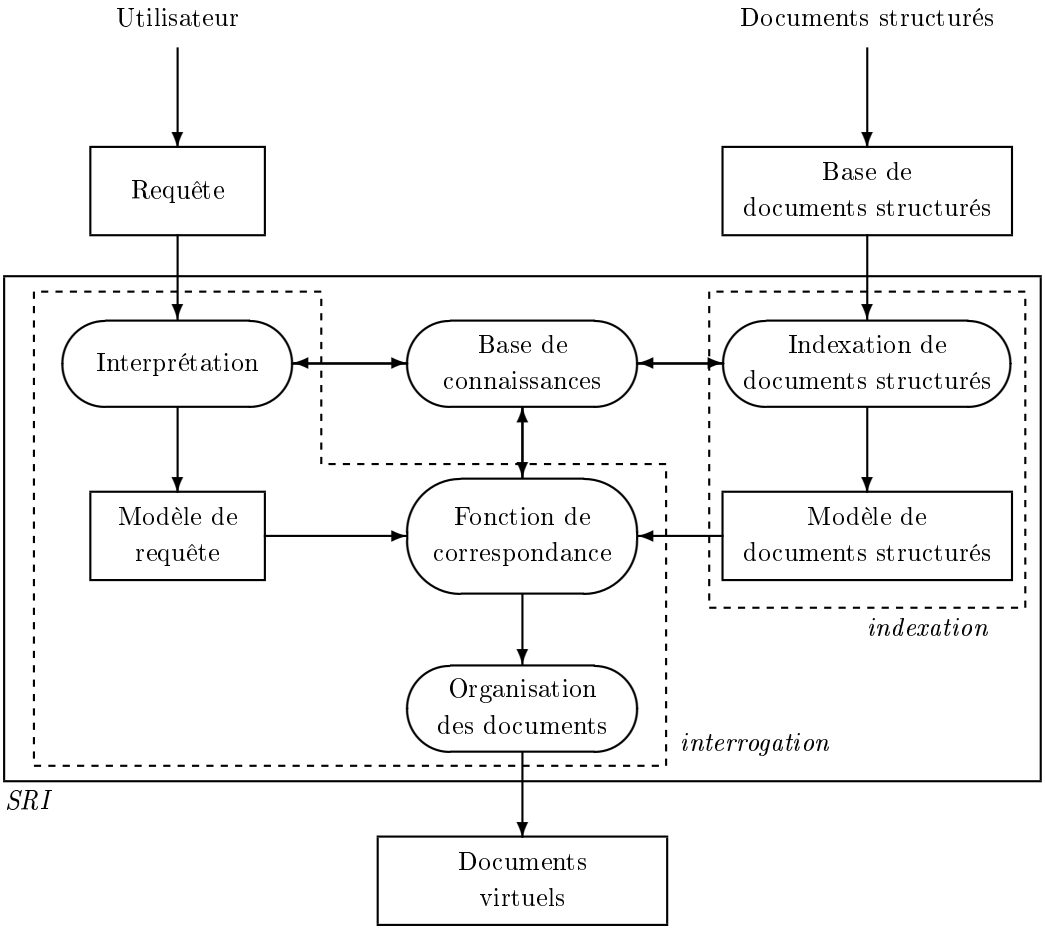

FIG. 4 – Système de recherche d'information pour des documents virtuels.

Pour le calcul de la fonction de correspondance, nous utilisons des notions d'exhaustivité et spécificité relatives entre doxels appartenant à un même environnement. Ces valeurs caractérisent les liens entre doxels et favorisent la prise en compte de certains doxels comme réponse à une requête donnée.

Nous nous focalisons sur les documents textuels, car le projet s'intéresse prioritairement à ces données.

4 Plan du manuscrit

Nous traitons la problématique de la recherche d'information pour des documents virtuels de la manière suivante.

La première partie est consacrée à l'état de l'art. Nous détaillons d'abord différents travaux de recherche qui considèrent des documents structurés plutôt que des documents atomiques au chapitre 1. Plusieurs types de structures sont ici envisagés. Puis nous présentons au chapitre 2 des études qui se sont intéressées au problème des documents virtuels. Tous ces travaux montrent que répondre à une requête par des doxels plutôt que par des documents complets permet d'améliorer la précision de la recherche en terme de focalisation structurelle des réponses.

La seconde partie du mémoire présente notre contribution avec la proposition d'un nouveau modèle de recherche d'information pour des documents virtuels. Ce modèle prend en compte les résultats d'analyse de l'état de l'art et propose de considérer l'environnement des doxels pour retrouver les éléments les plus pertinents :

- le chapitre 3 propose une description générale d'un corpus de documents structurés ;
- le chapitre 4 décrit notre modèle de documents, intégrant la dimension de structure des documents structurés et modélisant l'environnement des doxels ;
- le chapitre 5 modélise la phase d'interrogation, en proposant une fonction de correspondance adaptée aux documents structurés et à la prise en compte de l'environnement des doxels, et détermine un format de documents virtuels ;
- le chapitre 6 correspond à une instanciation du modèle précédemment décrit sur des documents structurés dans le format XML.

Tout au long de cette deuxième partie, nous présentons un exemple pour mieux illustrer nos choix.

La troisième partie présente la validation de notre approche par des expérimentations et l'analyse des résultats obtenus. Nous commençons par la présentation de la campagne INEX au chapitre 7 puis nous proposons des expérimentations au chapitre 8 afin de valider notre système et de le confronter à d'autres systèmes sur le corpus des collections d'INEX.

Ce manuscrit se termine par une conclusion générale qui rappelle le contenu de chaque partie et les grandes lignes suivies au cours de ce travail ; l'intérêt et les limites des travaux sont soulignés, pour ensuite présenter les perspectives d'avenir.

Première partie

État de l'art

Chapitre 1

Documents structurés

Sommaire

À partir du milieu des années 90, des chercheurs se sont penchés sur la prise en compte de la structure des documents dans le cadre de la recherche d'information, avec l'idée de focaliser davantage les réponses sur des parties de documents plutôt que sur des documents complets (ou atomiques). Ces travaux découlent de la prolifération des documents structurés (structures SGML [ISO86], ODA [Hor85], et plus récemment XML[12]) et des besoins de réponses plus précises pour répondre aux besoins des utilisateurs.

L'objectif de ce chapitre est de présenter les différents travaux qui considèrent le traitement des documents structurés plutôt que des documents atomiques comme élément de base d'un système de recherche d'information.

Nous commençons par nous intéresser à l'usage d'un système de recherche de documents structurés par des utilisateurs. Une telle étude sur le comportement des usagers est proposée en section 1.1. Dans cette étude, les auteurs partent du constat qu'une utilisation efficace d'un système de recherche d'information nécessite que les utilisateurs sachent quand arrêter - temporairement - de chercher pour lire des documents, et où ils doivent commencer à lire. Une conclusion est que les documents structurés hiérarchiquement sont intéressants dans le sens où ils permettent de profiter de différents points d'accès et ainsi de basculer facilement d'un mode à l'autre.

La structure des documents est donc utile et doit être prise en compte par les systèmes de recherche d'information. En section 1.2, nous définissons une classification des travaux en recherche de documents structurés, selon les relations qui peuvent exister entre documents structurés, et plus particulièrement entre deux doxels de documents structurés.

Dans ce chapitre, nous détaillons ainsi plusieurs formats de documents structurés et leur intégration dans le système de recherche d'information classique. La section 1.3 présente des approches qui ne considèrent aucune relation entre doxels de documents structurés tandis que les sections 1.4 et 1.5 postulent qu'il existe des relations entre doxels. En effet, dans la section 1.4, les approches proposées reposent sur le fait que deux doxels peuvent être liés d'un point de vue thématique, autrement dit sémantique : leur contenu les rapproche. Quant aux approches de la section 1.5, elles exploitent davantage la structure des documents, la structure de composition étant fournie par construction des documents.

1.1 Usage des documents structurés

Avant de synthétiser nos propos sur les documents structurés, nous présentons une étude sur le comportement des usagers mis face à un système de recherche de documents structurés, dans laquelle leur mode d'exploration, alternant recherche et lecture, est mis en évidence.

Hertzum, Lalmas et Frokjaer, dans [HLF01], s'appuient sur le système TeSS qui supporte l'exploration et l'interrogation d'une collection de documents selon 4 modes d'interaction : la navigation (BROWSE) à partir des entrées de la table des matières, la recherche booléenne classique (LOGICAL), le diagramme de Venn (VENN) qui est supposé renvoyer des réponses plus facilement accessibles que le modèle booléen, et le mode (ALL) qui est une combinaison des 3 modes précédents.

[12]XML : http ://www.w3.org/XML/

Deux outils de lecture existent, qui permettent d'ouvrir le texte dans des fenêtres, à tout niveau dans la hiérarchie des documents structurés :

- *View block* ouvre une fenêtre qui comporte toutes les sous-parties du texte correspondant à la partie sélectionnée dans la table des matières ;
- *View fragment* ouvre seulement le texte correspondant à la partie désirée, en excluant le texte de ses sous-parties.

Les expériences menées portent essentiellement sur le côté interface, et les 87 sujets interrogés ont été forcés par le système à utiliser tous les modes d'interaction pour résoudre 20 tâches de recherche d'information distinctes, ce qui correspond à répondre à des requêtes. Les mesures effectuées montrent que la qualité des réponses apportées par les sujets est équivalente d'un mode à l'autre, mais que, par contre, le temps passé pour atteindre ces solutions varie. D'autre part, une analyse plus précise du comportement de recherche et de lecture des utilisateurs est proposée, en fonction du mode de recherche auquel ces derniers ont accès.

Au niveau de l'outil de lecture, *View block* a été utilisé deux fois plus que *View fragment*, mais dans l'ensemble, le nombre total de fenêtres ouvertes est le même pour chaque mode. De plus, *View block* est préféré à *View fragment* en mode BROWSE alors que c'est l'inverse pour le mode LOGICAL.

D'autre part, sur le temps total nécessaire pour compléter une tâche, les utilisateurs passent globalement 25-28% du temps en recherche, et ensuite seulement ils commencent à lire. Ils s'arrêtent de lire bien avant la fin de la tâche qu'ils sont en train d'effectuer et passent les derniers 17-22% de leur temps à faire des recherches additionnelles et à écrire leurs réponses. Au final, le temps passé à lire représente en moyenne 35-39% du temps utilisé pour compléter une tâche : les sujets passent donc au moins un tiers de leur temps à lire, quelque soit le mode de recherche proposé.

Un autre aspect étudié est l'accès aux résultats, il s'effectue soit par *hits* soit par *textes cibles*. Les *hits* sont des points d'entrée présentés sous forme de liste hiérarchisée, les parties pertinentes étant repérées par des astérisques, ils dirigent le comportement de recherche de l'utilisateur. Un *hit* est en fait le texte correspondant à une partie, en excluant ses sous-parties, il correspond à ce qui peut être vu avec le mode de lecture *View fragment*, il est très spécifique. La lecture en mode *View block* permet d'accéder au contenu incluant les sous-parties et il est également possible d'obtenir davantage de contexte en ouvrant une des parties englobant le *hit*.
Une autre solution consiste à diviser le texte en 4 catégories de textes en fonction de leur position par rapport au texte cible, repéré en gras : les textes cibles (*target texts*) qui ont été identifiés par la recherche comme pertinents, les textes au-dessus du texte cible (*above-target texts*), les textes au-dessous (*below-target texts*) ou en dehors (*off-target texts*) des textes cibles qui ne constituent pas des réponses mais peuvent contenir des informations intéressantes par rapport à la réponse. Globalement, les sujets ont souvent fait appel à l'ouverture du texte situé au-dessus des textes cibles, quelque soit le mode de recherche utilisé, ce qui souligne l'importance du contexte pour l'utilisateur, et les utilisateurs préfèrent le *View block* au *View fragment*. À cela correspondent deux hypothèses : les utilisateurs préfèrent pouvoir différer leur décision d'approfondir après avoir eu un aperçu, ou bien, ils veulent tout simplement éviter d'ouvrir plusieurs fenêtres.

En conclusion, afin d'accéder au texte, les utilisateurs préfèrent les vues qui présentent la totalité des sous-parties du texte relatif à une partie (commande *View block*), ce qui fait qu'ils évitent d'alterner des périodes courtes de recherche et lecture, et assure une certaine continuité dans les tâches. Le *View block* est davantage utilisé pour obtenir plus de détails que de contexte. D'autre part, la tâche de lecture est utilisée pour supporter la recherche, voire même à la place de la recherche. Ainsi, la minimisation de la lecture en ne fournissant en réponse que les meilleurs points d'entrée n'est pas la solution d'après cet article : il faut donc trouver un juste milieu entre lecture et navigation. Accéder au document complet qui constitue le contexte des doxels apparaît comme important, même dans le cas de la recherche de documents structurés où la possibilité de répondre à un besoin d'information uniquement par une partie de document est offerte.

1.2 Axes d'analyse

Répondre à une requête par un doxel plutôt que par un document complet permet de renvoyer des parties de documents supposées plus précises pour un besoin donné, et améliore ainsi la précision de la recherche en terme de focalisation structurelle des réponses. Une question est donc de définir comment décrire le contenu des documents structurés en fonction de leurs composants. D'autre part, si l'on considère des doxels et non plus des documents complets, il devient intéressant de tirer profit du fait que les doxels peuvent avoir des similarités susceptibles d'être prises en compte. L'étude de l'état de l'art que nous avons menée nous permet de proposer une classification des approches existantes : elles se distinguent, comme le présente la figure 1.1, par les éléments qu'elles utilisent - aucune dépendance entre les éléments de structure, des dépendances basées sur le contenu des doxels, des dépendances entre doxels basées sur la structure - et le moment de leur utilisation - au cours du processus d'indexation ou au moment de l'interrogation-.

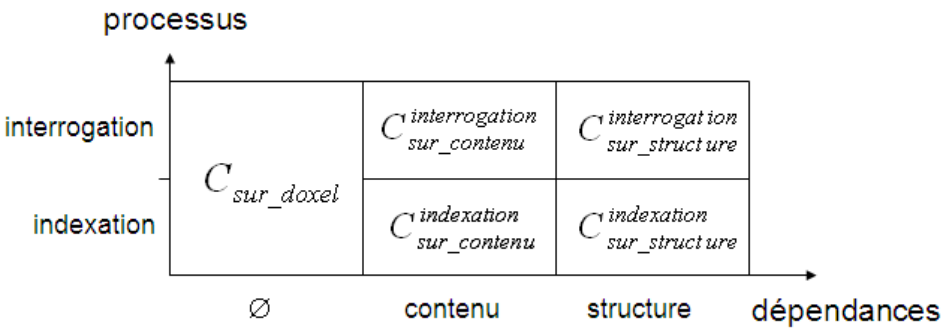

FIG. 1.1 – Axes d'analyse.

Les approches de la classe C_{sur_doxel} ne considèrent pas le fait que les doxels ont des relations entre eux, qu'elles soient de structure ou de contenu. Cette catégorie est donc davantage proche d'une simple transposition de travaux sur les documents complets vers des travaux sur les documents structurés.

Les approches de la classe $C_{sur\_contenu}$ utilisent des relations entre doxels basées sur le contenu des éléments ou sur des éléments de structure par exemple. Ces relations semblent plus pertinentes à utiliser pour des doxels que pour des documents complets, car des doxels étant (hypothétiquement) plus petits (en terme de nombre de mots pour un texte) que des documents complets, ils couvrent moins de sujets différents et cela réduit les risques de contresens dans l'interprétation des doxels et donc des relations entre doxels. Dans cette classe d'approches, nous distinguons les sous-classes $C^{indexation}_{sur\_contenu}$ et $C^{interrogation}_{sur\_contenu}$ en nous basant sur le moment auquel la création de ces relations est effectuée. Un exemple d'une approche de type $C^{indexation}_{sur\_contenu}$ consiste par exemple à créer des relations entre doxels traitant d'un même thème au moment de l'indexation. Un exemple d'une approche de type $C^{interrogation}_{sur\_contenu}$ consiste par exemple à créer et utiliser des relations de plus proches voisins entre doxels lors du traitement de requêtes.

Les approches de la classe $C_{sur\_structure}$ sont fondamentalement différentes car modélisent le contenu et la fonction de correspondance des doxels en s'appuyant sur des éléments de structure. Pour cela, ces approches utilisent les liaisons existantes entre les doxels (le plus souvent des relations de composition structurelle logique) pour formaliser des propagations d'information entre ces éléments. Dans cette classe d'approches, certaines, appelées $C^{indexation}_{sur\_structure}$, mettent l'accent sur la modélisation du contenu des documents. Un exemple d'approche de type $C^{indexation}_{sur\_structure}$ consiste par exemple à formaliser le contenu d'un doxel comme une fonction du contenu de ses composants structurels. D'autres approches, appelées $C^{interrogation}_{sur\_structure}$ insistent davantage sur l'utilisation de la structure au moment de l'interrogation. Un exemple d'une approche de type $C^{interrogation}_{sur\_structure}$ consiste à formaliser la valeur de pertinence d'un doxel comme une fonction des valeurs de pertinence de ses composants structurels.

Dans la suite, nous présentons tout d'abord les travaux étudiés dans l'état de l'art en les mettant en perspective des classes d'approches décrites ci-dessus. Certains travaux appartiennent à plusieurs classes, nous les avons classés par rapport à la connaissance qu'ils apportent pour notre étude.

1.3 Doxels indépendants dans un document

Cette section rassemble les approches de la classe C_{sur_doxel} qui considèrent les doxels comme indépendants vis à vis de la structure du document et/ou du contenu des autres doxels du corpus. En effet, effectuer une recherche sur des documents structurés n'implique pas forcément d'utiliser les relations entre les doxels.

Les approches présentées dans cette section permettent de définir des doxels de différentes manières afin de déterminer le meilleur découpage possible pour la recherche d'information. Dans [SJC93], Salton s'est interrogé sur les différentes approches possibles pour découper un texte. La première approche [MSDWZ93] considère deux découpages, un en pages et un en sections : alors que le second est issu de caractéristiques de structure prédéfinies, le premier s'appuie intuitivement sur le confort de lecture d'un élément à l'écran. Le deuxième découpage proposé [MSDWZ93] compare un découpage en paragraphes, qui sont des éléments de structure comme les sections de l'approche précédente, et un découpage en passages, qui se rapproche du découpage en pages ; dans cette deuxième approche, un test sur la longueur minimale des éléments découpés est proposé. La dernière approche [LC02] que nous considérons repose uniquement sur un découpage en passages, et se distingue des précédentes par l'utilisation d'un modèle de

recherche d'information plus récent, le modèle de langue ; de plus, les auteurs proposent une étude sur différentes longueurs de passages et sur différentes collections. D'autres approches de découpage en passages [ZMWSD95, MS94, KZ97, CCPT00] existent, que nous ne détaillerons pas dans cette section.

Le fait de considérer des pages, sections, paragraphes ou passages démontre l'intérêt du découpage pour la recherche d'information avant d'élaborer des techniques de découpage plus pointues. Les approches de cette section soulignent également l'importance de considérer la valeur de pertinence du document complet pour calculer la valeur de pertinence d'un doxel.

1.3.1 Pages et sections

La première approche que nous présentons est celle de Moffat, Sacks-Davis, Wilkinson et Zobel [MSDWZ93], qui souhaitent découper le texte de façon à ce que les parties répondant à la requête soient aisément lisibles à l'écran pour tout utilisateur. Ils proposent ainsi de tester l'efficacité de la recherche de documents complets en se basant sur des informations propres aux parties de ces documents, afin de déterminer s'il est préférable d'indexer plutôt des documents complets ou plutôt des parties de documents.

Idée Les auteurs considèrent plusieurs découpages possibles pour les documents :

1. un découpage en "pages" : ils fixent le nombre maximal de caractères lisibles à 1000 et considèrent le paragraphe comme l'unité minimale de découpage de documents. La stratégie de pagination consiste alors à découper les documents en paragraphes puis à les assembler de sorte que les parties de documents ne dépassent pas 1000 caractères, la longueur d'une page, soit environ 200 mots[13] ;
2. un découpage en sections suivant la structure logique des documents.

Indexation et recherche Les documents complets et les doxels sont indexés séparément et classiquement par un $tf.idf$. Différentes approches de recherche basées sur le modèle vectoriel sont alors proposées parmi lesquelles :

- E_1, approche de référence, classe les documents sur leur pertinence globale ;
- E_2 ordonne les documents complets sur la valeur de pertinence de leur section la plus pertinente ;
- E_{10} ordonne les documents complets sur la valeur de pertinence de leur page la plus pertinente.

Dans la plupart des travaux d'état de l'art, ce sont des documents complets qui sont retournés en réponse par le système de recherche d'information ; à la différences des approches issues des travaux fondateurs du domaine de la recherche d'information, les documents sont retrouvés en se basant sur des caractéristiques d'éléments de ces documents, les doxels, au lieu de considérer les documents dans leur atomicité. Le fait de renvoyer des documents complets plutôt que des parties de documents, bien que les pondérations ont été effectuées sur des parties de documents, permet de comparer les résultats basés sur des éléments de documents avec ceux obtenus en considérant les documents complets. Nous utilisons cette technique en section 8.3 de notre partie *Validation*.

[13] 1000 caractères correspondent environ à 200 mots (avec une moyenne de 5 caractères par mot, conformément aux chiffres de wikipedia http ://en.wikipedia.org/wiki/Wikipedia :Size_comparisons)

Expérimentations et résultats Les expérimentations sont menées sur un corpus de documents issus de la collection TREC-1 [Har93] pré-découpés en sections.

Les résultats des différentes expériences démontrent que les meilleurs résultats sont obtenus pour des recherches utilisant un découpage en pages. De plus, les résultats utilisant les documents complets (*référence*) sont meilleurs que ceux obtenus à partir des sections, probablement parce que ce découpage en sections était inapproprié au corpus considéré, d'autres études ayant conclu le contraire.

D'autre part, des expérimentations plus poussées ont également été décrites uniquement sur les sections. Les auteurs comparent des approches qui prennent en compte l'ordre des sections dans le document (les premières sections sont plus importantes que les suivantes), le type des sections, et la valeur du document complet pour classer les sections. Il résulte de cette étude que ces informations apportent des améliorations aux résultats en terme de précision moyenne, en particulier quand elles sont combinées.

Conclusion Les auteurs ont proposé de considérer deux critères pour le découpage des documents : la longueur des éléments de documents à considérer, pour qu'ils ne soient pas trop petits, et leur type, paragraphe ou section dans ce cas. Cette approche souligne l'intérêt du découpage des documents pour la recherche d'information ainsi que l'importance de considérer le score du document complet lors de l'affectation de poids aux doxels pour cette recherche. Ce travail considère l'appartenance d'une page ou d'une section à un document. Finalement, l'utilisation des documents complets combinée avec les parties de documents améliore les résultats.

1.3.2 Paragraphes et passages

La deuxième approche que nous avons choisie de développer est celle de Callan [Cal94]. Cette approche est relativement proche de la précédente [MSDWZ93] dans le sens où elle considère un découpage selon des paragraphes, mais elle démontre l'intérêt de fixer une longueur minimale aux doxels, et Callan propose une nouvelle solution pour le découpage de documents.

Idée Callan compare deux approches pour la recherche de documents : la première méthode consiste à utiliser des paragraphes et la deuxième technique utilise un découpage en fenêtres, également appelées passages.

Indexation et recherche

Découpage en paragraphes La première étude se base sur la structure des documents. Callan émet l'hypothèse que le paragraphe véhicule une information : il s'agit pour lui de l'entité de base, qui serait donc cohérente *a priori* vis-à-vis de la recherche d'information. Cependant, le paragraphe est étudié par une approche statistique et l'entité étant petite, ceci a un impact sur la qualité des résultats ; c'est pourquoi Callan choisit d'étendre la notion de paragraphe, en agglomérant plusieurs paragraphes. Le découpage en paragraphes est alors effectué de telle sorte que la taille des paragraphes soit comprise entre 50 et 200 mots, qu'il n'existe pas de paragraphes de longueur inférieure à 50 mots, et qu'il n'existe pas non plus de paragraphes de longueur supérieure à 200 mots. Ainsi, si un paragraphe est plus petit que 50 mots, il est fusionné avec le paragraphe suivant. Cette méthode de découpage est équivalente au découpage en "pages" de l'approche précédente [MSDWZ93].

Découpage en passages La deuxième étude proposée par Callan est une approche dite par passage. Le découpage ne tient absolument pas compte de la structure et consiste à utiliser des fenêtres coulissantes de n mots. Le découpage est *a priori* le même, quelque soit la requête considérée. Évidemment, le risque de diviser des passages pertinents en plusieurs fragments est présent. Alors, l'idée de Callan consiste à diviser le texte en fenêtres de longueur n, tous les $n/2$ mots, en commençant le premier passage d'un document au premier terme pertinent qui répond à la requête. Il se donne ainsi la chance de ne pas noyer le premier terme pertinent au milieu d'un passage, espérant qu'il est suivi de près par d'autres termes pertinents. Ce type de découpage est dynamique dans le sens où il suppose un redécoupage du texte à chaque nouvelle requête.

Ce découpage en passages d'un document produit un document qui peut être considéré comme structuré. En effet, Chatti et Calabretto [CC07] ont proposé un modèle multi-structuré contenant une structure de base qui organise le contenu partagé entre plusieurs structures - dans notre cas, plusieurs découpages du document initial - en fragments élémentaires disjoints ; un passage serait alors reconstruit par composition à partir de fragments dans la structure de base. Les documents découpés en passages peuvent alors être vus comme des documents structurés.

Pour ces expériences, la recherche de documents est basée sur la pertinence du doxel (paragraphe ou passage) qui correspond le mieux à la requête considérée, et les documents complets sont retournés en réponse.

Expérimentations et résultats Les expérimentations ont été menées sur des documents de différentes collections, et les approches sont testées avec des requêtes de TREC-1 [Har93] et TREC-2 [Har95], en utilisant un système basé sur le modèle probabiliste [RJ88].

Concernant la recherche basée sur des paragraphes, les résultats obtenus sont moins bons (-2,9% et -27,9% sur deux collections) qu'avec la recherche sur des documents complets. Les arguments avancés sont que peut-être la longueur des paragraphes des collections considérées n'est pas adaptée, ou encore que le texte n'est pas suffisamment organisé en fonction du contenu. Cependant, la combinaison des pertinences des documents et des paragraphes permet d'améliorer la précision moyenne de quelques pourcents (+1.0% et +4.3% sur les mêmes deux collections). Considérant la recherche basée sur les passages, la précision moyenne, par rapport à l'utilisation de documents complets, est nettement améliorée (+20,7%) en considérant la pertinence du meilleur passage des documents, de même qu'en combinant les pertinences des documents et des passages (+23.5%).

La comparaison des approches par paragraphe et passage, sur diverses collections homogènes et hétérogènes et pour des passages et fenêtres de tailles différentes, montre que les passages donnent de meilleurs résultats.

Conclusion Ce travail apporte une nouvelle technique de découpage des documents : le découpage en passages. Comme l'approche précédente, il ne considère que l'appartenance d'un paragraphe ou d'un passage à un document. Une nouvelle fois, la précision moyenne obtenue en combinant les documents et les parties de documents est meilleure qu'en considérant l'un ou l'autre pris seul. D'autre part, le fait de considérer un passage seul n'est pas suffisant pour la recherche d'information sur des documents structurés, comme l'a montré l'étude à partir du meilleur passage.

1.3.3 Passages avec taille fixe ou variable

La troisième approche que nous considérons ici utilise les modèles de langue pour la recherche de passages de documents, il s'agit de l'approche de Lavrenko et Croft [LC02], qui est dans la lignée des approches de Kise [KJDM01] et Melucci [Mel98] mais basée sur un modèle de langue. L'approche par modèles de langue a été utilisée sur des documents atomiques (par exemple [PC98]) et cet article l'applique à des passages. Nous expliquons tout d'abord le modèle de recherche d'information utilisé, avant de décrire les expérimentations de Lavrenko et Croft.

Contexte - Modèles probabilistes Les modèles probabilistes sont très utilisés en recherche d'information. Trois événements sont à considérer dans ce cadre : la requête Q, un document D et la pertinence binaire R. Les modèles diffèrent par leur estimation de la probabilité qu'un document soit pertinent pour une requête donnée.

Modèle classique Dans le modèle probabiliste classique de Robertson et Spärck-Jones [RJ88], le classement des documents est donné par la fonction $O_{MC}(D)$ basée sur le modèle classique (MC) et se fonde sur le ratio de vraisemblance entre $P(D|R)$, la probabilité que D fasse partie de l'ensemble des documents pertinents, et $P(D|\neg R)$ la probabilité que D ne fasse pas partie de l'ensemble des documents pertinents :

$$O_{MC}(D) = \frac{P(D|R)}{P(D|\neg R)} \approx \prod_{w \in D} \frac{P(w|R)}{P(w|\neg R)}$$

La difficulté dans le modèle probabiliste classique réside dans la nécessité d'entraîner le corpus pour calculer $P(w|R)$ et $P(w|\neg R)$.

Modèle de langue Les modèles de langue de Ponte et Croft [PC98] pallient cette difficulté en proposant un score sur chaque document. Ce score est calculé comme la probabilité que la requête soit issue de la même distribution qu'un document $P(Q|M_D)$, où M_D est un modèle de document et le classement des documents $O_{ML}(D)$ sur le modèle de langue (ML) est alors donné par la divergence de Kullbach-Leibler entre la probabilité empirique de la requête, $\hat{P}_Q$, et la probabilité du modèle de document, $P(M_D)$:

$$O_{ML}(D) = -KL(\hat{P}_Q, P(M_D)) = -\sum_i \hat{P}_Q(q_i) log \frac{P(q_i|M_D)}{\hat{P}_Q(q_i)} \propto log\ P(Q|M_D)$$

où $P(w|M_D)$ est défini par :

$$P(w|M_D) = \lambda \frac{tf(w,D)}{dl_D} + (1-\lambda)\frac{cf_w}{cs}$$

avec λ un paramètre de lissage dans $[0,1]$, $tf(w,D)$ la fréquence du terme w dans D, dl_D la taille de D (en nombre de termes), cf_w le nombre d'occurrences de w et cs le nombre total de termes dans le corpus.

Idée Plutôt que d'attribuer la génération de la requête Q aux modèles de documents M_D, Lavrenko et Croft modélisent la pertinence : ils supposent que, étant données une collection de documents et une requête utilisateur Q, il existe un modèle de pertinence qui attribue la probabilité $P(w|R)$ à un terme w dans l'ensemble des documents pertinents. Ce travail constitue un exemple d'utilisation des modèles de langue dans le modèle probabiliste classique. Ils proposent

de classer les documents sur la fonction $O_{RM}(D)$ basée sur le modèle de pertinence (R) comme la divergence de Kullbach-Leibler entre le modèle de pertinence, R, et la probabilité du modèle de document, $P(M_D)$:

$$O_{RM}(D) = KL(R, P(M_D)) = \sum_w P(w|R) log \frac{P(w|R)}{P(w|M_D)}$$

Un modèle de pertinence détermine la probabilité $P(w|R)$ d'observer un terme w dans les documents pertinents pour un besoin d'information donné exprimé par Q. Plus précisément, Lavrenko et Croft [LC01] proposent d'approximer $P(w|R)$ en utilisant la probabilité conjointe d'observer le terme w en même temps que les termes de la requête $q_1, \ldots, q_m$:

$$P(w|R) \approx P(w|Q) = \frac{P(w,q_1,\ldots,q_m)}{P(q_1,\ldots,q_m)} = \frac{P(w,q_1,\ldots,q_m)}{\sum_{v \in vocabulaire} P(v,q_1,\ldots,q_m)}$$

Le calcul de $P(w, q_1, \ldots, q_m)$ est effectué par des méthodes d'échantillonnage :

$$P(w, q_1, \ldots, q_m) = P(w) \prod_{i=1}^{m} \sum_{M_i \in \mathcal{M}} P(M_i|w) P(q_i|M_i)$$

où $\mathcal{M}$ est l'ensemble des 50 modèles M_D des 50 meilleurs documents retrouvés pour la requête $q_1, \ldots, q_m$, où la probabilité de choisir une distribution M_i basée sur w est calculée comme suit : $P(M_i|w) = P(w|M_i)P(w)/P(M_i)$, avec les $P(M_i)$ équiprobables et $P(w|M_i)$ est donnée par la formule habituelle des modèles de langue.

Passages Les auteurs ne proposent pas d'avancées sur la construction de passages, mais utilisent deux approches existantes :

1. des passages avec des fenêtres de taille fixe, avec recouvrement de moitié, comme vu précédemment [Cal94]. La première fenêtre commence à la position qui correspond au premier terme qui indexe le document ;
2. des passages de taille variable (entre 50 et 600 mots).

Expérimentations et résultats Dans les expérimentations, différentes méthodes de calcul de correspondance sont proposées, qui sont rassemblées dans la table 1.1.

Expérimentation	Correspondance
ML	$KL(Q\|M_{passage})$
R_1	$KL(R_{passage}\|M_{passage})$
R_2	$KL(R_{passage}\|M_{document})$
R_3	$KL(R_{document}\|M_{passage})$

TAB. 1.1 – Expérimentations proposées avec la valeur de correspondance choisie.

Les résultats sont évalués par rapport à la pertinence des documents classés sur leur meilleur passage pour ML, R_1 et R_2 et par rapport à la pertinence des documents pour R_3.

Pour les passages de taille fixe, les expérimentations ont porté sur des tailles de 50, 150 et 350 mots avec la méthode ML. Pour les passages de taille variable, le point de départ se situe tous les 25 mots, tous les modèles utilisent une taille fixée (de 50 à 600 mots), et les méthodes de calculs R_1, R_2 et R_3 sont utilisées.

Sur les fenêtres de taille fixe, des expérimentations avec différents paramètres de lissage λ sont effectuées, qui concluent à une amélioration des résultats pour la recherche de documents avec

le paramètre de Jelinek-Mercer, en particulier sur des documents du Federal Register (FR-12). Pour d'autres corpus (Associated Press et TREC), les résultats sont plus mitigés.

Pour un λ fixé, les expériences R_1, R_2 et R_3 sont menées sur des passages de 50 mots, les méthodes R_1 et R_2 produisent des résultats qui peuvent améliorer de plus de 100% les performances par rapport aux documents complets, pour la collection FR-12, comme le présente la table 1.2. Par contre, les résultats sont moins bons sur des collections de documents de longueur moyenne ou des collections hétérogènes, telles que les collections Associated Press ou TREC-45[14].

	Doc	R_1	R_2	R_3
Précision moyenne	0,1486	0,3177	0,2702	0,1501
Amélioration	/	+113,8%	+81,8%	+1,0%

TAB. 1.2 – Modèle de pertinence avec recouvrement de fenêtres de moitié, passage = 50 mots, collection FR-12, requêtes 51-100.

Sur les fenêtres de tailles variables, les résultats sont toujours plus mauvais que pour les documents complets, mais la méthode R_2, qui calcule la divergence de Kullbach-Leibler entre le modèle sur les paragraphes et le modèle sur les documents complets, donne les meilleurs résultats.

Conclusion Les résultats obtenus par ces travaux montrent que les approches avec des modèles de pertinence sont utilisables sur des collections de documents découpés en passages. Dans certains cas ces approches fournissent des résultats très supérieurs à des approches par modèles de langues sur les documents complets, selon la longueur des documents complets en fait et l'hétérogénéité de la collection. Dans tous les cas, les méthodes qui combinent des éléments venant des passages et des éléments venant des documents complets semblent donner de meilleurs résultats.

Suivant cette approche, il faut souligner la complexité de la mise en place de modèles de langues pour des documents, ainsi que celle liée à la création des modèles de pertinences utilisés. La question est alors de savoir si cette complexité de traitement induit un gain très significatif en terme de qualité des réponses, ce qui n'est pas encore démontré.

1.4 Doxels liés par leur contenu

Quelque soit le découpage choisi, les doxels peuvent être liés entre eux, par leurs contenus respectifs. Alors que la section précédente ne considérait aucun lien entre doxels, cette section rassemble des approches qui utilisent des relations de contenu entre doxels. Les travaux de cette classe d'approches $C_{sur\_contenu}$ se distinguent selon qu'ils prennent en compte le contenu à l'indexation et/ou à l'interrogation.

Les liens de contenu entre doxels peuvent dénoter des références bibliographiques ou des références hypertextes par exemple. Ces approches sont utilisées par Savoy [Sav96] et Shakery et Zhai [SZ06], sur des modèles de recherche différents.

[14]Text REtrieval Conference : http ://trec.nist.gov/pubs.html

Une autre idée consiste à considérer qu'un doxel est lié à un autre par son contenu thématique [SSBM96] ; Salton, Singhal, Buckley et Mitra créent ainsi des liens entre doxels d'un même document, pour une approche de résumé de textes. Nous n'approfondissons pas cette méthode dans notre étude, comme elle ne présente pas de résultats en termes de recherche d'information, nous retenons simplement que la proximité entre doxels est alors évaluée à partir de mesures de similarités basées sur le modèle vectoriel. Nous présentons par contre l'approche de Kurland et Lee [KL05] qui calculent la similarité entre documents préfiltrés par un système de recherche d'information.

1.4.1 Liens bibliographiques

Dans [DvR93], Dunlop et Van Rijsbergen ont étudié l'impact des citations sur l'indexation. Dans l'article [Sav96], Savoy se fonde sur l'hypothèse que les références bibliographiques lient le contenu de doxels, il suppose également qu'il existe des doxels proches sémantiquement dans un même corpus et il étudie comment utiliser ces liens entre documents comme élément pour la recherche d'information textuelle.

Idée Son idée est d'exploiter lors du traitement des requêtes des liens créés entre des documents. Les liens considérés entre les documents sont des liens de références bibliographiques, des liens de couplage bibliographique, qui mesure la similarité de listes de références bibliographiques entre documents, des liens de co-citation, c'est-à-dire des documents qui co-citent des documents, et des liens de plus proche voisin (PPV) au sens du contenu sémantique. Ces liens permettent de déterminer quels documents sont proches (au sens bibliographique) d'un document donné. Dans cette approche, nous soulignons que certains liens sont pré-existants, comme les liens bibliographiques, tandis que d'autres sont créés (PPV) ; nous reviendrons sur ces aspects dans notre modèle en section 4.2.

Indexation et recherche Savoy indexe classiquement avec un modèle vectoriel les documents du corpus. Il modifie la formule de correspondance pour prendre en compte les documents liés au noeud considéré, et pas seulement le contenu du noeud. La valeur de pertinence RSV d'un document D_i pour une requête Q est alors calculée comme suit :

$RSV(D_i, Q) = RSV(D_i^0, Q) + \sum_{k=1}^{r} \alpha_{ik} \cdot RSV(D_k^0, Q)$, avec r les documents cibles de liens partants de D_i, D_i^0 le document pris seul, D_i le document avec ses liens directs et α_{ik} un coefficient fixé expérimentalement ou dépendant du nombre de liens sortants.

Expérimentations et résultats Savoy utilise la collection ACM qui comporte 3200 documents comme corpus et 50 requêtes. Les résultats obtenus, en comparaison avec le modèle vectoriel (VSM) sans lien, sur la collection ACM sont présentés dans la table 1.3, où out_i représente le nombre de liens sortants d'un document.

Modèle	VSM	Référence	Couplage	Co-citation	PPV
Précision Moyenne	32.58	35.27 (+8.3%) $\alpha_{ik} = 0.2$	34.71 (+6.5%) $\alpha_{ik} = 0.2$	34.24 (+5.1%) $\alpha_{ik} = 1/out_i$	32.11 (-1.4%) $\alpha_{ik} = 0.1$

TAB. 1.3 – Précision moyenne sur les différents modèles.

Conclusion Savoy a émis l'hypothèse que les liens de références cachent un contenu sémantique semblable. Les résultats de ses expérimentations montrent que l'utilisation de liens de référence améliore de plus de 8% la précision moyenne, les liens de couplage améliorent de 6,5%, les liens de co-citation de 5,1% et les liens de plus proches voisins n'améliorent pas les résultats. Par conséquent, nous retenons que l'utilisation de liens bibliographiques avec un modèle vectoriel peut considérablement améliorer la recherche d'information, mais que ce n'est pas toujours le cas (PPV). Nous inscrivons ces travaux dans la classe $C^{indexation}_{sur\_contenu}$ et $C^{interrogation}_{sur\_contenu}$.

Depuis, d'autres travaux ont considéré des liens non structurels tels que des liens Internet ou des liens de similarité. Dans [SA07] par exemple, Smucker et Allan ont montré que des liens de similarité pouvaient aider à naviguer dans l'espace résultat et dans [SA06], les mêmes auteurs ont proposé de construire des liens de similarité entre documents basés sur des requêtes afin d'améliorer la qualité des résultats.

1.4.2 Liens hypertextes

Henzinger [Hen05] a analysé les liens hypertextes du Web. Des algorithmes tels que le Pagerank [BP98, PBMW98a] ou HITS [Kle99] utilisent les liens d'un réseau de pages web pour assigner des scores à chaque page du réseau, ils se basent sur des liens hypertextes. Ils permettent respectivement de calculer les scores de popularité des pages ou de classifier les sites web sur leurs valeurs de hubs et autorités. Cependant, ils séparent les caractéristiques provenant des liens et des contenus, et les relations ne servent pas à naviguer dans l'espace résultat. D'autres approches [KC95, CT89, FS95] ont considéré l'hypermédia comme supporte à la recherche pour intégrer la navigation et l'interrogation.

Dans l'article [SZ06], Shakery et Zhai supposent que les contenus de deux doxels sont probablement en relation s'il existe un lien hypertexte entre eux. Ils montrent comment utiliser ces liens comme éléments pour la recherche d'information textuelle.

Idée Leur idée est que, étant donnée une requête, un bon document résultat est un document dont le contenu répond bien à la requête et qui est entouré de bons documents également. Ils définissent alors un cadre de propagation des valeurs de pertinences des documents vers les doxels de leur voisinage. Les ensembles de voisins, pour Shakery et Zhai, ne sont pas forcément mutuellement exclusifs, ils peuvent être l'ensemble des liens entrants, l'ensemble des liens sortants, le document pris seul, l'ensemble des documents de toute la collection, etc. Cette approche est donc relativement proche de celle de Savoy [Sav96]. Ils montrent comment utiliser ces liens comme éléments pour la recherche d'information textuelle, en intégrant, en plus de Savoy [Sav96], une mesure sur la probabilité d'aller d'un document vers un autre.

Indexation et recherche La probabilité de visite d'un document x est calculée comme suit :

$$p(\mathrm{x}) = \sum_{k}^{i=1} \alpha_i \sum_{d \in D} p(d) p_i(d \rightarrow x)$$

où :

- $p(d)$, la probabilité d'hyper-pertinence : pour chaque document, la probabilité de visiter le document ;
- α_i, la probabilité de sélection de l'ensemble des voisins N_i : pour chaque ensemble de voisins, indique la probabilité de choisir ce type particulier de voisins en naviguant depuis le document courant ;

– $p_i(d \rightarrow x)$, la probabilité navigationnelle : pour chaque document d'un groupe spécifique, indique la probabilité de visiter une page particulière dans l'ensemble des voisins choisis.

Les auteurs se basent sur des approches probabilistes pour estimer les probabilités d'hyperpertinence, qui sont basées sur le contenu des documents, ils utilisent un modèle Okapi de Robertson [Rob02] et un modèle de langue de Zhai et Lafferty [ZL01], pour comparer les deux approches.

En pratique, α_i est approximé en utilisant la pertinence moyenne $rel(N_i)$ des documents de l'ensemble des voisins N_i :

$$\alpha_i \propto \frac{1}{|N_i|} \sum_{X \in N_i} rel(X), \text{ tel que } \sum_i \alpha_i = 1$$

les probabilités navigationnelles sur les liens sont estimées par : $p(d \rightarrow x) = p(x)$.

Expérimentations et résultats Les expérimentations menées sur des parties de TREC-2002 [Voo02], TREC-2003 [Voo03] et TREC-2004 [Voo04] utilisent uniquement des liens de navigation entrants et sortants (IN et OUT). La prise en compte de ces liens, c'est-à-dire ces types de voisins, améliore les résultats. De plus, la propagation pondérée améliore les propagations uniformes, avec un paramètre expérimentalement évalué. La combinaison de différents types de voisins améliore également les performances, en comparaison à n'importe quel ensemble de voisins considéré seul. Finalement, quelque soit l'approche de base utilisée, Okapi ou le modèle de langue, les performances sont meilleures avec l'utilisation de voisins.

Conclusion Cette approche permet de donner un cadre à l'ensemble des approches telles que PageRank [PBMW98b] ou les Hubs et Authorities [Kle99] et souligne l'intérêt de prendre en compte un voisinage pour les documents d'une collection. Nous inscrivons ces travaux dans la classe $C^{interrogation}_{sur\_contenu}$ puisqu'ils demandent de traiter le voisinage au moment de l'interrogation.

1.4.3 Liens de génération

Les travaux précédents réalisent la recherche de documents en fonction de leur "environnement" lors du calcul de pertinence des documents. Une autre idée est de réaliser deux étapes : la première étape permet de filtrer les éléments pertinents avant de proposer, en seconde étape, un traitement d'interrogation plus lourd, qui intègre la similarité entre les documents filtrés. Dans l'article [KL05], Kurland et Lee étudient comment réorganiser un ensemble de documents répondant à un besoin d'information, en s'appuyant sur des liens de contenu entre documents. De plus les auteurs comparent des résultats obtenus avec le modèle de langue à ceux rendus par la similarité vectorielle.

Idée Après une première recherche d'information pour une requête q sur un corpus donné, Kurland et Lee proposent de considérer l'ensemble des documents résultats. Leur idée consiste à modéliser, pour chacun de ces documents initialement résultats, ses relations avec les autres documents de cet ensemble par un modèle de langue. Ils créent alors des liens appelés liens de génération, à partir desquels est calculée une valeur de centralité pour chaque document initialement retrouvé. C'est sur cette valeur que se base le ré-ordonnancement des documents initialement pertinents pour q.

Indexation et recherche L'indexation et la recherche initiales peuvent être effectuées par n'importe quelle approche de recherche d'information, qui produit alors un ensemble $\mathcal{D}_{init} \in \mathcal{C}$ des documents du corpus $\mathcal{C}$ arrivant en tête pour une requête q.

Ré-ordonnancement Un graphe $G = (\mathcal{D}_{init}, wt)$ dit graphe de génération dont les noeuds sont les documents de $\mathcal{D}_{init}$ est calculé. Les arcs orientés d'un noeud o vers un noeud g sont pondérés par $wt(o \to g)$.

La centralité d'un document est vue comme le degré de responsabilité qu'un document a d'avoir généré les autres dans l'ensemble initial. La centralité d'un document d du graphe G est alors calculée à partir de ses arcs entrants, correspondant à son influx I, et est donnée par :

$$Cen_I(d;G) \stackrel{def}{=} \sum_{o \in \mathcal{D}_{init}} wt(o \to d)$$

Le ré-ordonnancement des documents est alors effectué selon le critère $Cen(d;G)$ ou $Cen(d;G) \cdot p_d(q)$.

Graphes uniformes ou pondérés Des expérimentations sont menées en utilisant deux graphes de génération différents $G_U = (\mathcal{D}_{init}, wt_U)$ et $G_W = (\mathcal{D}_{init}, wt_W)$ avec, pour $o, g \in \mathcal{D}_{init}$:

$$wt_U(o \to g) = \begin{cases} 1 & \text{si } g \in TopGen(o), \\ 0 & \text{sinon.} \end{cases}$$

où $TopGen(o)$ dénote les meilleurs générateurs de o.

$$wt_W(o \to g) = \begin{cases} p_g(o) & \text{si } g \in TopGen(o), \\ 0 & \text{sinon.} \end{cases}$$

où p_g dénote le modèle de langue lissé unigramme induit à partir de g.

G_U traite les meilleurs générateurs uniformément, tandis que G_W les pondère. Un influx uniforme et un influx pondéré peuvent alors être calculés.

Influx récursif Certaines expérimentations requièrent des graphes pour lesquels aucun arc n'est nul, un paramètre de lissage λ est appliqué pour définir un graphe $G^{[\lambda]} = (\mathcal{D}_{init}, wt^{[\lambda]})$ où :

$$wt^{[\lambda]}(o \to g) = (1 - \lambda) \cdot \frac{1}{|\mathcal{D}_{init}|} + \lambda \cdot \frac{wt(o \to g)}{\sum_{g' \in \mathcal{D}_{init}} wt(o \to g')}$$

Un influx récursif est également défini pour qu'un document qui est pointé par des documents dont la centralité est petite ne soit pas considéré avec une valeur de centralité élevée sous prétexte qu'il a été fortement généré par ces documents :

$$Cen_{RI}(d;G) \stackrel{def}{=} \sum_{o \in \mathcal{D}_{init}} wt(o \to d) \cdot Cen_{RI}(o;G)$$

Expérimentations et résultats Les expérimentations ont été menées sur des corpus de TREC (AP89, AP, WSJ, TREC8) [VH99] pour des graphes de génération uniformes et avec des pondérations différentes, en considérant des influx simples puis pour des influx récursifs, en réordonnant sur les valeurs de centralité. Les mêmes expérimentations ont été répétées en tenant compte cette fois-ci des pondérations obtenues dans le classement initial basé sur le modèle de langue.

Conclusion Le premier constat est que l'utilisation de cette mesure de centralité améliore les résultats du système de recherche d'information. De plus, les résultats démontrent que l'utilisation de graphes de génération utilisant des pondérations différentes pour les meilleurs documents générateurs conduit à des performances meilleures qu'avec l'utilisation de pondérations uniformes. En outre, une mesure de centralité récursive donne de meilleures performances. Enfin, l'utilisation du matching initial améliore également les résultats. Les résultats sont meilleurs dans environ 2/3 des 96 comparaisons proposées (8 algorithmes basés sur la centralité × 4 corpus × 3 mesures de performance).

Finalement, les auteurs ont comparé leur approche qui construit des graphes à l'aide du modèle de langue plutôt qu'en appliquant des mesures de similarité basées sur le cosinus, et les modèles de langue permettent d'obtenir de meilleurs résultats.

Nous inscrivons ces travaux dans la classe $C_{sur\_contenu}$ puisqu'ils considèrent des liens de génération basés sur le contenu. Nous classons plus précisément ces travaux dans la classe $C^{interrogation}_{sur\_contenu}$ dans la mesure où il s'agit d'un ré-ordonnancement de résultats, et que les calculs tiennent compte de la requête.

1.5 Doxels liés par la structure

Cette section rassemble des approches qui considèrent les doxels comme liés par des aspects de structure propres aux documents auxquels ils appartiennent. Les travaux présentés dans cette classe d'approches $C_{sur\_structure}$ se distinguent selon qu'ils prennent en compte la structure à l'indexation et/ou à l'interrogation. Ces travaux portent sur des structures plus complexes que les découpages basiques en pages, passages ou paragraphes vus précédemment.

Il existe bien des approches telles que celle de Lalmas et Rölleke [RLK+02] qui combinent des aspects contenu et des liens de structure pondérés en introduisant une dimension d'accessibilité pour retrouver les meilleures parties de documents. Gery, dans [Gér06], considère que le contenu des doxels dépend du contenu des précédents doxels dans l'ordre de lecture des doxels d'un document. Cette approche s'est classée 64ème/106 à INEX 2006. Beigbeder dans [Bei06, Bei07] limite les aspects structure aux titres et sections ; il utilise la proximité des termes pour pondérer les éléments de sections et propage les termes des titres dans les sections qu'ils encadrent, ce qui donne de bons résultats en terme de précision pour cette approche. Schenkel et Theobald dans [ST06] calculent le score d'un élément comme la concaténation des contenus textuels de tous ses nœuds ascendants dans l'ordre du document. Cette méthode a produit de bons résultats pour la tâche Focused Task d'INEX 2006 (système dans les 10 meilleurs systèmes pour la campagne d'évaluation). La structure est utilisée encore d'une autre façon par Lu, Robertson et MacFarlane dans [LRM06] : ils utilisent d'abord une recherche au niveau du document complet, puis décident d'une taille minimale pour éviter d'avoir des doxels trop petits comme résultats, comme dans [KdRS04]. Ce travail a obtenu les meilleures performances sur la tâche Focused Task de INEX 2006. Nous ne reviendrons pas sur ces approches dans la suite, mais nous avons choisi de présenter les travaux qui suivent pour leur originalité.

Une première idée consiste à effectuer une propagation des index sur la structure hiérarchique des documents, c'est ce que présentent Cui et Wen dans [CWC03]. Pour éviter de se poser des questions sur comment propager des termes entre composants de documents dans un document, Sauvagnat [Sau04] a proposé d'indexer uniquement les feuilles des documents structurés, puis de réaliser lors du traitement de requêtes des propagations de valeurs de pertinence entre parties de

documents. La prise en compte de la structure des documents peut également se baser sur une évaluation *a priori* d'une partie de document dans le document complet, comme dans [Hua07], qui utilise une structure dite de surface. Une autre idée consiste à s'appuyer sur les réseaux Bayésiens. Les réseaux Bayésiens peuvent supporter non seulement la structure des documents du corpus, mais également la structure des requêtes posées. Nous présentons deux approches, l'une simple [MJKZ98], et l'autre [PG04] plus complexe qui nécessite une étape d'apprentissage. Finalement, nous présentons l'approche de Lalmas [LV04] qui s'efforce de mettre un cadre théorique basé sur la théorie de Demptser-Shafer pour la recherche d'information sur des documents structurés : les représentations des documents et des requêtes peuvent alors s'intégrer dans ce cadre.

1.5.1 Structure hiérarchique

Dans le domaine de la recherche hiérarchique, Crouch a proposé un modèle dynamique flexible dans [Cro06], basé sur le modèle vectoriel étendu de Fox [Fox83]. A partir des scores de pertinence des éléments feuilles de documents structurés et de la connaissance de la structure des documents, elle retrouve les doxels les plus pertinents pour une requête donnée.

Avant elle, Cui et Wen ont proposé d'étudier l'impact de l'utilisation de chemins d'accès à des éléments de documents pour la recherche d'information en s'appuyant sur l'information structurelle des documents, les statistiques de distributions des termes, et un algorithme de classement des doxels basé sur l'index hiérarchique établi. Dans [CWC03], les auteurs présentent leur technique dite de *recherche flexible d'éléments*, qui retourne les doxels pertinents pour des requêtes avec une granularité arbitraire, afin d'améliorer l'efficacité de la recherche d'éléments structurés.

Idée La méthode d'indexation est basée sur une propagation hiérarchique des index représentant les parties de documents : un terme est propagé d'un composant vers un composé, si sa distribution dans les parties du composé est uniforme, afin de ne pas attribuer un terme comme représentant d'une section s s'il ne concerne qu'une des sous-sections de s par exemple.

Indexation L'indexation hiérarchique est établie avec la même structure que celle du document. Les termes d'indexation sont répartis dans les noeuds de l'arbre du document. L'idée clé est qu'un terme d'indexation caractérise un noeud-élément s'il est discriminant pour le concept de cet élément. Un bon terme d'indexation est un terme qui apparaît fréquemment et de manière répartie dans le texte de l'élément décrit, son rang est élevé comparativement à celui des autres termes de l'élément considéré.

Les poids des termes pour les éléments, pour l'étape de classement, sont calculés différemment selon que l'élément considéré est une feuille (un paragraphe P_j) ou un élément intermédiaire (E_j) de la structure du document global :

- $Weight(t_i, P_j) = ln(tf(t_i, E_j)) \times ln(N/n_i)$ est la mesure classique du $tf.idf$, employée ici avec N le nombre total de documents du corpus et n_i le nombre de documents contenant le terme t_i ;
- $Weight(t_i, E_j) = ln(1 + tf(t_i, E_j)) \times I(t_i, E_j)$ utilise la mesure de fréquence tf du terme t_i dans l'élément E_j, et la mesure d'entropie I, i.e. la distribution du terme t_i dans l'élément E_j. Cette valeur d'entropie se base en fait sur les valeurs de tf dans chacun des fils structurels de l'élément considéré.

Ces poids sont normalisés en les divisant par le maximum des poids de tous les termes d'un même élément pour être comparables.

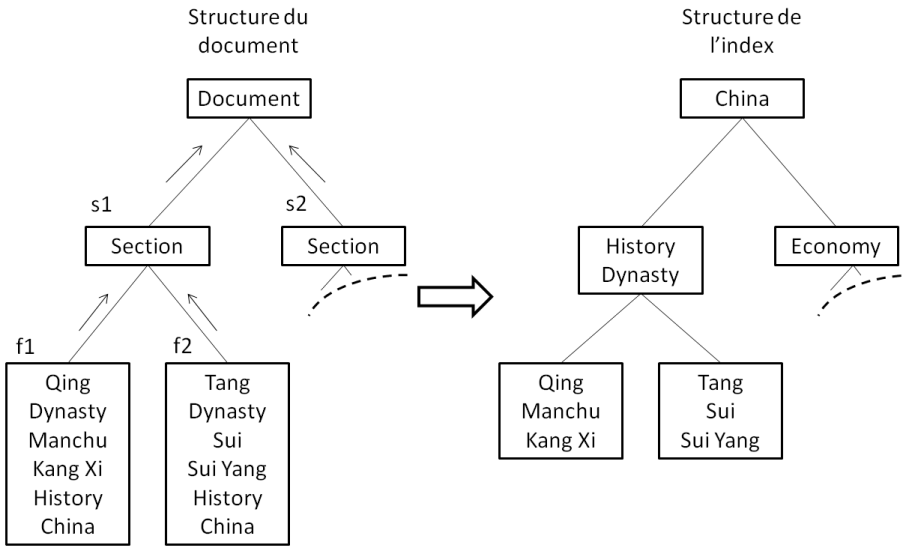

FIG. 1.2 – Propagation de l'index et mécanisme d'élagage [CWC03].

La sélection des termes d'indexation est effectuée par propagation et élagage, les poids de chaque terme étant connus par rapport à un élément. Un terme est propagé vers un élément plus haut placé dans la structure si son poids excède un certain seuil, et par conséquent, le terme est également enlevé des éléments descendants, puisqu'il apparaît alors comme caractérisant une plus grande partie. Ce procédé est appliqué récursivement et la figure 1.2 illustre la propagation de l'index et du mécanisme d'élagage pour un document exemple : les éléments feuilles $f1$ et $f2$ du document sont tous deux indexés par les termes *Dynasty*, *History* et *China*, si bien qu'ils indexent la section $s1$ par propagation ; puis le terme *China* indexe à la fois $s1$ et $s2$ si bien qu'il indexe le document complet. Un point clé de cette approche est que le seuil doit être contrôlé dynamiquement et ajusté pour chaque document en fonction des poids des termes d'un élément donné. Un terme est alors choisi comme terme-index pour un élément si et seulement si son poids est important pour l'élément de structure. Cette importance est avérée si le poids du terme est supérieur ou égal à la valeur moyenne des poids des termes de l'élément augmentée de l'écart type de ces poids. La solution proposée pour l'indexation utilise la structure interne des documents, mais permet de générer des index qui sont censés être représentatifs tout au long de la structure des documents.

Recherche Les éléments les plus pertinents sont sélectionnés avec l'algorithme de recherche flexible d'éléments : pour chaque élément, un algorithme de classement de chemin est utilisé pour calculer les valeurs de pertinence de chaque élément candidat face à une requête. Selon le mécanisme d'indexation hiérarchique, un élément ne partage pas de terme d'indexation avec ses prédécesseurs. Un élément est donc complètement représenté par les termes index des éléments représentés au cours du chemin. Et réciproquement, un chemin peut être caractérisé par l'élément situé au plus bas niveau du chemin. Le classement d'élément est alors équivalent à un classement de chemin.

$Relevance(Path_P) = \sum_{i=1}^{Q} Weight(t_i, Path_P) \times ln(N/n_i)$
où Q est le nombre de termes de requête dans une requête et $Weight(t_i, Path_P)$ est le poids du

terme de requête t_i sur le chemin P : le poids d'un terme pour un chemin est défini comme son poids pour l'élément contenant ce terme sur le chemin.

Dans ce modèle, les auteurs utilisent des méthodes de recherche de documents classiques pour obtenir une liste des documents pertinents (qui contiennent au moins un des mots de la requête) pour réduire l'espace de recherche, puis ils appliquent leur étude des éléments candidats pour chaque document, afin de retourner les chemins les mieux classés. La présentation du résultat est effectuée sous forme d'arbre de navigation.

Expérimentations et résultats Une évaluation a été menée sur un corpus de documents XML tirés de l'encyclopédie Encarta. Les expériences avec la méthode proposée sont comparées à un système de recherche basé sur des paragraphes seuls, avec un $tf.idf$ classique, avec la mesure de correspondance *cosinus*, identique à celle utilisée dans l'approche proposée. Deux expériences (que nous appelons $E1$ et $E2$) ont été menées pour établir les index : $E1$ tient compte des titres des éléments de la structure du document ce qui est pertinent *a priori* pour juger du contenu d'un élément ; $E2$ n'en tient pas compte. Quelque soit le seuil fixé pour la recherche d'élément flexible, on se rend compte que les résultats sont meilleurs avec la méthode proposée qu'avec la méthode classique de recherche de paragraphes. En fait, on note en moyenne une amélioration des résultats de $56,02\%$ (*F-value*) en tenant compte du titre des éléments ($E1$), et de $40,89\%$ (*F-value*) sans en tenir compte ($E2$).

Conclusion Les résultats sont meilleurs que ceux obtenus avec une recherche classique de document, comme la recherche de paragraphes par exemple. La méthode est peu sensible au changement de seuil, et l'utilisation d'un seuil dynamique améliore les performances. Les travaux décrits ici appartiennent à la classe $C^{indexation}_{sur\_structure}$, car les pondérations des termes des doxels dépendent des fils de ce doxel.

1.5.2 Structure de surface

Nous venons d'étudier une approche de type hiérarchique par propagation de terme. Une autre façon d'indexer les parties de documents est proposée par Zargayouna dans [Zar04] ; l'auteur considère que la valeur de pertinence d'un document est égale au produit des valeurs de pertinence de certaines de ses parties. Cependant, aucune expérimentation n'est proposée. Dans [Hua07], Huang propose une approche qui prend en compte des caractéristiques de structure de surface (*shallow structural features*) pour renvoyer les doxels pertinents pour les requêtes de la compétition INEX 2007. Huang s'appuie sur les différents composants d'un élément, mais les pondère différemment.

Idée Huang part du constat que les éléments de documents les plus souvent retrouvés appartiennent davantage au début des documents et que leur chemin d'accès sont souvent courts en terme de profondeur sur la collection INEX2007. Huang propose alors de pondérer les doxels indépendamment de leur contenu et selon ces critères qu'il qualifie "de surface". De plus, l'approche considère indépendamment une caractéristique sur le contenu des éléments, en proposant une représentation compacte de ces derniers : une représentation compacte d'un élément est constituée par extraction des mots provenant des titres, sous-titres, images, légendes et des mots écrits en gras, italique ou dans une police plus grosse que celle du texte qui les entourent.

Indexation La méthode d'indexation des doxels attribue ainsi des probabilités *a priori* à chaque doxel e basées sur la localisation de l'élément dans le document complet $|e_{location}|$ et la longueur de son chemin d'accès dans le document $|e_{path}|$:

$$P(e) = \frac{1}{5+|e_{location}|} \cdot \frac{1}{3+|e_{path}|}$$

Recherche Le modèle de recherche proposé s'appuie sur le modèle de langue. La pertinence d'un élément e pour une requête q est donnée par :

$$P(e|q) \propto P(e) \cdot P(q|e)$$

La probabilité de générer la requête q composée des termes $t_1, ..., t_k$ à partir d'un élément e est évaluée par l'utilisation d'un modèle de langue avec lissage sur le corpus C :

$$P(q|e) = \prod_{i=1}^{k} (\lambda \cdot P(t_i|C) + (1-\lambda) \cdot P(t_i|e))$$

L'approche proposée diffère du modèle de langue classique dans son évaluation de $P(t_i|e)$ en prenant en compte une représentation complète $P(t_i|e_{full})$ et une représentation compactée des éléments $P(t_i|e_{compact})$ limitée aux termes mis en évidence (italique, gras) dans le texte. Finalement,

$$P(q|e) = \prod_{i=1}^{k} 0.6P(t_i|C) + 0.3P(t_i|e_{full}) + 0.1P(t_i|e_{compact})$$

Expérimentations et résultats Les expérimentations ont été menées sur le corpus de documents XML de la campagne d'évaluation 2007 tirés de l'encyclopédie Wikipedia. L'approche est classée $5^{\text{ème}}$ en $MAiP$ sur 79 soumissions sur la tâche Focused, qui suppose que l'utilisateur préfère avoir en réponse l'élément de document le plus pertinent pour une requête donnée. Cependant, l'approche n'est pas comparée à une approche de base qui ne prendrait pas en compte cette pondération par $P(e)$.

Conclusion Les travaux décrits ici sont de classe $C^{indexation}_{sur\_structure}$, car les pondérations des éléments sont effectuées *a priori*, en se basant uniquement sur la structure. Etant donné le traitement de compactage des contenus des éléments, cette approche pourrait également appartenir à la classe $C^{interrogation}_{sur\_contenu}$, mais l'auteur mène plus particulièrement une étude sur l'utilisation d'éléments qui ne sont pas de contenu pour caractériser les doxels.

1.5.3 Structure de réseau Bayésien simple

Une autre façon de s'appuyer sur la structure des documents consiste à utiliser des réseaux Bayésiens. Myaeng, Jang, Kim et Zhoo ont proposé dans [MJKZ98] un modèle de recherche d'information, dérivé du modèle de réseau d'inférence de type réseau Bayésien développé pour la recherche d'information par Turtle et Croft [TC90] puis repris par Callan avec INQUERY [CCH92], pour des documents SGML [ISO86].

Contexte - Le réseau d'inférence pour la recherche de documents Un réseau d'inférence pour la recherche de documents est présenté en figure 1.3, il comporte deux sous-réseaux : un pour les documents et un pour les requêtes. Il est constitué de noeuds et de liens directionnels représentant respectivement des variables aléatoires booléennes, ce qui intuitivement représente le fait qu'un noeud est pertinent pour une requête donnée ou non, et des relations causales (dépendances) entre deux noeuds liés. Dès qu'un lien entre deux noeuds existe, la croyance qu'un noeud est en relation avec un autre est calculée en utilisant un mécanisme qui combine les probabilités :

- Le réseau de documents peut représenter un ensemble de documents d_i avec différents niveaux d'abstraction, par exemple, un réseau pourrait contenir un seul niveau d'abstraction, le niveau r, la représentation du contenu du document. Les noeuds de documents représentent le fait qu'un document répond à une requête utilisateur, ils sont assignés à $true$, et la probabilité vaut $P(d_i) = 1/(\text{nombre de documents})$. La valeur d'un lien entre d_i et un r_k est évaluée comme la probabilité $P(r_k|d_i)$. Les noeuds de représentation de contenu représentent le fait qu'un concept de requête c est observé et valent soit $true$ soit $false$. La valeur d'un arc entre la représentation du contenu d'un document r_k et un noeud de concept de requête c_l est la croyance dans cette observation.
- Dans le réseau de requêtes, le noeud q représente le besoin d'information et est marqué à $true$ puisqu'il doit être satisfait, tandis que les noeuds qui expriment les concepts c_i liés à q peuvent valoir $true$ ou $false$, selon qu'ils sont observés dans les documents ou non. Le réseau de requêtes est lié au réseau de documents par les arcs entre les noeuds de concepts et les noeuds de représentation du contenu des documents.
- Les réseaux d'inférence permettent de spécifier des fonctions pour calculer la croyance en une proposition connaissant la croyance en cette proposition sur ses noeuds parents.

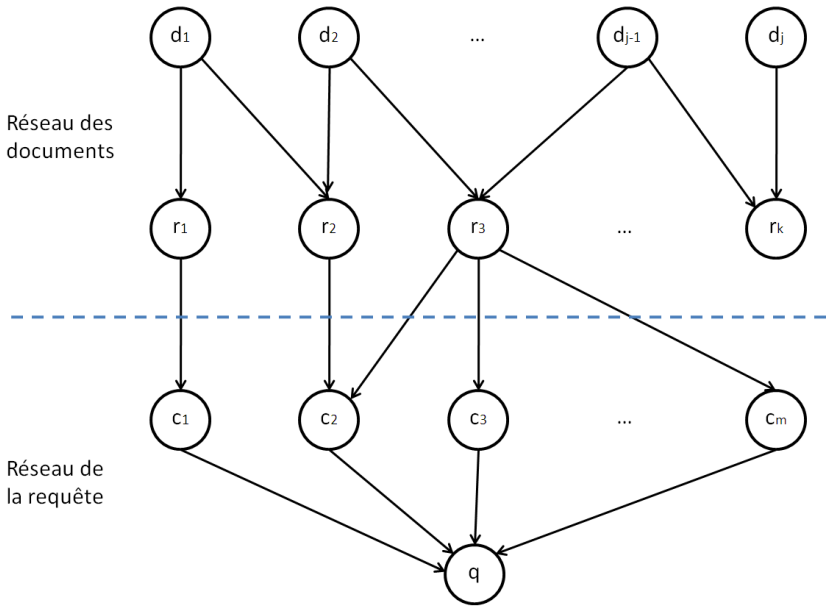

FIG. 1.3 – Un exemple de réseau d'inférence simple pour la recherche de documents [CCH92].

Idée Les auteurs proposent d'étendre le modèle développé pour INQUERY, le système de recherche d'information de Callan [CCH92], basé sur le modèle probabiliste, en y intégrant la structure des documents.

Indexation Un document est représenté par un arbre, dont les noeuds correspondent à une partie de la structure de ce document (chapitre, titre, section, etc.) et le texte qui lui est associé. Considérons la figure 1.4, la structure logique du document SGML d_1 est reflétée par la représentation. Par exemple, le fait que le doxel fils de type section s_{12} compose le doxel parent de type chapitre c_1 y est représenté par le fait que le noeud du réseau qui représente le père possède un lien vers le noeud du réseau représentant le composant. A la différence d'INQUERY, ce réseau est donc directement calqué sur la structure du document et l'information "descend" du document pour arriver jusqu'aux termes.

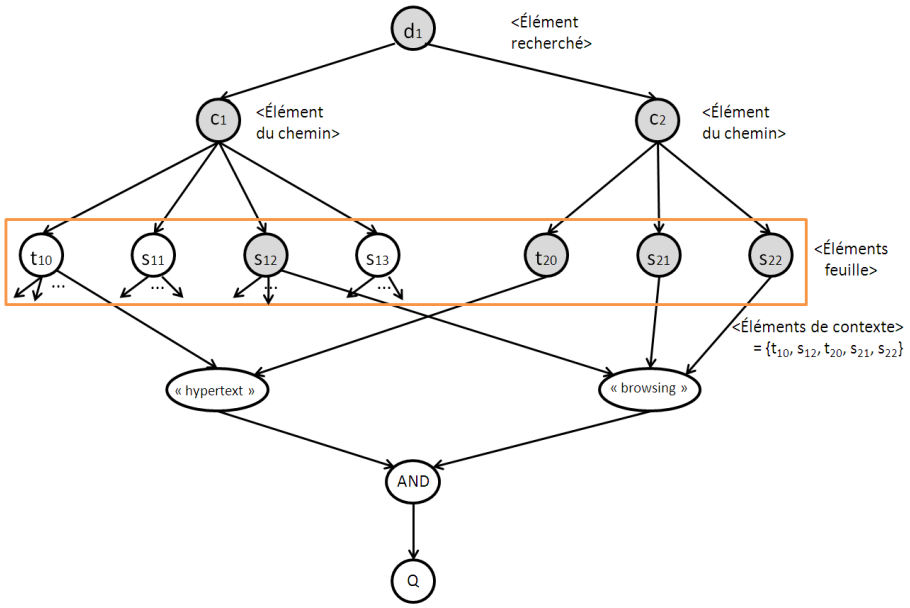

FIG. 1.4 – Un réseau d'inférence pour un document structuré et une requête [MJKZ98].

Recherche Lorsqu'une requête est posée au système, comme dans la figure 1.4, on calcule pour tout document d_i la probabilité que les termes de la requête représentent bien le document. Pour cela, il est nécessaire de calculer la probabilité qu'une section représente bien le document d_i, puis qu'un terme / concept représente bien une section, et enfin qu'une requête représente bien ce terme. Le sous-réseau "document" et le sous-réseau "requête" sont connectés pour représenter une requête contenant à la fois des restrictions de contenu (clause CONTAINS) et de structure (clause INCLUDES). L'élément recherché est spécifié par le terme de recherche (clause FIND). Le réseau présenté en figure 1.4 permet de répondre à la requête Q ="FIND documents that INCLUDES a chapter whose title CONTAINS the term *hypertext* AND whose section CONTAINS the term *browsing*".

Processus Le processus de recherche consiste à :

– marquer tous les candidats-éléments recherchés et les noeuds-feuilles qui contiennent au moins un des termes de la requête ainsi que les éléments situés sur le chemin. Les noeuds

grisés dans la figure 1.4 représentent des éléments contenant au moins un des termes recherchés ; ainsi, t_{20} comporte le terme "hypertext" et s_{12}, s_{21}, s_{22} comportent le terme "browsing" tandis que le noeud d_1 est marqué parce qu'il est un noeud candidat, de type "document" comme spécifié dans la requête. Les noeuds c_1 et c_2 sont marqués parce qu'appartenant au chemin entre les noeuds feuilles contenant les termes et un noeud candidat ;
- calculer le degré de croyance qu'un noeud terme de la requête soit appuyé par un ensemble de noeud-feuilles qui lui sont connectés ;
- calculer le degré de croyance pour une requête comme un tout, à l'aide de la combinaison des preuves des noeuds termes de la requête et en fonction des opérateurs booléens utilisés.

Calcul Bayésien de croyance Comme habituellement dans un réseau bayésien, si C est un noeud et F est l'ensemble des noeuds parents de C, alors la croyance de C, notée $B(C)$, est définie par la formule $B(C) = \Sigma_F P(C|F) \times P(F)$

Par rapport à la théorie, la méthode de calcul proposée est simplifiée et n'utilise que les événements probabilistes positifs et ainsi, pour un réseau comportant deux éléments section S_1 et S_2, noeud-parents d'un élément T et noeud-fils d'un élément C, le degré de croyance pour T est calculé comme suit :

$$\begin{aligned} B(T|C) &= P(T|S_1) \times P(S_1) + P(T|S_2) \times P(S_2) \\ &= P(T|S_1) \times P(S_1|C) \times P(C) + P(T|S_2) \times P(S_2|C) \times P(C) \end{aligned}$$

avec $P(S_i|C) = \lambda_i \times (S_i \cdot C)$ la mesure de similarité vectorielle entre l'élément parent C et chacun de ses fils S_i, le $\cdot$ représente le produit scalaire entre vecteurs, et le λ_i représente le poids du type de l'élément considéré. avec $P(T|S_i)$ calculée à l'indexation à l'aide de la formule normalisée de [TC90] : $P(T|S_i) = IDF_T \times TF_{i,S_i}$. L'utilisation d'événements probabilistes positifs permet de simplifier les calculs et de s'abstraire de difficultés pratiques sur les événements négatifs.

Le traitement de requêtes est réalisé de la manière suivante : la recherche sur le contenu est réalisée avant celle sur la structure afin d'améliorer le coût en terme d'espace de recherche. Des structures d'index indépendantes par fichiers inverses permettent des requêtes sur la structure ou sur les attributs. La recherche est effectuée de bas en haut (*bottom-up*), depuis les feuilles, guidée par le fait qu'il s'agit d'un réseau d'inférence et que le texte est contenu dans les feuilles uniquement. L'étude des segments du fichier inverse permet de rechercher les types d'éléments spécifiés par la requête : s'il n'est pas trouvé, même si un terme de la requête apparaît dans un noeud-feuille, on ne fait rien ; s'il est trouvé, le calcul de croyance est effectué.

Expérimentations et résultats Des expérimentations ont été menées sur les topics 51 à 150 de la collection TREC 97, sur des textes de brevets, documents sous forme d'arbres à 4 niveaux de profondeur au moins. 32 requêtes ont été utilisées, sur 1000 documents environ, dont environ 25% de documents pertinents.

Il résulte de l'analyse des résultats que les informations de structure SGML permettent d'améliorer les performances de recherche d'information. La précision moyenne est améliorée pour passer de 0.1563 (système de référence sur modèle vectoriel avec documents complets sans marqueurs SGML) à 0.2030 (approche proposée par les auteurs). Une expérimentation a été faite en n'utilisant que certaines parties de documents (comme <TEXT> ou <BSUM>), améliorant les résultats. Les auteurs en ont déduit que toutes les parties de documents n'avaient pas la même importance pour la recherche de document.

Conclusion Ce travail montre que l'utilisation de balises permet d'améliorer la précision moyenne lors de la recherche de documents structurés. Cependant, il est très difficile de savoir comment assigner des poids aux balises et les choix faits sont en grande partie pragmatiques, ce qui limite la portée de ce travail (par exemple, les probabilités entre parties de documents sont strictement basées sur des produits scalaires). De plus, l'utilisation du réseau d'inférence dans ce cas précis requiert de nombreux calculs. Il est également à signaler que les calculs en ligne sont simplifiés en éliminant les probabilités négatives. L'impact de cette simplification n'a pas été évalué. La propagation de probabilités est ainsi biaisée mais on ne sait pas dans quelle mesure.

Les propositions faites dans ce travail le placent dans les deux classes $C^{indexation}_{sur\_structure}$ et $C^{interrogation}_{sur\_structure}$:

- dans la classe $C^{indexation}_{sur\_structure}$, car le fait de calculer lors de l'indexation des documents les similarités entre doxel composant et composé sert de base pour le contenu du composant ;
- dans la classe $C^{interrogation}_{sur\_structure}$ lors du traitement d'une requête les propagations le long du réseau Bayésien sont effectuées.

1.5.4 Structure de réseau Bayésien avec apprentissage

Une autre manière d'utiliser les réseaux Bayésiens, plus proche de la théorie (sans appliquer de produit scalaire comme dans [MJKZ98] par exemple) a été proposée par Piwowarski et Gallinari dans [PG04]. Les auteurs présentent un modèle général pour la recherche de documents structurés basé sur les réseaux Bayésiens pour optimiser l'utilisation de la structure des documents par les systèmes de recherche d'information[15].

Idée Les réseaux utilisés intègrent la structure des documents XML étudiés comme dans l'approche de Myaeng [MJKZ98] présentée dans la section précédente. Mais les auteurs vont plus loin dans la mesure où ils représentent explicitement dans des noeuds du réseau Bayésien des résultats de correspondances suivant un ou plusieurs modèles de recherche d'information (vectoriel ou probabiliste par exemple). Les résultats sont sous forme de variable aléatoire booléenne représentant les états "pertinent" ou "non pertinent", pour chaque doxel. Ils proposent un cadre d'apprentissage des probabilités conditionnelles.

Indexation Les variables aléatoires $v_x \in V$ du réseau Bayésien pour les doxels prennent leurs valeurs dans l'ensemble d'états fini $V = \{I, B, E\}$:

- I (*Irrelevant*) quand l'élément n'est pas pertinent ;
- B (*Big*) quand l'élément est fortement exhaustif[16] et partiellement spécifique[17] ; ces mesures sont issues de la campagne INEX2005 [INE05] ;
- E (*Exact*) quand l'élément est fortement exhaustif et spécifique.

La probabilité qu'un doxel X soit dans un état $v_X \in V$ est donnée par la formule suivante, sachant que $M_i(X) \in \{R, \neg R\}$ représente le score local de X sur le modèle M_i et $P(M_i(X) = m_i)$ la probabilité de ce score local, n étant le nombre de modèles de référence considérés :

$$P(X = v_X \mid q) = \sum_{v_y \in V, m_1, \ldots, m_n \in \{R, \neg R\}} P(X = v_X \mid Y = v_Y,\ M_1(X) = m_1, \ldots, M_n(X) = m_n)$$

$$\times\ P(Y = v_Y \mid q)\ \times\ P(M_1(X) = m_1)\ \times\ \ldots\ \times\ P(M_n(X) = m_n)$$

[15] Plus de détails apparaissent dans la thèse de Piwowarski [Piw03].
[16] L'exhaustivité décrit la mesure dans laquelle un doxel discute des sujets de la requête.
[17] La spécificité décrit la mesure dans laquelle un doxel se limite aux sujets de la requête.

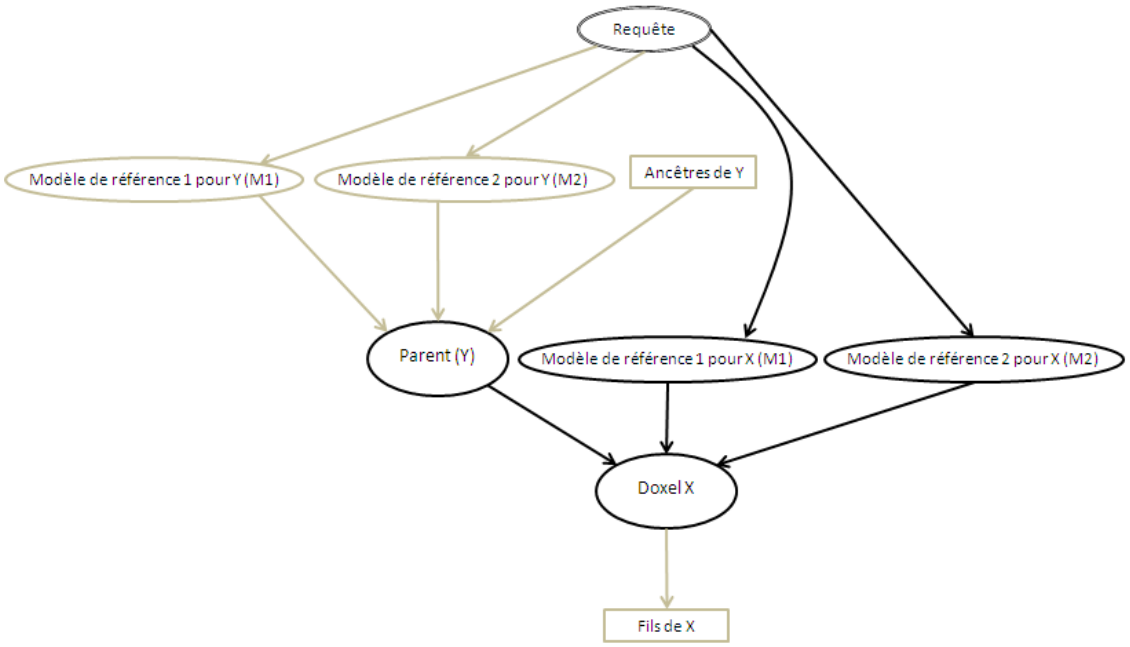

FIG. 1.5 – Vue locale d'un réseau Bayésien s'appuyant localement sur 2 modèles de recherche d'information M_1 et M_2 [PG04].

Recherche La valeur de pertinence d'un élément X relativement à une requête q est finalement calculée par :

$$\text{RSV}(X) = P(X = E \wedge X\text{'s parent} = B \mid q)$$

Ainsi, un élément ne peut être jugé pertinent que si son père a déjà été jugé pertinent mais pas totalement spécifique, ce qui semble sensé. On remarque ici que, par rapport à l'approche de [MJKZ98], le calcul utilisé pour déterminer la pertinence d'un doxel est fondamentalement différent. En effet, on utilise ici des probabilités pour des doxels sachant la requête et non pas l'inverse. Ce choix est possible à cause de l'explicitation des pertinences, ainsi que par l'inclusion de manière "cachée" de valeurs de pertinences booléennes incluant des modèles de recherche.

Pour que cette approche soit utilisable, il est nécessaire d'utiliser des algorithmes d'apprentissage sur un ensemble de documents et de requêtes. Ces apprentissages permettent de définir les probabilités conditionnelles entre les noeuds doxels et entre les noeuds "cachés" et les nœuds doxels. Les auteurs proposent une approche utilisant un principe de maximisation de vraisemblance itératif par maximisation de l'espérance. Aucune indication n'est fournie sur le temps d'apprentissage pour le système.

Conclusion L'approche décrite ici est complexe car elle nécessite un apprentissage compliqué. Les auteurs ont réalisé un apprentissage sur 15 requêtes pour un ensemble de 1000 documents sur le corpus de la campagne INEX 2003, qui est une collection de documents XML. Les auteurs indiquent que l'apprentissage semble converger après mille itérations, mais les résultats sur 15 requêtes avec une évaluation qui ne peut pas se comparer avec d'autres résultats de la bibliographie. Cette approche appartient aux deux classes $C^{indexation}_{sur\_structure}$ et $C^{interrogation}_{sur\_structure}$:

- dans la classe $C^{indexation}_{sur\_structure}$, car le fait de calculer lors de l'indexation les probabilités conditionnelles sur les pertinences entre composants et composés sert de base pour le contenu du composant ;
- dans la classe $C^{interrogation}_{sur\_structure}$, car lors du traitement d'une requête, des propagations le long du réseau Bayésien sont effectuées.

1.5.5 Structure et fonction de croyance

Les approches basées sur des réseaux Bayésiens lient les documents et les requêtes dans un même réseau. La dernière approche que nous avons choisi de présenter rapproche les représentations des documents et des requêtes afin de les intégrer dans un même cadre.

L'article [LV04] de Lalmas et Vannoorenberghe définit une approche théorique de recherche de documents structurés basée sur la théorie de l'évidence de Dempster-Schafer (décrite en particulier dans [Sme94]), dont Lalmas est une pionnière dans son utilisation pour la recherche d'information. Ce travail permet de formaliser des aspects liés à la combinaison de pertinences de parties de documents sur des requêtes de structure et de contenu.

Contexte - Théorie de Demptser-Shafer Les travaux de Shafer permettent de représenter la connaissance incertaine sur un ensemble fini Ω de valeurs possibles pour une variable x, appelé cadre de discernement. Selon la théorie de Demspter-Shafer, la connaissance partielle de la valeur prise par la variable x peut être représentée comme une fonction de masse $m : 2^{\Omega} \rightarrow [0, 1]$ telle que $\Sigma_{A \subseteq \Omega}\ m(A) = 1$. L'ignorance complète est représentée par $m(\Omega) = 1$, et une fonction de masse est dite *normale* si $m(\emptyset) = 0$. Les éléments A tels que $m(A) > 0$ sont appelés éléments focaux de m.

Pour combiner plusieurs sources de connaissances au sujet des valeurs de Ω prises par la variable x on agrège, par l'opérateur de combinaison de Dempster noté $m_{\oplus}$, les fonctions de masses m_1 et m_2 des deux connaissances :

$$m_{\oplus}(A) \ = \ (m_1 \oplus m_2)(A) \ = \ \frac{m_{\boxplus}(A)}{1 - m_{\boxplus}(\emptyset)} \ \ \forall A \subseteq \Omega$$

avec $m_{\boxplus}(A) \ = \ (m_1 \underline{\boxplus} m_2)(A) \ = \ \Sigma_{B \cap C = A}\ m_1(B) \cdot m_2(C)\ \forall A \subseteq \Omega$

Cette combinaison $m_{\oplus}$ est normalisée. En effet, si l'on pose que $m_{\oplus}(\emptyset) = 0$ alors $m_{\oplus}$ est bien une fonction de masse comme la masse de l'ensemble vide est distribuée aux éléments focaux de $m_{\boxplus}$. De plus, $m_{\boxplus}(\emptyset)$ dénote le degré de conflit entre m_1 et m_2. Ces opérations ci-dessus sont établies sur le même ensemble de valeurs Ω.

Il est possible de définir une combinaison entre deux fonctions de masse sur des cadres de discernement différents par :

$$(m^{\Omega_1} \underline{\cap} m^{\Omega_2})(A \times B) \;=\; m^{\Omega_1}(A) \cdot m^{\Omega_2}(B) \;\; \forall A \subseteq \Omega_1, B \subseteq \Omega_2$$

Les concepts de marginalisation et d'extension ont pour but de calculer des fonctions de masses sur des ensembles différents, en se basant sur une fonction de masse m^Ω définie sur un ensemble Ω égal au produit cartésien de deux ensembles Ω_1 et Ω_2. La fonction de masse marginale $m^{\Omega\downarrow\Omega_1}$ sur Ω_1 est définie, pour tout $A \subseteq \Omega_1$, par :

$$m^{\Omega\downarrow\Omega_1}(A) \;=\; \Sigma_{\{B\subseteq\Omega | Proj(B\downarrow\Omega_1)=A\}} m^\Omega(B)$$

avec $Proj(B \downarrow \Omega_1) \;=\; \{\omega_1 \in \Omega_1 \mid \exists \omega_2 \in \Omega_2; (\omega_1, \omega_2) \in B\}$

où $Proj$ représente la projection de B sur Ω_1. Le concept de marginalisation permet de passer à un cadre de discernement de granularité plus fine en propageant les valeurs de fonction de masse de Ω vers Ω_1. Une telle marginalisation peut bien sûr également être appliquée à Ω_2.

Le concept d'extension permet, de manière duale, d'élargir le cadre de discernement. Étant donnée une fonction de masse sur un cadre de discernement Ω_1, son extension sur $\Omega = \Omega_1 \times \Omega_2$ est définie pour tout $B \subseteq \Omega$ par :

$$m^{\Omega_1\uparrow\Omega}(B) = \begin{cases} m^{\Omega_1}(A) & \text{si } B = A \times \Omega_2 \text{ pour } A \subseteq \Omega_1, \\ 0 & \text{sinon.} \end{cases}$$

Cette extension se limite donc à propager aux B correspondants totalement à Ω_2.

Idée Les auteurs proposent de définir la représentation du contenu et de la structure de documents XML dans un même formalisme, basé sur la théorie des fonctions de croyance, et ils élaborent des opérateurs de recherche sur ces documents.

Indexation

Contenu Notons $e \in E$ un doxel du corpus E, et $t \in T$ un terme du vocabulaire T. Pour représenter le contenu d'un élément e, une fonction de masse notée $m^T[e]$ est définie pour chaque terme de T (avec la contrainte suivante : si $tf(e,t) > 0$ alors $m^T[e](\{t\}) \neq 0$, et donc $\{t\}$ est un élément focal pour la fonction de masse). Les auteurs utilisent également les composants d'un doxel pour qualifier son contenu, par l'utilisation de la règle de Dempster sur des fonctions de masses des composants d'un doxel :

$$m^T[children(e)] \;=\; \bigoplus_{e' \in E | part\_of(e')=e} \underline{m}^T[e']$$

avec $\underline{m}^T$ fonction de masse affaiblie, pour donner moins d'importance au contenu des composants du doxel. On combine ensuite cette valeur avec le contenu du document par la règle de Dempster :

$$m^T[e, children(e)] \;=\; m^T[children(e)] \oplus m^T[e]$$

Structure Pour représenter la structure des documents dans ce cadre, les auteurs définissent un cadre de discernement Y basé sur une fonction donnant une importance pour chaque type de doxel. Un type de doxel est par exemple "ARTICLE/ABSTRACT/PARA". La fonction de masse est notée m_1^Y. Dans le cas de requêtes avec des composants de structure, une fonction de masse m_2^Y est fournie par la requête, et la fonction de masse pour la structure m^Y est la combinaison par la règle de Dempster de m_1^Y et m_2^Y.

Combinaison du contenu et de la structure Toute l'information (contenu et structure) est agrégée dans une fonction de masse m^Ω sur $\Omega = T \times Y : m^\Omega(A \times B) = (m^T[e] \underline{\cap} m^Y)(A \times B) = m^T[e](A) \cdot m^Y(B)$.

À cause du produit cartésien entre T et Y, la fonction m^Ω n'est pas utilisable immédiatement car elle ne lie pas réellement les deux sous-ensembles T et Y. m^Ω est utilisée pour définir la fonction de masse $m^T[e, m^Y]$ consistant à obtenir un corpus d'évidence relatif au contenu du document XML ayant observé sa structure. On pose :

$$m^T[e, m^Y] = (m^\Omega \oplus m^{Y\uparrow\Omega})^{\downarrow T}$$

La fonction $m^T[e, m^Y]$ quantifie la croyance pour qu'un terme t de T indexe l'élément e ayant observé son contenu, celui de ses éléments fils ainsi que sa structure.

Recherche Pour traiter des requêtes sur le contenu seul, une fonction de masse $m^T[\text{CO}]$ sur T est définie. Elle représente l'importance relative des termes de la requête CO (*content only*). Pour le traitement de la requête la combinaison de Dempster entre $m^T[e]$ et $m^T[\text{CO}]$ est effectuée, et les résultats sont ordonnés suivant une transformation de la fonction de masse en probabilité suivant [Sme94]. Pour les requêtes de structure et de contenu, l'idée est la même : une fonction de masse $m^T[\text{CAS}]$ sur T est définie. Elle représente l'importance relative des termes de la requête CAS (*content and structure*). Pour le traitement de la requête la combinaison de Dempster entre $m^T[e, m^Y]$ et $m^T[\text{CAS}]$ est effectuée, et les résultats sont ordonnés suivant une transformation de la fonction de masse en probabilité.

Expérimentations et résultats Cette approche n'a pas été évaluée qualitativement. Au niveau quantitatif, on peut soulever le problème de la complexité potentielle de cette approche.

Conclusion Cette approche a le mérite de formaliser certains aspects de la recherche de documents structurés, mais on peut se poser la question des raisons qui font choisir les combinaisons de Dempster pour répondre au problème posé, en particulier au niveau de la gestion des structures de documents qui semblent être un point faible de la proposition faite.

Les propositions de cette approche basée sur la théorie de l'évidence utilisent, lors du calcul, des index des documents les composants des doxels, ce qui place ce travail dans la catégorie $C^{indexation}_{sur\_structure}$.

1.6 Synthèse

Dans ce chapitre, nous nous sommes d'abord intéressés au comportement des utilisateurs face à des documents structurés. Pour notre étude, nous retenons que le document présenté à l'utilisateur doit lui permettre de naviguer d'un doxel à un autre potentiellement en alternant les modes de recherche et de lecture. D'autre part, si nous choisissons de fournir des points d'accès, ces derniers doivent comporter suffisamment d'information sur le contexte et les détails doivent être accessibles à l'utilisateur.

Nous avons ensuite réalisé une classification des approches de recherche d'information dans le contexte de documents structurés, avant de dresser un état de l'art à partir d'approches sélectionnées parmi les approches fondatrices du domaine et d'autres toutes récentes. Enfin nous nous sommes interrogés sur le comportement des usagers face à des documents structurés.

Le tableau 1.4 récapitule les différents travaux étudiés dans cette partie et donne la classe d'approches à laquelle ils appartiennent.

Approche \ *Classe*	$\mathbf{C}_{sur\_doxel}$	$\mathbf{C}^{indexation}_{sur\_contenu}$	$\mathbf{C}^{interrogation}_{sur\_contenu}$	$\mathbf{C}^{indexation}_{sur\_structure}$	$\mathbf{C}^{interrogation}_{sur\_structure}$
Pages et sections [MSDWZ93]	×				
Paragraphes et passages [Cal94]	×				
Passages avec taille fixe ou variable [LC02]	×				
Liens bibliographiques [Sav96]		×	×		
Liens hypertextes [SZ06]		×	×		
Liens de génération [KL05]			×		
Structure hiérarchique [CWC03]				×	
Structure logique stockée [Sau04]					×
Structure de surface [Hua07]			×	×	
Structure de BN simple [MJKZ98]				×	×
Structure de BN avec apprentissage [PG04]				×	×
Structure évaluée par fonction de croyance [LV04]				×	

TAB. 1.4 – Les travaux par classe d'indexation et de recherche de documents non-atomiques

Les articles sur les pages et sections [MSDWZ93], et paragraphes et passages [Cal94] datent des années 90 et sont fondateurs du domaine. Ils nous ont permis de bien comprendre les problèmes rencontrés dès que l'on considère les documents d'un corpus comme des entités non atomiques. Le découpage des documents est un véritable "casse-tête" : faut-il établir une taille minimale pour un doxel ; si oui, la longueur est-elle fixée comme un nombre de caractères ou un type d'élément (paragraphe, section...). Un travail a été proposé sur la définition de passages de taille fixe ou de taille variable dans [LC02]. Finalement, on peut se demander si un découpage en passages doit être effectué statiquement à l'indexation ou plutôt dynamiquement. Toutes ces approches se situent dans la classe C_{sur_doxel}, elles ne prennent pas en compte des éléments de contenu ou de structure des documents.

Pour les travaux de classe $C_{sur\_contenu}$, nous avons répertorié des articles qui exploitent le contenu de différentes façons : tandis que l'approche sur les liens de génération [KL05] se fonde sur des liens sémantiques, d'autres présupposent qu'un lien bibliographique [Sav96] ou un lien hypertexte [SZ06] est forcément le reflet d'un contenu similaire. Les traitements de ces liens de contenu sont effectués à l'indexation et/ou à l'interrogation selon les approches.

Les approches de classe $C_{sur\_structure}$ sont surtout rencontrées dans des travaux plus récents qui tentent de s'émanciper des travaux antérieurs, et qui profitent certainement de la facilité de gérer rapidement des index de taille très importante grâce aux progrès informatiques. Ces approches considèrent des documents structurés tels que des documents SGML ou XML. Parmi ces travaux, certaines approches reposent sur une exploitation relativement simple de la structure hiérarchique [CWC03] par propagation d'index, de la structure logique stockée [Sau04], ou encore de caractéristiques de structure de surface [Hua07] telles que la localisation d'un élément dans le document, plutôt au début ou à la fin, ou la profondeur de son chemin d'accès. D'autres approches se basent sur les réseaux bayésiens (BN) et visent à appliquer ces réseaux d'inférence à des documents structurés. Les premiers travaux [MJKZ98] utilisent une structure de réseau d'inférence très proche de la structure logique des documents. Les travaux plus récents de Piwowarski et Gallinari [PG04] utilisent des réseaux dont la structure est plus éloignée de celle des documents, mais qui permettent de mieux intégrer les aspects structurels dans les réseaux, au prix il est vrai d'une complexité qui pourrait s'avérer rédhibitoire pour des grandes bases de documents. Cette remarque est aussi valide pour les travaux sur la théorie de l'évidence [LV04].

Dans les travaux listés ici, il apparaît un consensus concernant l'idée que pour la recherche de documents non-atomiques le contexte des doxels (i.e. les éléments reliés d'après leur contenu dans l'approche $C_{sur\_contenu}$) et les composants ou composés pour les approches de classe $C_{sur\_structure}$ peuvent s'avérer utiles. Nous prendrons ces dimensions en compte au moment de proposer notre modèle.

Chapitre 2

Documents virtuels

Sommaire

Les systèmes de recherche d'information actuels sont en mesure de renvoyer des doxels en réponse à une requête utilisateur comme le montre le chapitre 1. Ainsi, les résultats obtenus sont supposés être plus ciblés et donc plus pertinents que si les documents complets composés de ces doxels étaient retournés. La question se pose alors de savoir comment ces doxels sont organisés pour être présentés à l'utilisateur : un système peut présenter une simple liste de doxels, et ainsi transposer le cas classique de liste de documents à l'échelle des éléments de documents ou bien choisir une solution plus sophistiquée, telle qu'un document structuré composé de ces doxels-réponses.

Nous appelons *document virtuel* un document qui est généré en réponse à une requête utilisateur et qui est conçu spécialement pour un utilisateur donné à un moment donné : un document virtuel n'a pas de raison d'être en dehors de ce contexte. Watters, dans [Wat99], a caractérisé un document virtuel comme suit : "Un document virtuel est simplement un document pour lequel aucun état persistent n'existe et pour lequel certaines ou toutes les instances sont générées à l'exécution. [...] Un document électronique est à la fois défini comme du contenu et des liens associés à ce document. Un document virtuel peut alors être un ensemble de pages, une applet Java, ou des résultats d'application, et peut ou non avoir des liens associés. Le contenu peut être défini par des tags, un template, un programme, une requête de la base de données, ou par une application." Les éléments présentés par un système de recherche d'information en réponse à une requête constituent un *document virtuel*, quelque soit le format de présentation adopté.

L'objectif de ce chapitre est de présenter les différents travaux de recherche d'information qui considèrent la génération de documents virtuels au sens où nous l'avons définie en introduction et qui renvoient alors ces documents virtuels plutôt que des documents complets en réponse à un besoin d'information. En section 2.1, nous définissons une classification des travaux portant sur les documents virtuels, selon la flexibilité du modèle d'organisation qui les caractérise.

La section 2.2 présente les approches pour lesquelles le modèle d'organisation est fortement contraint, les résultats qui sont présentés à l'utilisateur sont alors intégrés au modèle d'organisation très formellement. La section 2.3 s'intéressent à des approches contraintes également selon un certain nombre de paramètres, mais qui offrent toutefois une certaine flexibilité dans l'organisation. Enfin la section 2.4 fait référence à quelques approches nullement contraintes, les documents virtuels créés via ces mécanismes ne respectent pas un modèle d'organisation contraint comme précédemment. À travers cette étude, nous détaillons plusieurs formats de documents virtuels existants, depuis la simple liste de doxels jusqu'au document virtuel répondant à un schéma d'organisation complexe afin de dégager clairement les avantages mais aussi les contraintes de la génération de documents virtuels plus ou moins complexes.

2.1 Axe d'analyse

La génération de documents virtuels représente un problème complexe : d'une manière globale, on peut se demander comment générer un document virtuel. En fait, il s'agit d'une part de garantir la cohérence du document virtuel en terme de structure argumentative des documents, et d'autre part d'intégrer des informations issues des algorithmes de recherche lors de la génération du document virtuel.

Cette problématique de génération de document virtuel vient ainsi s'ajouter aux étapes du système de recherche d'information : on peut se demander si elle est mise en oeuvre à l'indexation et/ou à l'interrogation, ou bien si elle est développée ensuite, au cours d'une nouvelle étape d'organisation des doxels pertinents.

Un modèle se dégage des différentes approches que nous avons étudiées, il est présenté à la figure 2.1. Un utilisateur exprime un besoin d'information pour lequel il spécifie le contenu attendu ainsi que la forme sous laquelle il aimerait recevoir ce contenu, il fournit en fait un modèle d'organisation. Une recherche est alors effectuée sur la partie contenu de la requête. Le résultat de la recherche est organisé en prenant en compte le modèle de document spécifié par l'utilisateur, et éventuellement d'autres données d'une base de connaissances. À l'issue de cette phase d'organisation, un document réponse virtuel est retourné à l'utilisateur.

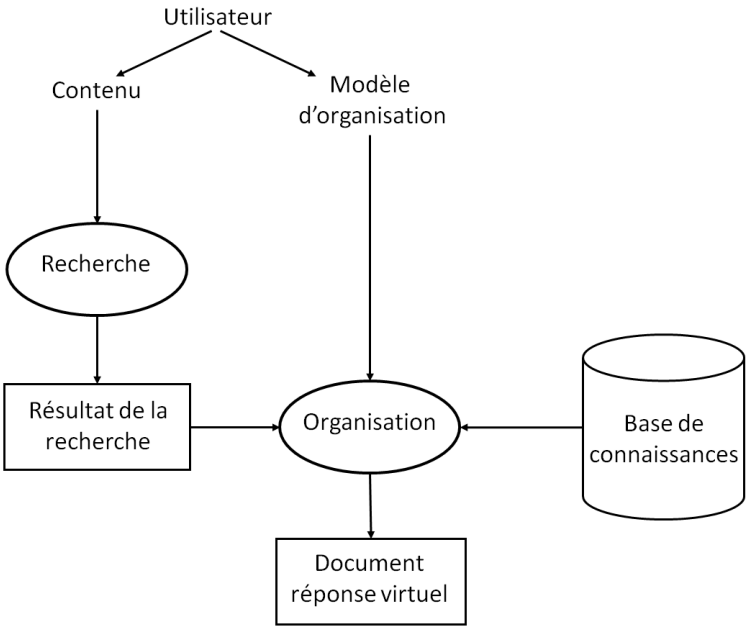

FIG. 2.1 – Génération d'un document réponse virtuel.

L'étude de l'état de l'art que nous avons menée nous permet de classer les approches existantes selon le degré de flexibilité du modèle d'organisation qu'elles proposent, comme le montre la figure 2.2. En effet, le modèle d'organisation fourni par l'utilisateur au moment de sa requête peut-être vu comme l'ensemble des contraintes à considérer lors de la génération du document virtuel réponse. Ce modèle est alors plus ou moins flexible. Notons que la flexibilité du modèle d'organisation est corrélée avec l'implication nécessaire de la part de l'utilisateur (aussi appelé auteur du document virtuel) dans l'approche.

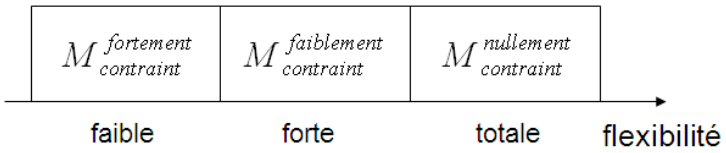

FIG. 2.2 – Axe d'analyse.

Les approches de la classe $\mathcal{M}^{fortement}_{contraint}$ considèrent que le document virtuel résultat suit un modèle d'organisation précis, complètement spécifié. Dans cette classe d'approches, on retrouve des applications qui génèrent des documents techniques répondant à un formalisme très contraint. Dans le domaine éducatif par exemple, les documents produits doivent répondre à de nombreux besoins de réutilisation : un enseignant veut pouvoir réutiliser un support de cours dans un cadre différent l'année suivante.

Les approches de la classe $\mathcal{M}^{faiblement}_{contraint}$ reposent sur un modèle contraint mais qui introduit une part de flexibilité pour prendre en considération d'éventuelles exceptions aux approches fortement contraintes. Les travaux rassemblés dans cette classe obéissent à un cadre formel qui laisse cependant une certaine liberté aux auteurs des documents virtuels. Ce contexte permet d'intégrer des documents administratifs dont le formalisme est avéré, tels que des articles de journaux, mais dont le contenu ne répond pas à aucune règle.

Les approches de la classe $\mathcal{M}^{nullement}_{contraint}$ ne répondent à aucun modèle contraint par des règles comme celles imposées dans les deux classes d'approches précédentes. Un document virtuel est produit, il répond à un besoin donné mais il ne suit pas un schéma d'organisation précis pour cela.

Dans les sections suivantes, nous réalisons une étude des travaux relatifs au domaine des documents virtuels en partant des modèles les plus contraints proposés dans la littérature, et en allant vers les modèles les plus flexibles existants. Les approches présentées ici ne sont pas des approches de recherche d'information *a priori*, mais bien des approches propres à la génération de documents virtuels.

2.2 Approches fortement contraintes

Cette section rassemble des approches dans lesquelles un document virtuel est généré à partir d'un modèle d'organisation fortement contraint. En effet, les éléments d'un document virtuel peuvent répondre à une organisation très précise, supposée donner tout son sens au document.

Pour illustrer ce concept, nous présentons les travaux de Chabert-Ranwez [CR00] qui sont considérés comme une référence dans le domaine et pour lesquels le modèle d'organisation est spécifié à l'aide de règles de grammaire. Nous nous intéressons ensuite à des travaux utilisant des spécifications déclaratives [IG02] et enfin à une étude basée sur des bases de connaissances [TA99].

Dans cette section, un document virtuel est limité à un document pédagogique ou un document journalistique, ce qui permet aux auteurs de spécifier des contraintes très fortes quant au

format des documents, qu'il ne serait probablement pas possible d'envisager dans un domaine où les documents ne répondent pas au même type de règles logiques.

2.2.1 Création de documents par des règles de grammaire

Chabert-Ranwez [CR00] s'est intéressée à la création de documents virtuels dans le cadre de l'enseignement. Pour elle, la création d'un document virtuel repose sur une organisation très précise, qui répond à des règles de grammaire reflétant le schéma traditionnel d'un document pédagogique. Cette application pédagogique s'appuie ainsi sur les études qui ont démontré formellement qu'il existe un schéma pour enseigner.

L'idée de Chabert-Ranwez consiste à fixer des règles qui permettent de spécifier de manière stricte les différents rôles des doxels attendus dans le document virtuel à créer. Ces règles sont présentées dans la table 2.1 en utilisant la notation BNF étendue.

```
R0   :  <Document pédagogique> ::= <Début> <Corps> <Fin>
R1   :  <Début> ::= <Résumé> <Tables des matières> <Introduction>
R2   :  <Corps> ::= <Instruction descendante> | <Instruction ascendante>
R3   :  <Fin> ::= <Conclusion> [<Références>] [<Glossaire>] [<Annexes>]
R211 :  <Instruction ascendante> ::= {<Exemple> | <Contre-exemple>} |
        <Illustration>} <Théorisation> [<Théorisation>] {<Exercice>} |
        [<Instruction ascendante>]
R212 :  <Théorisation> ::= <Théorème> | <Description> | <Définition> |
        <Preuve> | <Formule>
R221 :  <Instruction descendante> ::= <théorisation> [<théorisation>]
        {<Exemple> | <Contre-exemple> | <Illustration>} {<Exercice>} |
        [<Instruction descendante>]
```

TAB. 2.1 – Règles d'organisation pour des documents pédagogiques [CR00].

La règle `R0` spécifie par exemple qu'un document pédagogique est constitué d'un `Début` de document, puis d'un `Corps` de document et d'une `Fin` de document. Le `Corps` d'un document pédagogique est décrit par une `Instruction descendante` ou une `Instruction Ascendante` (règle `R2`). En fait, ces règles modélisent différentes pédagogies, respectivement celle qui part de la théorie puis l'illustre, versus celle qui à partir d'exemple construit la théorie.

Pour respecter le schéma d'enseignement, les doxels retrouvés par la recherche d'information doivent alors être imbriqués dans cette structure de document virtuel formellement définie par les règles de grammaire. Dans ce but, la caractérisation initiale des documents parmi lesquels la recherche est effectuée est une étape indispensable, afin d'exploiter ces données pour produire le document virtuel réponse.

Indexation des briques d'information D'après Chabert-Ranwez [CR00], tout document initial possède les caractéristiques suivantes qu'il faut extraire à l'indexation :

– Il est composé de une ou plusieurs parties [Roi99] appelées briques d'information (BI) par la suite. Chabert-Ranwez a donné la définition suivante d'une brique d'information : "Une Brique d'Information (BI) est un fragment de document, disponible sous (au moins) un

média, caractérisé par un modèle conceptuel, i.e. une représentation abstraite, et pouvant être inséré dans un document réel. Ces briques peuvent être composites, c'est-à-dire subdivisées en plusieurs sous-briques, sous réserve que les briques ainsi obtenues aient un modèle conceptuel consistant. La consistance traduit la cohérence de leur sémantique par rapport à la description du domaine auquel elles se réfèrent ainsi que la signification plausible de la brique" ;
- Chacune de ces parties peut avoir un style particulier, et joue un rôle particulier au sein de la narration. On peut considérer que le style de chaque partie est fortement dépendant de son rôle (l'illustration d'une robe rouge sera certainement une image), mais également que le rôle que va jouer la partie est fortement lié à son style (si l'on dispose du texte ayant pour commencement "définition : ", on peut difficilement lui faire jouer le rôle d'exemple). Ce rôle dépend entre autres choses de la place de la partie considérée dans le document ;
- Chaque partie possède des caractéristiques propres - la taille d'une image, d'un texte ou la durée d'un enregistrement, d'une vidéo ;
- Ces différentes parties sont organisées suivant une stratégie argumentative précise ;
- Entre chaque partie il y a des transitions qui expriment des liens de causalité, de contradiction, de chronologie. Ces transitions peuvent être implicites (souvent un lien de chronologie) ou explicites (locution adverbiale) et pourraient être générées à l'aide de méthodes de génération de langage naturel par exemple ;
- Le vocabulaire utilisé dans chaque partie est issu d'une ou plusieurs ontologies, suivant que le document traite d'un ou plusieurs domaines.

Précisons que les objets d'information dont parle Sylvie Chabert-Ranwez correspondent aux doxels que le système de recherche d'information retourne. En fait, elle utilise des paires conjonctives conceptuelles, par exemple : $[Sonate](partiede)[Mouvement]$, pour décrire le contenu de ses doxels. Ces paires conjonctives conceptuelles sont pondérées.

Un modèle d'organisation appartenant au domaine qui est traité est alors complété à l'aide des doxels de la base, en utilisant des calculs de distances sémantiques, comme proposé par Crampes [Cra97]. Une fois les informations sélectionnées, elles doivent être organisées. Le rôle des doxels nécessite donc d'être fixé *a priori* pour qu'ils soient catégorisés.

Conclusion Afin de présenter un document virtuel réponse, cette approche s'appuie sur des informations acquises à l'indexation, pour caractériser les doxels retrouvés par le système avant l'étape d'organisation. Ces doxels sont alors assemblés suivant un modèle d'organisation spécifique, qui produit le document virtuel réponse. Bien que le modèle dans son ensemble semble relativement générique, il est fortement dépendant du contexte d'application, dans ce cas le domaine pédagogique. En effet, la caractérisation des doxels nécessite une très bonne connaissance des ontologies et la pondération des doxels est parfois manuelle. Dans le contexte pédagogique, l'auteur peut se permettre de développer une approche fortement contrainte. Cependant, ce type d'approche est très exigente dans le sens où l'auteur est mis à contribution pour pondérer son document par exemple. De même les transitions entre parties de document virtuel nécessiteraient l'utilisation de génération de langage naturel, mais ces aspects n'ont pas été abordés dans ces travaux à cause de leur complexité. Les expérimentations ne sont pas décrites en détails, nous savons simplement que la recherche d'information s'effectue sur la proximité sémantique des éléments, mais aucun résultat n'est présenté, l'auteur s'est focalisée sur la présentation du modèle d'organisation.

2.2.2 Génération de documents par spécifications déclaratives

Pour Iksal et Garlatti [IG02], un document est composé de fragments, et l'organisation de ces fragments pour générer un document virtuel passe par des spécifications déclaratives. Cette approche a été utilisée pour deux applications : un projet dans le domaine du journalisme pour la création de dossiers thématiques, et un projet de création, partage et réutilisation de matériaux pédagogiques.

Iksal et Garlatti proposent un environnement de conception de documents virtuels personnalisables (DVP) qui possède une spécification déclarative des processus de sélection et d'organisation intégrant des critères d'adaptation/personnalisation :

1. la spécification du processus d'organisation est une description déclarative d'un document générique - un graphe orienté dont les arcs sont des relations sémantiques et les noeuds des composants sans contenu -, effectuée par l'auteur, et la spécification de la sélection est réalisée également par l'auteur en définissant les propriétés sémantiques de contenu associées à chaque noeud ;

2. les critères d'adaptation sont spécifiés par des propriétés associées aux documents génériques et sont définies pour le lecteur, le système prend en compte les caractéristiques de l'utilisateur.

Le document générique est spécifié par les auteurs des documents eux-mêmes : chaque auteur modélise son propre document en créant une ou plusieurs structures narratives composées de fragments choisis ; ces structures sont un ensemble de composants et de relations. Des méta-données permettent de spécifier le contenu des noeuds, elles indiquent le concept correspondant au domaine auquel les fragments s'identifient, ainsi que le type de fragment dont il s'agit ou d'autres informations sur la source du fragment. La spécification du contenu d'un noeud peut être donnée par une requête dont le résultat est constitué de tous les fragments pertinents pour décrire ce noeud, ou l'auteur du document virtuel peut sélectionner tous les fragments pertinents pour décrire ce noeud et le système génère alors une description commune aux fragments.

L'étape d'assemblage consiste en l'instanciation de cette structure à l'aide des spécifications de contenu et des schémas des méta-données pour générer un document XML puis un document HTML.

Conclusion Cet article est essentiellement orienté sur les aspects de personnalisation et ne fournit pas de résultats quantitatifs relatifs à la génération de document virtuel, mais l'utilisation de spécifications déclaratives correspondant à un format de *template* nous paraît intéressante. Cependant, ces travaux montrent une fois de plus que la génération fortement contrainte de documents virtuels est pesante pour les auteurs, comme elle les oblige à une contribution importante, par des annotations de contenu de documents dans le cas présent. Dans un contexte de recherche d'information où le passage à l'échelle est une réalité, envisager des annotations à la main serait une aberration.

2.2.3 Construction de supports de cours à partir de bases de connaissances

Tazi et Altawki s'intéressent dans [TA99] à la création de supports de cours, qu'ils considèrent comme des documents virtuels. Il s'agit une fois de plus d'une application pédagogique.

La conception d'un support de cours repose sur des fragments de cours pré-existants. Dans ce cadre, un document virtuel peut être généré à partir de connaissances sur le domaine d'enseignement, de connaissances sur la pédagogie et de connaissances sur l'élève.

La base de connaissances du domaine est un réseau sémantique qui décrit le contenu de chaque document de la base, ainsi qu'un descripteur qui influencera le choix du document adéquat, dans le sens où il comporte des informations sur l'objectif pédagogique ainsi que le niveau minimum requis pour que l'élève accède à ce contenu. Considérons l'exemple d'un document qui vise à enseigner les bases de la logique binaire, il pourrait être décrit par un objectif, connaître les opérateurs booléens, et un niveau minimum pour être en mesure d'acquérir cette notion : connaître la notion de proposition.

La base de connaissances pédagogiques est constituée de stratégies pédagogiques. Voici l'exemple d'une stratégie pédagogique pour traiter un exercice[18] :

```
# DonnerExercice   → FaireExercice, VérifierExercice, CorrigerExercice
  FaireExercice    → PrésenterExercice, QuestionRéponse
  VérifierExercice → ChercherRéponse, InformerEtudiant
  CorrigerExercice → Expliquer, DonnerSolutionCorrigée
                     | DévelopperSousProblème, Recommencer
                     | DonnerUneIndication, Recommencer
  Expliquer        → ParLExemple | ParDescription | ParAnalogie
```

Cet ensemble de règles rappelle les règles proposées par Chabert-Ranwez [CR00]. En effet, nous restons dans le domaine pédagogique, et donner un exercice (`DonnerExercice`) consiste à d'abord à concevoir l'exercice (`FaireExercice`), puis à le vérifier (`VérifierExercice`), et enfin à le corriger `CorrigerExercice`. La règle `CorrigerExercice` souligne le savoir-faire pédagogique et d'une manière générale, ces règles garantissent le respect des grands principes pédagogiques.

La base de connaissance sur le lecteur donne son niveau de connaissances dans le domaine et sa trace dans le système, ainsi que son profil personnalisé.

Le document virtuel est utilisé dans cette approche pour répondre au besoin d'un élève, en situation d'apprentissage. Dès que cet élève veut accéder à un nouveau cours, la sélection des documents à lui présenter s'effectue en tenant compte de son niveau via le descripteur (connaître la notion de proposition est un préliminaire à l'apprentissage des opérateurs booléens) et en appliquant les règles pédagogiques élémentaires qui guident l'apprentissage.

[18] Dans l'exemple, | signifie *OU* et la virgule signifie *ET*.

Connaissant ce processus de génération de documents virtuels, le projet de Tazi et Altawki consiste à remplir une base de documents pédagogiques avec de nouveaux supports de cours, qui sont des modifications de supports de cours existants, en les adaptant au niveau des élèves : ainsi un élève qui connaît déjà la notion de proposition ne se verra pas proposer le même document d'apprentissage des opérateurs booléens qu'un élève qui n'aurait pas déjà acquis la notion de proposition.

Conclusion Les travaux de Tazi et Altawkir reposent sur ce contexte contraint que représente le domaine de l'enseignement. Tout comme les travaux de Chabert-Ranwez, ces travaux pourraient être génériques, mais dans un domaine aussi modélisable que le domaine pédagogique. Pour cette approche, la difficulté consiste à remplir la base initiale de supports de cours ; selon toute vraisemblance, cette étape nécessite l'intervention de professeurs pour qualifier les éléments de documents.

2.3 Approches faiblement contraintes

Cette section s'intéressent à des approches de création de documents virtuels qui sont un peu plus flexibles quant à la définition du modèle d'organisation des éléments qui constituent les documents virtuels, comme l'interpréteur dans [PVH98] qui se veut générique pour faciliter la réutilisation d'informations. Cette flexibilité devrait entre autres réduire le nombre d'interventions humaines pour compléter l'indexation des documents.

Nous développons d'abord une approche par modules fixes et/ou contraints décrite dans [BVNN+02] puis nous nous intéressons à une approche de génération de documents virtuels basée sur des plans de discours [DGM+98]. Enfin, nous présentons l'approche des listes de documents résultats comme un exemple de document virtuel.

Dans cette section, un document virtuel est faiblement contraint comme le contenu des fragments de documents virtuels n'est pas toujours complètement spécifié, bien que chaque fragment respecte un formalisme défini. Les applications présentées dans ce cadre sont encore fortement dépendantes du domaine auquel elles se rattachent, sauf pour la liste de documents résultats vue comme un document virtuel.

2.3.1 Création de documents par des modules contraints et/ou flexibles

Dans [BVNN+02], Boukottaya, Vanoirbeek, Nguyen Ngoc, Rekik et Zeramdini utilisent la notion de "module" pour définir la décomposition d'un document virtuel. À chaque module d'un document virtuel donné correspond une description de sa structure et de son contenu. Une instance d'une structure modulaire est appelée "document composite", ce sont des fragments de documents. Définir ces structures modulaires permet d'augmenter la réutilisabilité et d'améliorer la collaboration, en facilitant l'échange de fragments de documents. Cette démarche peut être vue comme la définition d'un contrat contraignant la décomposition du futur document entre plusieurs auteurs.

En effet, les objectifs des auteurs sont de favoriser la combinaison et la réutilisation des fragments de documents structurés dans la génération collaborative de documents virtuels structurés. Leur projet est essentiellement développé dans un contexte pédagogique.

L'idée des auteurs consiste à utiliser un langage de modèle de document flexible (FDM), qui définit la syntaxe d'une famille de documents XML auxquels il faudra se conformer dans la description. La spécification FDM est elle-même un document XML.

La flexibilité a été introduite en partant du constat qu'un concepteur n'a pas toujours une idée précise de la définition de certains modules. Certes, la structure globale d'une classe de document peut être définie, mais la micro-structure peut rester imprécise ; un "module contraint" est alors défini, sa structure n'est pas décrite par une séquence d'éléments et d'attributs, mais par un ensemble de contraintes appelé "contrat", ce qui introduit la notion de flexibilité. En plus des modules contraints, il existe les "modules fixés" qui sont entièrement décrits par leur structure.

Un document est donc décrit selon la syntaxe proposée à la table 2.2, conforme à la syntaxe XML. Un document est composé de modules fixés et de modules contraints ; ce sont les modules contraints qui permettent d'introduire la notion de flexibilité, avec une déclaration de contrat (Contract_dec) au lieu d'un contenu fixe (Content_Dec).

Documentclass_dec →	< DocumentClass name = namevalue > (GlobalAttribute_dec)* (Module_dec)+ < DocumentClass >
Module_dec →	((FixedModule_dec)|(ConstrainedModule_dec))
GlobalAttribute_dec →	< GlobalAttribute name = namevalue type = attype use = (optional | required | fixed | default) value = attvalue/ >
FixedModule_dec →	< FixedModule name = namevalue > (GlobalAttribute_dec)* (Content_dec) < FixedModule >
ConstrainedModule_dec →	< ConstrainedModule name = namevalue > (GlobalAttribute_dec)* (Contract_dec) < ConstrainedModule >
Contract_dec →	< contract > ((SConstraint_dec)|(DepConstraint_dec)) < contract >

TAB. 2.2 – Syntaxe des documents

Ainsi, les auteurs veulent proposer un modèle qui supporte à la fois la réutilisation et la collaboration en se basant sur des concepts de modularité et de flexibilité.

Conclusion Cet article est consacré au contexte pédagogique, ce qui garantit un schéma de document virtuel relativement stable. On peut se poser la question de l'utilisation du langage de modèle de document flexible en dehors de ce contexte. Ce modèle pourrait constituer un intermédiaire entre la liste de doxels pertinents et le document virtuel réponse, en offrant une certaine structure ; par contre, les auteurs ne s'intéressent pas du tout au problème de la recherche d'information.

2.3.2 Génération de documents à partir de la langue naturelle

L'article [DGM+98] s'intéresse à la création de documents virtuels en utilisant des techniques de génération de la langue naturelle. Cette approche se limite à la génération de documents virtuels pour décrire ou comparer les objets d'un musée. Pour répondre à cet objectif de discours (*discourse goal*), les auteurs utilisent un plan de discours (*discourse plan*) choisi dans une librairie de plans, qui constitue la ressource linguistique du système. Cette stratégie propre aux techniques de génération de la langue naturelle est adaptée pour supporter l'utilisation d'un environnement hypertexte.

Les auteurs différencient clairement les travaux sur la génération de langue des travaux sur le traitement de documents pour générer des documents : les travaux sur la génération de langue partent habituellement d'informations de base conceptuelle et non de texte, alors que les travaux de génération de textes par traitement de textes utilisent comme source des documents textuels, éventuellement annotés. D'après les auteurs, le problème de l'existence de données pour la génération de langue n'est pas habituellement soulevé, alors que ce problème est crucial pour l'utilisation de telles approches dans des cas réels. C'est dans ce cadre que se situe la proposition de cet article : utiliser des sources de données pour la génération de langue qui viennent de textes. Le contexte applicatif de ces travaux est un système d'information pour un musée, avec génération de textes décrivant des objets du musée.

Un système de génération de langue utilise habituellement en entrée des objectifs de discours et une base de connaissances pour générer un plan de discours. Dans le cadre des travaux décrits, les sources pour la génération de la base de connaissance sont une base de données de 5000 articles, et un thésaurus de types d'objets. À partir de ces informations une liste de quatre étapes permet une extraction de connaissances :

1. extraction des dimensions de l'objet ;

2. extraction des catégories du thésaurus : cette étape vise à retrouver dans quelle catégorie du thésaurus est un objet de la base de données ;

3. extraction de noms propres, matériau, créateurs, lieux et date de création : cette information extraite de la description textuelle des objets. Cette extraction utilise le thésaurus fourni par le musée ;

4. extraction d'information de PARTIE-DE et TYPE. La notion de partie se limite à déterminer les objets en fonction de la collection à laquelle ils appartiennent (par exemple une paire de chaussures d'une collection de chaussures du XIXème siècle). Le TYPE d'objet est facilement extrait d'un champ de la base de données.

On se rend compte ici que les auteurs utilisent beaucoup la base de données fournie par le musée, et que les traitements d'extraction de noms ne sont pas très sophistiqués.

Un plan de discours est généré à partir d'un schéma de discours qui est un *template* de paragraphes qui fournissent une structure, un contenu et un ordre prédéfini. Un exemple de tel schéma est décrit dans [iads03], dans lequel un *template* de discours explicatif est composé de l'exposé du problème, puis de la solution, puis d'explications, et enfin d'une évaluation. Dans [DOMK98], les auteurs décrivent comment caractériser un schéma appelé "CompareAndContrast" pour deux animaux. Ce schéma est décrit par la grammaire suivante :

CompareAndContrast →
LinneaenRelationship CompareProperties
CompareProperties →
CompareProperty CompareProperties
CompareProperties → ∅

Dans ce schéma, une première étape est liée à la comparaison des propriétés suivant la taxinomie de Linneaen. Ensuite, une seconde comparaison est faite sur une liste prédéfinie de propriétés (CompareProperty). Une fois le schéma choisi, une instance de ce schéma est composée à partir de la base de connaissance en utilisant le *template*. Si nous reprenons l'exemple du schéma "CompareAndContrast", on peut donc avoir une phrase de cette grammaire qui est "LinneaenRelationship CompareProperty CompareProperty" s'il existe deux propriétés comparables entre animaux en plus de la taxinomie. Le détail de ces éléments, en dehors de toute contrainte sur la syntaxe de la génération de texte, contient toutes les informations pour créer le texte par la suite. Ensuite, un composant de création de surface utilise le plan de discours, un lexique et une grammaire pour générer le texte. Dans leurs travaux, les auteurs définissent dans le plan de discours les moyens de représenter des liens hypertextes. Quand un lien est sélectionné par un utilisateur, l'action de l'utilisateur est utilisée comme entrée pour définir le but du discours utilisé pour générer le document suivant.

Conclusion On remarque que dans ces travaux une grande partie des éléments sont construits manuellement : les schémas de discours (*templates*), la traduction de schéma de discours vers le plan de discours. Dans le cas de nombreux schémas de discours (passage à l'échelle) cette approche devient difficile à contrôler. De plus, la mise en place de la base de connaissances nécessite d'avoir beaucoup d'informations de bonne qualité ainsi que des connaissances additionnelles (thésaurus) par exemple.

Un point qui est cependant intéressant à noter est que dans l'approche considérée, il est possible de tenir compte de l'historique du discours pour améliorer l'organisation des réponses : il est en particulier possible d'indiquer explicitement des éléments comme "dans l'item précédent" pour faciliter la lecture. Sans pour autant proposer des solutions identiques, il est intéressant de se poser la question pour savoir si cet aspect pourrait être utile dans le cas de l'organisation des réponses en temps que contexte de présentation d'un document. Ce genre de "signe" est souvent proposé dans les navigateurs Web par un changement de couleur des ancres dans les pages Web pour distinguer les pages visitées de celles non encore vues, mais on peut se poser la question de raffiner cela dans notre cas.

2.3.3 Liste considérée comme un document virtuel

Une première approche consiste à considérer les listes de résultats habituellement renvoyés par un système de recherche d'information comme des documents virtuels. Les utilisateurs se voient proposer les documents qui répondent le mieux à leur requête, sous forme de liste. Le modèle consiste à associer à chaque document résultat un snippet ; sous les documents résultats, des extraits de contenu apparaissent par exemple dans le cas du moteur de recherche Google[19].

Un "snippet" Google est un court extrait de page répondant à la requête de l'utilisateur, qui lui permet de voir les termes de recherche qu'il a utilisé en caractères gras et en contexte dans les résultats Google, et de sélectionner la page qui les intéresse. En général, les utilisateurs sélectionneront plus volontiers (et plus rapidement !) une page qui est présentée avec des termes dans le contexte du document[20].

Conclusion L'ensemble des documents pertinents et des snippets associés ces documents constitue un document virtuel.

2.4 Approches nullement contraintes

Dans cette troisième classe d'applications de génération de documents virtuels, les modèles d'organisation sont très flexibles.

Nous nous intéressons à un problème de génération de résumé [MSB97], le nombre de paragraphes étant fixé, mais aucune contrainte d'organisation n'est précisée autrement. Nous proposons ensuite une approche de création de documents Web virtuels [DB99] puis une approche qui permet l'interconnection de documents pour créer un document virtuel.

Les approches présentées dans cette section peuvent *a priori* s'appliquer à n'importe quel domaine. Elles ne sont d'ailleurs pas du tout orientées recherche d'information, ni spécifiques au domaine pédagogique comme beaucoup d'approches pré-citées.

2.4.1 Génération de résumé de document

Dans [MSB97], M. Mitra, A. Singhal et C. Buckley proposent une méthode de résumé de documents par extraction de paragraphes. Alors qu'ils ont le même objectif, celui de refléter tout le document, deux humains génèrent des extraits différents de ce document, d'où l'idée de vouloir rendre cette opération automatique.

Les documents sont indexés sur la base du modèle vectoriel et une carte des relations entre paragraphes et des segments par thèmes est produite qui permettra de ré-assembler les paragraphes.

La contrainte pour générer le document virtuel porte sur le nombre de paragraphes que comportera le résumé. Si n paragraphes doivent constituer le résumé, on va choisir les n paragraphes qui ont le plus grand nombre de connexions, au sens de Salton [SSBM96] (connexions basées sur la mesure de similarité du cosinus, supérieure à un seuil fixé), avec les autres paragraphes (*bushy nodes*), comme ils sont *a priori* plus centraux et les concaténer pour former un "bushy path". La transition entre ces paragraphes n'est cependant pas assurée par ce simple mécanisme, et la lecture du résumé peut s'avérer difficile.

[19] Google : http ://www.google.fr/
[20] http ://www.google.fr/intl/fr/remove.html

Les "bushy" paragraphes ne permettent pas d'extraire des paragraphes ayant trait à un sujet en particulier si le thème est très spécialisé. C'est pourquoi, les auteurs proposent de poser une contrainte, celle de créer des "bushy paths" pour chaque segment ("segmented bushy paths"), et de leur ajouter des paragraphes pour atteindre le nombre de paragraphes souhaités dans le résumé, proportionnellement à leur taille relative dans le texte.

Une autre technique consiste à garder le paragraphe le plus interconnecté ou le premier paragraphe et à lui concaténer ceux qui lui sont le plus connectés et le suivent dans le texte, pour former un tout intelligible, un "depth first path". Mais, cette technique ne permet pas de couvrir tous les aspects d'un article.

Dans [MSB97], l'heuristique que les auteurs de documents introduisent un thème dans les premiers paragraphes d'une partie de document est prise en compte : cette pratique garantit une meilleure lisibilité. Afin de constituer un meilleur résumé, il est alors possible d'adjoindre le paragraphe introduction d'un segment au paragraphe ayant le plus de connexions avec les paragraphes du-dit segment pour former un "augmented segmented bushy path", toujours en fonction du nombre de paragraphes souhaités pour constituer le résumé.

Des expériences ont été menées pour tester ces propositions de résumé auprès d'utilisateurs. La solution la plus satisfaisante consistait à demander à un utilisateur-rédacteur de produire un résumé manuellement en suivant des instructions bien claires qui permettaient d'obtenir un résumé du même type que celui produit par la machine. Ensuite, un utilisateur-lecteur compare les deux résumés à sa propre vision du résumé idéal pour donner son degré de satisfaction. Cette mesure consiste en fait à calculer le recouvrement entre les deux résumés générés.

En pratique, deux utilisateurs sont rédacteurs. Les analyses montrent un taux de recouvrement de seulement 46% entre les deux extraits manuels ainsi produits, résultats probablement dus au fait que les utilisateurs peuvent marquer les paragraphes qui leur paraissent les plus importants en première lecture, puis ensuite décider de ne trier que dans ce lot de paragraphes préselectionnés afin de réduire leur sélection au nombre de paragraphes préconisés pour le résumé.

De manière automatique, les meilleurs extraits sont donnés par les méthodes de "global bushy paths" et "augmented segmented bushy paths" avec un taux de satisfaction utilisateur de 55% et le système semble obtenir des résultats équivalents à une sélection humaine, avec un taux de de recouvrement 46%, ce qui permet d'affirmer que le système concurrence le résumé humain.

En revanche, une autre technique consiste à former le résumé en extrayant les premiers paragraphes d'un document, c'est la méthode appliquée par Google ; cette expérience donne les mêmes résultats que le meilleur résumé produit automatiquement, et est bien plus lisible pour un utilisateur : quelle est alors l'utilité du résumé automatique par extraction de texte ?

Conclusion Cet article s'intéresse à générer un résumé de document à partir d'un document unique. Les paragraphes sont assemblés dans l'ordre du texte et sans ajouter de mots ou phrases de liaison entre doxels extraits. En outre, les auteurs ont choisi de réutiliser les paragraphes tels quels, ce qui nous conforte dans l'idée de réutiliser les doxels plutôt que de générer de nouveaux doxels. La question de générer un document virtuel à partir de plusieurs documents n'est cependant pas abordée dans cet article.

2.4.2 Documents web virtuels

L'objectif de l'article [DB99] n'est pas de générer un ou des documents virtuels qui répondent à des requêtes, mais de proposer une manière de structurer des sites Web en se basant sur le contenu des documents. Les auteurs définissent un document Web virtuel comme un ensemble de pages Web d'un site représentant un espace logique d'information.

Un document Web virtuel est associé à une liste d'attributs numériques ou symboliques (provenant d'une ontologie par exemple), et permet de représenter par exemple les métadonnées du Dublin Core[21]. Les attributs peuvent être statiques, ascendants ou descendants le long des liens de composition entre les documents ou les pages Web. Un document Web virtuel est composé d'autres documents virtuels et/ou de pages web. Lors du traitement de requêtes, l'utilisation des attributs permet de poser des requêtes à la fois sur les documents virtuels et les pages Web, par exemple on peut rechercher des documents dont le document virtuel est indexé par "environnement", plutôt que de rechercher toutes les pages Web qui traitent de ce sujet d'après leur contenu. Les auteurs n'ont pas fourni d'explications sur la manière de générer les attributs des documents virtuels et ont favorisé des descriptions manuelles. La structuration proposée par ce travail peut donc à la fois considérer le contenu sémantique des documents ou bien d'autres aspects (comme un auteur par exemple).

Conclusion L'idée source de la proposition de cet article vise à mettre en place des structures lors de l'indexation des documents. Ces structures définissent de nouveaux documents virtuels en dehors de toute requête. Le point faible de cette proposition réside dans la nécessité d'utiliser des données fournies manuellement pour valoriser les attributs, même si l'on peut espérer pouvoir utiliser des sources d'informations existantes pour structurer automatiquement ces pages Web. On peut également se poser la question de la complexité du langage de requête à utiliser pour obtenir des réponses pertinentes.

2.4.3 Documents virtuels à base de documents interconnectés

Les travaux de Das-Neves, Fox et Yu [DNFY05] s'inscrivent dans le cadre de la recherche d'information. Ils proposent un modèle pour retrouver et présenter les résultats pour des requêtes "X et Y" auxquelles il n'est pas possible de répondre par une liste de documents résultats, parce que la pertinence des résultats est évaluée comme trop faible. L'objectif des travaux de [DNFY05] est alors d'estimer la relation entre le sujet X et le sujet Y. Ces travaux peuvent être vus comme un outil pour la génération de documents virtuels dans le sens où des liaisons entre documents sont effectuées, ce qui se rapproche de certaines de nos préoccupations.

Les auteurs proposent de considérer des requêtes qui incluent différents sujets qui sont reliés. Les expérimentations sont menées sur deux sujets par requête. L'idée n'est plus de présenter une liste triée de documents résultats, mais une ou plusieurs séquences de documents qui décrivent un ensemble valide de relations entre les sujets.

L'entrée du système est un ensemble de mots, mais qui représentent deux sujets. La réponse est un réseau de chaînes d'évidences (*chain of evidences*). Chaque chaîne est composée d'une liste de sujets additionnels (*stepping stones*), une chaîne est une explication (*pathway*) pour la connexion entre les sujets de la requête. L'approche est ainsi dynamique, les sujets sont interconnectés au moment de la requête.

[21] Dublin Core : http ://dublincore.org/documents/dcmi-terms/

L'approche pour créer des *stepping stones* et des *pathways* est décrite comme suit :

1. Lancer une recherche par sujets s_1 et s_2, en tentant d'inclure tous les termes des sujets, et si cela n'est pas satisfaisant en terme de taille de réponse, retirer les termes considérés comme moins importants.
2. Création de points d'arrêt
   (a) Retrouver les ensembles des réponses pour s_1 et s_2, appelés E_{s_1} et E_{s_2}.
   (b) Création d'un cluster, C_{s_1} pour E_{s_1} et C_{s_2} pour E_{s_2}.
   (c) Calcul du centroïde CC_{s_1} pour les 10 premiers documents de C_{s_1}, de même pour CC_{s_2} à partir de C_{s_2}.
3. Création de Stepping Stones et de Pathways par
   (a) L'utilisation des centroïdes des points d'arrêt comme deux requêtes pour trouver deux nouveaux ensembles de documents E'_{s_1} et E'_{s_2}.
   (b) Création d'un ensemble de documents intermédiaires, $E'_{s_1 s_2}$, contenant les documents communs à E'_{s_1} et E'_{s_2}.
   (c) Création de connexions
      i. Chaque connexion possède un document de chaque points d'arrêt et un document de $E'_{s_1 s_2}$.
      ii. L'importance d'une connexion est donnée en utilisant la probabilité des documents suivant le sujet et la probabilité d'un document sachant son précédent dans la connexion.
      iii. Elimination des documents qui n'appartiennent à aucune connexion.
      iv. Regroupement des documents restants et étiquetage (basé sur une approche Suffix-Tree Clustering [ZE98] qui utilise des arbres de suffixe dans lesquels un noeud identifie l'ensemble des documents qui partagent une séquence de mots). L'étiquetage sert de stepping stone.
      v. Présentation des pathways.

Comme le cadre applicatif est les documents scientifiques, les auteurs utilisent leur contenu, leurs co-citations et leurs co-références.

Les auteurs proposent une évaluation de leur proposition. Les expérimentations sont menées sur 4 requêtes (paires de sous-requêtes) et 12 sujets. Durant ces expérimentations, les utilisateurs doivent trouver des documents qui servent de connexions entre les deux sous-requêtes. D'après les résultats obtenus, les utilisateurs trouvent davantage de documents de connexion avec l'approche des auteurs comparée à une approche courante de recherche d'information.

Cet article fournit donc un cadre pour déterminer *a posteriori* des relations entre documents, c'est-à-dire que cette approche crée des relations entre documents qui n'existent pas initialement. Et nous pensons pouvoir nous appuyer sur de telles relations dans le cadre de la génération de documents virtuels.

2.5 Synthèse

Dans cette partie, nous avons décrit les problèmes liés à la génération de documents virtuels, sans nous borner au contexte du domaine de la recherche d'information. Nous avons décrit un modèle de génération de documents virtuels couramment utilisé, qui permet de créer des documents virtuels réponses en fonction de résultats de recherche et d'une étape d'organisation basée sur un modèle d'organisation. Cette organisation permet de décrire la structure discursive du document et doit instancier cette structure discursive. Nous avons ensuite dressé un état de l'art sur certaines approches actuelles, en choisissant de séparer les travaux sur la flexibilité du modèle d'organisation qui les caractérise.

L'étude de ces articles a permis de dégager les principaux problèmes relatifs à la génération de documents virtuels, et plus particulièrement le problème lié à l'organisation des documents virtuels. Le tableau 2.3 récapitule les différents travaux étudiés dans cette partie et donne la classe d'approches à laquelle ils appartiennent.

Approche \ *Classe*	$\mathcal{M}_{contraint}^{fortement}$	$\mathcal{M}_{contraint}^{faiblement}$	$\mathcal{M}_{contraint}^{nullement}$
Règles de grammaire [CR00]	×		
Spécifications déclaratives [IG02]	×		
Bases de connaissances [TA99]	×		
Modules contraints et/ou flexibles [BVNN+02]		×	
Langue naturelle [DGM+98]		×	
Liste		×	
Résumé de documents [MSB97]			×
Documents Web virtuels [DB99]			×
Documents interconnectés [DNFY05]			×

TAB. 2.3 – Les travaux par classe de modèle d'organisation des documents virtuels.

Dans la classe d'approches fortement contraintes $\mathcal{M}_{contraint}^{fortement}$, les applications étudiées [CR00, BVNN+02, IG02] appartiennent au domaine pédagogique et définissent un principe de construction basé sur un modèle de documents : les *templates*. Un template fournit un squelette pour les documents virtuels, et le cadre pédagogique est suffisamment contraint pour avoir une vision quasie exhaustive de l'organisation de tels documents.

Les approches plus faiblement contraintes de la classe $\mathcal{M}_{contraint}^{faiblement}$ permettent d'adoucir la rigidité des templates dans une approche pédagogique collaborative [TA99] par le fait que ces templates sont généraux et que plusieurs auteurs peuvent, à partir d'un même template, créer des documents virtuels très différents. Il en va de même pour l'application [DGM+98] qui considère l'introduction d'un plan de discours pour générer un document virtuel, le plan de discours se veut généraliste. L'approche par listes est très généraliste, et bien connue des utilisateurs de systèmes de recherche d'information.

Parmi les approches nullement contraintes de la classe $\mathcal{M}_{contraint}^{nullement}$ et non limitées au domaine de la pédagogie, les travaux [MSB97] mettent en exergue les grandes difficultés à construire automatiquement des documents à partir de fragments de documents qui sont compréhensibles. Cette difficulté est diminuée dans le cas où on ne tente pas de recréer de toutes pièces des documents, mais où l'on se "limite" à considérer des liens entre les doxels comme dans [DB99] ou dans [DNFY05].

Les approches étudiées dans l'état de l'art sont limitées en terme de généralisation à cause des difficultés d'organisation des documents virtuels. Ces approches sont très spécifiques, centrées sur un domaine (éducation, musée), et elles ne sont pas toujours automatiques, ce qui pose problème pour le passage à l'échelle dans d'autres contextes applicatifs moins bien délimités. Par exemple, les templates tels que présentés sont très spécifiques à un domaine fixé *a priori*, même si ces templates sont suffisamment généraux pour permettre une certaine flexibilité lors de la création des documents virtuels en terme de doxels composants. La création de ces templates doit donc marier à la fois généralité et spécificité, ce qui est difficile à faire dans un cadre général. De même, une fois une organisation et un template instanciés, l'étape de génération du document virtuel textuel facilement lisible est également une étape très compliquée dans un cadre général, ce qui nuit à une utilisation de telles approches dans de nombreux contextes applicatifs.

Dans le cas de la recherche d'information, nous ne visons pas *a priori* un cadre applicatif spécifique, et donc l'utilisation de templates ne paraît pas être une solution envisageable. Par contre, comme l'ont montré les travaux [DB99] hors traitement de requête et [DNFY05] dans le cas de traitement de requêtes, la mise en place d'organisations plus simples, sans utiliser de traitements linguistiques sur les aspects discursifs des documents, semble être une solution viable qui permette le passage à l'échelle aussi bien en terme de rapidité de traitement que de contextes applicatifs.

Deuxième partie

Proposition d'un modèle pour des documents virtuels

Préambule

Notre cadre d'étude est la recherche d'information. Usuellement, un système de recherche d'information tente de répondre à des besoins utilisateur par des documents complets. Nous proposons dans cette partie un modèle de recherche d'information pour des documents virtuels. Comme le signale Salton [SJC93], le fait de renvoyer des éléments de documents permet de satisfaire davantage le besoin d'information des utilisateurs.

Les chapitres 1 et 2 nous ont permis de décrire l'existant d'une part dans le domaine de la recherche d'information sur des documents structurés et d'autre part en ce qui concerne la production de documents virtuels :

- à partir de documents structurés, un système de recherche d'information sur des documents structurés renvoie les éléments de documents pertinents pour un besoin utilisateur donné. Nous avons considéré qu'une simple liste de doxels pertinents constitue un document virtuel : cette approche nullement contrainte, comme présenté dans l'état de l'art sur les documents virtuels, ne demande aucun effort à l'utilisateur, mais ne produit "rien de nouveau" pour favoriser l'exploration des résultats en dehors la liste ;
- à partir d'un besoin d'information, d'éléments de documents quelconques et d'un modèle d'organisation, un système d'organisation peut assembler des éléments de documents de sorte qu'un document virtuel soit produit. L'étude de l'état de l'art sur les documents virtuels a d'abord souligné la complexité d'organisation et la nécessité d'intervention des utilisateurs pour produire des documents virtuels fortement contraints, répondant à un modèle d'organisation laissant peu de libertés et l'application de tels modèles n'est envisageable que dans des domaines très spécifiques, tels que l'enseignement. D'un autre côté, cet état de l'art sur les documents virtuels propose un processus de création de documents virtuels : sous réserve de définir de nouveaux modèles d'organisation, peut-être moins contraints, la génération de documents virtuels pourrait s'effectuer de manière plus automatisée. Cette structuration est très complexe et nécessite idéalement une compréhension profonde des documents ainsi que des outils de génération de documents.

Le défi dans la production de documents virtuels repose sur l'organisation de ces éléments de documents étiquetés pertinents, pour créer un tout cohérent. Notre objectif consiste à favoriser la navigation des utilisateurs dans l'espace des résultats. Ainsi, nous voulons rassembler les doxels semblables ou complémentaires et permettre aux utilisateurs de satisfaire leur besoin d'information plus rapidement. Nous proposons dans ce but d'adopter une solution intermédiaire, un modèle d'organisation faiblement contraint $\mathcal{M}_{contraint}^{faiblement}$ comme présenté à la figure 1, entre les modèles fortement contraints et les modèles nullement contraints. Cette approche n'est pas spécifique à un domaine particulier.

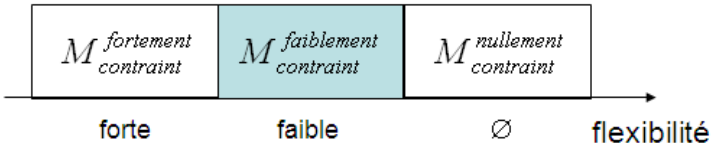

FIG. 1 – Positionnement de nos travaux sur les documents virtuels.

L'état de l'art côté documents structurés a montré que nous savons produire des éléments de documents, qui se veulent plus ciblés que des documents complets, pour répondre aux requêtes utilisateurs. Les approches existantes se sont appuyées sur les caractéristiques de structure et de contenu propres aux documents structurés.

Suite à ces constatations, par rapport à l'état de l'art, notre approche se situe donc comme présenté à la figure 2 à la fois dans les classes $C_{sur\_contenu}$ et $C_{sur\_structure}$:

- nous choisissons de nous baser sur les caractéristiques des éléments de documents à la fois pour retrouver les éléments les plus pertinents pour la recherche d'information, puis pour favoriser la navigation des utilisateurs au niveau des documents virtuels.
- nous proposons d'exploiter les liens entre éléments de documents et de les caractériser par des valeurs d'exhaustivité et de spécificité relatives (section 4.3.2) qui permettront de guider l'utilisateur dans sa navigation, selon qu'il souhaite retrouver toutes les informations qui répondent à son besoin d'information ou bien qu'il souhaite retrouver les éléments qui s'intéressent exclusivement à sa requête.

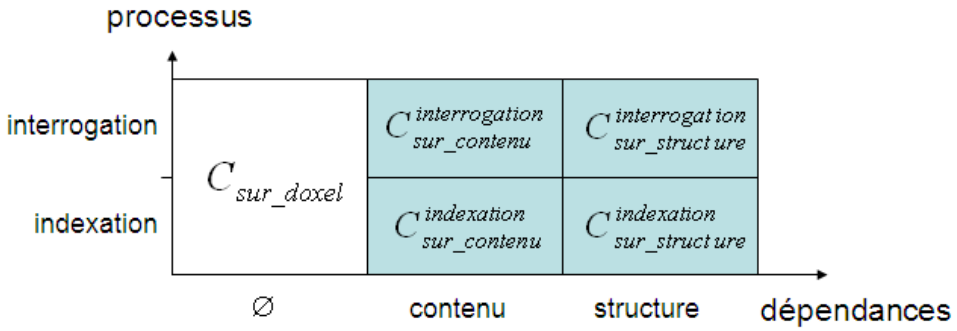

FIG. 2 – Positionnement de nos travaux sur les documents structurés.

Nous voulons construire un modèle de recherche d'information pour des documents virtuels basé sur un corpus de documents structurés. L'organisation des éléments de documents étiquetés pertinents suite à la recherche d'information devra produire un document virtuel répondant à notre objectif, comme présenté à la figure 3 (qui est une copie de la figure 4 présentée en partie *Introduction* page 9). Dans cette figure, les étapes qui incluent des éléments relatifs à l'organisation sont marquées par un astérisque : l'organisation est préparée dès l'indexation des documents, par la mise en place de liens entre éléments puis au moment de l'interrogation, la fonction de correspondance prend en compte des caractéristiques pour les aspects organisation, avant de finaliser l'organisation pour générer les documents virtuels résultats.

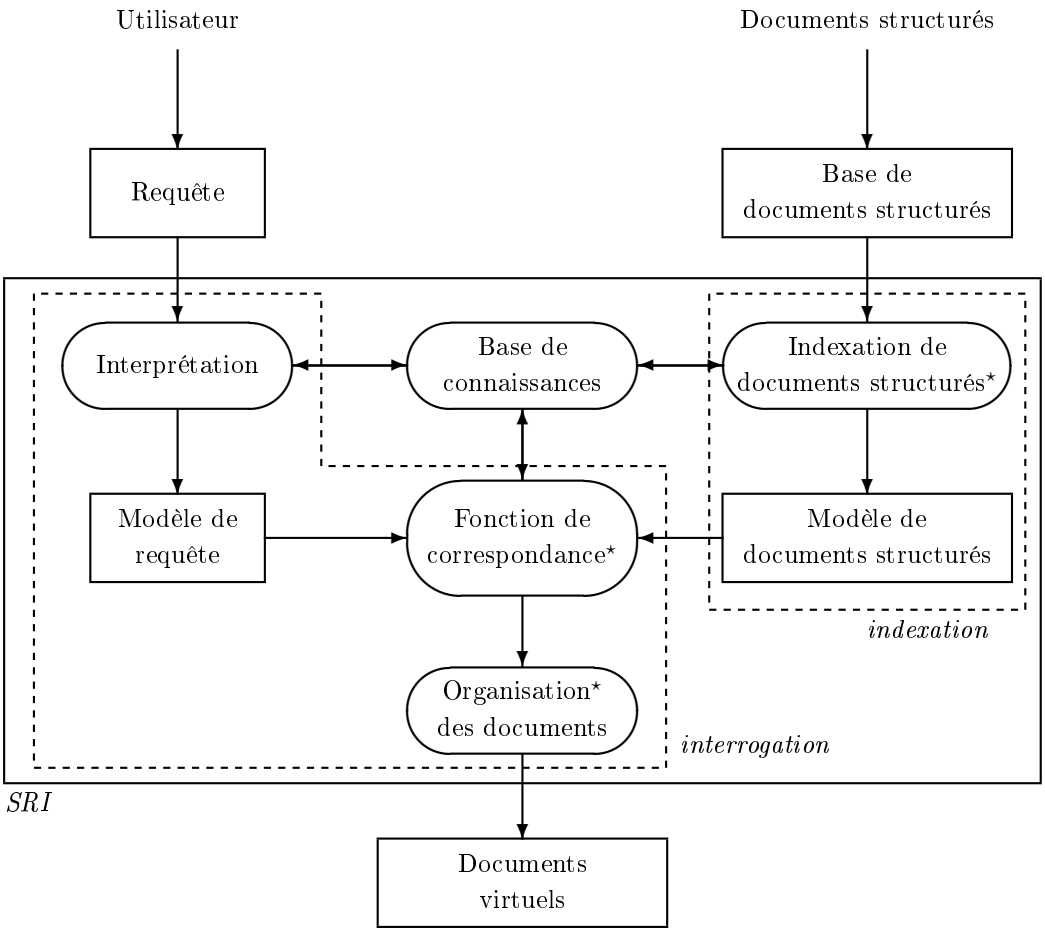

FIG. 3 – Système de recherche d'information pour des documents virtuels.

Afin d'atteindre l'objectif que nous venons de décrire, nous proposons un modèle de recherche d'information pour des documents virtuels. Avant de présenter les parties indexation et interrogation de ce modèle aux chapitres 4 et 5, nous définissons dans un premier temps le corpus sur lequel repose ce modèle au chapitre 3. Cette étape est fondamentale dans le sens où la génération des documents virtuels réponses dépend de l'indexation et par extension des propriétés du corpus. Le chapitre 6 propose une instanciation de notre modèle sur des documents structurés XML.

Chapitre 3

Corpus de documents structurés

Sommaire

Dans la partie I consacrée à l'état de l'art, nous avons d'abord souligné l'intérêt des documents structurés pour la recherche d'information. Par la suite, nous avons montré que la génération de documents virtuels ne peut que profiter d'un prédécoupage des documents du corpus pour compléter des schémas d'organisation prédéfinis, comme rappelé en *Préambule*. Nous choisissons par conséquent de décrire un corpus de documents structurés pour notre modèle.

Ce chapitre décrit formellement le corpus, noté $\mathcal{C}_{tot}$, sur lequel repose le modèle de documents structurés et le modèle d'interrogation présentés respectivement aux chapitres 4 et 5.

Définition 4. *Le corpus total, $\mathcal{C}_{tot}$, est l'ensemble de tous les doxels d (en anglais document elements) de la collection de documents structurés sur lesquels porte le processus de recherche.*

Le chapitre 1 a permis de déterminer les éléments utiles pour la recherche d'information sur des documents structurés. Le corpus sur lequel porte la recherche est ainsi entièrement défini par :

$$CORPUS =< \mathcal{C}_{tot}, \mathcal{C}_{feuille}, E_{type}, f_{type}, f_{contenu}, R_{comp}, E_{rel} >$$

Les éléments suivants sont décrits dans la suite de ce chapitre :
- $\mathcal{C}_{tot}$ est l'ensemble des doxels de la collection de documents structurés,
- $\mathcal{C}_{feuille} \in \mathcal{C}_{tot}$ est l'ensemble des doxels feuilles,
- E_{type} est l'ensemble des types de doxels de la collection,
- f_{type} et $f_{contenu}$ sont les fonctions qui à un doxel associent respectivement son type et son contenu,
- R_{comp} est la relation de composition entre doxels,
- E_{rel} est l'ensemble des relations non compositionnelles.

La prise en compte des liens de structure [PG04, CWC03] est une pratique courante, source d'amélioration des résultats : nous modélisons ces informations à l'aide de relations de composition R_{comp} en section 3.1. D'autre part, le contenu des éléments du corpus ($f_{contenu}$) doit évidemment être représenté et les types des éléments considérés [MSDWZ93] (E_{type}, f_{type}) se sont révélés intéressants pour la recherche ; nous intégrons ces données dans notre modèle comme présenté en section 3.2 (. Nous avons également montré l'intérêt de relations autres que les relations de composition [SZ06] (E_{rel}) et nous proposons de les modéliser en section 3.3.

Toutes les notions abordées dans ce chapitre, et dans les chapitres 4 et 5 qui suivent, sont illustrées sur la mini-collection extraite de l'encyclopédie Wikipédia, $\mathcal{C}^{exemple}$ présentée en figure 3.1. La collection compte 5 documents portant sur la définition de la recherche d'information (*Document 1*), les moteurs de recherche (*Document 2* et *Document 3*), ce qu'est un moteur (*Document 4*) et la définition d'une base de données (*Document 5*). Notons que le *Document 2* est très gros comparativement aux autres documents, et qu'il n'est pas entièrement représenté en figure 3.1. Le corpus total correspondant à la collection $\mathcal{C}^{exemple}$, $\mathcal{C}^{exemple}_{tot}$, comporte 24 doxels d_{ij} représentés ici auxquels s'additionnent les doxels de la partie du *Document 2* non développée. Nous appellerons d_{ij} le $j^{\text{ième}}$ doxel, numéroté dans l'ordre préfixé, du document i de la collection $\mathcal{C}^{exemple}$. Cette numérotation est appliquée par exemple au document en figure 3.2.

Document 1 – Recherche d'information

```
<document nom="Doc1">
<p> <text> La recherche d'information est la science qui consiste à rechercher l'information dans des
documents, dans des </text> <lien vers="Doc5" chemin="/document[1]"> bases de données </lien>.
</p>
<p> La recherche d'information est un domaine historiquement lié aux sciences de l'information et à la
bibliothéconomie qui ont toujours eu le souci d'établir des représentations des documents dans le but
d'en récupérer des informations, à travers la construction d'index. L'informatique a permis le
développement d'outils pour traiter l'information et établir la représentation des documents au moment
de leur indexation, ainsi que pour rechercher l'information. On peut aujourd'hui dire que la recherche
d'information est un champ transdisciplinaire, qui peut être étudié par plusieurs disciplines, approche
qui devrait permettre de trouver des solutions pour améliorer son efficacité.
</p>
</document>
```

Document 2 – Moteur de recherche

```
<document nom="Doc2">
<p> <text> Un moteur de recherche est un logiciel permettant de retrouver des ressources associées
à des mots quelconques. Dès qu'un utilisateur veut effectuer une </text> <lien vers="Doc1"
chemin="/document[1]"> recherche </lien> <text> , il peut utiliser cet outil. </text>
</p>
<p> <text> Les principaux moteurs de recherche sont </text> <lien vers="Doc3"
chemin="/document[1]"> le système Google </lien> <text> , Yahoo!, Baidu, "le Google chinois" qui
monte en puissance, et Live Search, le moteur de recherche de Microsoft. </text>
</p>
...
</document>
```

Document 3 – Google – moteur de recherche

```
<document nom="Doc3">
<p> <text> Le </text> <lien vers="Doc2" chemin="/document[1]/p[1]"> moteur de recherche </lien>
<text> Google, qui a donné le nom à la société Google, est le moteur de recherche d'information sur
Internet le plus utilisé au monde. </text>
</p>
</document>
```

Document 4 - Moteur

```
<document nom="Doc4">
<p> Un moteur est un dispositif transformant une énergie non-mécanique en une énergie mécanique
ou travail (moteur chimique, moteur électrique…).
</p>
</document>
```

Document 5 – Base de données

```
<document nom="Doc5">
<p> Une base de données, usuellement abrégée en BD ou BDD, est un ensemble structuré et
organisé permettant le stockage de grandes quantités d'informations afin d'en faciliter l'exploitation
(ajout, mise à jour, recherche de données).
</p>
<p> Une base de données se traduit physiquement par un ensemble de fichiers présent sur une
mémoire de masse (bien souvent un disque). Certaines peuvent être accessibles via les réseaux, on
parle alors de base de données en ligne.
</p>
</document>
```

FIG. 3.1 – Collection de documents $\mathcal{C}^{exemple}$.

3.1 Structure dans $\mathcal{C}_{tot}$

Par définition, un doxel de $\mathcal{C}_{tot}$ est un élément qui apparaît dans une structure arborescente représentant un document structuré. Si l'on s'intéresse à l'aspect structurel des doxels, un doxel est éventuellement composé de doxels fils et un doxel fils compte au plus un doxel père. Notons que dans ce modèle, nous choisissons de nous limiter à des documents structurés dont les éléments ne se recouvrent pas.

Dans cette section, nous définissons formellement la relation de composition R_{comp} entre doxels, puis nous décrivons ce que nous entendons par doxel *document* et doxel *feuille*.

Relation de composition R_{comp} La structure des doxels d'un corpus est définie par une relation de composition R_{comp} avec $R_{comp} \subset \mathcal{C}_{tot} \times \mathcal{C}_{tot} \times \mathbb{N}^*$. Un triplet $(d_1, d_2, n) \in R_{comp}$ dénote que le doxel $d_1 \in \mathcal{C}_{tot}$ est composé directement par le doxel $d_2 \in \mathcal{C}_{tot}$ et que le doxel d_2 est le $n^{\text{ième}}$ fils du doxel d_1. La relation de composition R_{comp} est non-réflexive, anti-symétrique et non-transitive.

Dans le triplet, le n permet de mémoriser la place des doxels les uns par rapport aux autres ; de la sorte, l'ordre des doxels est conservé, ce qui est utile pour reconstituer le contenu d'un doxel en fonction du contenu de ses composants ou encore pour le cas où notre modèle voudrait davantage pondérer les premiers éléments d'un doxel, comme c'est le cas dans certaines approches.

Exemple Dans $\mathcal{C}^{exemple}$, le doxel d_{12} est composé de deux doxels fils, d_{13} et d_{14} : $(d_{12}, d_{13}, 1) \in R^{exemple}_{comp}$ et $(d_{12}, d_{14}, 2) \in R^{exemple}_{comp}$; d_{11} est le doxel père des doxels d_{12} et d_{15} : $(d_{11}, d_{12}, 1) \in R^{exemple}_{comp}$ et $(d_{11}, d_{15}, 2) \in R^{exemple}_{comp}$.

Doxels feuilles Nous appelons doxel *feuille* un doxel qui n'a aucun fils suivant la relation de composition R_{comp}. $\mathcal{C}_{feuille}$ représente l'ensemble des doxels *feuilles*.

$$\mathcal{C}_{feuille} = \{d \in \mathcal{C}_{tot} |\ \nexists d' \in \mathcal{C}_{tot}\ ;\ \forall n \in \mathbb{N}^*,\ (d, d', n) \in R_{comp}\}$$

L'ensemble des doxels non feuilles du corpus est donné par $\mathcal{C}_{\neg feuille} = \mathcal{C}_{tot} \backslash \mathcal{C}_{feuille}$.

Exemple Sur l'exemple de la figure 3.2, $d_{13}, d_{14}, d_{15} \in \mathcal{C}^{exemple}_{feuille}$.

Doxels documents Nous définissons un *document* comme la racine d'un document structuré formant un tout cohérent au niveau de sa structure logique de composition, correspondant à une instance d'un format de document (un fichier XML par exemple, ou un fichier texte). À un tel document complet correspond un doxel qui n'a aucun père suivant la relation de composition R_{comp}. $\mathcal{C}_{document}$ représente l'ensemble des doxels *documents*.

Exemple Dans $\mathcal{C}^{exemple}_{tot}$, le doxel d_{11} constitue un *document*. Formellement, $d_{11} \in \mathcal{C}_{document}$.

3.2 Caractéristiques des doxels de $\mathcal{C}_{tot}$

À tout doxel, nous devons d'abord être capables d'associer son contenu, car la recherche d'information se base sur une description du contenu des éléments et son type pour apporter des informations complémentaires quant au rôle du doxel considéré dans un document et/ou dans la collection.

Comme nous nous limitons à des doxels textuels, un doxel est décrit par :

- un contenu : ce contenu, une chaîne de caractères[22], est formé du texte correspondant à ce doxel (avec éventuellement des marqueurs de structure) ; la fonction $f_{contenu}$ renvoie pour un doxel son contenu brut :

$$f_{contenu} \ : \ \mathcal{C}_{tot} \ \rightarrow \ \text{String}$$

- un type : le type d'un doxel que nous considérons correspond au rôle que joue ce doxel dans un document ; soit E_{type} l'ensemble des types des doxels, la fonction f_{type} associe à un doxel son type :

$$f_{type} \ : \ \mathcal{C}_{tot} \ \rightarrow \ E_{type}$$

Cette description nous permet donc d'avoir l'information de base pour la recherche d'information, à partir du contenu des doxels. De plus, par l'intermédiaire du type du doxel, nous pouvons également prendre en considération le fait qu'un doxel apparaît plutôt en introduction d'un document ou bien qu'il représente un élément de trop petite taille pour être retrouvé, comme certaines approches le proposaient en pondérant différemment les doxels d'un corpus selon leur type pour améliorer la recherche. Le fait de typer un doxel peut aussi éventuellement permettre de filtrer les doxels retrouvables dans une collection. Notons que dans le modèle ainsi défini, nous ne nous sommes pas intéressés à reproduire ce qui correspond aux attributs des balises XML.

Typiquement, dans le cas d'une application XML, l'ensemble des types de doxels peut être constitué par l'union des noms des marqueurs apparaissant dans les documents du corpus.

Exemple Le *Document 1* de la collection $\mathcal{C}^{exemple}$ est représenté sous forme arborescente en figure 3.2 afin d'illustrer ces concepts. Les doxels $d_{11}, d_{12}, d_{13}, d_{14}$, et d_{15} sont éléments de $\mathcal{C}^{exemple}_{tot}$.

Exemple d'une fonction $f^{exemple}_{contenu}$ basée sur la structure Nous appelons $s(d)$ la fonction qui pour un doxel feuille $d \in \mathcal{C}^{exemple}_{feuille}$ retourne son contenu sous forme d'une chaîne de caractères.

Le contenu d'un doxel de $\mathcal{C}^{exemple}_{tot}$ est défini comme la concaténation des contenus de ses composants par la relation R_{comp}, et ceci en suivant un parcours préfixé, basé sur les indices de R_{comp}.

[22] String est l'ensemble de toutes les chaînes de caractères.

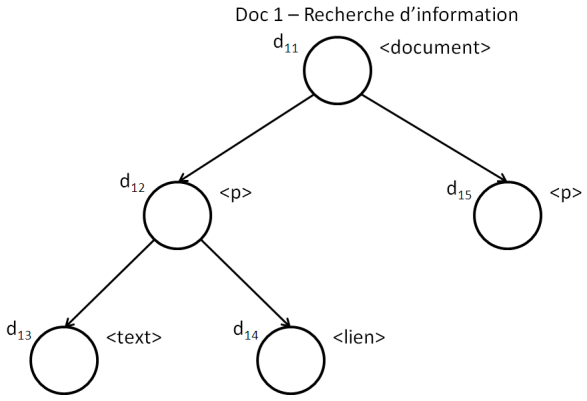

FIG. 3.2 – Représentation du document 1 de $\mathcal{C}^{exemple}$ sous forme d'arbre.

$$f^{exemple}_{contenu} : \mathcal{C}^{exemple}_{tot} \rightarrow \text{String}$$
$$d \mapsto \begin{cases} s(d) & \text{si } d \in \mathcal{C}^{exemple}_{feuille} \\ f^{exemple}_{concat-contenu}(\{d' | (d, d', n) \in R_{comp}\}) & \text{sinon} \end{cases}$$

sachant que la fonction de concaténation $f^{exemple}_{concat-contenu}$ est définie comme suit, pour des éléments ayant le même père d_0 :

$$f^{exemple}_{concat-contenu} : 2^{\mathcal{C}^{exemple}_{tot}} \rightarrow \text{String}$$
$$\mathcal{D} \mapsto \begin{cases} \text{“ ”} & \text{si } \mathcal{D} = \emptyset \\ f^{exemple}_{contenu}(d_1) \cdot f^{exemple}_{concat-contenu}(\mathcal{D} \backslash \{d_1\}) | & \\ \quad (d_0, d_1, n_1) \in R_{comp} \wedge & \\ \quad \nexists d \in \mathcal{D} \backslash \{d_1\} \ ; \ (d_0, d, n) \in R_{comp} \wedge n < n_1 & \text{sinon} \end{cases}$$

Le “$\cdot$” correspond à la fonction de concaténation de chaînes de caractères.
Sur l'exemple de la figure 3.2, le contenu de d_{12} est donné par :

$$f^{exemple}_{contenu}(d_{12}) = f^{exemple}_{concat-contenu}(\{d_{13}, d_{14}\}) = f^{exemple}_{contenu}(d_{13}) \cdot f^{exemple}_{contenu}(d_{14}) = s(d_{13}) \cdot s(d_{14})$$

ainsi, $f^{exemple}_{contenu}(d_{12})$ = (“La recherche d'information est la science qui consiste à rechercher l'information dans des documents dans des”) $\cdot$ (“bases de données”) = “La recherche d'information est la science qui consiste à rechercher l'information dans des documents dans des bases de données”.
L'ensemble des types de la collection $\mathcal{C}^{exemple}_{tot}$ est donné par :

$$E^{exemple}_{type} = \{\texttt{document, p, lien, text}\}$$

Pour le doxel d_{11}, qui est un document, nous avons $f^{exemple}_{type}(d_{11}) = \texttt{document}$ et le type du doxel d_{13} est $f^{exemple}_{type}(d_{13}) = \texttt{text}$.

3.3 Relations non-compositionnelles dans $\mathcal{C}_{tot}$

En section 3.1, nous avons proposé une modélisation du fait que les doxels de $\mathcal{C}_{tot}$ sont reliés entre eux par les relations de structure propres aux documents auxquels ils appartiennent, i.e. des relations de composition. Mais les éléments d'un corpus ne sont pas inter-reliés uniquement par des relations de composition. Des relations autres, telles que des relations de bibliographiques [Sav96], existent et nous choisissons de les refléter dans notre modélisation. Ces relations sont susceptibles d'être utiles pour la recherche d'information. Elles peuvent exister *a priori* ou non. Finalement, les relations non-compositionnelles servent à établir l'environnement d'occurrence des doxels, non plus dans le document, mais dans le corpus.

Nous proposons de définir des relations non-compositionnelles entre doxels. Ces relations sont éléments d'un ensemble de relations E_{rel}, tel que chaque relation rel de E_{rel} est incluse dans $\mathcal{C}_{tot} \times \mathcal{C}_{tot}$. Il peut exister différentes sortes de relations et nous les distinguons dans le modèle car c'est potentiellement utile pour la recherche d'information de les pondérer selon leur nature.

Soit rel une relation de E_{rel}, un élément $(d_1, d_2) \in rel$ dénote le fait que le doxel d_1 est la source d'un lien rel avec le doxel d_2. Nous choisissons délibérément de ne pas expliciter certaines caractéristiques de relations (comme la transitivité, la symétrie), en posant que la construction des relations devra générer ces éventuels aspects : les relations sont donc représentées en extension.

Exemple Dans $\mathcal{C}^{exemple}$, nous posons que la balise < lien > représente des relations non-compositionnelles entre doxels.

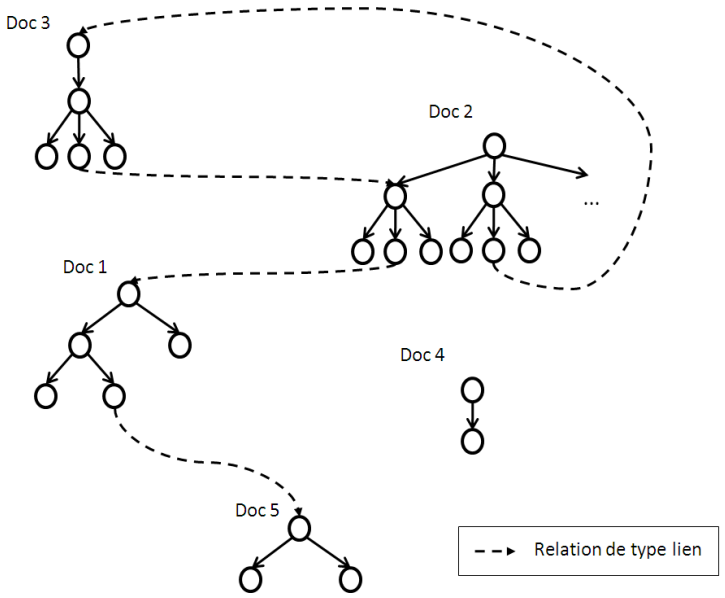

FIG. 3.3 – Relation non-compositionnelle entre doxels de $\mathcal{C}^{exemple}$.

Le résultat de la génération de $E^{exemple}_{\texttt{lien}}$ est représenté à la figure 3.3 ; il existe un lien du doxel d_{14} vers le doxel d_{51}, comme le doxel d_{14} est de type < `lien` >.

Nous appelons $E^{exemple}_{\texttt{lien}}$ l'ensemble des relations de type $<$ `lien` $>$ dans la collection $\mathcal{C}^{exemple}$.

$$E^{exemple}_{\texttt{lien}} \;=\; \{(d_{14}, d_{51}), (d_{23}, d_{11}), (d_{28}, d_{31}), (d_{34}, d_{22})\}.$$

Chapitre 4

Modèle d'indexation de documents structurés

Sommaire

Dans ce chapitre, nous définissons les éléments importants pour effectuer l'indexation des doxels des documents structurés. Ce chapitre correspond à l'étape d'indexation de la figure 4 présentée au chapitre *Introduction* page 9. Il s'agit de modéliser l'indexation du corpus décrit au chapitre 3, c'est-à-dire les doxels ainsi que les relations entre doxels.

Nous nous intéressons à l'environnement des doxels parmi les autres doxels du corpus, et pas uniquement à des aspects de compositions entre doxels. D'une part, nous posons que les aspects structurels des doxels sont porteurs de sens quand au contenu du document, et nous utilisons pour en tenir compte un schéma d'indexation qui se base sur cette structure. D'autre part, pour les aspects non-compositionnels, qui ne sont quasiment pas pris en compte pour les documents structurés dans l'état de l'art, nous les intégrons par l'utilisation de valeurs d'exhaustivité et de spécificité relatives entre un doxel et les éléments auxquels il est relié, c'est-à-dire les éléments de son environnement d'occurrence.

Dans l'indexation des documents structurés, afin de favoriser plus tard l'organisation des résultats, nous choisissons d'exploiter l'environnement d'occurrence des doxels. Dans cette optique, nous utilisons non seulement la structure des documents, mais également des relations non-compositionnelles issues du corpus et/ou établies à l'indexation. Un post-traitement est proposé pour ne garder dans l'index que des éléments retrouvables. Toutes ces notions sont présentées dans les sections suivantes.

Étapes d'indexation Pour l'étape d'indexation, nous avons à notre disposition les doxels, leurs relations de composition structurelle et les relations non-compositionnelles issues du corpus. Nous considérons, à l'instar de ce qui existe dans de nombreux travaux sur les documents structurés [Wil94, PG04], que la structure des documents est un élément primordial à prendre en compte pour l'indexation, c'est pour cela que nous décrivons dans la suite les aspects liés à ces relations de composition structurelle (section 4.1), puis nous présentons l'environnement d'occurrence des doxels en section 4.2 avant de nous intéresser à la caractérisation des aspects non-compositionnels en section 4.3.

De manière graphique, le modèle d'indexation proposé suit le schéma présenté en figure 4.1. Cette figure est essentiellement divisée en deux parties distinctes : d'une part, un cylindre qui représente les données, et à partir du corpus, nous obtenons un index des documents considérés ; d'autre part, des blocs fonctionnels pour les étapes d'indexation structurelle, de définition d'un environnement d'occurrence pour les doxels, d'indexation non-compositionnelles et de filtrage sur les éléments retrouvables, qui utilisent en entrée les éléments qui sont liés par des flèches en pointillés et qui produisent en sortie les éléments pointés par des flèches pleines :

- La première étape d'indexation locale utilise les doxels seuls ; dans une seconde étape l'indexation locale à un document utilise les pré-index des doxels et la structure des documents pour établir un pré-index des documents complets. Ensuite une étape d'indexation globale intègre des informations globales sur le corpus pour construire les index de tous les doxels suivant leur structure de composition.

- La nature des informations non-structurelles diffère de celles de structure, c'est pourquoi nous considérons ces éléments relatifs à des liaisons entre les doxels de manières différentes. La création d'un environnement d'occurrence des doxels permet d'accéder à davantage d'informations sur le corpus et devrait permettre de mieux organiser les doxels en enrichissant les relations entre eux.

– L'indexation non-compositionnelle vise à établir une mesure des doxels en fonction de leur environnement en terme de doxels non reliés par une relation de composition. Elle a pour objectif de déterminer dans quelle mesure un doxel est utilisable dans l'organisation d'une réponse fournie par le système.

– Enfin, nous sélectionnons les éléments retrouvables dans la collection selon plusieurs critères, en appliquant pour cela un filtre à partir de sources d'information.

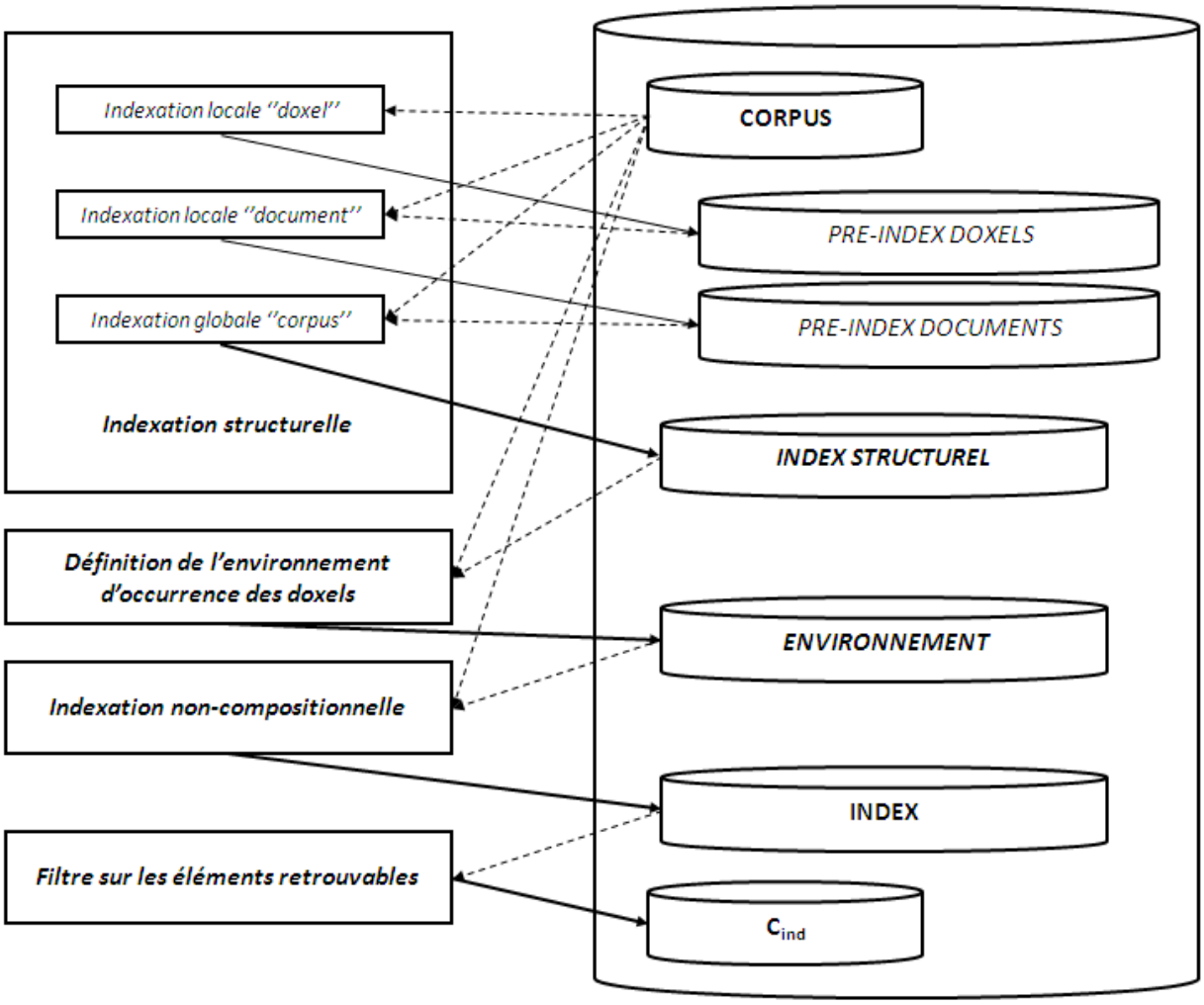

FIG. 4.1 – Modèle d'indexation de documents structurés.

4.1 Indexation structurelle

Rappelons que notre objectif est de prendre en compte le fait que les doxels d'un document structuré ne sont pas indépendants. Ainsi, nous émettons comme hypothèse forte que le contenu d'un doxel est lié au contenu de ses composants, et nous voulons par conséquent pouvoir propager le contenu des doxels feuilles et par extension des doxels composants vers leurs composés

respectifs. Notre solution consiste alors à introduire la notion de propagation, en nous appuyant sur la structure arborescente que nous avons présentée.

Idée Le processus d'indexation de documents atomiques se déroule classiquement en deux étapes :

1. une première étape définit le vocabulaire qui indexe un document et détermine le tf de ces termes ;
2. ensuite, une seconde étape définit l'*idf* de ces termes, et une opération combinant tf et *idf* définit la pondération des termes pour les documents.

Dans notre cas, nous intégrons le fait que les documents sont structurés en définissant une étape supplémentaire, et en modifiant la première étape en tenant compte de la structure compositionnelle des documents. Pour l'indexation de documents structurés, nous posons donc que la modélisation de la propagation entre les doxels se passe comme suit :

1. La première étape consiste à définir une description du contenu de chaque doxel de $\mathcal{C}_{tot}$ en réalisant des propagations d'index, elle se déroule en deux temps :
   - une pré-indexation des doxels feuilles, qui correspond à l'étape "Indexation locale doxel" de la figure 4.1 ;
   - puis une seconde pré-indexation sur les doxels non feuilles qui utilise la relation de composition R_{comp} de manière récursive, et qui correspond à l'étape "Indexation locale document" de la figure 4.1. Dans le cas de l'utilisation de liens de composition structurelle, la structure est arborescente (et donc sans cycle), ce qui évite de faire face à des cycles dans le calcul de termes et des pondérations locales.

   Ce calcul permet d'exploiter les aspects locaux des doxels, qui ne dépendent pas de caractéristiques autres que le contenu du doxel et de ses composants.
2. La finalisation de l'indexation intègre des éléments non-locaux d'indexation, liés à la collection par exemple, et correspond à l'étape "Indexation globale corpus" de la figure 4.1.

Les raisons qui ont motivé ces étapes sont multiples :

- l'indexation est plus rapide car il n'y a pas de duplication de calculs. Prenons le cas où nous voulons calculer les tf pour les termes qui indexent les documents : une solution pour calculer ces tf est de réaliser l'extraction des termes puis le calcul de tf pour chaque doxel considéré indépendamment (en extrayant le contenu textuel de chaque doxel), ce calcul duplique énormément d'efforts de découpage en mots, lemmatisation etc., car les doxels composés contiennent les doxels composants ; la seconde solution est de calculer les tf pour les doxels feuilles, puis de fusionner pour les doxels non-feuilles les listes de termes avec le tf de leurs composants, dans ce cas on n'extrait qu'une fois chaque mot provenant d'un doxel quelconque.
- dans le cas d'une collection qui évolue (ajouts/retraits de documents complets ou bien de doxels dans des documents existants), une partie des calculs ne sont pas à refaire, ce qui est primordial si on veut proposer des approches qui traitent de nombreux documents.
- si différents modèles de propagations sont proposés, il est alors possible de les définir à partir d'éléments communs non-dupliqués. Par exemple, si une propagation se réalise à l'indexation en considérant uniquement les termes communs aux sous-parties, et une autre sur tous les termes, ces propagations utilisent comme base des index sur les feuilles des documents qui sont communs aux deux processus de propagation.

Cette étape a pour objectif de prendre en compte les relations de composition entre les doxels pour établir un pré-index qui va permettre de définir l'index finalisé lors de l'étape globale.

4.1.1 Étape locale - doxel feuille

Durant cette étape, les feuilles de structures de $\mathcal{C}_{leaf}$ sont analysées pour générer un pré-index, défini comme index intermédiaire, suivant un langage d'indexation prédéfini $\mathcal{L}_{pre-index}$. Nous notons *pre-index-dox* la fonction telle que :

$$\textit{pre-index-dox} : \quad \mathcal{C}_{feuille} \quad \rightarrow \quad \mathcal{L}_{pre-index}$$

qui permet pour un élément de $\mathcal{C}_{feuille}$ d'obtenir son pré-index suivant le langage $\mathcal{L}_{pre-index}$. Nous posons que le langage $\mathcal{L}_{pre-index}$ possède un élément "vide", noté $\emptyset_{pre-index}$, qui indique qu'un doxel n'a pas de contenu. $\emptyset_{pre-index}$ est un élément neutre pour la fonction $\oplus_{\textit{pre-index-doc}}$ décrite en section 4.1.2.

$$\text{PRE-INDEX-DOXELS} = \{(d, \textit{pre-index-dox(d)}) | d \in \mathcal{C}_{feuille}\} \subset \mathcal{C}_{feuille} \times \mathcal{L}_{pre-index}.$$

Doc 3 – Google – moteur de recherche

<document>
Le <lien vers: « Doc2Dox2»>moteur de recherche</lien> Google, qui a donné le nom à la société Google, est le moteur de recherche d'information sur Internet le plus utilisé au monde.

</document>

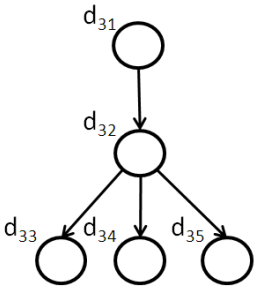

FIG. 4.2 – Le document 3 et sa structure arborescente pour l'étape d'indexation.

À l'issue de l'étape locale - doxel, la table des tf est remplie de telle sorte que seuls les doxels feuilles sont indexés.

Exemple Considérons par exemple le document 3 de la mini-collection reproduit à la figure 4.2. L'ensemble $\mathcal{C}^{exemple}_{feuille}$ des feuilles du document 3 comporte trois éléments, d_{33}, d_{34} et d_{35}.
Soit le langage de pré-indexation $\mathcal{L}_{pre-index} = \mathcal{V}oc \times \mathbb{N}$, où le vocabulaire $\mathcal{V}oc$ associé au langage ne contient pas les mots vides de sens pour l'indexation, tels que "le", "à", "est", etc., et le nombre d'occurrences des termes du vocabulaire est compté.

$$\begin{aligned}
\textit{pre-index-dox}\ (d_{33}) &= \emptyset_{pre-index} \\
\textit{pre-index-dox}\ (d_{34}) &= \{(moteur, 1), (recherche, 1)\} \\
\textit{pre-index-dox}\ (d_{35}) &= \{(Google, 2), (donne, 1), (nom, 1), (societe, 1), (moteur, 1), (recherche, 1), \\
&\quad (Internet, 1), (utilise, 1), (monde, 1), \}
\end{aligned}$$

Ainsi, à l'issue de l'étape locale - doxel feuille, la table des index est remplie comme présenté à la table 4.1, où d_{3j} représente le doxel j du document 3.

	d_{31}	d_{32}	d_{33}	d_{34}	d_{35}
moteur			0	1	1
recherche			0	1	1
Google			0	0	2
donne			0	0	1
nom			0	0	1
societe			0	0	1
information			0	0	1
Internet			0	0	1
utilise			0	0	1
monde			0	0	1

TAB. 4.1 – Table des index pour le document 3 à l'issue de l'étape locale - doxel.

4.1.2 Étape locale - document

L'étape locale sur les documents permet de définir les pré-index pour les doxels non-feuilles structurellement. Cette étape opère par propagation des fils vers le père des pré-index et fusion de ces pré-index. Pour tout élément de $\mathcal{C}_{tot}$, la fonction *pre-index-doc* donne le pré-index après fusion. Pour un doxel feuille d, la fonction *pre-index-doc* fournit le même résultat que la fonction *pre-index-dox* :

$$\forall d \in \mathcal{C}_{leaf}, \textit{pre-index-doc}(d) = \textit{pre-index-dox}(d).$$

L'étape de propagation permet d'obtenir, pour un doxel non feuille donné, les pré-index de tous ses doxels composants structurellement, pour les fusionner suivant une fonction, notée $\oplus_{\textit{pre-index-doc}}$.

Ainsi, pour un doxel d, la fonction *pre-index-doc* est définie telle que :

$$\begin{array}{rccl} \textit{pre-index-doc} & : & \mathcal{C}_{tot} & \rightarrow & \mathcal{L}_{pre-index} \\ & & d & \mapsto & \begin{cases} \textit{pre-index-dox}(d) \text{ si } d \in \mathcal{C}_{leaf} \\ \oplus_{\textit{pre-index-doc}}(\{(\textit{pre-index-doc}(d'), n) | (d, d', n) \in R_{comp}\}) \text{ sinon} \end{cases} \end{array}$$

sachant que la fonction $\oplus_{\textit{pre-index-doc}}$ effectue la fusion des index des doxels récursivement.

Un exemple très simple pour $\oplus_{pre-index-doc}$, dans le cas d'une représentation utilisant le modèle vectoriel de recherche d'information sans tenir compte de la position n des composants structurels d'un doxel, consiste à faire la somme, pondérée par $1/n$ des vecteurs des doxels composants, récursivement.

Le résultat de l'étape locale sur les documents est l'ensemble de couples de doxels d pré-indexés :

$$\text{PRE-INDEX-DOCUMENTS} = \{(d, \textit{pre-index-doc(d)}) | d \in \mathcal{C}_{tot}\} \subset \mathcal{C}_{tot} \times \mathcal{L}_{pre-index}.$$

Revenons sur notre exemple en utilisant la fonction $\oplus_{pre-index-doc}$ décrite ci-dessus :

$$\begin{aligned}
\textit{pre-index-doc}\ (d_{33}) &= \textit{pre-index-dox}\ (d_{33}) \\
\textit{pre-index-doc}\ (d_{34}) &= \textit{pre-index-dox}\ (d_{34}) \\
\textit{pre-index-doc}\ (d_{35}) &= \textit{pre-index-dox}\ (d_{35})
\end{aligned}$$

$$\textit{pre-index-doc}\ (d_{32}) = \oplus_{pre-index-doc}(\{\textit{pre-index-doc}\ (d_{33}), \textit{pre-index-doc}\ (d_{34}), \textit{pre-index-doc}\ (d_{35})\})$$

$$\textit{pre-index-doc}\ (d_{31}) = \oplus_{pre-index-doc}(\{\textit{pre-index-doc}\ (d_{32})\})$$

Ainsi, à l'issue de l'étape locale - document, la table des index est remplie comme présenté à la table , où d_{3j} représente le doxel j du document 3. Tous les doxels du document 3 sont indexés et en fait, tous les documents sont indexés.

	d_{31}	d_{32}	d_{33}	d_{34}	d_{35}
moteur	2	2	0	1	1
recherche	2	2	0	1	1
Google	2	2	0	0	2
donne	1	1	0	0	1
nom	1	1	0	0	1
societe	1	1	0	0	1
information	1	1	0	0	1
Internet	1	1	0	0	1
utilise	1	1	0	0	1
monde	1	1	0	0	1

TAB. 4.2 – Table des index pour le document 3 à l'issue de l'étape locale - document.

4.1.3 Étape globale - corpus

L'étape globale vise à intégrer, pour obtenir un index des doxels suivant leur structure de composition, des éléments relatifs au corpus complet. Ces éléments peuvent être mis en parallèle avec le calcul d'*idf* dans le cas de documents atomiques. Pour avoir les informations d'*idf* finales, il est nécessaire d'avoir réalisé une étape d'indexation de toute la collection. Séparer cette étape permet d'avoir une plus grande liberté sur les opérations réalisées : on peut par exemple réaliser de simples calculs d'*idf* sur tous les doxels (indépendamment), ou bien des calculs d'*idf* par type de doxel par document [SB06]. De manière générale, nous définissons cette étape globale à l'aide de la fonction *pre-index-corpus* qui utilise les pré-index de documents pour définir l'index compositionnel des documents.

$\mathcal{L}_{index}$ est le langage d'indexation final. Le résultat intermédiaire utilisait $\mathcal{L}_{pre-index}$: en effet, si un terme de $\mathcal{L}_{pre-index}$ apparaît partout avec une valeur d'*idf* petite, alors ce terme pourrait être filtré hors du vocabulaire de $\mathcal{L}_{index}$, d'où l'intérêt d'avoir deux langages distincts. On peut avoir $\mathcal{L}_{pre-index} = \mathcal{L}_{index}$.

$$\textit{pre-index-corpus}\ :\ \mathcal{C}_{tot} \times 2^{\mathcal{C}_{tot} \times \mathcal{L}_{pre-index}} \rightarrow \mathcal{L}_{index}.$$

Il est à noter que le résultat de cette indexation opère sur tout élément de $\mathcal{C}_{tot}$ et pas seulement de $\mathcal{C}_{ind}$, car l'étape d'indexation non-compositionnelle peut utiliser ces index, comme présenté à la section 4.3, et l'ensemble des couples $(d, \textit{pre-index-corpus}(d))$ constitue l'ensemble INDEX-STRUCTUREL.

L'index est complètement défini par l'ensemble INDEX-STRUCTUREL $\subset \mathcal{C}_{tot} \times \mathcal{L}_{index}$:

$$\text{INDEX-STRUCTUREL} = \{(d, \textit{pre-index-corpus(d,pre-index-doc(d))}) | d \in \mathcal{C}_{tot}\}.$$

Par souci de simplification, on appellera $index(d)$ le résultat de la fonction *pre-index-corpus* pour un doxel d particulier.

En ce qui concerne notre exemple, pour le document 3, l'index du corpus complet, en utilisant un simple calcul d'$idf = log\frac{N}{df_i}$ sur les doxels pris indépendamment, est donné par la table 4.3.

	d_{31}	d_{32}	d_{33}	d_{34}	d_{35}
moteur	0.1538	0.1538	0	0.5	0.09
recherche	0.1538	0.1538	0	0.5	0.09
Google	0.1538	0.1538	0	0	0.18
donne	0.0769	0.0769	0	0	0.09
nom	0.0769	0.0769	0	0	0.09
societe	0.0769	0.0769	0	0	0.09
Internet	0.0769	0.0769	0	0	0.09
utilise	0.0769	0.0769	0	0	0.09
monde	0.0769	0.0769	0	0	0.09
norme	1	1	0	1	1

TAB. 4.3 – Table des index normalisée pour le document 3 à l'issue de l'étape locale - document.

4.2 Définition de l'environnement d'occurrence des doxels

Nous nous sommes intéressés jusqu'à présent à des relations de composition entre les doxels. Nous considérons intéressant de générer de nouvelles relations entre doxels qui favoriseront l'organisation des résultats. L'environnement d'occurrence des doxels peut être déterminé à partir des index des doxels, c'est-à-dire de leur contenu sémantique, mais aussi à partir de liens préexistants. D'une part, cet environnement est exploité dans la littérature sous différentes formes (voisinage, liens entrant et/ou sortant par exemple) et apporte une amélioration des résultats de recherche d'information. D'autre part, rappelons que notre objectif est de permettre à un utilisateur ayant un besoin d'information à satisfaire de naviguer efficacement dans l'ensemble des éléments résultats ; ainsi, l'environnement des doxels a pour objectif de favoriser cette navigation, d'où l'intérêt de les définir puis de les indexer.

Environnement complet

Chaque doxel d de $\mathcal{C}_{tot}$ est associé, via les relations de E_{rel}, à un ensemble de doxels (cf section 3.3 page 78). Considérons une relation rel de E_{rel}. L'environnement immédiat du doxel d suivant cette relation, c'est-à-dire son environnement d'occurrence, noté $env_{d,rel}$, est donné par : $env_{d,rel} = \{d' | (d, d') \in rel\}$.

L'environnement complet d'un doxel d est défini par $env_d = \bigcup_{rel \in E_{rel}} env_{d,rel}$.

Tous les environnements sont décrits dans ENVIRONNEMENT $= \cup_{d \in \mathcal{C}_{tot}} env_d >$.

4.2.1 Exemple de voisinage

Similairement à Savoy [Sav96] qui prend en compte les documents plus proches voisins au sens sémantique, nous pouvons par exemple choisir d'utiliser les doxels plus proches voisins afin de créer une relation $E_{NN} \subset \mathcal{C}_{tot} \times \mathcal{C}_{tot}$ qui reflète la notion de plus proches voisins.

La fonction $sim : \mathcal{L}_{index} \times \mathcal{L}_{index} \rightarrow [0, 1]$ définit la notion de similarité entre deux phrases du langage $\mathcal{L}_{index}$ qui correspondent à la représentation du contenu de deux doxels. Informellement, $(d_1, d_2) \in E_{NN}$ si la valeur de $sim(\text{index}(d_1), \text{index}(d_2))$ est supérieure à un certain seuil. Ces relations de plus proches voisins sont orientées.

L'ensemble $env_{d,NN}$ définit le voisinage d'un doxel d :

$$env_{d,NN} = \{d' | d' \in \mathcal{C}_{tot}, (d, d') \in E_{NN}\}.$$

On prend au maximum k voisins tant que la valeur de similarité est supérieure à un certain seuil, Th.

Ces relations de plus proches voisins sont un exemple de relations non-compositionnelles.

Considérons par exemple une relation de plus proche voisin définie par la mesure de similarité cosinus, avec un nombre de voisins $k = 3$ et un seuil $Th = 0.6$. L'environnement $env_{d,NN}$ pour la collection $\mathcal{C}^{exemple}$ est complètement défini par les éléments suivants qui le composent :

$env_{d_{11},NN} = \{d_{24}\}$,
$env_{d_{12},NN} = \{d_{24}\}$,
$env_{d_{13},NN} = \{d_{24}\}$,
$env_{d_{14},NN} = \{d_{51}\}$,
$env_{d_{15},NN} = \{d_{24}\}$,
$env_{d_{21},NN} = \{d_{31}, d_{32}, d_{34}\}$,
$env_{d_{24},NN} = \{d_{11}, d_{12}, d_{13}, d_{15}\}$,
$env_{d_{26},NN} = \{d_{31}, d_{32}, d_{34}\}$,
$env_{d_{27},NN} = \{d_{34}\}$,
$env_{d_{29},NN} = \{d_{34}\}$,
$env_{d_{31},NN} = \{d_{21}, d_{26}\}$,
$env_{d_{32},NN} = \{d_{21}, d_{26}\}$,
$env_{d_{34},NN} = \{d_{21}, d_{26}, d_{27}, d_{29}\}$,
$env_{d_{51},NN} = \{d_{14}\}$,

et $env_{d_{22},NN} = env_{d_{23},NN} = env_{d_{25},NN} = env_{d_{28},NN} = env_{d_{29},NN} = env_{d_{33},NN} = \emptyset$,
$env_{d_{35},NN} = env_{d_{41},NN} = env_{d_{42},NN} = env_{d_{52},NN} = env_{d_{53},NN} = \emptyset$,

4.2.2 Liens

Par la relation non-compositionnelle issue du traitement des balises de type *lien* et décrite en section 3.3, l'environnement $env_{d,lien}$ est généré et comporte les éléments suivants :

$env_{d_{12},lien} = \{d_{51}\}$,
$env_{d_{22},lien} = \{d_{11}\}$,
$env_{d_{26},lien} = \{d_{31}\}$,
$env_{d_{32},lien} = \{d_{22}\}$,

Ainsi l'environnement complet sur l'exemple est complètement décrit par

$$\text{ENVIRONNEMENT} = env_{d,NN} \cup env_{d,lien}.$$

4.3 Indexation non-compositionnelle

Nous avons jusqu'à présent pris en compte des relations de composition entre doxels et nous avons défini des environnements d'occurrence pour les doxels, via des relations non-compositionnelles. Nous décrivons ici une caractérisation de ces relations non-compositionnelles. L'idée que nous défendons est que les relations non-compositionnelles sont porteuses de sens et doivent être intégrées lors de la description des doxels pour réaliser des opérations de recherche d'information sur des documents structurés par la suite.

Une solution pour utiliser ces relations entre doxels pourrait être d'utiliser la même idée que celle sur les structures de documents, c'est-à-dire la propagation d'index en suivant les liens. Dans ce cas, de nombreux écueils se présentent : les relations entre doxels ne sont pas forcément hiérarchiques, ce qui peut provoquer des cycles suivant une (ou plusieurs) relations(s). De plus, la prise en compte de propagations d'index se révèle coûteuse et très difficile à contrôler, comme nous ne connaissons pas *a priori* l'ordre des cycles potentiels. C'est pourquoi nous proposons de tenir compte de ces relations non-compositionnelles en ne propageant pas des index de doxels, mais plutôt en calculant des "différentiels" d'index qui sont des qualifications des similarités/dissimilarités entre doxels. Représenter des différences n'a pas d'impact direct sur les index des doxels, ce qui permet d'éviter les propagations et leurs désagréments cités ci-dessus.

Nous avons créé des liens entre doxels et nous nous intéressons à caractériser la pertinence du choix d'un doxel lié à un doxel pertinent en réponse à une requête. L'intérêt du doxel lié pour la recherche d'information peut se mesurer selon deux dimensions : la première correspond à la notion de présence de l'information demandée dans l'information apportée par le doxel lié ; la seconde détermine si l'information contenue dans le doxel lié a une taille appropriée par rapport à la quantité et à la disposition des informations importantes qu'elle contient. Ces deux informations sont reprises dans les mesures de degré de pertinence et de couverture puis d'exhaustivité et de spécificité mises en place pour l'évaluation de la recherche d'information dans des documents XML.

Idée

Pour caractériser les relations entre doxels, nous proposons de définir des exhaustivité et spécificité relatives entre doxels. Si nous sommes en mesure de caractériser si un doxel est plus exhaustif ou plus spécifique pour une requête donnée qu'un autre doxel, cet élément devrait permettre de favoriser l'organisation des réponses à cette requête.

On définit R^T_{comp} la fermeture transitive de l'environnement d'un doxel via R_{comp} afin de créer l'ensemble env^T_d.

$$\text{Soient } d, d' \in \mathcal{C}_{tot},$$
$$(d, d'') \in R^T_{comp} \Leftrightarrow ((d, d'', n) \in R_{comp} \vee \exists d' \in \mathcal{C}_{tot}, (d, d') \in R^T_{comp} \wedge (d', d'', n') \in R_{comp}).$$

$$env^T_d = env_d \cup \{d'' | d'' \in env_{d'} \wedge (d, d') \in R^T_{comp}\}.$$

Pour chaque doxel d, nous définissons alors une fonction qui mesure un taux d'exhaustivité et de spécificité de d par rapport à son environnement env_d^T.

4.3.1 Exhaustivité et spécificité

Les définitions de l'exhaustivité et de la spécificité ont été proposées à INEX 2005 [PL04] et s'inspirent de [CMF96]. D'après l'état de l'art, elles sont davantage liées à une estimation qui caractérise le contenu d'un document pour une requête donnée :

- un document est exhaustif pour une requête s'il traite de tous les aspects de la requête,
- un document est très spécifique pour une requête s'il ne traite que d'éléments de la requête.

Ces deux considérations sur le contenu de documents ne sont pas totalement corrélées :

- un document peut être très exhaustif mais peu spécifique : dans ce cas le document parle de tous les aspects de la requête mais également de beaucoup d'autres aspects non relatifs à la requête ;
- un document peut être très spécifique et peu exhaustif : dans ce cas le document parle quasiment totalement de la requête, mais ne traite qu'une partie de la requête ;
- un document "idéal" pour une requête est donc très exhaustif et très spécifique.

Une difficulté liée à ces définitions est qu'elles ne peuvent être calculées que lors du traitement d'une requête, ce qui implique un traitement dynamique coûteux en temps. Nous souhaitons régler ce problème en introduisant un calcul à l'indexation, et éventuellement en introduisant des calculs statiques de prise en compte de requêtes utilisateurs pour effectuer une forme d'apprentissage des valeurs d'exhaustivité et de spécificité.

4.3.2 Exhaustivité et spécificité relatives

Nous définissons dans la suite des éléments qui permettent de proposer des notions d'"exhaustivité d'environnement" et de "spécificité d'environnement" pour un doxel suivant une relation de non composition *rel*, de ces éléments. Le succès de telles caractéristiques des doxels est lié à la qualité des valeurs d'exhaustivité et de spécificité définies. Par contre, si ces valeurs sont correctes, nous en attendons un atout sur la qualité de la structuration des résultats de requêtes lors de la phase d'interrogation.

Définition 5. *Soit une relation non-compositionnelle du doxel d_1 vers le doxel d_2 représentée à la figure 4.3 :*

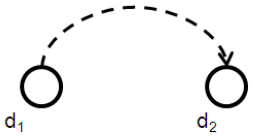

FIG. 4.3 – Deux doxels d_1 et d_2 liés.

- *L'exhaustivité relative de cette relation, notée $Exh(d_1, d_2) \in [0, 1]$, dénote la mesure avec laquelle d_2 traite de tous les sujets de d_1.*
- *La spécificité relative de cette relation, notée $Spe(d_1, d_2) \in [0, 1]$, dénote la mesure avec laquelle d_2 traite uniquement des sujets de d_1.*

Par exemple, si d_2 aborde tous les éléments de d_1, alors $Exh(d_1, d_2)$ devrait avoir une valeur proche de 1. Par exemple, si d_2 parle uniquement d'éléments de d_1, alors $Spe(d_1, d_2)$ devrait avoir une valeur proche de 1.

Cette définition ne tient *a priori* pas compte des requêtes utilisateurs. Alors, les valeurs pré-calculées pourraient ne pas avoir l'effet souhaité. En effet, si un utilisateur a un comportement qui vise à favoriser un type de doxel plutôt qu'un autre habituellement dans ses recherches, il serait préférable d'en tenir compte. C'est pourquoi nous utilisons les éléments relatifs à des requêtes. Nous pouvons par exemple exploiter les logs pour obtenir des requêtes types pour estimer les valeurs d'exhaustivité et de spécificité.

Si nous revenons aux doxels d_1 et d_2 vus en figure 4.3, et en utilisant leurs index respectifs, nous définissons les exhaustivité et spécificité relatives du doxel d_2 par rapport au doxel d_1 en nous basant sur deux éléments :

1. le contenu des doxels : dans ce cas on se base sur le contenu des doxels pour estimer une exhaustivité et une spécificité du doxel source de la relation par rapport au doxel cible de cette relation. Dans ce cas, nous obtenons une estimation *a priori* de l'exhaustivité des doxels reliés, appelée $Exh_{apriori}(d1, d2)$ et $Spe_{apriori}(d1, d2)$.
2. un ensemble $S_{Qpre} = \{q_{pre}\}$ de requêtes pré-établies. Ces requêtes doivent être suffisamment nombreuses pour concerner toutes les relations non-compositionnelles de la collection. Dans ce cas, pour les doxels reliés, nous estimons des valeurs d'exhaustivité $Exh_{sample\restriction q_{pre}}(d1, d2)$ et de spécificités $Spe_{sample\restriction q_{pre_i}}(d1, d2)$ restreintes aux requêtes prédéfinies q_{pre} de S_{Qpre}. Puis nous moyennons les valeurs d'exhaustivités et de spécificités entre ces doxels pour toutes les requêtes de S_{Qpre}, de manière à obtenir $Exh_{sample\_set\restriction S_{Qpre}}(d1, d2)$ et $Spe_{sample\_set\restriction S_{Qpre}}(d1, d2)$. Cette proposition est inspirée des travaux de Callan et Connell [CC01]. Dans ces travaux, les auteurs visent à définir, en se basant sur des requêtes pré-établies, une description de ressources ; une ressource étant un système de recherche d'information pour lequel nous n'avons pas de détail particulier. Après avoir décrit les ressources, un méta-moteur peut pondérer les résultats des ressources lors du traitement de requêtes. Dans notre cas, nous utilisons l'idée de ces travaux en :
   (a) Posant des requêtes $q_{pre} \in S_{Qpre}$ composées de termes du vocabulaire,
   (b) Calculant l'exhaustivité et la spécificité des doxels du corpus pour les requêtes $q_{pre} \in S_{Qpre}$.

Cette idée d'utilisation de "Query-Based sampling" permet de ne pas calculer toutes les requêtes possibles, ce qui serait trop coûteux voire impossible, mais de se limiter à certaines requêtes, qui peuvent être vues comme des contextes de recherche. La quantité de requêtes utilisées pour faire cet échantillonnage est un paramètre de notre approche.

Ensuite, pour un couple $(d1, d2)$, une combinaison des exhaustivités $Exh_{apriori}(d1, d2)$ et $Exh_{sample\_set\restriction S_{Qpre}}(d1, d2)$ est établie, de même pour les spécificités. Le résultat est donc la valeur d'exhaustivité globale $Exh_{globale}(d1, d2)$ et la valeur de spécificité globale $Spe_{globale}(d1, d2)$ où $d_2 \in env_{d_1}$. Avec $\lambda, \lambda' \in [0, 1]$:

$$Exh_{globale}(d1, d2) = \lambda * Exh_{apriori}(d_1, d_2) + (1 - \lambda) * Exh_{sample\_set\restriction S_{Qpre}}(d_1, d_2)$$

$$Spe_{globale}(d1, d2) = \lambda' * Spe_{apriori}(d_1, d_2) + (1 - \lambda') * Spe_{sample\_set\restriction S_{Qpre}}(d_1, d_2)$$

Les notions d'exhaustivité et de spécificité sont respectivement complètement définies par :

$$\text{EXH-SPE} =< Exh_{apriori}, Spe_{apriori}, Exh_{sample-set\restriction S_{Qpre}}, Spe_{sample-set\restriction S_{Qpre}}, S_{Qpre}, Exh_{globale}, Spe_{globale} >.$$

Considérons par exemple la relation "lien" : deux doxels sont liés entre eux par la relation "lien" si il existe un lien XML entre eux. Ces relations sont obtenues à partir du doxel père du doxel *<lien>* dans le code des documents. Cette relation a été présentée à la figure 3.3. Ainsi, les relations de E_{lien} sont les suivants : $(d_{12}, d_{51}), (d_{22}, d_{11}), (d_{26}, d_{31}), (d_{32}, d_{22}) \in E_{lien}$.

Nous supposons que les exhaustivités et spécificités relatives sont calculées pour toutes ces relations :

1. Pour les exhaustivités et spécificités *a priori*, les résultats sont synthétisés dans la table 4.6. Les calculs ont été effectués à partir de la formule qui est présentée en section 6.2.2.

$Doxelsource$	$Doxelcible$	$Exh_{apriori}$	$Spe_{apriori}$
d_{12}	d_{51}	0.7692	0.5192
d_{22}	d_{11}	0.4	0.2098
d_{26}	d_{31}	0.6429	0.6316
d_{32}	d_{22}	0.4737	0.4

TAB. 4.4 – Exhaustivité et spécificité *a priori* pour les liens de $\mathcal{C}^{exemple}$.

2. Pour notre exemple, nous supposons que l'ensemble de requêtes pré-établies contient une seule requête S_{Qpre} = *"moteur de recherche d'information"* ; ainsi, en posant $\lambda = \lambda' = 0.5$, les valeurs d'exhaustivité et de spécificité obtenues en tenant compte des requêtes de S_{Qpre} sont synthétisées dans la table 4.5.

Doxel source	*Doxel cible*	$Exh_{sample-set\restriction S_{Qpre}}$	$Spe_{sample-set\restriction S_{Qpre}}$
d_{12}	d_{51}	1	1
d_{22}	d_{11}	0.8	0.2809
d_{26}	d_{31}	1	0.8889
d_{32}	d_{22}	0.8889	1

TAB. 4.5 – Exhaustivité et spécificité sur $S_{Q_{pre}}$ pour les liens de $\mathcal{C}^{exemple}$.

termes de q_{pre}	d_{12}	d_{51}	d_{22}	d_{11}	d_{26}	d_{31}	d_{32}	d_{22}
$information$	2	1	0	8	0	1	1	0
$moteur$	0	0	1	0	3	2	2	1
$recherche$	2	1	2	5	3	2	2	2

TAB. 4.6 – Occurrence des termes de q_{pre} dans les doxels liés de $\mathcal{C}^{exemple}$.

Connaissant la table des index pour les doxels liés 4.5, la table 4.5 peut être interprétée comme suit :

- concernant le premier lien allant de d_{12} vers d_{51} : l'exhaustivité vaut 1 comme le doxel d_{51} parle de *moteur de recherche d'information* de la même façon que le doxel d_{12} (2 dimensions sur 3), et la spécificité vaut 1 également ce qui signifie que le doxel d_{51} parle de *moteur de recherche d'information* sur les mêmes dimensions exactement ou sur moins de dimensions que le doxel d_{12} (en l'occurrence 2 dimensions sur 2 présentes dans d_{12} apparaissent dans d_{51}) ;
- concernant le deuxième lien allant de d_{22} vers d_{11} : l'exhaustivité vaut 0.8 comme le doxel d_{11} parle moins de *moteur de recherche d'information* que le doxel d_{22} (1 seule dimension sur 2 apparaissant dans d_{22}, mais cette dimension est plus pondérée, d'où une valeur de 0.8 et pas de 0.5, comme on aurait pu s'y attendre en cas de pondération identique sur les 2 dimensions), et la spécificité vaut 0.2809 ce qui signifie que le doxel d_{11} ne parle pas de *moteur de recherche d'information* suivant les mêmes dimensions que le doxel d_{22}, d'autres dimensions apparaissent dans d_{11} et avec une pondération importante *a priori* vue la valeur affaiblie à 0.2809 ;
- concernant le troisième lien allant de d_{26} vers d_{31} : l'exhaustivité vaut 1 comme le doxel d_{31} parle de *moteur de recherche d'information* de la même façon que le doxel d_{26}, tandis que la spécificité vaut 0.8889 ce qui signifie que le doxel d_{31} ne parle pas de *moteur de recherche d'information* suivant les mêmes dimensions que le doxel d_{26}, d'autres dimensions apparaissent dans d_{31} et avec une pondération faible *a priori* vue la valeur relativement forte à 0.8889 ; ;
- concernant le quatrième lien allant de d_{32} vers d_{22} : l'exhaustivité vaut 0.8889 comme le doxel d_{22} parle moins de *moteur de recherche d'information* que le doxel d_{32}, tandis que la spécificité vaut 1 ce qui signifie que le doxel d_{22} parle de *moteur de recherche d'information* sur moins de dimensions que le doxel d_{32}.

Les valeurs d'exhaustivité et de spécificité globale sont synthétisées dans la table 4.7, pour $\lambda = \lambda' = 0.75$.

Doxel source	*Doxel cible*	$Exh_{globale}$	$Spe_{globale}$
d_{12}	d_{51}	0.8269	0.6394
d_{22}	d_{11}	0.5	0.2276
d_{26}	d_{31}	0.7322	0.6959
d_{32}	d_{22}	0.5775	0.55

TAB. 4.7 – Exhaustivité et spécificité globale pour les liens de $\mathcal{C}^{exemple}$.

4.4 Doxels de $\mathcal{C}_{ind}$

Tous les doxels de $\mathcal{C}_{tot}$ sont tous indexés à l'issue de l'étape d'indexation. Cependant, l'étude de l'état de l'art nous a par exemple montré que l'utilisation de doxels trop petits nuit à la pertinence des résultats [MSDWZ93]. À l'évidence, renvoyer le titre d'une partie de document en réponse à un utilisateur ne constitue pas nécessairement une réponse adaptée, même si le titre comporte tous les termes de la requête utilisateur, dans ce cas, il vaut probablement mieux renvoyer à l'utilisateur le titre ainsi que la sous-partie correspondant à ce titre.

Il semble alors intéressant de pouvoir limiter les doxels retrouvables par le système de recherche d'information à un sous-ensemble de $\mathcal{C}_{tot}$. Nous choisissons de nommer $\mathcal{C}_{ret}$ l'ensemble des doxels qui sont retrouvables *a priori*. L'ensemble $\mathcal{C}_{ret}$ est inclus dans l'ensemble $\mathcal{C}_{tot}$.

Dans tous les cas, il est possible que certains doxels soient déclarés (par une source d'information extérieure) comme *a priori* non-pertinents (en utilisant par exemple des critères de type ou de taille). Il s'agit donc d'une forme de filtrage *a priori*. Ainsi, nous définissons un ensemble de sources d'informations $S = \{s_i\}$, et pour chaque source d'information s_i nous définissons une fonction booléenne f_{s_i} telle que :

$$f_{s_i} : \mathcal{C}_{tot} \rightarrow \{vrai, faux\}$$
$$d \mapsto \begin{cases} vrai & \text{si } d \text{ est un doxel qui peut être retrouvé comme réponse} \\ & \text{d'après la source d'information } s_i \\ faux & \text{sinon} \end{cases}$$

On appelle $\mathcal{C}_{ret/S}$ l'ensemble de tous les doxels qui peuvent être retrouvés dans la collection, par rapport à un ensemble de sources d'informations S :

$$\mathcal{C}_{ret/S} = \{d \mid d \in \mathcal{C}_{tot},\ \wedge_{s \in S}\ f_s(d)\}$$

Dans le cas où il n'existe pas de source d'information externe, on pose qu'il existe une source existant *a priori*, appelée $s_{a-priori}$, telle que la fonction $f_{s_{a-priori}}$ est la fonction constante *vraie*. Il en résulte que S n'est jamais vide.

Pour faciliter la lecture, nous remplacerons la notation $\mathcal{C}_{ret/S}$ par $\mathcal{C}_{ret}$.

Sur la collection $\mathcal{C}^{exemple}$, nous posons que l'ensemble des éléments retrouvables est constitué des doxels de type *document* et p, les doxels de type *lien* ne peuvent pas être retrouvés. Ainsi, $d_{11}, d_{24}, d_{28}, d_{51} \notin \mathcal{C}_{ret}$.

Notre approche repose, comme pour tout modèle de recherche d'information, sur le fait que les doxels éléments de $\mathcal{C}_{ret}$ sont indexés et donc potentiellement retrouvés par le processus de recherche. Cependant, notre modèle intègre également le fait que tout doxel, en étant cible de relations non-compositionnelles, peut également, même s'il n'appartient pas à $\mathcal{C}_{ret}$, jouer sur la pertinence d'un doxel de $\mathcal{C}_{ret}$ comme faisant partie de l'environnement d'un tel doxel.

Cet état de fait nous conduit à définir que les doxels pour lesquels on utilise l'indexation sont les doxels de $\mathcal{C}_{ret}$, mais également les doxels cibles de relations non-compositionnelles ayant pour source un doxel de $\mathcal{C}_{ret}$.

L'ensemble des doxels cibles de relations non-compositionnelles ayant pour source un doxel de $\mathcal{C}_{ret}$, appelé $\mathcal{C}_{env}$, tel que :

$$\mathcal{C}_{env} = \bigcup_{d \in \mathcal{C}_{ret}} env_d.$$

Pour $\mathcal{C}^{exemple}$, $\mathcal{C}_{env} = \{d_{11}, d_{22}, d_{31}, d_{51}\}$.

On utilise dans la suite l'ensemble des doxels indexés et cibles $\mathcal{C}_{ind} = \mathcal{C}_{ret} \cup \mathcal{C}_{env}$; c'est pour ces doxels qu'on va utiliser l'index dans les étapes d'interrogation. On remarque qu'il n'est pas utile de stocker l'index des autres doxels.

4.5 Récapitulatif

Le modèle de document est donné par :

$$\text{MODELE-DOC} =< \text{CORPUS}, \text{INDEX-STRUCTUREL}, \text{VOISINAGE}, \text{EXH-SPE} >.$$

où :
- CORPUS est le corpus de documents structurés,
- INDEX-STRUCTUREL est l'index structurel des doxels du corpus,
- ENVIRONNEMENT est la description de l'environnement des doxels du corpus,
- EXH-SPE est la description des notions d'exhaustivité et de spécificité des doxels du corpus.

L'index d'un doxel est donc complètement défini par son contenu, son environnement ainsi que des valeurs d'exhaustivité et de spécificité globales avec les doxels de son environnement.

$$\text{INDEX} =< \text{INDEX-STRUCTUREL}, \text{ENVIRONNEMENT}, \text{EXH-SPE} >.$$

Suite à cette étape, tous les doxels de $\mathcal{C}_{tot}$ sont indexés.

Chapitre 5

Modèle d'interrogation

Sommaire

Nous considérons dans ce chapitre uniquement des requêtes sur le contenu seul de documents structurés (*Content Only* selon la terminologie Inex [INE05]). En particulier, nous n'intégrons pas ici de prise en compte des types de doxels cibles d'une requête. Nous considérons en effet que l'utilisation explicite de structure dans une requête n'est pas un usage courant pour rechercher des documents structurés dans le cas général, un utilisateur cherchant des documents sans connaître la structure des documents structurés.

Comme nous l'avons expliqué au chapitre 4, les doxels d'un corpus sont indexés selon des propagations de contenu basées sur la structure de composition, ainsi que sur des informations d'exhaustivité et de spécificité relatives des doxels voisins (selon des relations non-compositionnelles). Ces représentations permettent d'utiliser tout l'environnement d'un doxel pour représenter son contenu.

La modélisation de l'interrogation que nous proposons permet d'utiliser ces représentations, pour retrouver des doxels en se basant à la fois sur la composition de ces doxels et sur leurs relations non-compositionnelles. L'objectif est à la fois de retrouver des doxels pertinents, et de proposer une organisation des réponses qui permette simultanément une visualisation claire de l'espace résultat, et une navigation rapide dans cet espace.

L'interrogation sur le contenu de documents structurés que nous proposons se base sur les étapes suivantes :

1. tout d'abord une recherche basée sur le contenu des doxels, en utilisant uniquement de manière statique les aspects compositionnels du modèle de documents.

2. ensuite, une étape de recherche par l'environnement détermine la pertinence des doxels en prenant en compte les relations non compositionelles, en se basant sur les exhaustivité et spécificité relatives des doxels reliés.

3. enfin, une dernière étape, qualifiée d'étape d'organisation, propose une vue sur les relations qui sont utiles pour explorer l'espace résultat de la requête.

Nous passons en revue ces différents étapes dans les sections suivantes, dans l'ordre indiqué ci-dessus.

5.1 Représentation de la requête

Comme indiqué précédemment, nous traitons ici le cas d'une requête utilisateur sans explicitation d'éléments de structure cibles. Nous posons que la requête est modélisée selon le même langage d'indexation $\mathcal{L}_{index}$ que les documents. Cette hypothèse est celle qui est couramment faite en recherche d'information (dans le modèle vectoriel par exemple).

L'index d'une requête q est alors obtenu par par la fonction $index(q)$.

Pour l'exemple, posons la requête q = *"moteur de recherche"*. Pour cette requête, $index(q) =< (moteur, 1), (recherche, 1) >$, si nous considérons l'utilisation d'un anti-dictionnaire contenant "de", sans utiliser d'étape de lemmatisation des termes.

5.2 Recherche par le contenu

Dans le cadre de nos travaux, la recherche de doxels passe par l'obtention des doxels qui répondent le mieux à une requête utilisateur. Dans la liste obtenue, nous choisissons de garder des doxels possiblement non indépendants structurellement, car la liste obtenue peut contenir des doxels qui peuvent composer d'autres éléments de cette même liste. Cette étape est donc relativement simple dans la mesure où le contenu des doxels est connu car il a été défini lors de l'étape d'indexation : on rappelle que l'index d'un doxel élément de $\mathcal{L}_{index}$ est obtenu par la fonction $index(d)$.

De manière plus formelle, le résultat de l'étape de recherche par le contenu est une liste L_q^0 de couples $(d, RSV_{d,q}^0)$ avec d dans $\mathcal{C}_{ind} \cup \mathcal{C}_{env}$, triée par ordre de valeur de pertinence décroissante et $RSV_{d,q}^0 = matching(index(d), index(q))$ la valeur de pertinence du doxel d pour la requête q où :

$matching : \mathcal{L}_{index} \times \mathcal{L}_{index} \rightarrow \mathbb{R}$.

Dans le cas de l'utilisation du modèle vectoriel par exemple, la fonction de correspondance peut être le cosinus, couramment employé. Nous utilisons d'ailleurs le cosinus comme fonction de matching pour notre exemple.

On constate que cette étape de recherche par le contenu est similaire à ce qui existe pour la recherche de documents non-structurés en recherche d'information. Cependant, l'indexation de ces doxels a utilisé la structure compositionnelle des doxels, comme nous l'avons expliqué en partie 4.1, lors de l'indexation structurelle des doxels. Il en résulte que cette étape intègre la représentation de la structure des doxels d'un point de vue statique, c'est-à-dire sans dépendance par rapport à une requête.

Sur l'exemple, l'étape de recherche par le contenu produit les résultats présentés en colonnes 2 et 3 de la table 5.1, qui correspondent aux valeurs de pertinence $RSV_{d,q}^0$ et au rang déduit de ces valeurs. Par conséquent, la liste $L_{d,q}^0$ vaut :

$$\begin{aligned} L_q^0 \;=\; [\; & (d_{34}, 1), (d_{27}, 0.8165), (d_{21}, 0.6713), (d_{26}, 0.6547), (d_{31}, 0.6489), (d_{32}, 0.6489), \\ & (d_{29}, 0.6172), (d_{22}, 0.5477), (d_{41}, 0.4523), (d_{42}, 0.4523), (d_{23}, 0.4472), (d_{13}, 0.4264), \\ & (d_{12}, 0.3922), (d_{35}, 0.3922), (d_{11}, 0.2957), (d_{15}, 0.2261), (d_{52}, 0.1667), (d_{51}, 0.0981)]. \end{aligned}$$

5.3 Recherche par l'environnement

Suite à la recherche par le contenu, notre objectif est de définir une organisation de l'espace résultat qui facilite une présentation et une exploration de cet espace. Cette organisation doit servir de support à la présentation des doxels résultats ainsi que de relations entre ces doxels choisis comme importants pour la requête.

Dans cette étape, l'utilisation de l'environnement des doxels de L_q^0 est effectuée pour obtenir une valeur de pertinence tenant compte des relations non-compositionnelles.

Nous obtenons une liste L_q^1 qui est un ensemble de couples $(d, RSV_{d,q}^1)$ tels que $RSV_{d,q}^1$ est la valeur de pertinence d'un doxel d dans son environnement env_d^T pour chaque doxel de L_q^0.

La valeur de pertinence d'un doxel d dans son environnement est le résultat d'une fonction f_{env}^{RSV} qui utilise en entrée un n-uplet comprenant la valeur de pertinence du doxel pris seul, la liste

d	$RSV^0_{d,q}$	rang	$RSV^{1^-}_{d,q}$	rang	$RSV^1_{d,q}$	rang
d_{26}	0.6547	4	0.5322	1	0.5740	1
d_{32}	0.6489	6	0.4340	3	0.5161	2
d_{34}	1	1	0.5	2	0.5	3
d_{27}	0.8165	2	0.4082	4	0.4082	4
d_{21}	0.6713	3	0.3357	5	0.3357	5
d_{31}	0.6489	5	0.3244	6	0.3244	6
d_{22}	0.5477	8	0.3049	8	0.3101	7
d_{29}	0.6172	7	0.3086	7	0.3086	8
d_{12}	0.3922	13	0.2216	12	0.2334	9
d_{41}	0.4523	9	0.2261	9	0.2261	10
d_{42}	0.4523	10	0.2261	10	0.2261	11
d_{23}	0.4472	11	0.2236	11	0.2236	12
d_{13}	0.4264	12	0.2132	13	0.2132	13
d_{35}	0.3922	14	0.1961	14	0.1961	14
d_{11}	0.2957	15	0.1478	15	0.1478	15
d_{15}	0.2261	16	0.1131	16	0.1131	16
d_{52}	0.1667	17	0.0833	17	0.0833	17
d_{51}	0.0981	18	0.0490	18	0.0490	18
d_{25}	0	19	0	19	0	19
d_{53}	0	20	0	20	0	20

TAB. 5.1 – Valeurs de pertinence pour $q =$ *moteur de recherche*.

L^0_q, et l'ensemble de triplets contenant les doxels $d' \in env^T_d$ étant indexés et donc appartenant à $\mathcal{C}_{ind} \cup \mathcal{C}_{env}$, avec leurs valeurs de spécificité et d'exhaustivité relatives. Cette fonction renvoie un résultat de type réel, ce qui donne[23] :

$$f^{RSV}_{env} : \mathbb{R} \times 2^{\mathcal{C}_{ind} \cup \mathcal{C}_{env} \times \mathbb{R}}_{[\,]} \times 2^{\mathcal{C}_{ind} \cup \mathcal{C}_{env} \times \mathbb{R} \times \mathbb{R}} \rightarrow \mathbb{R}$$

Cette fonction est appelée avec les paramètres suivants pour un doxel d de $\mathcal{C}_{ind} \cup \mathcal{C}_{env}$:

$$RSV^1_{d,q} = f^{RSV}_{env}(RSV^0_{d,q}, L^0_q, \{(d', Exh_{globale}(d,d'), Spe_{globale}(d,d')) | d' \in env^T_d \cap (\mathcal{C}_{ind} \cup \mathcal{C}_{env})\}).$$

La fonction f^{RSV}_{env} a pour objectif de définir la valeur de pertinence globale d'un doxel en considérant non seulement sa pertinence par rapport à la requête, mais également en prenant en compte sa pertinence en temps que point de départ de navigation dans l'espace des résultats.

Un exemple de fonction, dans le cas de l'utilisation du modèle vectoriel de recherche d'information, peut s'inspirer des travaux sur la propagation lors du traitement de requête de Savoy [Sav96], en utilisant une combinaison linéaire de la valeur de pertinence du doxel et des valeurs de pertinence des doxels cibles combinés avec les valeurs d'exhaustivité et de spécificités établies durant l'étape d'indexation.

[23] $2^E_{[]}$ dénote l'ensemble de toutes les listes possibles composées d'éléments de E dans la signature de f^{RSV}_{env}

Ainsi, pour l'exemple, nous proposons d'utiliser la fonction suivante :

$$RSV^1(d,q) = \alpha * RSV^0_{d,q} + (1-\alpha) * RSV_{env}(d,q),$$

avec $\alpha \in [0,1]$

$$RSV_{env}(d,q) = \sum_{d' \in env^T_d} \frac{\beta * Exh_{globale}(d,d') + (1-\beta) * Spe_{globale}(d,d')}{|env^T_d|} RSV^0_{d',q}$$

avec $\beta \in [0,1]$ fixé de telle sorte que l'exhaustivité ou la spécificité est privilégiée.

Sur notre corpus $\mathcal{C}^{exemple}$, si nous fixons les paramètres aux valeurs $\alpha = 0.5$ ce qui signifie que nous accordons autant d'importance au contenu des doxels qu'à leur environnement, et si nous posons $\beta = 0$, ce qui signifie que nous privilégions les réponses les plus spécifiques, la liste $L^1_{d,q}$ obtenue est la suivante :

$$\begin{aligned} L^1_q = [\ & (d_{26}, 0.5740), (d_{32}, 0.5161), (d_{34}, 0.5), (d_{27}, 0.4082), (d_{21}, 0.3357), (d_{31}, 0.3244), \\ & (d_{22}, 0.3101), (d_{29}, 0.3086), (d_{12}, 0.2334), (d_{41}, 0.2261), (d_{42}, 0.2261), (d_{23}, 0.2236), \\ & (d_{13}, 0.2132), (d_{35}, 0.1961), (d_{11}, 0.1478), (d_{15}, 0.1131), (d_{52}, 0.0833), (d_{51}, 0.0490)\ ] \end{aligned}$$

Les résultats sont détaillés dans la table 5.1, en colonnes 6 et 7, correspondant à la valeur $RSV^1_{d,q}$ et au rang qui lui est associé. Les colonnes 4 et 5 de ce même tableau produisent des résultats intermédiaires, $RSV^{1-}_{d,q}$ et le rang correspondant, qui seraient obtenus en n'utilisant que les valeurs d'exhaustivité et de spécificité évaluées *a priori*, en ne considérant pas l'ensemble de requêtes $S_{Q_{pre}}$. Nous remarquons que par exemple, le doxel d_{32} est mis en avant en comparaison avec le doxel d_{34} dans le classement final, alors que c'était l'inverse en ne considérant que les valeurs *a priori*. Le fait d'avoir interrogé la collection avec $q_{pre} =$ *moteur de recherche d'information* a permis de réajuster les valeurs d'exhaustivité et de spécificité dans ce contexte et de permettre un réponse plus appropriée.

La comparaison des classements entre les colonnes 3 et 7 est beaucoup plus intéressante : alors que le doxel d_{26} était classé $4^{\text{ième}}$ après l'interrogation sur le contenu, il est le meilleur doxel pour répondre à la requête en le considérant pris dans son contexte. En effet, il comporte un lien qui lui apporte de l'information pertinente pour la requête considérée et qui le fait remonter dans le classement. Il en va de même pour le doxel d_{32} qui passe de la $6^{\text{ième}}$ position à la $2^{\text{ième}}$.

5.4 Organisation des résultats

Une fois la liste L^1_q obtenue, il devient possible de définir un schéma permettant d'organiser les réponses. Nous choisissons d'organiser les réponses en une liste L^2_q. Cette étape ne calcule pas à proprement parler de nouveaux résultats ou de nouvelles valeurs, mais structure les résultats de manière à permettre la présentation de documents virtuels qui prennent en compte la pertinence des résultats à une requête.

Cette liste à deux objectifs,

- organiser les doxels résultats en utilisant le fait qu'ils composent des documents ayant une structure compositionnelle,
- organiser les doxels résultats en utilisant leur environnement non structurel.

Nous proposons, de nouveau, de considérer que la structure compositionnelle des documents structurés est la plus importante, et de considérer la structure non-compositionnelle comme étant le second élément à considérer dans l'organisation. Il en résulte que la liste L_q^2 est composée elle-même de listes, notées $L_{q,i}^2$.

Une liste $L_{q,i}^2$ regroupe les doxels de la liste L_q^1 (éléments de $\mathcal{C}_{ind} \cup \mathcal{C}_{env}$) appartenant à un même document structuré. Un élément d'une liste $L_{q,i}^2$ est un triplet $(d, rsv, \{(d', rsv')\})$ de $\mathcal{C}_{ind} \times \mathbb{R} \times 2^{\mathcal{C}_{ind} \cup \mathcal{C}_{env} \times \mathbb{R}}$ qui dénote pour un doxel d de $\mathcal{C}_{ind}$ provenant de L_q^1, avec une valeur de pertinence rsv (en fait la valeur $RSV_{d,q}^1$), l'ensemble des doxels d' de $\mathcal{C}_{ind} \cup \mathcal{C}_{env}$ qui sont dans l'environnement de d avec leur valeur de pertinence rsv' (en fait la valeur $RSV_{d',q}^1$). Une liste $L_{q,i}^2$ est triée par ordre de pertinence rsv décroissante. Tous les doxels en première position d'un triplet, pour une même liste $L_{q,i}^2$, appartiennent au même document structuré. Le fait de réaliser une telle structure permet de disposer de manière simple de toutes les valeurs de pertinences des doxels d'un document qui répondent à la requête, ainsi que des doxels liés par des relations non-compositionnelles à ces doxels de documents.

La liste L_q^2 est elle-même triée sur la première valeur de pertinence rsv de ces sous-listes $L_{q,i}^2$. Il en résulte que cette liste possède en tête la liste correspondant au document ayant un doxel avec une valeur de correspondance maximale pour la requête. Nous nous retrouvons dans le cas habituel de la liste de résultats pour un modèle de recherche d'information.

La liste L_q^2 obtenue pour notre exemple est la suivante :

$$
\begin{aligned}
L_q^2 \;=\; [\; & [(d_{26}, 0.5740, \{(d_{31}, 0.3244)\}), (d_{27}, 0.4082), (d_{21}, 0.3357), (d_{22}, 0.3101, \{(d_{11}, 0.1478)\}),\\
& (d_{29}, 0.3086), \{(d_{23}, 0.2236)\}],\\
& [(d_{32}, 0.5161, (d_{22}, 0.3101)), (d_{34}, 0.5), (d_{31}, 0.3244), (d_{35}, 0.1961)],\\
& [(d_{12}, 0.2334, (d_{51}, 0.0490)), (d_{13}, 0.2132), (d_{11}, 0.1478), (d_{15}, 0.1131)],\\
& [(d_{41}, 0.2261), (d_{42}, 0.2261)],\\
& [(d_{52}, 0.0833), (d_{51}, 0.0490)]\ ]
\end{aligned}
$$

5.5 Récapitulatif

Ce chapitre décrit le processus d'interrogation mis en œuvre dans notre modèle. Il s'effectue en plusieurs étapes : nous tenons compte dans un premier temps à la fois du contenu des documents et de leur structure (par transitivité vu le processus d'indexation développé) ; puis nous considérons l'environnement des doxels en utilisant des mesures d'exhaustivité et de relativité que nous avons définies. Intégrer l'environnement des doxels au moment de la recherche remet en cause le classement des doxels répondant le mieux à une requête.

Considérations sur une mise en oeuvre réaliste Les propositions ci-dessus nécessitent la mise en place de processus de calculs potentiellement coûteux, cela nuisant à l'utilisation d'un tel modèle, tel quel, dans le cas de grandes bases de documents structurés. Pour éviter de trop grands temps de calculs lors du traitement de requêtes, il peut s'avérer nécessaire de limiter la taille des listes intermédiaires, L_q^1 ou L_q^2, de manière à ne pas générer de trop nombreux calculs lors de la prise en compte des relations non-compositionnelles. Ces limitations peuvent porter sur le nombre d'éléments dans les listes, ou bien plus finement sur des seuils liés aux valeurs de pertinences obtenues pour les doxels.

Dans tous les cas, des expérimentations sont nécessaires pour définir quelle solution est la plus adaptée, en terme de qualité des réponses et de qualité d'exploration de l'espace résultat.

Présentation des résultats Une fois une organisation des réponses définie, par l'intermédiaire de la liste L_q^2, nous définissons ici une ébauche de présentation des résultats, apte à refléter l'espace résultat obtenu.

Un principe simple dans ce cas est de permettre d'accéder directement aux premiers documents structurés de la liste L_q^2 (car ils sont triés par ordre de doxel le plus pertinent), L_q en présentant la liste à l'écran, puis lors de la présentation d'un document de la liste L_q on présente le doxel de ce document qui est le plus pertinent, en indiquant visuellement d'une manière ou d'une autre les doxels de env_d à la fois pertinents pour la requête et liés fortement à d.

Il résulte d'un tel affichage que l'organisation obtenue au travers de la liste L_q^2 est bien reflétée à l'écran. Un aperçu de l'interface fournie en prototype est proposée à la figure 5.1. Dans cette figure, les documents pertinents pour la requête "vauban fortifications" sont listés à gauche ; quand un document est sélectionné, sa structure est développée sous forme d'arbre dans la colonne centrale, et les éléments de l'arbre précédés d'un petit carré correspondent aux doxels pertinents. Dès qu'un doxel est sélectionné dans cet arbre, son contenu est affiché au format HTML dans la fenêtre supérieure droite et son environnement apparaît simultanément sous forme de liste en bas à droite de l'arbre. Le contenu des doxels de l'environnement est affiché dans la fenêtre en bas à droite de l'interface. L'utilisateur peut naviguer dans la structure des documents pertinents pour atteindre rapidement les doxels pertinents, il peut aussi naviguer aisément dans l'environnement des doxels pertinents.

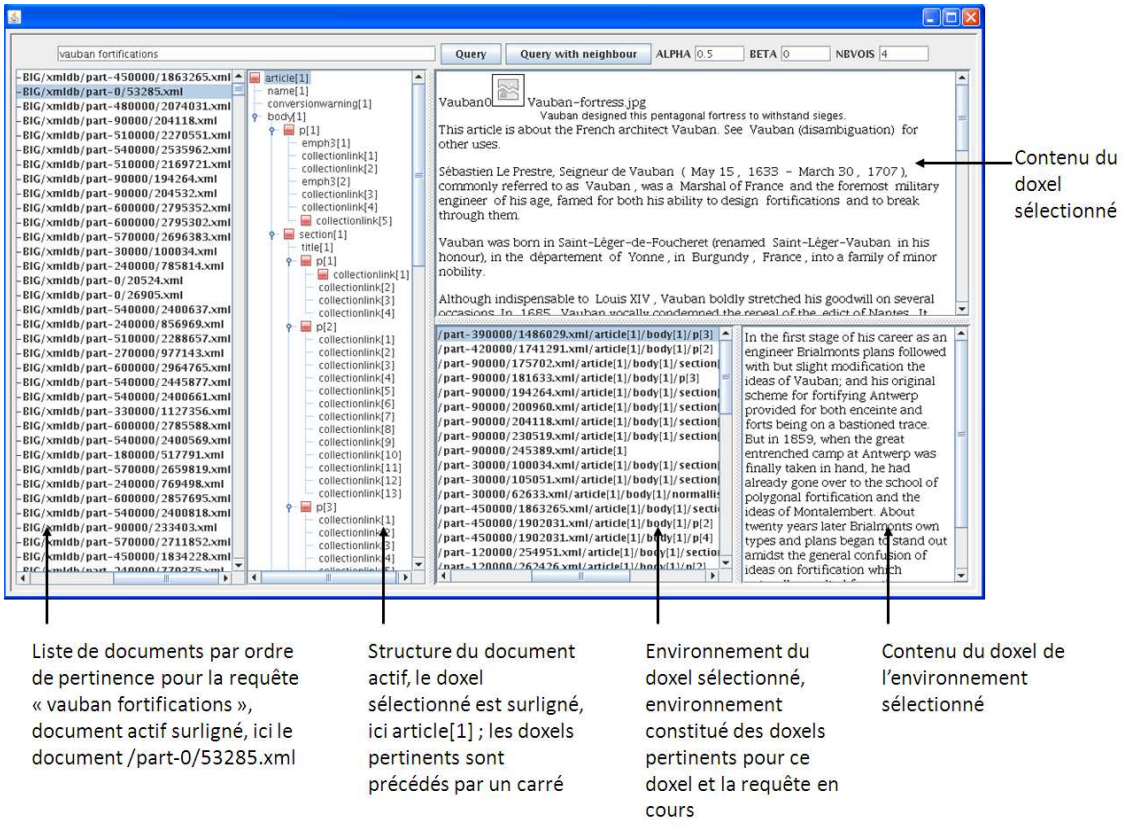

FIG. 5.1 – Exemple de présentation à l'écran.

Chapitre 6

Instanciation du modèle sur des documents XML

Sommaire

Dans les chapitres précédents, nous avons proposé un modèle abstrait de recherche d'information qui se base sur des documents structurés. La définition de ce modèle reste générale si bien que le modèle peut s'appliquer à différents formats de documents, dans différents domaines et à différents médias à condition que ces derniers permettent l'utilisation de structure et de relations entre éléments de structure.

XML (*eXtensible Markup Language*) définit les moyens de représenter sous forme textuelle des données hiérarchiques. Parce qu'il est à la fois très simple et expressif, XML est devenu le format le plus populaire pour la représentation de l'information et l'échange de données sur le Web. Le modèle de données XML est caractérisé par sa structure en forme d'arbre, avec la possibilité de décrire *a priori* la structure et les composants d'un ensemble de documents en utilisant des DTD (*Document Type Definition*) ou bien des schémas XML. Il en résulte que XML est un moyen simple de représenter des documents structurés.

Nous choisissons pour ces raisons d'appliquer notre modèle au média textuel et en particulier à des documents structurés en XML. Ce chapitre décrit rapidement une instanciation possible de notre modèle pour des documents XML. Dans une première section, nous détaillons les caractéristiques d'une collection de documents XML, considérée comme le corpus de documents structurés. Puis nous développons les processus d'indexation et d'interrogation dans ce contexte. Le modèle peut alors s'utiliser sur n'importe quelle collection de documents XML à condition que la DTD soit fournie.

6.1 Corpus de documents XML

Nous avons choisi d'appliquer notre modèle pour des documents virtuels à des documents XML. La force de XML réside dans sa capacité à pouvoir décrire n'importe quel domaine de données grâce à son extensibilité. Nous restons donc très généraux.

Considérons un document XML quelconque tel que présenté à la figure 6.1. Ce document comporte l'entête type d'un document XML. Et le corps du document est ensuite constitué de parties de documents identifiées par les balises `article`, `body`, `p`, `author`, `book` et `year`.

```
<?xml version="1.0" encoding="UTF-8"?>
<article>
  <body>
      <p> Il y a des circonstances de la vie où l'homme ressemble
        effectivement à un ordinateur :
        <it>tout lisse à l'extérieur, mais clignotant des neurones
          avec frénésie.</it>
      </p>
      <author> Daniel Pennac </author>
      <book> La fée carabine </book>
      <year> 1987 </year>
  </body>
</article>
```

Fig. 6.1 – Un document XML.

Les doxels du corpus $\mathcal{C}_{tot}$ correspondent à tous les éléments identifiés par une balise dans les documents XML.

Les éléments sont identifiés par leur chemin dans le document donné dans la syntaxe XPath. Pour des raisons d'unicité de la description, les chemins sont complètement définis par la grammaire suivante :

Path ::= '/'*ElementNode Path* | '/'*ElementNode* | '/'*AttributeNode*
ElementNode ::= ElementName *Index*
AttributeNode ::= @AttributeName
Index ::= '['integer']'

Ces éléments sont clairement identifiés par les chemins XPath[24] qui leur correspondent et ne se recouvrent pas, par construction. Par exemple, le doxel correspondant à l'auteur dans le document présenté en exemple dans la figure 6.1 est identifié par /article[1]/body[1]/author[1].

Les types des éléments sont donnés par les noms des balises des documents de la base.

Finalement, un doxel est caractérisé par son type et son contenu ; le contenu d'un doxel est obtenu comme la concaténation des contenus de tous ses doxels, sans pondération particulière. Notons que le texte *Il y a des circonstances de la vie où l'homme ressemble effectivement à un ordinateur :* n'est pas directement encadré par une balise mais est une sous-partie de la balise `p` et il devra être identifié comme tel par le modèle au moment de l'indexation, pour répondre correctement à la fonction qui permet d'obtenir le contenu d'un élément, et ainsi conserver l'ordre des éléments.

6.2 Modèle d'indexation de documents structurés XML

6.2.1 Indexation structurelle

Afin d'indexer des documents structurés XML, nous utilisons un analyseur syntaxique : les données encapsulées dans chaque document sont ainsi récupérées en parcourant les documents pour extraire les informations qu'il contient.

Nous avons choisi une API utilisant une approche hiérarchique : en effet, les analyseurs utilisant cette technique construisent une structure hiérarchique contenant des objets représentant les éléments du document, et dont les méthodes permettent d'accèder aux propriétés. La principale API utilisant cette approche est DOM[25].

Plus exactement DOM est un langage normalisé d'interface (API, Application Programming Interface), indépendant de toute plateforme et de tout langage, permettant à une application de parcourir la structure du document et d'agir dynamiquement sur celui-ci.

À l'aide de DOM, pour chaque élément rencontré (balise, commentaire, texte), nous recueillons les données qui nous intéressent pour l'indexation.

L'indexation structurelle mise en place dans cette instanciation correspond à ce qui est décrit en section 4.1.

Nous choisissons de ne considérer aucune propagation le long de la structure : nous indexons un élément de document simplement par son contenu, en utilisant un modèle vectoriel. Ainsi les fonctions $pre - index - dox$ et $pre - index - doc$ sont identiques et renvoient simplement

[24] XML Path Language : http ://www.w3.org/TR/xpath, consulté le 16 mai 2008

[25] DOM (*Document Object Model*) est une spécification du W3C (World Wide Web Consortium) : http ://www.w3schools.com/dom/default.asp

le contenu des doxels sans tenir compte de la position des composants structurels de ce dernier. L'indexation finale est obtenue en appliquant un simple calcul d'*idf*.

En résumé, la représentation du contenu d'un doxel d_i est donnée par un vecteur généré à partir du modèle vectoriel usuel : $d_i = \{w_{i,1}, \cdots, w_{i,k}\}$ où les poids sont donnés par un simple $tf.idf$:

$w_{i,j} = tf_{i,j} \times log\frac{N}{df_j}$

où $tf_{i,j}$ représente la fréquence du terme j dans le doxel i, N est le nombre de documents dans le corpus et df_j le nombre de documents du corpus comportant le terme j. Cette pondération serait notée `tfc` dans l'estimation de SMART [SB88].

6.2.2 Indexation non-compositionnelle

L'environnement d'occurrence des doxels est donné par l'attribut XLink[26] qui peut être utilisé avec n'importe quelle balise. L'attribut XLink permet d'associer à un doxel un document.

Les liens ainsi indexés sont caractérisés en utilisant des fonctions de calcul d'exhaustivité et de spécificité.

Une liste non-exhaustive des fonctions permettant de calculer la similarité entre deux vecteurs est proposée par Salton [SM86] : le coefficient de Dice, le coefficient de Jaccard, le cosinus, le coefficient de recouvrement(*overlap*). Ces opérateurs peuvent être définis pour des vecteurs binaires (similaires à des ensembles) ou réels. Nous décrivons nos choix sur des ensembles pour plus de simplicité, mais l'objectif est de proposer des fonctions prenant en compte des vecteurs dans l'espace des réels.

Cas ensembliste Considérons deux ensembles E_1 et E_2, visualisés sur la figure 6.2.

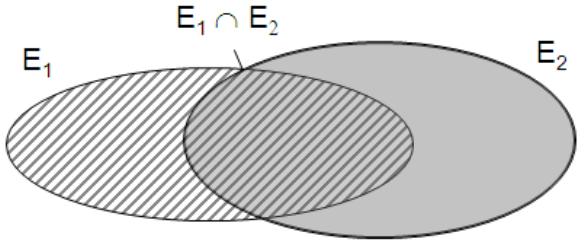

FIG. 6.2 – Deux ensembles E_1 et E_2 et leur intersection.

Le coefficient de recouvrement de deux ensembles, $F^b_{overlap}$ (pour fonction binaire de recouvrement) mesure le ratio entre la norme de l'intersection de deux ensembles divisés et le minimum des normes des deux ensembles comparés [SM86] :

$$F^b_{overlap}(E_1, E_2) = \frac{|E_1 \cap E_2|}{\min(|E_1|, |E_2|)}$$

Cette fonction semble intéressante si l'on a une vision ensembliste des notions d'exhaustivité et de spécificité relative, mais elle a le principal défaut (pour nos travaux) d'être symétrique alors que les exhaustivité et spécificité ne le sont pas d'après nos hypothèses.

[26] XLink : http ://www.w3.org/TR/xlink/

Nous proposons de modifier cette valeur de recouvrement pour calculer les exhaustivité et spécificité relatives, en considérant toujours E_1 comme la source et E_2 comme la cible d'un lien. Nos propositions sont :

- Pour l'exhaustivité : $F^b_{exh}(E_1, E_2) = \frac{|E_1 \cap E_2|}{|E_1|}$. Dans ce cas, l'exhaustivité de l'ensemble E2 par rapport à l'ensemble E1 est mesurée par la norme de l'intersection des deux ensembles divisée par la norme de E_1 : en nous plaçant dans une vision de recherche d'information, si E_2 parle de tous les sujets de E_1, alors le numérateur sera égal à $|E_1|$, et donc la valeur d'exhaustivité relative de E_2 par rapport à E_1 sera de 1 ; dans le cas dual où le document E_2 ne traite d'aucun des sujets de E_1, alors le numérateur est nul et donc la valeur d'exhaustivité relative également. Cette fonction a donc bien ses valeurs dans l'intervalle $[0, 1]$.
- Pour la spécificité : $F^b_{spe}(E_1, E_2) = \frac{|E_1 \cap E_2|}{|E_2|}$. La spécificité relative de l'ensemble E_2 par rapport à l'ensemble E_1 est mesurée par la norme de l'intersection des deux ensembles divisée par la norme de E_2, c'est-à-dire la proportion de E_2 qui est dans E_1 : si E_2 ne parle que des sujets de E_1, alors le numérateur sera égal à $|E_2|$, et donc la valeur de spécificité de E_2 par rapport à E_1 sera de 1 ; dans le cas dual où le document E_2 ne traite d'aucun sujet de E_1, alors le numérateur est nul et donc la valeur d'exhaustivité également. Cette fonction a donc bien ses valeurs dans l'intervalle $[0, 1]$.

Cas de vecteurs réels Dans la réalité, nous nous situons dans un cas où nous manipulons des vecteurs de réels et non binaires. Nous étendons donc les propositions ci-dessus à des vecteurs réels. D'après Salton [SM86], le calcul de recouvrement $F^r_{overlap}$ pour deux vecteurs de réels V_1 et V_2 est :

$$F^r_{overlap}(V_1, V_2) = \frac{\sum_i (V_1)_i \cdot (V_2)_i}{\min(\sum_i (V_1)_i^2, \sum_i (V_2)_i^2)}$$

L'utilisation de multiplication des coefficients sur les mêmes dimensions fait sens, dans la mesure où si les deux valeurs sont grandes (termes importants pour les deux documents) alors le produit l'est aussi, et si l'une des deux valeurs est petite alors le produit aussi. On pourrait aussi penser à des *min* par exemple, en se rapprochant de calculs d'intersection d'ensembles flous, mais le produit donne une interprétation correcte des choses.

Pour les calculs d'exhaustivité et de spécificité, la transposition est moins directe, étant donné le sens donné aux deux mesures. Considérons le cas courant où des termes sont communs aux deux vecteurs V_1 et V_2 ; en général, leurs poids sont différents dans chacun des vecteurs. Les mesures d'exhaustivité et de spécificité permettent de manière duale, d'évaluer d'abord la présence ou l'absence des termes dans l'intersection, puis de calculer alors proportionnellement leur importance relativement à V_1 et à V_2 respectivement.

La transposition des fonctions de calcul d'exhaustivité et de spécificité relatives sur des ensembles à des fonctions sur des vecteurs de réels est alors décrite comme suit :

$$F^r_{exh}(V_1, V_2) = \frac{\sum_{i|(V_2)_i \neq 0} (V_1)_i^2}{\sum_i (V_1)_i^2}$$

$$F^r_{spe}(V_1, V_2) = \frac{\sum_{i|(V_1)_i \neq 0} (V_2)_i^2}{\sum_i (V_2)_i^2}$$

Si nous revenons aux doxels d_1 et d_2 vus en figure 4.3, et en utilisant leurs index respectifs, nous définissons les exhaustivité et spécificité relatives du doxel d_2 par rapport au doxel d_1 en nous basant sur deux éléments :

1. le contenu des doxels et leur type : dans ce cas on se base sur le contenu des doxels pour estimer une exhaustivité et une spécificité du doxel source de la relation par rapport au doxel cible de cette relation. Dans ce cas, nous obtenons une estimation *a priori* de l'exhaustivité des doxels reliés, appelée $Exh_{apriori}(d1, d2)$ et $Spe_{apriori}(d1, d2)$. Ces valeurs sont définies comme des extensions de la fonction de recouvrement (*overlap*) [SM86] des index de d_1 et d_2 : elles reflètent la proportion de recouvrement entre la source et la cible de la relation. Nous définissons l'exhaustivité relative et la spécificité relative sur la base de vecteurs de doxels non normalisés $w_{1,i}$ et $w_{2,i}$ (respectivement pour d_1 et d_2) comme suit :

$$Exh_{apriori}(d_1, d_2) = \frac{\sum_{i|w_{2,i}\neq 0} w_{1,i}^2}{\sum_i w_{1,i}^2}$$

$$Spe_{apriori}(d_1, d_2) = \frac{\sum_{i|w_{1,i}\neq 0} w_{2,i}^2}{\sum_i w_{2,i}^2}$$

   En effet, le fait de normaliser ne refléterait pas la taille des doxels considérés et nous souhaitons garder ces informations.

2. un ensemble $S_{Qpre} = \{q_{pre}\}$ de requêtes pré-établies. Ces requêtes doivent être suffisamment nombreuses pour concerner toutes les relations non-compositionnelles de la collection. Dans ce cas, pour les doxels reliés, nous estimons des valeurs d'exhaustivité $Exh_{sample\restriction q_{pre}}(d1, d2)$ et de spécificités $Spe_{sample\restriction q_{pre_i}}(d1, d2)$ restreintes aux requêtes prédéfinies q_{pre} de S_{Qpre}. Ces valeurs sont définies comme suit, et uniquement si $w_{1,i} \cdot w_{2,i} \cdot q_{pre_i} \neq 0$.

$$Exh_{sample\restriction q_{pre}}(d1, d2) = \frac{\sum_{i|w_{2,i}\cdot q_{pre_i}\neq 0} w_{1,i}^2}{\sum_i w_{1,i}^2}$$

$$Spe_{sample\restriction q_{pre}}(d1, d2) = \frac{\sum_{i|w_{1,i}\cdot q_{pre_i}\neq 0} w_{2,i}^2}{\sum_i w_{2,i}^2}$$

6.2.3 Doxels de $\mathcal{C}_{ind}$

Les éléments retrouvables peuvent être facilement limités à un certain nombre de balises, d'autant plus que la DTD des documents XML est supposée connue, ainsi qu'aux éléments cibles de liens.

6.3 Modèle d'interrogation

La fonction de correspondance que nous choisissons est le cosinus couramment employé dans la littérature. Nous posons donc $RSV^0_{d,q} = cos(d, q)$, ce qui permet de calculer $RSV_{env}(d, q)$ en tenant compte à la fois du contenu des doxels et du contenu des doxels de leur environnement :

$$RSV^1(d, q) = \alpha * cos(d, q) + (1-\alpha) * \sum_{d' \in env_d^T} \frac{\beta * Exh_{globale}(d, d') + (1-\beta) * Spe_{globale}(d, d')}{|env_d^T|} cos(d', q)$$

6.4 Récapitulatif

Le modèle de recherche d'information pour des documents virtuels que nous avons proposé a été instancié à des documents structurés. Nous avons en particulier défini les éléments d'indexation structurelle et non-compositionnelles ainsi que des fonctions qui permettent de prendre en compte les différents éléments. Nous avons également défini une fonction de correspondance pour l'interrogation.

Troisième partie

Validation du modèle

Préambule

Nous avons présenté en partie II un modèle abstrait et générique pour l'indexation et la recherche de documents virtuels à partir de documents structurés. Ce chapitre est préalablement motivé par une étude de l'état de l'art sur les documents structurés et les documents virtuels. Nous proposons un modèle basé sur des documents structurés et qui renvoient des éléments de documents organisés en exploitant l'environnement de ces éléments, pour produire des documents virtuels résultats.

Au chapitre 6, nous avons proposé une instance de ce modèle, où nous limitons les documents structurés à des documents XML. En particulier, nous avons décrit les phases d'indexation et d'interrogation dans ce contexte.

Cette partie est consacrée à l'évaluation et à la validation de ce modèle de recherche d'information sur des documents structurés pour des documents virtuels. Nous choisissons d'expérimenter notre modèle sur une collection de documents XML.
Tout d'abord, nous présentons au chapitre 7 les campagnes d'évaluation INEX qui constituent le cadre général des expérimentations conduites lors de cette thèse. Nous décrivons le corpus de documents ainsi que les requêtes considérés pour la campagne en 2007 ; nous expliquons en quoi consistent les jugements de pertinence avant de détailler spécifiquement la tâche Ad Hoc qui nous intéresse. Finalement, nous explicitons certaines métriques d'évaluation de cette campagne.

Dans un second temps, au chapitre 8, nous décrivons les expérimentations visant à valider nos hypothèses concernant l'apport de l'environnement des éléments pour la recherche d'information sur des documents structurés. Nous montrons dans un premier temps que notre modèle est valide et peut servir de référence, puis nous appliquons diverses configurations pour déterminer les meilleures valeurs de paramètres pour α, β et le nombre de voisins et montrer l'intérêt de la prise en compte des voisins. Ensuite nous améliorons nos résultats en les fusionnant avec des résultats sur des documents complets. Finalement, nous évaluons l'intérêt de notre modèle pour la navigation de l'utilisateur dans l'espace résultat.

Chapitre 7

Les campagnes d'évaluation INEX

Sommaire

INEX (INitiative for the Evaluation of XML retrieval[27]) est une campagne d'évaluation visant, à la manière de TREC (Text REtrieval Conference[28]) ou CLEF (Cross Language Evaluation Forum[29]) pour les documents atomiques, à permettre une évaluation et une comparaison des systèmes de recherche d'information qui considèrent des collections de documents structurés XML.

Ainsi, INEX fournit un ensemble de documents, un ensemble de requêtes et des jugements de pertinence, c'est-à-dire les estimations humaines des éléments pertinents pour chaque requête. Cette section décrit le fonctionnement global d'INEX, la campagne d'évaluation internationale dédiée à promouvoir le progrès en recherche d'information XML. Nous nous limitons à la tâche Ad Hoc, bien qu'INEX se soit développé depuis quelques années vers des tâches annexes (*Book*, *Efficiency*, *Entity Ranking*, *Interactive*, *Question-Answering*, *Link-the-Wiki*, *XML-Mining* pour 2008 par exemple).

7.1 Documents utilisés par INEX en 2007

La campagne INEX depuis 2002 et les corpus pour supporter la recherche de documents structurés ont évolué. Alors que les premières campagnes étaient basées sur un corpus de 16819 documents de la collections IEEE Computer Society pour 764 Mo, le corpus lors de la campagne de 2007 regroupait des articles généraux issus de l'encyclopédie Wikipedia [DG06]. Le corpus est constitué de 8 collections correspondant à 8 langues différentes. Nous nous intéressons spécialement au corpus en anglais, qui comporte 659 388 documents, soit une collection de 4.6 Go. Un article contient en moyenne 161 nœuds XML, et la profondeur d'un nœud dans la représentation arborescente d'un document vaut 6.72.

La figure 7.1 présente un extrait de document de la collection INEX2007. Les documents de la collection INEX sont considérés comme des *articles* et identifiés de manière unique par l'attribut *id* du champ *name*. Ici l'identifiant de l'article vaut *53285* ce qui correspond au nom du fichier *53285.xml*. Cet article s'intéresse à Vauban. Le corps du document (étiquette *body*) est subdivisé en sous-parties : une sous-partie *figure* dans notre cas, une sous-partie *indentation1*, des paragraphes *p* et des sous-parties *section*, qui sont elles-mêmes subdivisées. Il s'agit de la structure hiérarchique du document. Notons par ailleurs que la collection compte un certain nombre de liens, des *collectionlinks* et des *outsidelinks* :

- les *collectionlink*s, par exemple ceux de la section "biography" du document *53285.xml* (milieu de la figure 7.1) correspondent à des liens internes à la collection : des doxels de la collection pointent vers des documents complets de la collection, identifiés par leur nom de fichier (associé à leur *id*) ;
- les *outsidelink*s consistent en des liens externes : ils pointent vers des pages Internet identifiées par leur url ; un exemple d'un tel lien est en bas de la figure 7.1.

Ainsi, la syntaxe originelle du Wiki a été convertie en XML en utilisant à la fois des balises générales qualifiant la structure hiérarchique des documents (par exemple *article, section, paragraph, title, list* et *item*), des balises typographiques (par exemple *bold, emphatic*), et des balises fréquentes de représentation de liens.

[27] http ://inex.is.informatik.uni-duisburg.de/2007/
[28] http ://trec.nist.gov/
[29] http ://clef-campaign.org/

```
<?xml version="1.0" encoding="UTF-8" ?>
- <article>
    <name id="53285">Vauban</name>
    <conversionwarning>0</conversionwarning>
  - <body>
    - <figure>
        <image xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="../pictures/Vauban-fortress.jpg" id="245404" xlink:actuate="onLoad"
          xlink:show="embed">Vauban-fortress.jpg</image>
        <caption>Vauban designed this pentagonal fortress to withstand sieges.</caption>
      </figure>
    - <indentation1>
      - <emph2>
          This article is about the French architect Vauban. See
          <unknownlink src="Vauban (disambiguation)">Vauban (disambiguation)</unknownlink>
          for other uses.
        </emph2>
      </indentation1>
    - <p>
        <emph3>Sébastien Le Prestre, Seigneur de Vauban</emph3>
        (
        <unknownlink src="May 15">May 15</unknownlink>
        ,
        <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="35162.xml">1633</collectionlink>
        –
        <unknownlink src="March 30">March 30</unknownlink>
        ,
        <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="34709.xml">1707</collectionlink>
        ), commonly referred to as
        <emph3>Vauban</emph3>
        , was a
        <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="310824.xml">Marshal of France</collectionlink>
        and the foremost
        <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="100034.xml">military engineer</collectionlink>
        of his age, famed for both his ability to design
        <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="204118.xml">fortifications</collectionlink>
        and to break through them.
      </p>
    - <section>
        <title>Biography</title>
      - <p>
          Vauban was born in Saint-Léger-de-Foucheret (renamed
          <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="1863265.xml">Saint-Léger-Vauban</collectionlink>
          in his honour), in the
        - <emph2>
            <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="2892207.xml">département</collectionlink>
          </emph2>
          of
          <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="90611.xml">Yonne</collectionlink>
          , in
          <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="36895.xml">Burgundy</collectionlink>
          ,
          <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="10581.xml">France</collectionlink>
          , into a family of minor nobility.
        </p>

[···]

      </section>
    - <section>
        <title>Fortifications</title>

[···]

      - <p>
          He designed the fortifications of
          <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="86918.xml">Besançon</collectionlink>
          .
        </p>
      </section>
    - <section>
        <title>Works</title>
      - <normallist>
        - <item>
            <emph2>De l'attaque et de la défense des places</emph2>
            ("On Siege and Fortification") (1737, many reprints since).
          </item>
        - <item>
            <emph2>A Manual of Siegecraft and Fortification</emph2>
            , transl.
            <collectionlink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple" xlink:href="194264.xml">George A. Rothrock</collectionlink>
            , (Ann Arbor, 1968).
          </item>
        </normallist>
      </section>
    - <section>
        <title>External link</title>
      - <normallist>
        - <item>
            <outsidelink xmlns:xlink="http://www.w3.org/1999/xlink" xlink:type="simple"
              xlink:href="http://www.geocities.com/Pentagon/6750/vauban_frame.htm">Marshal Vauban Homepage</outsidelink>
            (much detail)
          </item>
        </normallist>

[···]

      </section>
    </body>
  </article>
```

FIG. 7.1 – Extraits du document 53285.xml.

7.2 Requêtes

Une requête est la représentation la plus fidèle possible d'un besoin d'information. Dans INEX, ces requêtes (appelées aussi *topics*) sont créées par les participants, experts du domaine de la recherche d'information, sur la collection généraliste Wikipedia selon des instructions très précises [Tea08].

La figure 7.2 présente une requête rédigée dans le format spécifique à INEX. Les champs importants d'une requête sont :

- <title> dans lequel une liste de mots-clés décrivent le besoin d'information, cette partie de la requête porte uniquement sur le contenu des documents (Content Only (CO) query) : dans l'exemple, l'utilisateur qui a posé la requête 414 cherche des documents dont le contenu porte sur "hip hop beat".
- <castitle> est un autre champ titre, mais qui comporte une description dans la syntaxe NEXI [TS08], qui permet de prendre en considération des aspects contenu et structure pour la recherche (Content And Structure (CAS) query).
- <description> une description en une ou deux phrases du langage naturel du besoin d'information.
- <narrative> une description détaillée du besoin d'information qui motive la requête et définit ce qui est pertinent pour la requête et ce qui ne l'est pas.

```
<?xml version="1.0" encoding="ISO-8859-1" ?>
<!DOCTYPE inex_topic (View Source for full doctype...)>
- <inex_topic topic_id="414" ct_no="3">
    <title>hip hop beat</title>
    <castitle>//*[about(., hip hop beat)]</castitle>
    <description>what is a hip hop beat?</description>
    <narrative>To solve an argument with a friend about hip hop music and beats, I want to learn all there is to
      know about hip hop beats. I want to know what is meant by hip hop beats, what is considered a hip hop
      beat, what distinguishes a hip hop beat from other beats, when it was introduced and by whom. I consider
      elements relevant if they specifically mention beats or rythm. Any element mentioning hip hop music or
      style but doesn't discuss abything about beats or rythm is considered not relevant. Also, elements
      discussing beats and rythm, but not hip hop music in particular, are considered not relevant.</narrative>
  </inex_topic>
```

FIG. 7.2 – Requête n°414 de la campagne INEX 2007.

130 requêtes ont été sélectionnées pour la campagne INEX2007.

7.3 Jugements de pertinence

Les jugements concernant la pertinence des éléments renvoyés par les systèmes sont effectués par les participants eux-mêmes, à l'aide d'une interface mise à disposition en ligne. Les différents niveaux et degrés de pertinence, ainsi que la forme prise par l'interface d'évaluation ont changé chaque année depuis les débuts d'INEX.

Deux dimensions étaient utilisées en 2005 :

- l'exhaustivité, qui décrit à quel point un élément traite du sujet d'une requête ;
- la spécificité, qui décrit à quel point un élément ne traite que du sujet d'une requête.

En pratique, la tâche de l'assesseur humain consistait, pour une requête donnée et un document proposé, à surligner le contenu qu'il juge pertinent, puis à attribuer un score d'exhaustivité (choisi sur une échelle à quatre niveaux) aux doxels contenant du texte surligné. La spécificité était alors calculée automatiquement comme un ratio entre la longueur de texte surligné dans chaque élément et la longueur totale du texte. De plus, certaines règles automatiques spécifient des contraintes intuitives concernant l'exhaustivité. En particulier, les ancêtres d'un élément doivent avoir une exhaustivité égale ou supérieure à cet élément.

Cette tâche très lourde a incité les organisateurs d'INEX à revenir à une méthodologie d'évaluation moins complexe. En 2007, les assesseurs devaient uniquement surligner tous les passages pertinents, et seulement ces derniers dans un échantillon (*pool*) de documents. La granularité des passages pertinents est approximativement la phrase. Après avoir évalué chaque article, le meilleur point d'entrée dans les documents pertinents pour une requête donnée était marqué par les assesseurs.

7.4 Tâche Ad Hoc

La tâche Ad Hoc est composée de trois sous-tâches :
- *Focused Task* : l'objectif de cette tâche est de retourner à un utilisateur une liste triée des doxels les plus pertinents pour satisfaire sa requête.
- *Relevant In Context Task* : l'objectif de cette tâche consiste à renvoyer à un utilisateur l'information pertinente pour sa requête dans le contexte des documents complets ; ainsi, les doxels pertinents sont regroupés par articles classés selon leur score de pertinence.
- *Best In Context Task* : pour cette tâche, seul le doxel correspondant au meilleur point d'entrée dans un document pertinent est fourni en réponse à un utilisateur, les résultats étant ordonnés selon la pertinence des documents complets.

Les tâches *Focused Task* et *Relevant In Context Task* sont évaluées en utilisant les parties de documents surlignées par les assesseurs, tandis que la tâche *Best In Context Task* est évaluée en utilisant les meilleurs points d'entrée.

7.5 Métriques d'évaluation

Nous ne présentons ici que les métriques d'évaluation pour la tâche *Focused Task* de INEX2007, sur laquelle nous avons choisi de nous concentrer pour les expérimentations. Les métriques d'évaluation sont basées sur le postulat suivant : la quantité d'information pertinente retrouvée est mesurée proportionnellement à la longueur de texte retrouvé pertinente.

Les participants renvoient une liste triée des 1500 doxels les plus pertinents pour chaque requête. L'évaluation est effectuée sur cette liste, en comparaison avec les éléments marqués comme pertinents lors des jugements de pertinence.

Les mesures d'évaluation sont basées sur des mesures classiques de rappel et de précision utilisées en recherche d'information. Des mesures de précisions interpolées à des points de rappel donnés sont calculées et une mesure de la précision interpolée moyenne est également produite.

Formellement, soit p_r la partie de document classée au rang r dans la liste triée des parties de documents L_q renvoyées par un système de recherche d'information pour une requête q donnée ($|L_q| = 1500$ doxels). Soit $rsize(p_r)$ la longueur de texte, en nombre de caractères, évaluée comme pertinente pour la requête dans l'élément de document p_r. Soit $size(p_r)$ le nombre total de caractères de p_r et soit $Trel(q)$ la quantité totale de texte pertinent pour q selon les assesseurs, sur tous les documents de la collection. la précision interpolée iP au taux de rappel x est calculée par :

$$iP[x] = \begin{cases} max_{i \leq r \leq |L_q|}(P[r] \wedge R[r] \geq x) & \text{si } x \leq R[|L_q|] \\ 0 & \text{si } x > R[|L_q|] \end{cases}$$

avec :

$$P[r] = \frac{\Sigma_{i=1}^{r}\ rsize(p_i)}{\Sigma_{i=1}^{r}\ size(p_i)}$$

et

$$R[r] = \frac{\Sigma_{i=1}^{r}\ rsize(p_i)}{Trel(q)}$$

où $P[r]$, la précision au rang r est définie comme la fraction du texte retrouvé qui est pertinent et $R[r]$, le rappel au rang r est défini comme la fraction de texte pertinent qui est retrouvé.

En plus des valeurs de précision interpolée à des points de rappel donnés, un score de performance globale est attribué, basé sur la mesure de précision interpolée moyenne AiP :

$$AiP = \frac{1}{101}\Sigma_{x=0.00,0.01,...,1.00} iP[x].$$

La performance sur l'ensemble des topics t est mesurée comme la moyenne des valeurs AiP obtenue sur chaque topic. Supposons qu'il existe n topics :

$$MAiP = \frac{1}{n}.\Sigma_t AiP(t)$$

7.6 Choix de la collection INEX 2007

Dans cette section, nous souhaitons apporter quelques précisions sur le choix de cette collection INEX pour nos expérimentations.

Comme nous venons de le décrire, INEX fournit un ensemble de documents, un ensemble de requêtes et des jugements de pertinence, qui permettent d'évaluer les systèmes. L'avantage de cette collection est qu'elle comporte beaucoup de documents (plus de 650 000) et beaucoup de requêtes (plus de 200 requêtes en 2006 et 2007). De plus, le fait de participer à cette campagne permet de confronter notre système à d'autres systèmes académiques.

Finalement, la collection comporte initialement des liens (collectionlinks) ce qui nous offre la possibilité de tester notre modèle sans avoir besoin de créer notre propres liens, au moins dans un premier temps.

Chapitre 8

Expérimentations

Sommaire

Dans ce chapitre, notre objectif consiste à valider l'instanciation du modèle et de la fonction de correspondance proposée sur le corpus de documents structurés fourni pour la compétition INEX 2007. Ce corpus est suffisamment large pour permettre des tests réalistes, dans la mesure où le passage à l'échelle est une préoccupation constante en recherche d'information, et la campagne d'évaluation permet de confronter de manière réaliste notre système à d'autres systèmes de recherche d'information.

Nous appelons Gycori le système qui implémente le modèle que nous avons proposé en partie II et instancié au chapitre 6. En section 8.1, nous précisons quelques points d'instanciation de notre modèle pour la campagne INEX 2007 puis nous décrivons Gycori en section 8.2.

Nous avons ensuite déterminé trois axes pour mener nos expérimentations :

1. **Validation du modèle de recherche d'information sur des documents structurés** - nous avons tout d'abord choisi de valider notre modèle de recherche d'information pour des documents structurés en contrôlant la validité de notre modèle par rapport à des modèles existants, afin d'assurer des valeurs de références correctes pour supporter les expérimentations suivantes ;

2. **Validation de l'intérêt de l'environnement des doxels pour la recherche** - nous nous sommes intéressés à vérifier que la prise en compte de l'environnement des doxels est profitable à la recherche de documents structurés : nous proposons différentes configurations, avec des liens sortants et des liens entrants dans les doxels, nous cherchons également à ajuster les paramètres de la fonction de correspondance ; nous ajoutons un lien entre chaque doxel et le document complet qui l'englobe et nous analysons les nouveaux résultats obtenus ; nous adaptons la fonction de correspondance aux requêtes.

3. **Validation des documents virtuels** - nous montrons l'intérêt de la création de documents virtuels pour l'utilisateur : nous proposons alors de simuler une navigation dans l'espace des résultats et de vérifier que la navigation guidée par l'environnement permet d'améliorer les performances de recherche.

8.1 Instanciation de l'environnement pour INEX

L'instanciation de notre modèle que nous avons utilisée pour répondre à la campagne d'évaluation INEX correspond à l'instanciation proposée au chapitre 6, qui porte sur des documents XML.

Nous listons dans cette section quelques précisions relatives à cette instanciation pour l'environnement :

– les éléments retrouvables, c'est-à-dire les éléments de $\mathcal{C}_{ind}$ sont limités aux six balises `article`, `p`, `collectionlink`, `title`, `section` et `item`, cibles les plus fréquentes pour la collection INEX 2007 [NF07]. Ainsi, le nombre de doxels indexés est réduit à un peu plus de 29 millions d'éléments.

- l'environnement d'occurrence des doxels est donné par l'attribut XLink[30], quand il est utilisé par la balise `collectionlink`, et par le voisinage au sens sémantique du terme des doxels pères de ces `collectionlink`s avec des éléments de même type qu'eux dans le document cible. Considérons par exemple la figure 8.1 : le doxel d_{17} est de type `collectionlink` et pointe vers le document dont le nœud racine est le doxel d_{21}. Nous effectuons un calcul de plus proche voisin entre le doxel d_{14} père du doxel d_{17} et tous les doxels de d_{21} du même type que d_{14}, c'est-à-dire des paragraphes `p`. Si la similarité au sens du cosinus entre d_{14} et ces doxels est supérieure à un certain seuil, alors nous considérons que ces doxels sont dans l'environnement de d_{17}. De plus, nous créons un lien de d_{14} vers d_{21} pour conserver l'idée du lien initial.
- Au moment de l'indexation, nous avons utilisé $\lambda = 1$ pour des raisons de simplification des calculs ; par suite, $Exh_{globale}(d1, d2) = Exh_{apriori}(d_1, d_2)$ et $Spe_{globale}(d1, d2) = Spe_{apriori}(d_1, d_2)$.
  En pratique, le nombre de `collectionlink`s existant dans la collection INEX 2007 s'élève à près de 12 millions de liens et l'approche décrite ci-dessus produit près de 115 millions de relations d'environnement entre doxels.

8.2 Le système Gycori

Le système Gycori correspond à l'implémentation du modèle ainsi instancié pour la campagne INEX 2007. Il a pour objectif de permettre la recherche de parties de documents structurés, en prenant en compte les liens entre les doxels.

Comme tout système de recherche d'information, il est constitué d'une partie dédiée à l'indexation et d'une partie dédiée à l'interrogation, comme indiqué dans la figure 8.2. Ces parties, relativement indépendantes, sont en fait associées par le fait qu'elles partagent un certain nombre de données, nécessaires au bon fonctionnement du système entier. En figure 8.2, nous voyons que le processus d'indexation utilise le corpus initial de documents XML, pour générer un certain nombre de données intermédiaires dans une base de données. Le processus d'indexation utilise cette base de données pour générer un fichier inverse permettant une recherche rapide dans une grande base de documents. Le processus de recherche, lui, utilise les fichiers inverses générés pour retrouver rapidement les doxels pertinents, puis utilise la base de données pour retrouver le voisinage de ces doxels, afin d'afficher par l'intermédiaire de son interface les documents avec les doxels pertinents.

La base de données proposée comporte 7 tables :
- doxels : cette table comporte la liste de l'ensemble des doxels de la collection, avec leur identifiant *id_dox*, le *fichier* auquel ils appartiennent, le *chemin* d'accès au doxel dans ledit fichier, le *type* de doxel dont il s'agit (son tag), s'il s'agit d'un doxel feuille ou non avec le paramètre *est_feuille*, s'il s'agit d'un doxel retrouvable ou non avec le paramètre *est_tag*, la *norme* du doxel et son identifiant de vector quantization *id_vq* ;
- fichiers : pour un identifiant de fichier *fichier*, le *chemin* d'accès au fichier dans la collection est donné ;
- compose_dox : cette table associe à chaque doxel *id_compose* ses composants, ligne par ligne les *id_composant* ;

[30] XLink : http ://www.w3.org/TR/xlink/

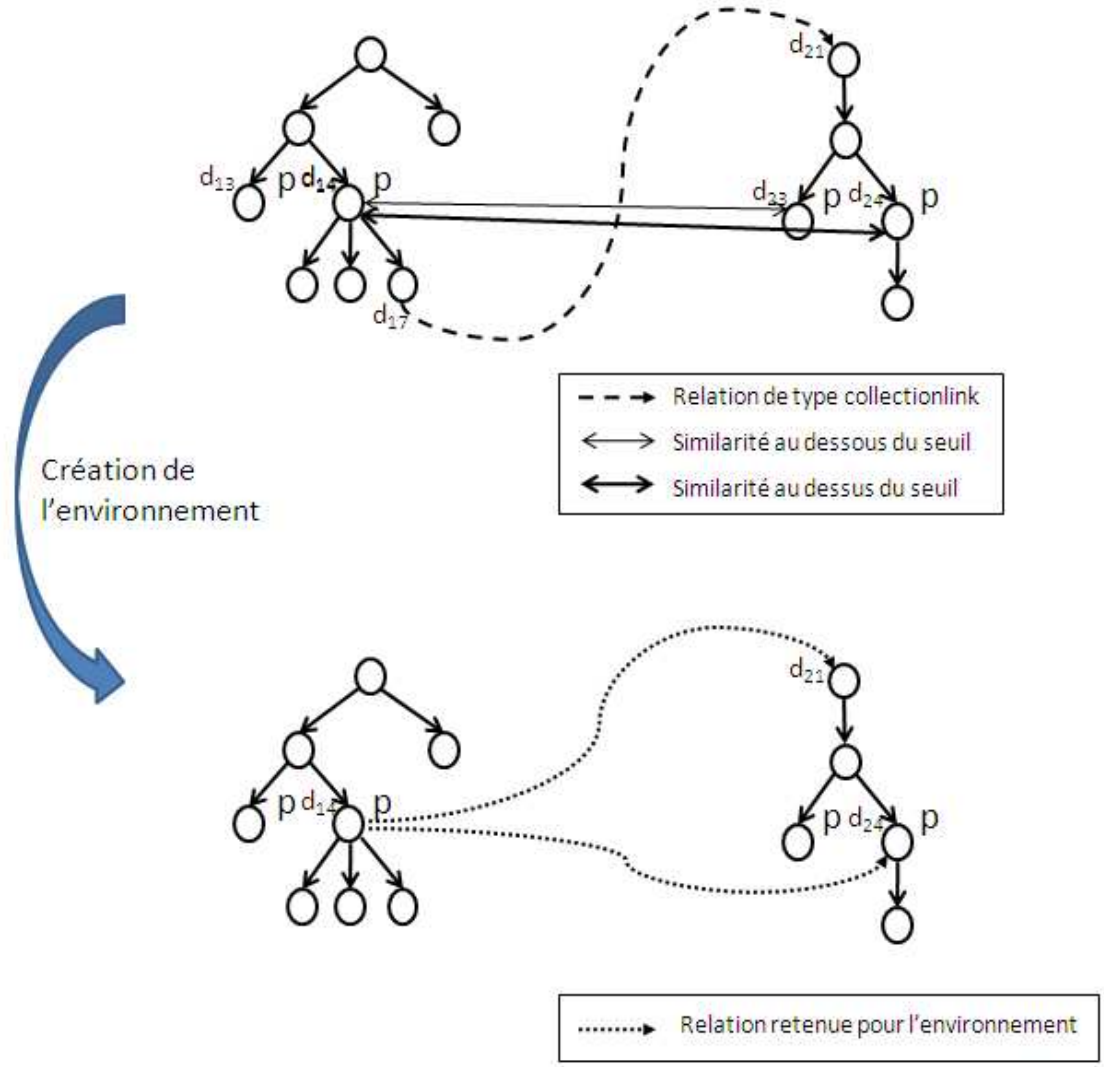

FIG. 8.1 – Création de l'environnement à partir d'un lien `collectionlink`.

- vocabulaire : à chaque *term* correspond un identifiant id_voc ;
- tf : pour un doxel id_dox_tf et un terme id_voc_tf de ce doxel donné, la fréquence du terme dans le doxel est donnée par $valeur\_tf$, et on connaît le *poids* associé ;
- idf : l'inverse doxel frequency est donné dans cette table, pour un terme de vocabulaire, id_voc_idf on lui associe sa valeur $valeur\_idf$;
- voisin : à un doxel id_dox_v1 est associé un voisin de ce doxel id_dox_v2, le *cosinus* entre ces deux doxels est stocké, ainsi que les valeurs d'exhaustivité *exh* et de spécificité *spe* correspondant à la relation orientée de id_dox_v1 vers id_dox_v2.

La figure 8.3 présente d'une manière graphique ces tables et leurs liaisons.

Des index ont été posés pour accélérer l'interrogation des tables en raison de leur taille (de l'ordre de millions d'éléments) pour de grandes collections.

Les travaux de développement réalisés sont relativement importants car nous avons recréé un système de recherche d'information complet, ce qui était une contrainte dans le cadre du projet

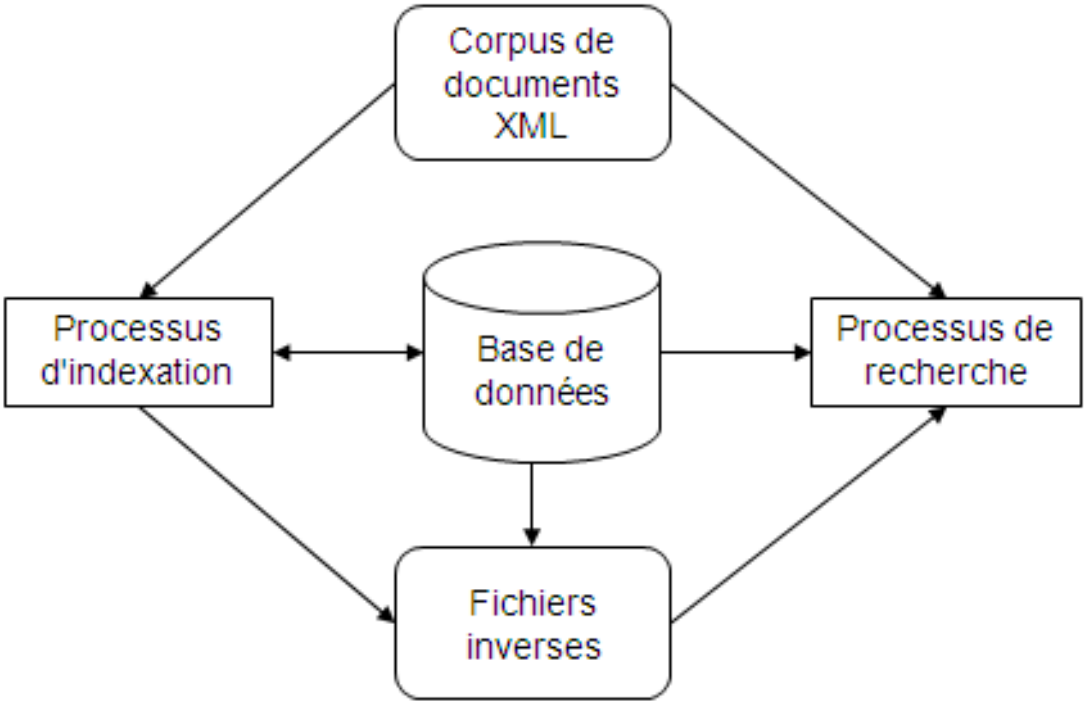

FIG. 8.2 – Description générale du système

dans lequel s'inscrit ce travail (projet France Telecom). Nous avons écrit tous les composants nécessaires à la démonstration de faisabilité de nos propositions. Ce développement est codé en Java (*jdk*1.6.0_01) dans un environnement de développement NetBeans[31]. De plus, afin de générer les grandes quantités d'informations telles que les liens entre les doxels, nous avons utilisé une base de données relationnelle (PostGresQL 8.2.[32]) interfacée avec Java par JDBC[33].

Les paquetages (packages) Java développés pour le prototype sont :

- DBLink : pour l'interface avec la base de données, afin de créer, de mettre à jour et d'accéder aux données intermédiaires et finales nécessaires.
- outils : paquetage regroupant l'ensemble des outils nécessaires à la mise en oeuvre du prototype. Ces outils vont de l'analyse des documents XML, à la génération des fichiers inverses, en passant par le calcul du voisinage des doxels.
- sri : paquetage qui contient le programme principal et qui permet d'exécuter l'indexation ou la recherche de documents.
- interface : gère l'interface avec l'utilisateur pour effectuer des recherches et présenter des résultats.

[31] Netbeans IDE 5.5 : http ://www.netbeans.org/
[32] PostgreSQL 8.2 : http ://www.postgresql.org/
[33] PostgreSQL JDBC Driver : http ://jdbc.postgresql.org/

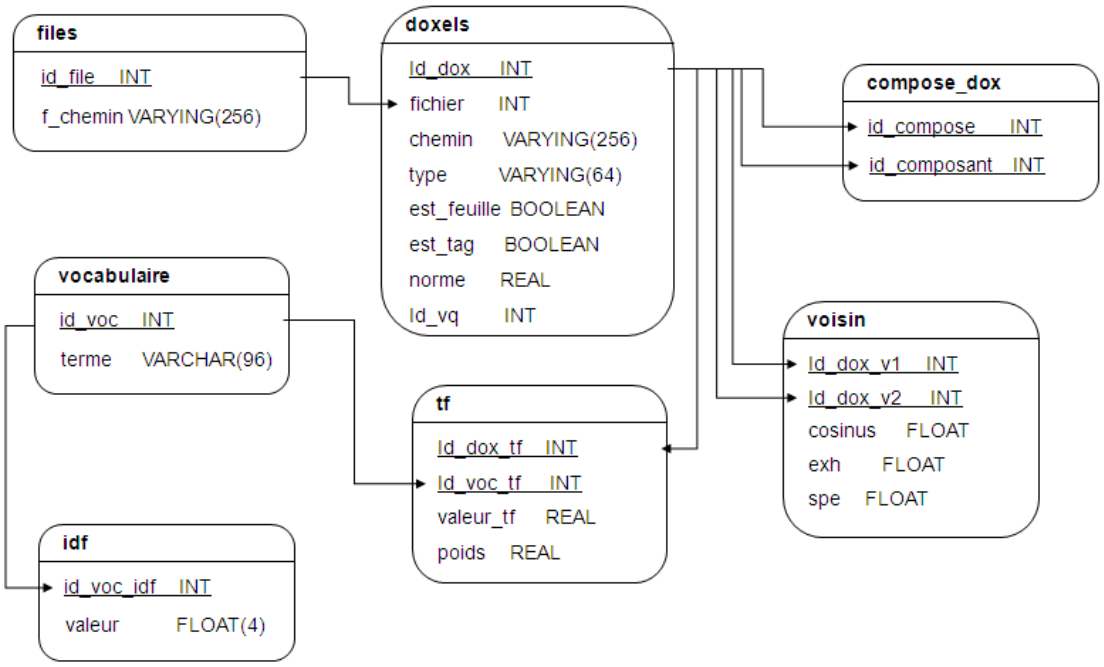

FIG. 8.3 – Schéma de la base de données

De manière plus globale, les caractéristiques du code développé sont les suivantes :

- Nb de lignes : 11418. Ce nombre correspond aux lignes de code avec les commentaires.
- Nb de classes : 58. Ce nombre correspond aux classes de l'application ainsi qu'à des classes de "travail" permettant par exemple d'opérer sur des couples de valeurs.
- Nb de méthodes : 354.

Tous les éléments utilisés ont pour avantage d'être aisément portables sur différentes plateformes, nous avons en particulier expérimenté avec succès ce portage sur machine Unix 64 Bits (Fedora Core 6) et sur machines sous Windows XP, sur lesquelles PostgreSQL et Java existent.

8.3 Validation du modèle de référence

Nous comparons dans cette section notre système avec un système implantant plusieurs systèmes de recherche d'information, Zettair [Zse], afin de valider le modèle de référence.

Zettair est un moteur de recherche Open Source qui permet l'indexation et la recherche de fichiers au format HTML. L'intérêt principal de ce moteur est sa capacité à manipuler des collections de documents de grande échelle (100 Go ou plus), de façon très rapide et efficace. Il est développé par le groupe de recherche d'information RMIT University à Melbourne.

Étant donné que Zettair ne propose pas la prise en compte des documents structurés, nous choisissons de comparer une version de notre système limitée au traitement de documents complets avec Zettair. Nous utilisons Zettair implémenté avec un modèle vectoriel puis avec un modèle de langue.

Rappelons que le modèle de langue, basé sur des calculs de probabilités, est un modèle dont les résultats ont, dans certains cas, dépassé les résultats obtenus par le modèle vectoriel. Inspiré de travaux sur la reconnaissance de parole, il estime la pertinence d'un document pour une requête comme la probabilité que le document génère la requête.

Dans le modèle de langue, la valeur de correspondance entre un document D et une requête Q est estimée par :

$$RSV(Q,D) \ = \ P(Q|D) \ = \ P(tq_1,\ldots,tq_n|M_D)$$

avec M_D le modèle de document. Si l'on considère, comme couramment en recherche d'information, que tous les termes sont indépendants conditionnellement au document, on a :

$$P(tq_1,\ldots,tq_n|M_D) \ = \ \Pi_{t_{q_i}} P(tq_i|M_D)$$

Une des grandes forces des modèles de langue est que les probabilités sont estimées en se basant sur les informations du document, mais également en se basant sur des informations tirées du corpus complet par l'intermédiaire d'un lissage. Le lissage de Dirichlet, utilisé par Zettair, est le suivant :

$$P(t|M_D) \ = \ (tf_{t,D} + \mu.P(t|C)) \ / \ (|D| + \mu)$$

avec μ fixé a priori, $tf_{t,D}$ le nombre d'occurrences de t dans D, et $|D|$ le nombre de termes de D.

$$P(t|C) = \frac{\#t \text{ dans le corpus de documents } C}{\text{nombre de termes total dans } C}$$

En utilisant cette approche, nous pouvons calculer la correspondance entre un document et une requête.

Nous avons donc utilisé Zettair pour calculer la correspondance entre une requête Q et les documents entiers avec le modèle de langue. Le paramètre μ a été fixé à 1500.

TAB. 8.1 – Comparaison de notre système avec Zettair (INEX 2007).

Système (modèle)	iP@0.00R	iP@0.01R	iP@0.05R	iP@0.10R	MAiP
Gycori (modèle vectoriel)	0.2867	0.2838	0.2694	0.2597	0.1354
Zettair (modèle vectoriel)	0.2412	0.2038	0.1643	0.1355	0.0720
Zettair (modèle de langue)	0.3394	0.3392	0.3209	0.2973	0.1599

Les résultats obtenus sont récapitulés dans le tableau 8.1 et sur les courbes de la figure 8.4.

Ces résultats montrent que notre approche basée sur les documents complets fournit des résultats meilleurs ($+87,9\%$) en précision moyenne que ceux obtenus par l'utilisation de Zettair implanté avec le modèle vectoriel. De même, les valeurs de précision aux points de rappel 0.00, 0.01, 0.05 et 0.10 sont améliorées pour notre approche face à l'approche Zettair basée sur le modèle vectoriel. Pour ces résultats, les calculs de significativité statistique ne sont pas utiles.

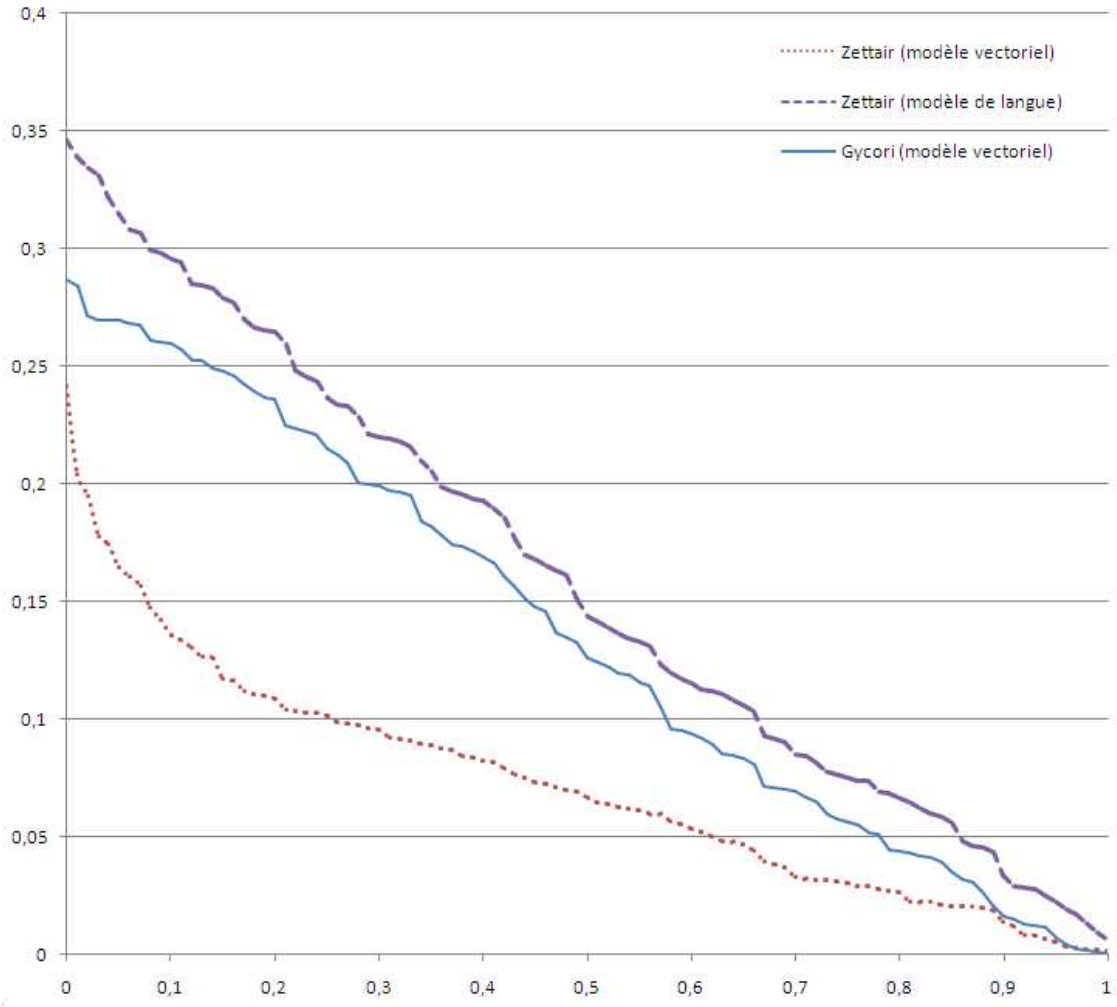

FIG. 8.4 – Comparaison de notre système avec Zettair (INEX 2007).

Par contre, les différentes configurations que nous avons expérimentées sur le modèle Zettair montrent que l'approche par modèle de langue présente des performances de précision moyenne de l'ordre de 120% supérieures à celle de l'approche par modèle vectoriel de Zettair et de 18% supérieures à celle de Gycori. Le comportement de Gycori est cependant plus proche du comportement de Zettair avec le modèle de langue que Zettair avec le modèle vectoriel pour lequel la précision diminue très rapidement dès la valeur de rappel $iP@0.01R$.

Les courbes de rappel / précision de la figure 8.4 confirment le comportement des différents systèmes : notre système, Gycori, correspond à la courbe située au milieu, entre deux configurations du système Zettair expérimentées. Notre système donne de meilleures performances que Zettair basé sur le modèle vectoriel, ce qui provient probablement du fait que Gycori utilise un modèle vectoriel avec des *idf* basés sur les doxels et non pas sur les documents complets.

Finalement, le modèle que nous avons implémenté présente des résultats intermédiaires ; le modèle est ainsi validé, pour supporter les différentes expérimentations qui suivent.

8.4 Validation du modèle utilisant l'environnement

Dans cette section, nous souhaitons valider le modèle vis-à-vis des aspects de prise en compte du voisinage. Pour ces expérimentations, nous nous appuyons sur les liens préexistants dans la collection de documents, les *collectionlinks*, présentés en section 8.1.

Les liens construits constituent l'environnement des doxels : les doxels possèdent des liens sortants et des liens entrants, ils sont sources ou cibles de liens ou encore à la fois sources et cibles de liens.

Nous effectuons d'abord des expérimentations qui considèrent ces liens pris séparément, les liens sortants puis les liens entrants sont pris en compte, puis nous poursuivons les expérimentations en considérant les deux types de liens.

8.4.1 Prise en considération des liens

Liens sortants

Nous considérons dans un premier temps les liens sortants des doxels comme décrivant l'environnement de ces derniers. Ces travaux ont fait l'objet de publications dans [VM08] et dans [VM07].

Nous choisissons d'effectuer deux générations de résultats différentes :

- Type FOC_1 : Les premiers résultats que nous retournons sont générés en répondant à la tâche *Focused Task* de la manière la plus simple possible : les doxels sont classés par ordre de pertinence décroissante relativement à la requête posée (Tableau 8.2 et Figure 8.5).
- Type FOC_2 : Nous choisissons de nous baser sur les éléments renvoyés pour la tâche *Relevant In Context Task* pour produire de nouveaux résultats pour la tâche *Focused Task* : les doxels sont alors regroupés par article et ordonnés sur le score des articles ; les doxels sont pondérés par le score de l'article auquel ils appartiennent et ils sont listés selon leur ordre d'apparition dans le document. Par ce biais, nous prenons en compte le score des documents lors de l'interrogation (Tableau 8.3 et Figure 8.6).

Pour les expérimentations, nous considérons différentes configurations qui prennent en compte :

- le nombre de voisins sortants ν variant de 0 (notre référence) à tous les voisins pris en compte (*all*) en considérant quelques valeurs intermédiaires ;
- l'importance accordée au contenu des doxels via le paramètre α, α indiquant l'importance de la correspondance relative du doxel par rapport à son voisinage ;
- l'importance accordée à l'exhaustivité face à la spécificité du voisinage des doxels par l'intermédiaire du paramètre β.

Pour un α fixé... Nous choisissons dans cette première série d'expérimentations de fixer $\alpha = 0.5$. Nous effectuons ce choix afin de donner autant d'importance aux informations provenant des documents complets qu'aux informations provenant de l'environnement des doxels. Le nombre de voisins ν varie.

TAB. 8.2 – Résultats de la prise en compte du voisinage - liens sortants (type FOC_1).

$Run(\nu, \alpha, \beta)$	iP@0.00R	iP@0.01R	iP@0.05R	iP@0.10R	MAiP
$(0, x, x)$	**0.3237**	0.1968	0.1416	0.1079	0.0364
$(1, 0.5, 0.5)$	0.2816	**0.2272**	**0.1735**	0.1309	0.0573
$(3, 0.5, 1)$	0.2571	0.2089	0.1662	**0.1324**	0.0567
$(4, 0.5, 1)$	0.2570	0.2100	0.1599	0.1259	0.0561
$(all, 0.5, 0.25)$	0.2726	0.2096	0.1635	0.1281	**0.0573**

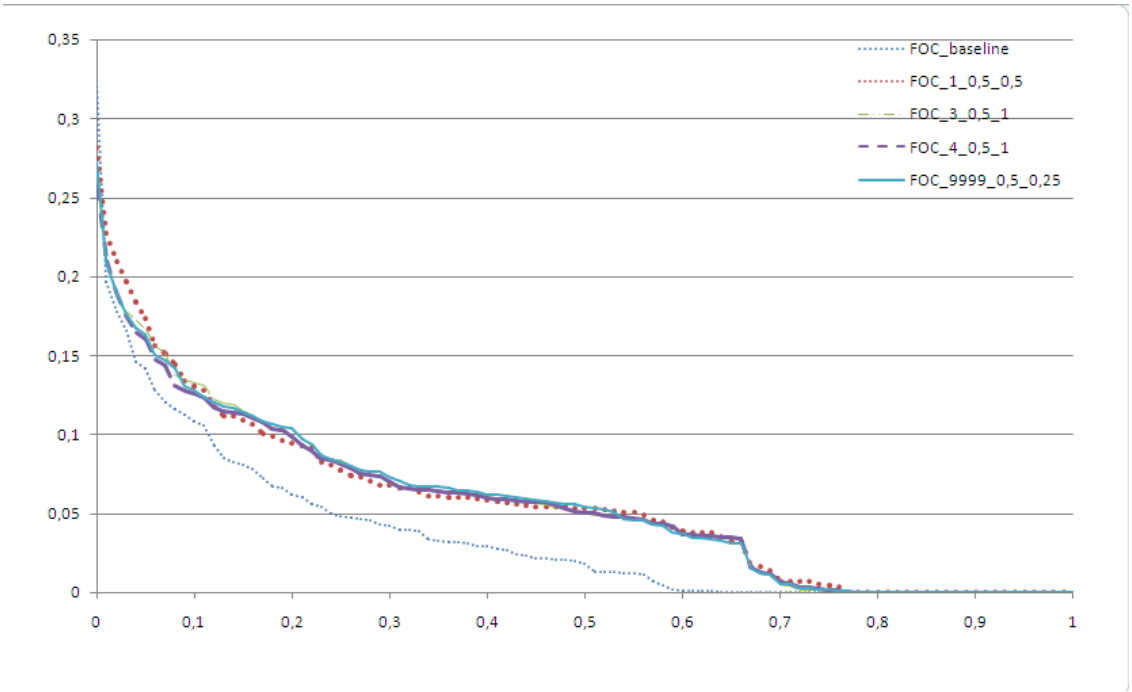

FIG. 8.5 – Courbe avec prise en compte du voisinage - liens sortants (type FOC_1).

Nous présentons dans les tables 8.2 et 8.3 les meilleurs résultats en terme de précision moyenne pour un nombre de voisins fixé, pour des raisons de clarté de lecture des résultats. Les meilleures valeurs pour chaque mesure sont mises en gras. Les tableaux comportent également les résultats du t-test de Student bilatéral pour des échantillons appariés, qui est un test courant en recherche d'information [Hul93] : les résultats statistiquement significatifs par rapport à la meilleure valeur sont soulignés. Ce test sera utilisé tout au long de ce chapitre pour comparer les données.

Dans le cas FOC_1 (table 8.2), le système sans voisinage donne le meilleur iP au point de rappel 0.00. Par contre, dès que l'on considère un voisinage avec $\alpha = 0.5$, avec un voisin et plus, on améliore le $MAiP$ et les iP après le point de rappel 0.00 sont tous meilleurs avec des voisins. Nous remarquons plus précisément que le système utilisant 1 seul voisin renvoie de meilleurs résultats si l'on considère les premiers doxels retournés. En moyenne, les résultats sont significativement meilleurs ($MAiP$) quand le système exploite l'environnement complet des doxels pour la recherche.

La courbe de la figure 8.5 confirme ces résultats : entre 0 et 0.1, au départ, c'est la courbe correspondant au système sans voisinage qui se situe au-dessus, elle repasse au-dessous de toutes les autres courbes juste après et la courbe correspondant à la prise en compte d'un voisin est au-dessus des autres ; après 0.1, les performances en tenant compte du voisinage sont meilleures, et les résultats sont significatifs en $MAiP$. Sur la courbe, à partir de $iP@0.01R$, les courbes avec voisinage se rapprochent et il existe un décrochement qui correspond au fait que notre système ne renvoie plus de résultats pertinents. Nous voyons cependant que les valeurs de précisions après $iP@0.06R$ sont positives lorsqu'on considère l'environnement alors qu'elles sont nulles pour le système sans environnement, ce qui veut dire que l'environnement permet de retourner plus de 70% des pertinents alors que seulement 60% des pertinents sont renvoyés sans le prendre en compte : nous améliorons donc le rappel global.

TAB. 8.3 – Résultats de la prise en compte du voisinage - liens sortants (type FOC_2).

$Run(\nu, \alpha, \beta)$	iP@0.00R	iP@0.01R	iP@0.05R	iP@0.10R	MAiP
$(0, x, x)$	**0.3173**	0.2918	0.2484	0.2189	0.0777
$(1, 0.5, 0.5)$	0.3154	0.3020	0.2653	0.2346	0.0939
$(3, 0.5, 0)$	0.3171	0.3041	0.2677	0.2314	0.0940
$(4, 0.5, 0)$	0.3165	**0.3048**	**0.2686**	0.2340	0.0942
$(all, 0.5, 0)$	0.3115	0.3000	0.2675	**0.2360**	**0.0962**

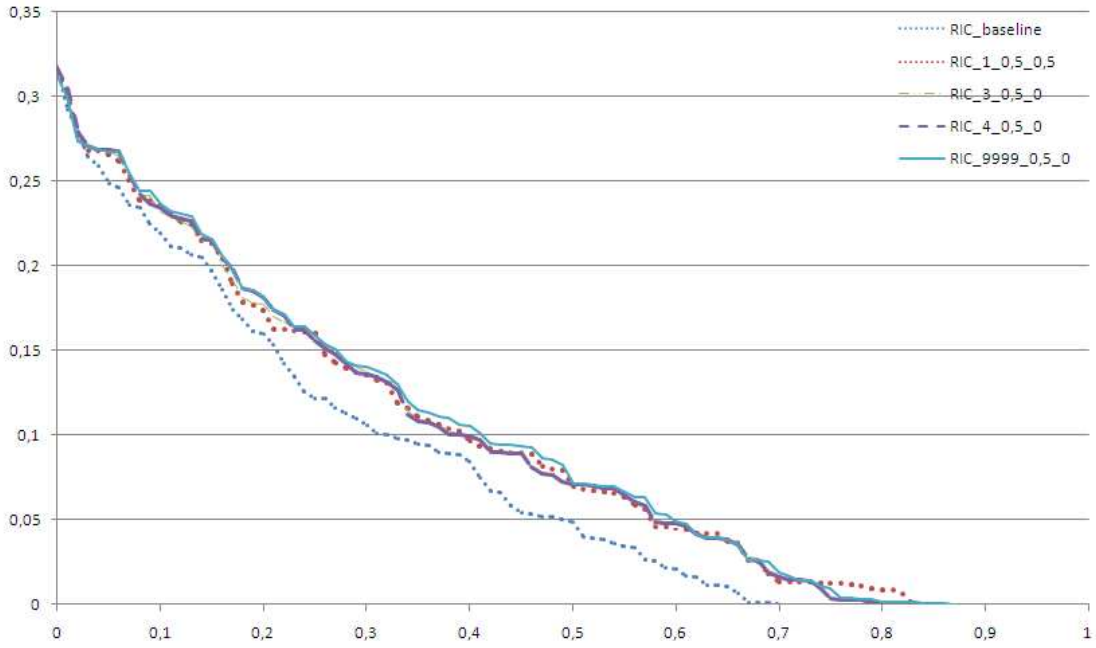

FIG. 8.6 – Courbe avec prise en compte du voisinage - liens sortants (type FOC_2).

Dans le cas FOC_2 (table 8.3), le système sans voisinage donne également un meilleur iP au point de rappel 0.00. Les résultats avec voisinage pour $\alpha = 0.5$ permettent d'améliorer la valeur de iP pour les autres points de rappel, et nous notons que les scores augmentent légèrement avec le nombre de voisins pris en compte. Les résultats sont significatifs à $iP@0.05R$ et $iP@0.10R$ par rapport au résultat sans voisinage. La meilleure moyenne $MAiP$ est obtenue en considérant tous les voisins sortants d'un doxel, tout comme dans le cas FOC_1 de la table 8.2, et ces résultats sont significatifs.

La courbe de la figure 8.6 montre ces mêmes résultats : jusqu'à $iP@0.05R$, les résultats avec voisinage restent très proches de la référence, puis l'écart se creuse. Vers $iP@0.02R$ la courbe correspondant au système avec un voisin est très proche de la référence. Ces courbes sont cependant plus linéaires que dans la première expérimentation (FOC_1), ce qui souligne le fait que la contiguïté des éléments de documents joue un rôle positif. Au final, c'est le système tenant compte de tous les voisins possibles qui est le plus performant sur les plus petites valeurs de rappel et également entre 0.03 et 0.05 où la courbe se distingue clairement au dessus des autres courbes.

Pour les deux types de résultats produits, FOC_1 et FOC_2, les résultats obtenus en prenant en compte un voisinage améliore la référence sans voisin. Les courbes présentées respectivement en figure 8.5 et 8.6 montrent que le comportement du système avec considération du voisinage est située nettement au-dessus de la courbe de référence, ce qui montre que les résultats avec voisinage permettent d'augmenter les performances. La prise en compte du voisinage permet d'améliorer les résultats en moyenne de 57,42% dans l'expérimentation de type FOC_1, et de 23,81% dans la configuration FOC_2.

De plus, en FOC_2 comme en FOC_1, nous remarquons que le rappel global est amélioré comparativement au système de référence ; globalement, la prise en compte des voisins permet d'affiner les résultats obtenus par la référence et les résultats sont meilleurs que ceux atteints en FOC_1 pour les résultats avec voisinage (1, 3, 4 ou tous les voisins) ce qui valide l'intérêt de notre modèle. D'autre part, les résultats obtenus en considérant une réorganisation par document (FOC_2) sont meilleurs ; dans la suite, nous ne considérons de ce fait que des résultats de type FOC_2.

Pour un ν fixé... Nous fixons maintenant ν à 4 voisins sortants, comme c'est la configuration qui nous donnait les meilleurs résultats pour les premiers points de rappel dans la table 8.3 et faisons varier α et β. Les valeurs choisies pour α sont 0.5 et 0.75, ce qui permet de changer l'importance relative des documents complets par rapport à l'environnement ; nous ne prenons pas de valeurs plus petites, les résultats présentés en état de l'art soulignaient l'importance des documents pour le calcul de la fonction de correspondance. Les valeurs pour β varient par pas de 0.25 entre 0 et 1, afin de donner plus ou moins d'importance à l'exhaustivité et/ou à la spécificité. Les résultats obtenus sont présentés dans la table 8.4.

Lors des expérimentations, nous avons constaté que les résultats, pour un nombre de voisins donné et pour des paramètres α et β variables, sont similaires et non statistiquement significatifs. Ainsi, dans les expérimentations qui suivent, nous ne faisons plus varier β et nous choisissons $\beta = 0$ pour $\nu = 4$, car les résultats sont meilleurs en $iP@0.00R$, $iP@0.01R$, $iP@0.05R$, $iP@0.10R$ et $MAiP$ même sans être significatifs statistiquement.

TAB. 8.4 – Résultats de la prise en compte du voisinage - liens sortants (type FOC_2).

$Run(\nu, \alpha, \beta)$	iP@0.00R	iP@0.01R	iP@0.05R	iP@0.10R	MAiP
$(4, 0.5, 0)$	**0.3165**	**0.3048**	**0.2686**	0.2340	**0.0942**
$(4, 0.5, 0.25)$	0.3115	0.2994	0.2664	0.2347	0.0929
$(4, 0.5, 0.5)$	0.3111	0.2991	0.2658	0.2341	0.0930
$(4, 0.5, 0.75)$	0.3123	0.3006	0.2669	**0.2350**	0.0923
$(4, 0.5, 1)$	0.3126	0.3003	0.2668	0.2349	0.0920
$(4, 0.75, 0)$	0.3139	0.3017	0.2648	0.2297	0.0845
$(4, 0.75, 0.25)$	0.3135	0.3013	0.2646	**0.2299**	0.0846
$(4, 0.75, 0.5)$	0.3147	0.3026	0.2656	0.2293	**0.0866**
$(4, 0.75, 0.75)$	0.3147	0.3026	0.2656	0.2288	0.0866
$(4, 0.75, 1)$	**0.3149**	**0.3027**	**0.2657**	0.2290	0.0864

Liens entrants

Pour la série d'expérimentations précédentes, nous avons considéré les liens sortants des doxels : en effet, nous estimons que si un auteur a référencé un autre doxel, ce dernier présente un intérêt pour le document en cours d'étude. De la même manière, nous considérons que si un élément est référencé par un autre, l'élément source de la référence est potentiellement intéressant pour la recherche. Pour cette raison, nous étudions l'impact des liens entrants pour la recherche d'information.

TAB. 8.5 – Résultats de la prise en compte du voisinage - liens entrants (type FOC_2).

$Run(\nu, \alpha, \beta)$	iP@0.00R	iP@0.01R	iP@0.05R	iP@0.10R	MAiP
$(0, x, x)$	0.3173	0.2918	0.2484	0.2189	0.0777
$(1, 0.5, 0.5)$	**0.3283**	0.3029	**0.2714**	0.2482	0.0903
$(3, 0.5, 0)$	0.3244	**0.3033**	0.2703	0.2490	0.0947
$(4, 0.5, 0)$	0.3180	0.2972	0.2674	**0.2494**	**0.0954**
$(all, 0.5, 0)$	0.3182	0.3006	0.2698	0.2470	0.0944

Les résultats décrits dans la table 8.5 montrent que les liens entrants apportent des améliorations : les résultats sont meilleurs mais non significativement à $iP@0.00R$ et $iP@0.01R$, et sont significativement meilleurs quelque soit le nombre de voisins entrants considérés, après le point de rappel 0.05. En moyenne, les résultats sont meilleurs en considérant 4 voisins et cette valeur est significativement différente par rapport à la valeur de référence et à la valeur de $MAiP$ correspondant à 1 lien entrant.

D'autre part, les résultats obtenus augmentent les performances atteintes avec la prise en compte des voisins sortants, aux points de rappel 0.00, 0.05 et 0.10. Les valeurs en $MAiP$ restent cependant inférieures de près de 1% à celles obtenues pour les liens sortants.

Environnement complet

Nous avons étudié l'impact des liens sortants et des liens entrants indépendamment ; nous voulons maintenant connaître l'impact de ces liens considérés simultanément pour déterminer si les informations des liens entrants complètent celles des liens sortants ou non.

Ces résultats sont calculés et ordonnés sur la RSV suivante :

$$RSV(d,q) = \alpha * RSV^0_{d,q} + (1-\alpha) * \frac{RSV_{env_{OUT}}(d,q) + RSV_{env_{IN}}(d,q)}{2}$$

où $RSV_{env_{OUT}}(d,q)$ et $RSV_{env_{IN}}(d,q)$ sont les valeurs de pertinence des environnements issus des liens sortants et entrants correspondant à la formule RSV_{env} présentée en section 5.3 et limitée à des liens respectivement uniquement sortants ou uniquement entrants.

Les résultats obtenus sont présentés dans la table 8.6. Dans la colonne $Run(\nu_{out}, \nu_{in}, \alpha, \beta)$, ν_{out} et ν_{in} correspondent respectivement au nombre de liens sortants et entrants. Pour ces premières expérimentations sur l'environnement complet, nous avons considéré le même nombre de liens sortants et entrants.

TAB. 8.6 – Résultats de la prise en compte du voisinage - liens entrants et sortants (type FOC_2).

$Run(\nu_{out}, \nu_{in}, \alpha, \beta)$	iP@0.00R	iP@0.01R	iP@0.05R	iP@0.10R	MAiP
$(0, 0, x, x)$	0.3173	0.2918	0.2484	0.2189	0.0777
$(1, 1, 0.5, 0.5)$	**0.3173**	**0.3038**	**0.2675**	**0.2483**	0.0915
$(3, 3, 0.5, 0)$	0.3152	0.3003	0.2651	0.2454	0.0912
$(4, 4, 0.5, 0)$	0.3091	0.2945	0.2641	0.2482	0.0933
$(all, all, 0.5, 0)$	0.3031	0.2906	0.2622	0.2477	**0.0938**

La prise en considération de l'environnement complet des doxels montre que le meilleur lien sortant et le meilleur lien entrant apportent de meilleurs résultats que toutes les autres configurations pour les premiers points de rappel, pour α et β correspondant aux meilleurs scores pour des liens sortants (cf section 8.4.1). Par contre, pour le $MAiP$, les meilleures performances sont obtenues en considérant l'ensemble des liens sortants et entrants pour chaque doxel.

Cependant, ces résultats sont moins bons que les meilleurs résultats obtenus en ne considérant que les liens sortants ($-2,7\%$ en $MAiP$) ou que les liens entrants ($-1,6\%$ en $MAiP$).

8.4.2 Validation du modèle utilisant les documents complets

Dans cette section, nous nous sommes intéressés à ajouter un lien entre chaque doxel et le document complet qui l'englobe, comme l'état de l'art avait souligné l'intérêt de la prise en compte des documents complets pour les documents structurés. Nous avons utilisé *a posteriori* ce lien pour le calcul de correspondance et en appliquant le système Zettair [Zse] déjà présenté en section 8.3 pour calculer la valeur de pertinence des documents complets. La nouvelle valeur de pertinence ainsi obtenue est la suivante :

$$RSV_{avecDoc}(d,q) = RSV(d,q) + [\beta * Exh(d,D) + (1-\beta) * Spe(d,D)] * RSV(D,q)$$

avec D le document complet englobant d, *rang* la fonction qui donne la position du document D dans la liste des résultats de Zettair et $RSV(D,q) = rang(RSV_{Zettair}(D,q))$.

Un lien d'un doxel vers le document complet qui l'englobe a une valeur d'exhaustivité égale à 1 par définition et nous posons $\beta = 1$ pour cette expérimentation.

D'une part, en section 8.3, nous avons en effet montré que le modèle de langue donne des performances de base sur les documents meilleures que le modèle vectoriel. D'autre part, les études de l'état de l'art telles que [MSDWZ93], [Cal94] ou encore [LC02] ont montré l'intérêt de prendre en considération la valeur de pertinence des documents complets pour une recherche d'information sur des documents structurés. Certes, nous avons intégré cet aspect dans notre modèle, mais dans la mesure où le modèle de langue offre de meilleurs résultats, nous choisissons de tenir compte de l'ordre proposé par Zettair appliqué à la collection avec un modèle de langue, même si cette expérimentation nous fait prendre quelques libertés avec le modèle défini.

Nous avons ainsi réordonné nos résultats initiaux en nous basant sur l'ordre des documents de Zettair dont les résultats sont rappelés à la première ligne du tableau. Nous avons testé cette fusion avec les résultats de base et certains résultats obtenus en prenant en compte l'environnement des doxels, comme le montre le tableau 8.7. Dans cette table, des tests de significativité ont été effectués pour les sous-tables correspondant à un regroupement en fonction de ν, les valeurs qui sont statistiquement significatives par rapport au meilleur score pour chaque colonne sont soulignées ; d'autres tests ont été effectués sur les valeurs en gras pour chaque colonne, comparativement à la meilleure valeur par colonne (suivie d'un astérisque), les résultats statistiquement significatifs apparaissent alors en italique (et en gras).

TAB. 8.7 – Réorganisation des doxels pertinents pour Gycori avec liens sortants, selon les résultats de Zettair sur les documents complets avec un modèle de langue.

$Run(\nu,\alpha,\beta)$	iP@0.00R	iP@0.01R	iP@0.05R	iP@0.10R	MAiP
Zettair (modèle de langue)	0.3394	0.3392	0.3209	0.2973	0.1599
$(0,x,x)$ *initial*	0.3173	0.2918	0.2484	0.2189	0.0777
$(0,x,x) + Zettair - FOC_1$	**0.4037**	0.2441	0.1256	0.0829	0.0347
$(0,x,x) + Zettair - FOC_2$	0.3418	***0.3306***	**0.2822**	**0.2598**	***0.0809***
$(1,0.5,0.5)$ *initial*	0.3154	0.3020	0.2653	0.2346	0.0939
$(1,0.5,0.5) + Zettair - FOC_1$	**0.4169***	0.3476	0.2896	0.2378	0.0907
$(1,0.5,0.5) + Zettair - FOC_2$	0.3950	**0.3648**	***0.3106***	**0.2777**	**0.1072**
$(all,0.5,0)$ *initial*	0.3115	0.3000	0.2675	0.2360	0.0962
$(all,0.5,0) + Zettair - FOC_1$	**0.4137**	0.3444	0.2942	0.2375	0.0932
$(all,0.5,0) + Zettair - FOC_2$	0.3940	**0.3794***	**0.3140***	**0.2846***	**0.1096***

Finalement, en appliquant une réorganisation des doxels selon la valeur de pertinence des articles auxquels ils appartiennent calculée avec Zettair sur un modèle de langue, nous avons amélioré les résultats obtenus uniquement à partir des valeurs de pertinence des doxels et de leur voisinage éventuel, résultats ayant été obtenus en s'appuyant sur un système basé sur le modèle vectoriel.

Au point de rappel $iP@0.00R$, les meilleurs résultats sont obtenus en combinant les résultats de la recherche de type FOC_1 avec le modèle de langue sur les documents complets, en ne prenant

en compte aucun voisin, et 1 seul voisin ou tous les voisins, et les valeurs sont statistiquement significatives. Pour tous les autres points de rappel et pour le $MAiP$, les résultats sont les plus performants et statistiquement significatifs en combinant les résultats selon l'approche FOC_2 et les résultats de Zettair (modèle de langue) sur les documents complets.

Les résultats obtenus montrent que cette approche permet d'obtenir de meilleurs résultats : pour 0 voisin +27.23% à $iP0.00$ et +4.12% en précision moyenne ; pour 1 voisin, +32.18% à $iP0.00$ et +14.16% en précision moyenne ; en considérant tous les voisins, +32.81% à $iP0.00$ et +13.93% en précision moyenne. Ceci confirme donc l'intérêt de la prise en compte des documents complets et l'apport du modèle de langue pour la recherche de documents structurés.

8.4.3 Adaptation du modèle aux requêtes

Dans cette section, nous cherchons à établir une corrélation entre le type de requête posée et le paramètre β appliqué dans la fonction de correspondance.

Par exemple, la requête 427 d'INEX 2007 est décrite par "Find a list of Hugo awarded best novels" : l'utilisateur qui pose cette requête s'attend à recevoir la liste complète (*all*) des meilleurs romans de Science Fiction ayant reçu le prix *Hugo Award*. La requête 465 d'INEX 2007 est quant à elle décrite par "I'm looking for differences between American English and British English", requête pour laquelle l'utilisateur ne veut que les différences les plus remarquables (best) entre l'anglais américain et l'anglais de Grande Bretagne. Sur l'ensemble des requêtes d'INEX 2007, 60% des requêtes sont des requêtes de type *all*, 35% sont des requêtes de type *best* et les autres requêtes ne sont pas typées.

À partir de ces informations, nous avons décidé de moduler le paramètre β et d'analyser les résultats engendrés dans ce cas, en prenant en compte $\beta = 0$ et $\beta = 1$, d'un côté sur les requêtes pour lesquelles l'utilisateur avait estimé qu'il préférait recevoir tous les résultats pertinents (*all*) et de l'autre sur les requêtes pour lesquelles l'utilisateur avait estimé qu'il ne voulait que les meilleurs résultats (*best*). Les résultats obtenus pour un nombre de liens sortants fixé à 4, et un alpha fixé à 0.5 apparaissent dans la table 8.8.

Nous nous attendions à obtenir les scores les meilleurs pour un paramètre $\beta = 0$ qui favorise l'exhaustivité et un mode *all*, et les scores les meilleurs pour un paramètre $\beta = 1$ qui favorise la spécificité et un mode *best*.

Les résultats que nous obtenons montrent que l'hypothèse concernant la relation entre le paramètre $\beta = 0$ et le mode *all* semble confirmée, bien que les résultats ne soient significatifs que pour le $MAiP$. Les scores les meilleurs à $iP@0.00R$, $iP@0.01R$, $iP@0.05R$ et $iP@0.10R$ apparaissent pour le paramètre $\beta = 0$ (3 premières lignes du tableau). Par contre la relation entre $\beta = 1$ et le mode *best* n'est pas validée, aucun des résultats n'est significatif, cela est peut-être dû au petit nombre de requêtes (32). D'autre part, nous pouvons inférer que notre modèle est potentiellement mieux adapté pour récupérer tous les résultats que pour récupérer les meilleurs résultats : en effet, pour trouver les meilleurs résultats, nous n'avons *a priori* pas besoin de liens. Nous rappelons tout de même que, pour les $iP@0.00R$ et $iP@0.01$ la configuration $\beta = 1$ donne les meilleurs résultats pour que la configuration $\beta = 0$ pour des requêtes de type *best* et la configuration $\beta = 0$ donne les meilleurs résultats pour que la configuration $\beta = 1$ pour des requêtes de type *all*.

TAB. 8.8 – Résultats de la prise en compte du type de requêtes (type FOC_2).

$Run(\nu, \alpha, \beta)$	*mode*	iP@0.00R	iP@0.01R	iP@0.05R	iP@0.10R	MAiP
$(4_{OUT}, 0.5, 0)$	*all*	**0.3493**	**0.3416**	0.3030	0.2432	0.1103
$(4_{IN}, 0.5, 0)$	*all*	0.3398	0.3326	**0.3045**	0.2763	**0.1126**
$(4_{IN\&OUT}, 0.5, 0)$	*all*	0.3398	0.3326	0.3043	**0.2779**	0.1095
$(4_{OUT}, 0.5, 1)$	*all*	0.3391	0.3314	0.3010	0.2465	0.1094
$(4_{IN}, 0.5, 1)$	*all*	0.3386	0.3313	0.3010	0.2709	0.1039
$(4_{IN\&OUT}, 0.5, 1)$	*all*	0.3393	0.3322	0.3013	0.2542	0.1034
$(4_{OUT}, 0.5, 0)$	*best*	0.2487	0.2363	**0.2181**	**0.2133**	**0.0739**
$(4_{IN}, 0.5, 0)$	*best*	0.2451	0.2324	0.2088	0.2030	0.0720
$(4_{IN\&OUT}, 0.5, 0)$	*best*	0.2356	0.2229	0.2021	0.1972	0.0724
$(4_{OUT}, 0.5, 1)$	*best*	**0.2516**	**0.2382**	0.2137	0.2087	0.0689
$(4_{IN}, 0.5, 1)$	*best*	0.2413	0.2141	0.2008	0.1932	0.0616
$(4_{IN\&OUT}, 0.5, 1)$	*best*	0.2392	0.2269	0.2023	0.1938	0.0668

Les résultats sont discutables dans le sens où nous avons basé nos expérimentations sur les choix de typage (*all* ou *best*) effectués par les concepteurs de requêtes INEX 2007 et qu'il semble exister quelques incohérence : par exemple, la requête 414 en description narrative énonce :"I want to learn **all** there is to know about hip hop beats" alors que l'auteur de cette requête l'a classée de type *best*.

8.5 Évaluation des documents virtuels résultats

La campagne d'évaluation INEX ne permet pas d'évaluer les documents virtuels résultats de notre système. Dans ce but, nous avons choisi d'examiner les résultats que nous obtenons en simulant une navigation de la part d'un utilisateur dans l'espace des doxels résultats produits par le système. Nous émettons l'hypothèse qu'à chaque fois qu'un utilisateur a la possibilité de suivre un lien, il le fait. La figure 8.7 illustre cette notion : à partir d'une liste de résultats permettant de naviguer vers un certain nombre de voisins (1 seul dans le cas de la figure 8.7) pertinents pour la requête en cours, nous créons une nouvelle liste de résultats qui suit l'ordre des doxels visualisés en tenant compte de la navigation vers les meilleurs voisins. Ainsi, sur l'exemple, l'utilisateur consulte d'abord le doxel 1, puis il a la possibilité de visualiser le doxel 11, nous supposons qu'il le visualise, puis l'utilisateur parcourt le doxel 2, puis le 21, et ainsi de suite. Cette simulation de liste de résultats nous permet de réutiliser les mesures d'évaluation d'INEX.

Nous évaluons la navigation à partir de deux listes de résultats différentes :
- le cas de référence correspond à l'utilisation de la liste de résultats produite par un système de recherche d'information qui ne tient pas compte du voisinage dans sa fonction de correspondance ;
- le second cas correspond aux résultats produits par le système de recherche d'information que nous proposons et qui tient compte des voisins dans sa fonction de correspondance.

Ainsi, nous proposons de tenir compte non seulement des doxels qui répondent le mieux, mais également des liens vers l'environnement jugé pertinent de ces doxels. Notre évaluation consiste à simuler le fait que l'utilisateur a suivi les n meilleurs liens proposés avant de revenir à la liste

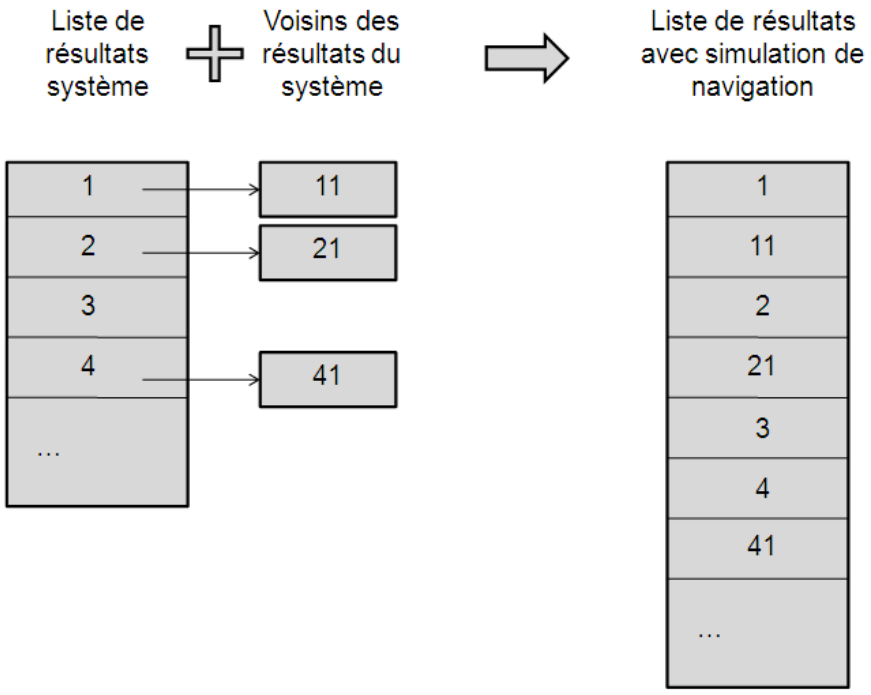

FIG. 8.7 – Simulation de navigation.

des résultats et refaire la même démarche avec les documents suivants de la liste. Ce parcours crée en fait une liste de résultats qui peut être évaluée par les mesures proposées par INEX 2007.

Les résultats que nous obtenons sont présentés dans la table 8.9 :

- d'un côté, nous simulons une navigation parmi les meilleurs liens à partir d'un résultat produit par le système sans utiliser les relations de voisinage.
- Les lignes étiquetées *n voisins* simulent une navigation parmi les meilleurs liens résultats, sur la base de résultats produits en tenant compte du voisinage dans la fonction de correspondance.

Les résultats statistiquement significatifs par rapport au t-test sont soulignés, les tests ayant été effectués sur chaque sous tableau rassemblant un même nombre de liens suivis. Les meilleurs scores du tableau sont suivis d'un astérisque et sont comparés avec les meilleurs scores de chaque sous-tableau pour le test statistique.

Les résultats obtenus montrent que, dans une configuration où un utilisateur naviguerait dans les doxels qui lui sont présentés par le système de recherche d'information, la navigation en suivant les liens proposés améliore les résultats en terme de précision entre 0.00 et 0.10, ce qui est le but visé. Cependant, les améliorations ne sont pas statistiquement significatives sur les premières valeurs de rappel alors qu'elles le sont en $MAiP$; il serait sans doute intéressant d'expérimenter cette pratique à partir d'un modèle prenant en considération le voisinage d'une manière différente.

TAB. 8.9 – Simulation de navigation.

$Run(\nu, \alpha, \beta)$	*nb liens suivis*	iP@0.00R	iP@0.01R	iP@0.05R	iP@0.10R	MAiP
$(0, x, x)_{FOC_2}$	1	0.2794	0.2444	0.2038	0.1682	0.0603
$(1, 0.5, 0.5)_{FOC_1}$	1	**0.3028***	0.2153	0.1616	0.1249	0.0500
$(1, 0.5, 0.5)_{FOC_2}$	1	0.2815	**0.2644***	**0.2120***	**0.1856***	**0.0749***
$(0, x, x)_{FOC_2}$	3	0.2757	0.2223	0.1759	0.1419	0.0519
$(3, 0.5, 0)_{FOC_1}$	3	0.2756	0.1911	0.1625	0.1435	0.0580
$(3, 0.5, 0)_{FOC_2}$	3	**0.2793**	**0.2343**	**0.1780**	**0.1549**	**0.0655**

D'autre part, les résultats obtenus soulignent le fait que la navigation vers un unique voisin (en suivant 1 lien) donne de meilleures performances que la navigation vers 3 voisins, même si la différence entre les deux configurations n'est pas statistiquement significative ; la navigation améliore globalement les résultats de toute façon.

Cette simulation de navigation constitue un autre mode d'exploration de l'espace des résultats. Nous constatons que l'utilisation du modèle que nous proposons permet de renvoyer à l'utilisateur des doxels qui possèdent effectivement des voisins pertinents, davantage qu'un système classique.

8.6 Discussion générale

Dans ce chapitre, nous avons d'abord validé le système Gycori implémentant notre modèle en le comparant sur des documents complets au système Zettair. Notre système donne des résultats intermédiaires entre ceux correspondant à l'instance modèle vectoriel et ceux correspondant à l'instance modèle de langue de Zettair ; ainsi, nous estimons que notre système peut servir de référence.

En second lieu, nous avons évalué l'impact de l'environnement sur les résultats de recherche. Nous avons successivement étudié les liens sortants, les liens entrants puis une combinaison des deux, avec différentes configurations de prise en compte de cet environnement ; le nombre de liens (paramètre ν), la dimension à accorder à l'environnement versus aux documents complets (paramètre α), et le choix des meilleurs liens favorisant plutôt l'exhaustivité ou plutôt la spécificité (paramètre β) étaient les facteurs de notre étude. Nous avons ensuite choisi de donner encore plus d'importance aux documents complets en fusionnant nos résultats avec ceux de Zettair basé sur le modèle de langue. Et nous avons étudié la relation entre le type de réponse attendue pour les requêtes et les paramètres d'exhaustivité et de spécificité.

Nous concluons que le voisinage est bénéfique pour la recherche d'information sur des documents structurés, que ce soit en utilisant des liens sortants, des liens entrants ou les deux à la fois. Il se dégage des expérimentations que les liens sortants sont meilleurs pour l'instanciation proposée, sur la collection test utilisée. Selon les expérimentations, le nombre de voisins idéal à prendre en considération est plus difficile à déterminer ; pour les premiers points de rappel, le fait de considérer le voisin le plus pertinent (un seul voisin) semble préférable, alors que la prise en compte de l'ensemble des voisins apporte de meilleurs résultats en $MAiP$. Les performances en prenant en compte 4 voisins sortants sont également bonnes sur l'ensemble des mesures, pour les liens sortants. Le fait de considérer uniquement 4 liens plutôt que l'ensemble des liens permet

de limiter les calculs, comparativement à un cas où nous choisirions de considérer tous les liens. Le fait de pondérer différemment les valeurs d'exhaustivité et de spécificité par le paramètre β donne un avantage marginal *a priori*, les résultats ne sont pas statistiquement significatifs, que ce soit dans le cas où le modèle est adapté aux requêtes ou dans les premières expériences où nous faisions varier les paramètres. De plus, nous avons constaté que le fait d'utiliser les documents complets améliore considérablement les résultats, c'est le cas dans nos expérimentations où les résultats initiaux sont fusionnés avec les résultats de Zettair sur le modèle de langue.

Finalement, nous nous sommes intéressés aux aspects documents virtuels que nous générons, afin de faciliter l'accès aux éléments pertinents aux utilisateurs en simulant une navigation dans l'espace des résultats.

Tab. 8.10 – Comparaison de nos résultats avec INEX 2007.

Scores	Officiel	Rang	Nouveau	Rang	Meilleur	Moins bon	Moyenne	Médiane
MAiP	0.0593	59	0.1096	**29**	0.1804	0.0037	0.0893	0.0851
iP@0.00R	0.2847	66	0.4169	**18**	0.4780	0.0276	0.3563	0.3722
iP@0.01R	0.2554	63	0.3794	**15**	0.4259	0.0171	0.3112	0.3312
iP@0.05R	0.2126	57	0.3140	**21**	0.3482	0.0081	0.2465	0.2620
iP@0.10R	0.1706	58	0.2846	**20**	0.3238	0.0007	0.2104	0.2207

Le tableau 8.10 présente l'ensemble de nos meilleurs résultats comparés aux systèmes de la campagne INEX 2007. Dans les premières colonnes, nous rappelons les meilleurs résultats que nous avons obtenus pour nos soumissions officielles, avec les classements obtenus sur 79 soumissions de l'ensemble des participants ; puis nous ajoutons les résultats que nous avons obtenus depuis, avec le rang correspondant parmi les systèmes officiels. Finalement, nous rappelons les scores des meilleures soumissions, moins bonnes soumissions, et la moyenne et la médiane des soumissions pour chaque mesure, pour comparaison.

Le système a été corrigé depuis la soumission des résultats officiels et les performances actuelles (amélioration de plus de 84% en $MAiP$) nous permettraient d'espérer un classement dans les 30 premiers systèmes en $MAiP$, voire dans le premier quart des systèmes pour des petites valeurs de rappel.

Conclusion générale

Synthèse et contribution

État de l'art Nous avons présenté en partie I un état de l'art sur les documents structurés et sur les documents virtuels respectivement aux chapitres 1 et 2.

L'état de l'art sur les documents structurés propose trois axes d'étude des travaux existants qui s'appliquent à décrire du contenu et de la structure des documents structurés, pour une réutilisation de ces informations au moment de l'interrogation. Alors que les approches de la classe C_{sur_doxel} n'utilisent aucune dépendance entre les éléments de structure pour la recherche d'information, les approches des classes $C_{sur\_contenu}$ et $C_{sur\_structure}$ prennent en compte respectivement des dépendances basées sur le contenu des doxels et des dépendances entre doxels basées sur la structure. Nous avons constaté qu'à la fois le contenu et la structure des documents présentent un intérêt pour la recherche d'information sur des documents structurés.

L'état de l'art sur les documents virtuels distingue les approches existantes selon le degré decontraintes d'organisation qu'elles imposent pour la production de document(s) virtuel(s) résultat(s) d'une recherche d'information. Des approches s'appliquant à des domaines très spécifiques proposent globalement des modèles d'organisation fortement contraints ($\mathcal{M}_{contraint}^{fortement}$) tandis que les approches plus générales s'appuient sur des modèles d'organisation faiblement contraints ($\mathcal{M}_{contraint}^{faiblement}$) voire nullement contraints ($\mathcal{M}_{contraint}^{nullement}$). La création de documents virtuels dans des contextes généraux est encore un domaine à explorer et pour notre proposition, nous devons trouver une approche intermédiaire pour obtenir des résultats satisfaisants sans trop de contraintes.

Modèle de recherche d'information pour des documents virtuels En partie II, nous avons proposé un modèle abstrait de recherche d'information qui se base sur des documents structurés. Ce modèle répond au problème de la génération de documents virtuels en proposant une solution intermédiaire, qui favorise la navigation dans l'espace des résultats. Lorsqu'un utilisateur consulte un élément e qui satisfait son besoin d'information, si e est lié à d'autres doxels pertinents $\{d_{pert}\}$ qui font partie de son environnement - qui n'apparaîtraient pas forcément à proximité de e qu'il consulte - il lui est possible d'accéder directement à l'ensemble de doxels $\{d_{pert}\}$. De plus, notre proposition regroupe les doxels proposés en réponses par document en adéquation avec l'étude [HLF01] qui stipule que les utilisateurs préfèrent avoir une vue globale des documents auxquels les doxels qu'ils consultent appartiennent.

La définition de ce modèle est générale et le modèle peut s'appliquer à différents formats de documents, dans différents domaines et à différents médias à condition que ces derniers permettent l'utilisation de structure et de relations entre éléments de structure. Nous avons montré que ce modèle peut s'appliquer à des documents XML textuels dans le chapitre 6.

Le modèle proposé comporte un certain nombre de paramètres qui doivent être ajustés en fonction des collections sur lesquelles porte l'interrogation ou bien en fonction des besoins de l'utilisateur :

- α qui permet de donner plus ou moins d'importance au contenu des doxels face à leur environnement ;
- β qui permet d'accorder un poids plus fort à l'exhaustivité globale d'un doxel ou à sa spécificité ;
- le nombre maximum de doxels à considérer dans l'environnement d'un doxel donné, ν.

Validation du modèle Nous avons effectué des expérimentations sur la collection INEX 2007 au chapitre 8, afin de valider le modèle. À l'issue de ces expérimentations, nous avons montré que :

1. notre système, implémentant le modèle proposé, produit des résultats sur des documents complets meilleurs que ceux renvoyés par un modèle vectoriel de référence, validant ainsi notre approche sur les documents atomiques ;
2. notre modèle permet d'obtenir de meilleurs résultats quand il prend en compte l'environnement d'occurrence des doxels que sans cet environnement : exploiter l'environnement des doxels en appliquant les mesures d'exhaustivité et de spécificité que nous avons employées améliore la recherche d'information sur des documents structurés (+24% en précision moyenne) ; les résultats sont encore meilleurs quand on les combine avec ceux renvoyés par un système implantant un modèle de langue sur des documents complets (+41% en précision moyenne) ;
3. de plus, les résultats obtenus en simulant un comportement de navigation systématique dans les documents virtuels générés permet de mieux répondre à l'utilisateur si les résultats sont ordonnés sur les scores donnés par le modèle avec environnement que sans environnement (+24%).

Les résultats que nous avons obtenus nous auraient situés dans le premier quart des 80 soumissions officielles de juillet 2007.

Perspectives

Le modèle proposé dans ce manuscrit a été instancié et nous avons montré qu'il apportait des améliorations pour la recherche d'information pour des documents virtuels : les résultats nous encouragent à poursuivre dans cette voie. Ce travail ouvre alors de nombreuses perspectives pour favoriser la navigation dans l'espace des résultats répondant à un besoin d'information.

À court terme, nous souhaitons proposer des améliorations suivant deux directions : au niveau de l'instanciation du modèle et au niveau expérimental.

Par rapport au modèle, nous souhaitons étudier plus finement les liens utilisables pour notre approche : en effet, nous avons considéré tous les liens `collectionlinks` préexistants dans la collection INEX 2007, par facilité pour ces premières expérimentations. Cependant, nous pouvons nous interroger sur les critères qui font qu'un lien peut être considéré comme facilitant *in fine* la navigation dans l'espace résultat. Ces liens doivent-ils être le résultat d'une observation du comportement d'utilisateurs ? Nous envisageons de détecter, éventuellement après avoir catégorisé les documents de la collection pour éviter l'explosion combinatoire, des liens de proximité sémantique, des liens de co-citation, des liens de chemin de lecture... Selon le type de liens choisis, un travail ultérieur devrait être mené pour déterminer si les mesures d'exhaustivité et de spécificité que nous avons proposées sont adaptées.

Notre modèle définit une configuration des paramètres α, β et ν indépendamment de toute requête. On peut se poser la question de savoir si ce choix est valide pour une collection donnée. En effet, il est raisonnable de penser que suivant la requête, il est préférable de favoriser l'exhaustivité ou la spécificité. Lors des expérimentations réalisées, nous avons déjà effleuré cette problématique en distinguant deux classes de requêtes (*all* et *best* en section 8.4.3) extraites d'INEX 2007, mais ce travail doit être approfondi.

Nous pouvons même aller plus loin en nous demandant si les termes d'une requête ne pourraient pas être séparés en plusieurs classes, qui seraient traitées par des configurations de paramètres α, β et ν fixés, lors du calcul de la correspondance. Nous aimerions favoriser l'utilisation du contenu des documents pour des termes de requête discriminants et favoriser l'utilisation de l'environnement pour des termes de requête plus vagues.

Au niveau des expérimentations, de nouvelles évaluations pourront être mises en place pour mesurer la validité des nouvelles hypothèses effectuées. D'autre part, si nous considérons l'instanciation proposée dans cette thèse, de nouvelles valeurs de paramètres seront testées afin de régler les paramètres du modèle plus finement. Nos travaux se poursuivent actuellement par la soumission de résultats à la campagne d'évaluation INEX 2008, les paramètres adoptés correspondant à ceux qui renvoyaient les meilleurs résultats pour INEX 2007 comme la collection est restée stable.

À plus long terme, nous souhaiterions explorer des aspects mobilité, la dimension multimédia et des aspects personnalisation.

Nous avons cherché à favoriser la navigation des utilisateurs dans l'espace des résultats ; nous pensons que le modèle pourrait évoluer pour être adapté à une utilisation en mobilité en diminuant l'importance de la liste et en mettant l'accent sur la navigation dans l'espace des résultats. Pour cette navigation, les liens mis en exergue pourraient dépendre, entre autres, de la taille des éléments cibles des liens, de façon à privilégier les éléments adaptés au dispositif d'affichage utilisé.

Ce travail s'applique à des documents structurés textuels ; nous aimerions l'élargir aux documents multimédia en général. Nous pouvons utiliser des liens entre les textes et les images comme dans [HSD97] afin de retrouver des images, des textes ou des documents structurés combinant les deux. Notre modèle s'adapterait à ce type d'approche, tout en sachant que le niveau de description des textes et le niveau de description des images sont très différents. L'expression des valeurs d'exhaustivité et de spécificité devront être redéfinies pour prendre en compte ces différents média.

Les systèmes informatiques intègrent de plus en plus des éléments de personnalisation. Nous voudrions étudier quel impact la personnalisation pourrait avoir sur le modèle qui a été proposé. Nous pourrions prendre en compte les relations qui sont pertinentes pour un utilisateur donné dans ses tâches de recherche courantes. Si l'utilisateur favorise des liens caractérisés plutôt exhaustif ou plutôt spécifique, alors le modèle pourrait mémoriser ces informations pour en tenir compte dans les propositions des résultats suivantes.

Le travail réalisé et ses extensions futures s'inscrivent par conséquent bien dans la tendance actuelle orientée vers des systèmes capables de s'adapter à la fois aux utilisateurs, aux matériels et aux données.

Liste des publications

Conférence internationale avec comité de lecture

Delphine Verbyst and Philippe Mulhem, Doxels in context for retrieval : from structure to neighbours, in ACM SAC 2008 - IAR Track, Fortaleza, Brésil, pp1122-1126, March 16-20, 2008.

Delphine Verbyst and Philippe Mulhem, LIG at INEX2007 Ad Hoc Track : Using Collectionlinks as Context, in Workshop INEX 2007, Schloss Dagstuhl International Conference and Research Center for Computer Science, Deutschland, pp94-104, December 17-19, 2007.

Conférence nationale avec comité de lecture

Philippe Mulhem and Delphine Verbyst, Recherche de documents structurés en mobilité : un modèle et une mesure d'évaluation, in CORIA 2008, Trégastel, France, pp211-223, March 12-14, 2008.

Delphine Verbyst, Indexation relationnelle pour la recherche de documents structurés interreliés, in RJCRI'07, Saint-Etienne, France, pp485-490, 28-30 mars, 2007.

Workshop international sans comité de lecture

Philippe Mulhem and Delphine Verbyst, LIG at Mosaic - Contextual IR and STOIC 101 results, in Workshop on Mobile Search and Annotation using Images in Context, Second Mosaic Meeting), Taipei, Taiwan, 15-16 Novembre, 2007.

Rapports

Delphine Verbyst and Philippe Mulhem, Rapport final de contrat, rapport de recherche MRIM, Groupe MRIM - LIG, Mai, 2008.

Delphine Verbyst and Philippe Mulhem, WP9 : Développement, Expérimentation, Intégration sur la RI en contexte, rapport de recherche MRIM, Groupe MRIM - LIG, Mai, 2008.

Delphine Verbyst and Philippe Mulhem, WP8 : Modélisation - Intégration documents virtuels et contexte pour la recherche d'information, rapport de recherche MRIM, Groupe MRIM -

LIG, Décembre, 2007.

Delphine Verbyst and Philippe Mulhem, WP7 : Développement et Expérimentation sur la RI en contexte, rapport de recherche MRIM, Groupe MRIM - LIG, Août, 2007.

Delphine Verbyst and Philippe Mulhem, WP5-6 : Développements et résultats d'expérimentations sur documents virtuels pour la RI hors contexte, rapport de recherche MRIM, Groupe MRIM - LIG, Mai, 2007.

Delphine Verbyst and Philippe Mulhem, WP4 : Modélisation du contexte pour la recherche d'information, rapport de recherche MRIM, Groupe MRIM - CLIPS-IMAG, Août, 2006.

Delphine Verbyst and Philippe Mulhem, WP3 : Modélisation de documents virtuels pour la recherche d'information, rapport de recherche MRIM, Groupe MRIM - CLIPS-IMAG, Août, 2006.

Delphine Verbyst and Philippe Mulhem, WP2 : Etat de l'art sur la recherche d'information en contexte, rapport de recherche MRIM, Groupe MRIM - CLIPS-IMAG, Mai, 2006.

Delphine Verbyst and Philippe Mulhem, WP1 : Etat de l'art sur les documents virtuels pour la recherche d'information, rapport de recherche MRIM, Groupe MRIM - CLIPS-IMAG, Février, 2006.

Bibliographie

[Bat86] Marcia J. Bates. Terminological assistance for the online subject searcher. In Proceedings of the Second Conference on Computer Interfaces and Intermediaries for Information Retrieval, 1986.

[Bei06] Michel Beigbeder. Structured content-only information retrieval using term proximity and propagation of title terms. In INEX 2006 Workshop Proceeding, pages 200–212, 2006.

[Bei07] Michel Beigbeder. Ensm-se at inex 2007 : Scoring with proximity. In Norbert Fuhr, Mounia Lalmas, and Andrew Trotman, editors, INitiative for the Evaluation of XML Retrieval (INEX). Pre-Proceedings of INEX 2007, Dagstuhl, Germany, December 2007.

[BP98] Sergey Brin and Lawrence Page. The anatomy of a large-scale hypertextual Web search engine. In Proceedings of the 7th International World Wide Web Conference, pages 107–117, Brisbane, Australia, 1998.

[BVNN+02] A. Boukottaya, C. Vanoirbeek, A. V. Nguyen Ngoc, Y. Rekik, and K. Zeramdini. A contract-based model for creating structured virtual documents : key issues for reusability and collaboration. In Documents Virtuels Personnalisables 2002, 2002.

[Cal94] James P. Callan. Passage-level evidence in document retrieval. In SIGIR '94 : Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, pages 302–310, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[CC01] Jamie Callan and Margaret Connell. Query-based sampling of text databases. ACM Trans. Inf. Syst., 19(2) :97–130, 2001.

[CC07] Noureddine Chatti and Sylvie Calabretto. Adaptation de XML et XQuery pour la représentation et l'interrogation des documents multi-structurés. In Quatrieme COnference en Recherche d'Information et Applications (CORIA'07), pages 109–124, Saint-Etienne, France, March 2007. Publications de l'Université de Saint-Etienne.

[CCH92] James P. Callan, W. Bruce Croft, and Stephen M. Harding. The INQUERY retrieval system. In Proceedings of DEXA-92, 3rd International Conference on Database and Expert Systems Applications, pages 78–83, 1992.

[CCPT00] Gordon V. Cormack, Charles L. A. Clarke, Christopher R. Palmer, and Samuel S. L. To. Passage-based query refinement (multitext experiments for trec-6). Information Processing and Management, 36(1) :133–153, 2000.

[CMF96] Yves Chiaramella, Philippe Mulhem, and Frank Fourel. A model for multimedia information retrieval. Technical report, FERMI ESPRIT BRA 8134, University of Glasgow, 1996.

[CR00] Sylvie Chabert-Ranwez. Composition automatique de documents hypermédia adaptatifs à partir d'ontologies et de requêtes intentionnelles de l'utilisateur. PhD thesis, Université Montpellier II - France, 2000.

[Cra97] Michel Crampes. Auto-adaptive illustration through conceptual evocation. In DL '97 : Proceedings of the second ACM international conference on Digital libraries, pages 247–254, New York, NY, USA, 1997. ACM.

[Cro06] Carolyn J. Crouch. Dynamic element retrieval in a structured environment. ACM Trans. Inf. Syst., 24(4) :437–454, 2006.

[CT89] W. B. Croft and H. Turtle. A retrieval model incorporating hypertext links. In HYPERTEXT '89 : Proceedings of the second annual ACM conference on Hypertext, pages 213–224, New York, NY, USA, 1989. ACM.

[CWC03] H. Cui, J. Wen, and T. Chua. Hierarchical indexing and flexible element retrieval for structured documents. In 25th European Conference on Information Retrieval Research (ECIR'03), pages 73–87. Springer-Verlag Berlin Heidelberg, 2003.

[DB99] Bich-Liên Doan and Michel Beigbeder. Virtual WWW Documents : a concept to explicit the structure of WWW sites. In BCS-IRSG Annual Colloquium on IR Research, 1999.

[DG06] Ludovic Denoyer and Patrick Gallinari. The Wikipedia XML Corpus. SIGIR Forum, 2006.

[DGM+98] Robert Dale, Stephen J. Green, Maria Milosavljevic, Cécile Paris, Cornelia Verspoor, and Sandra Williams. Using natural language to produce virtual documents. In Proceedings of the 3rd Australasian Document Computing Symposium, Sydney, Australia, 1998.

[DNFY05] Fernando Das-Neves, Edward A. Fox, and Xiaoyan Yu. Connecting topics in document collections with stepping stones and pathways. In CIKM '05 : Proceedings of the 14th ACM international conference on Information and knowledge management, pages 91–98, New York, NY, USA, 2005. ACM Press.

[DOMK98] Robert Dale, Jon Oberlander, Maria Milosavljevic, and Alistair Knott. Integrating natural language generation and hypertext to produce dynamic documents. Interacting with Computers, 11(2) :109–135, 1998.

[DvR93] M. D. Dunlop and C. J. van Rijsbergen. Hypermedia and free text retrieval. Information Processing and Management, 29(3) :287–298, 1993.

[Fox83] Edward Alan Fox. Extending the boolean and vector space models of information retrieval with p-norm queries and multiple concept types. PhD thesis, Ithaca, NY, USA, 1983.

[FS95] H. P. Frei and D. Stieger. The use of semantic links in hypertext information retrieval. Information Processing and Management, 31(1) :1–13, 1995.

[Gér06] Mathias Géry. Indexing reading paths for a structured information retrieval at INEX 2006. In INEX 2006 Workshop Proceeding, pages 160–164, 2006.

[GVR97] Thomas R. Gruber, Sunil Vemuri, and James Rice. Model-based virtual document generation. Int. J. Hum.-Comput. Stud., 46(6) :687–706, 1997.

[Har93] Donna Harman. Overview of the first trec conference. In SIGIR '93 : Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval, pages 36–47, New York, NY, USA, 1993. ACM.

[Har95] Donna Harman. Overview of the second text retrieval conference (trec-2). Information Processing and Management, 31(3) :271–289, 1995.

[Hen05] Monika Henzinger. Hyperlink analysis on the world wide web. In HYPERTEXT '05 : Proceedings of the sixteenth ACM conference on Hypertext and hypermedia, pages 1–3, New York, NY, USA, 2005. ACM.

[HLF01] Morten Hertzum, Mounia Lalmas, and Erik Frokjaer. How are searching and reading intertwined during retrieval from hierarchically structured documents ? In INTERACT 2001, Japan, July 2001.

[Hor85] Wolfgang Horak. Office document architecture and office document interchange formats : Current status of international standardization. Computer, 18(10) :50–60, 1985.

[HSD97] V. Harmandas, M. Sanderson, and M. D. Dunlop. Image retrieval by hypertext links. In SIGIR '97 : Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval, pages 296–303, New York, NY, USA, 1997. ACM.

[Hua07] Fang Huang. The Role of Shallow Features in XML Retrieval. In Norbert Fuhr, Mounia Lalmas, and Andrew Trotman, editors, INitiative for the Evaluation of XML Retrieval (INEX). Pre-Proceedings of INEX 2007, Dagstuhl, Germany, December 2007.

[Hul93] David Hull. Using statistical testing in the evaluation of retrieval experiments. In SIGIR '93 : Proceedings of the 16th annual international ACM SIGIR conference on Research and development in information retrieval, pages 329–338, New York, NY, USA, 1993. ACM Press.

[iads03] What is a discourse schema ? LinguaLinks Library, version 5.0 - published on CD-ROM by SIL International - www.sil.org/linguistics/glossaryoflinguisticterms/whatisadiscourseschema.htm, 2003.

[IG02] S. Iksal and S. Garlatti. Spécification déclarative pour des documents virtuels personnalisables. In Documents Virtuels Personnalisables 2002, 2002.

[INE05] INEX. INitiative for the Evaluation of XML retrieval 2005, http ://inex.is.informatik.uni-duisburg.de/2005/, 2005.

[ISO86] ISO 8879. Information processing – text and office systems – standard generalized markup language (sgml). 1986.

[KC95] Ammar Kheirbek and Yves Chiaramella. Integrating hypermedia and information retrieval with conceptual graphs formalism. In HIM, pages 47–60, 1995.

[KdRS04] Jaap Kamps, Maarten de Rijke, and Börkur Sigurbjörnsson. Length normalization in xml retrieval. In SIGIR '04 : Proceedings of the 27th annual international ACM SIGIR conference on Research and development in information retrieval, pages 80–87, New York, NY, USA, 2004. ACM.

[KJDM01] Koichi Kise, Markus Junker, Andreas Dengel, and Keinosuke Matsumoto. Experimental evaluation of passage-based document retrieval. In International Conference on Document Analysis and Recognition, pages 592–596, 2001.

[KL05] Oren Kurland and Lillian Lee. Pagerank without hyperlinks : structural re-ranking using links induced by language models. In SIGIR '05 : Proceedings of the 28th

annual international ACM SIGIR conference on Research and development in information retrieval, pages 306–313, New York, NY, USA, 2005. ACM.

[Kle99] Jon M. Kleinberg. Authoritative sources in a hyperlinked environment. Journal of the ACM, 46(5) :604–632, 1999.

[KZ97] Marcin Kaszkiel and Justin Zobel. Passage retrieval revisited. In SIGIR '97 : Proceedings of the 20th annual international ACM SIGIR conference on Research and development in information retrieval, pages 178–185, New York, NY, USA, 1997. ACM.

[LC01] Victor Lavrenko and W. Bruce Croft. Relevance-based language models. In Proceedings of the 24th annual international ACM SIGIR conference on Research and development in information retrieval, pages 120–127, 2001.

[LC02] X. Liu and W. Croft. Passage retrieval based on language models. In Proceedings of the eleventh international Conference on Information and Knowledge Management, McLean, USA, pages 375–382. ACM, 2002.

[LRM06] Wei Lu, Stephen E. Robertson, and Andrew MacFarlane. Cisr at inex 2006. In INitiative for the Evaluation of XML Retrieval (INEX). Proceedings of INEX 2006, pages 57–63, 2006.

[LV04] Mounia Lalmas and Patrick Vannoorenberghe. Indexation et recherche de documents XML par les fonctions de croyance. In Premiere COnference en Recherche d'Information et Applications (CORIA'04), Toulouse, France, March 2004. Hermès.

[MA96] Philippe Martin and Laurence Alpay. Conceptual structures and structured documents. In International Conference on Conceptual Structures, pages 145–159, 1996.

[Mel98] Massimo Melucci. Passage retrieval : a probabilistic technique. Information Processing and Management, 34(1) :43–68, 1998.

[MJKZ98] Sung-Hyon Myaeng, Don-Hyun Jang, Mun-Seok Kim, and Zong-Cheol Zhoo. A flexible model for retrieval of sgml documents. In SIGIR '98 : Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28 1998, Melbourne, Australia, pages 138–145. ACM, 1998.

[MS94] Elke Mittendorf and Peter Schäuble. Document and passage retrieval based on hidden markov models. In SIGIR '94 : Proceedings of the 17th annual international ACM SIGIR conference on Research and development in information retrieval, pages 318–327, New York, NY, USA, 1994. Springer-Verlag New York, Inc.

[MSB97] M. Mitra, A. Singhal, and C. Buckley. Automatic text summarization by paragraph extraction. In Mani and Maybury (Eds.), Advances in automatic text summarization. MIT Press, pages 31–36, 1997.

[MSDWZ93] Alistair Moffat, Ron Sacks-Davis, Ross Wilkinson, and Justin Zobel. Retrieval of partial documents. In Text REtrieval Conference, pages 181–190, 1993.

[NF07] M. Lalmas S. Malik A. Trotman N. Fuhr, J. Kamps. Overview of the inex 2007 ad hoc track. In Norbert Fuhr, Mounia Lalmas, and Andrew Trotman, editors, INitiative for the Evaluation of XML Retrieval (INEX). Pre-Proceedings of INEX 2007, Dagstuhl, Germany, December 2007.

[PBMW98a] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The pagerank citation ranking : Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[PBMW98b] Lawrence Page, Sergey Brin, Rajeev Motwani, and Terry Winograd. The PageRank Citation Ranking : Bringing order to the web. Technical report, Stanford Digital Library Technologies Project, 1998.

[PC98] Jay M. Ponte and W. Bruce Croft. A language modeling approach to information retrieval. In Research and Development in Information Retrieval, pages 275–281, 1998.

[PG04] Benjamin Piwowarski and Patrick Gallinari. A Bayesian Network for XML Information Retrieval : Searching and learning with the INEX collection. Information Retrieval, December 2004.

[Piw03] Benjamin Piwowarski. Techniques d'apprentissage pour le traitement d'informations structurées : application à la recherche d'information. PhD thesis, LIP6 - Paris - France, 2003.

[PL04] Benjamin Piwowarski and Mounia Lalmas. Interface pour l'évaluation de systèmes de recherche sur des documents XML. In Premiere COnference en Recherche d'Information et Applications (CORIA'04), pages 109–120, Toulouse, France, March 2004. Hermès.

[PVH98] François Paradis, Anne-Marie Vercoustre, and Brendan Hills. A virtual document interpreter for reuse of information. In EP '98/RIDT '98 : Proceedings of the 7th International Conference on Electronic Publishing, Held Jointly with the 4th International Conference on Raster Imaging and Digital Typography, pages 487–498, London, UK, 1998. Springer-Verlag.

[RJ88] Stephen E. Robertson and Karen Sparck Jones. Relevance weighting of search terms. In Document retrieval systems, pages 143–160, London, UK, UK, 1988. Taylor Graham Publishing.

[RLK+02] Thomas Roelleke, Mounia Lalmas, Gabriella Kazai, Ian Ruthven, and Stefan Quicker. The accessibility dimension for structured document retrieval. In Proceedings of the 24th BCS-IRSG European Colloquium on IR Research, pages 284–302, London, UK, 2002. Springer-Verlag.

[Rob02] Stephen Robertson. Threshold setting and performance optimization in adaptive filtering. Information Retrieval, 5(2-3) :239–256, 2002.

[Roi99] Cécile Roisin. Documents structurés multimédia. In Mémoire en vue de l'obtention d'une habilitation à diriger les recherches, septembre 1999.

[SA06] Mark D. Smucker and James Allan. Find-similar : similarity browsing as a search tool. In SIGIR '06 : Proceedings of the 29th annual international ACM SIGIR conference on Research and development in information retrieval, pages 461–468, New York, NY, USA, 2006. ACM Press.

[SA07] Mark D. Smucker and James Allan. Using similarity links as shortcuts to relevant web pages. In SIGIR '07 : Proceedings of the 30th annual international ACM SIGIR conference on Research and development in information retrieval, pages 863–864, New York, NY, USA, 2007. ACM Press.

[Sal68] Gerard. Salton. Automatic Information Organization and Retrieval. McGraw Hill Text, 1968.

[Sau04] Karen Sauvagnat. XFIRM : Un modèle flexible de recherche d'information pour le stockage et l'interrogation de documents xml. In Premiere COnference en Recherche d'Information et Applications (CORIA'04), pages 121–142, Toulouse, France, March 2004. Hermès.

[Sav96] Jacques Savoy. An extended vector-processing scheme for searching information in hypertext systems. Information Processing and Management, 32(2) :155–170, 1996.

[SB88] Gerard Salton and Christopher Buckley. Term-weighting approaches in automatic text retrieval. Information Processing and Management, 24(5) :513–523, 1988.

[SB06] Karen Sauvagnat and Mohand Boughanem. Propositions pour la pondération des termes et l'évaluation de la pertinence des éléments en recherche d'information structurée. In Troisième COnference en Recherche d'Information et Applications (CORIA'06), pages 29–40, Lyon, France, March 2006.

[SJC93] Gérard Salton, Allan J., and Buckley C. Approaches to Passage Retrieval in Full Text Information Systems. In SIGIR '93 : Proceedings of the 16th Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, New York, NY, USA, 1993. ACM Press.

[SM86] Gerard Salton and Michael J. McGill. Introduction to Modern Information Retrieval - Chapter 6, page 203. McGraw-Hill, Inc., New York, NY, USA, 1986.

[Sme94] Philippe Smets. What is dempster-shafers model ? In Advances in the Dempster-Shafer Theory of Evidence, Chapter 1, pages 5 – 34. John Wiley an Sons, 1994.

[SSBM96] Gerard Salton, Amit Singhal, Chris Buckley, and Mandar Mitra. Automatic text decomposition using text segments and text themes. In UK Conference on Hypertext, pages 53–65, 1996.

[ST06] Ralf Schenkel and Martin Theobald. Structural feedback for keyword-based xml retrieval. In 28th European Conference on Information Retrieval Research (ECIR'06), pages 326–337, 2006.

[SZ06] Azadeh Shakery and ChengXiang Zhai. A probabilistic relevance propagation model for hypertext retrieval. In CIKM '06 : Proceedings of the 15th ACM international conference on Information and knowledge management, pages 550–558, New York, NY, USA, 2006. ACM.

[TA99] Saïd Tazi and Yahya Altawki. Création de documents virtuels : Cas des supports de cours. In Atelier sur les Documents Virtuels Personnalisables : De la Définition à l'Utilisation, 11ème Conférence Francophone sur l'Interaction Homme-Machine. IHM'99 - Montpellier - France, novembre 1999.

[TC90] H. Turtle and W. B. Croft. Inference networks for document retrieval. In SIGIR '90 : Proceedings of the 13th annual international ACM SIGIR conference on Research and development in information retrieval, pages 1–24, New York, NY, USA, 1990. ACM Press.

[Tea08] Andrew Trotman and Birger Larsen et al. INEX 2007 guidelines for topic development. In Focused access to XML documents : Sixth Workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2007), 2008.

[TS08] Andrew Trotman and Borkur Sigurbjornsson. Narrowed Extended XPath I (NEXI). In Focused access to XML documents : Sixth Workshop of the INitiative for the Evaluation of XML Retrieval (INEX 2007), 2008.

[VH99] Ellen M. Voorhees and Donna Harman. Overview of the eighth text retrieval conference (trec-8). In TREC, 1999.

[VM07] Delphine Verbyst and Philippe Mulhem. Lig at inex 2007 ad hoc track : Using collectionlinks as context. In INEX 2007 Workshop Proceeding, 2007.

[VM08] Delphine Verbyst and Philippe Mulhem. Doxels in context for retrieval : from structure to neighbours. In SAC '08 : Proceedings of the 2008 ACM symposium on Applied computing, pages 1122–1126, New York, NY, USA, 2008. ACM.

[Voo02] Ellen M. Voorhees. Overview of trec 2002. In TREC, 2002.

[Voo03] Ellen M. Voorhees. Overview of trec 2003. In TREC, pages 1–13, 2003.

[Voo04] Ellen M. Voorhees. Overview of trec 2004. In TREC, pages 1–12, 2004.

[Wat99] Carolyn Watters. Information retrieval and the virtual document. Journal of the American Society for Information Science, pages 1028–1029, 1999.

[Wil94] Ross Wilkinson. Effective retrieval of structured documents. In Research and Development in Information Retrieval, pages 311–317, 1994.

[Zar04] Haifa Zargayouna. Contexte et sémantique pour une indexation de documents semi-structurés. In Premiere COnference en Recherche d'Information et Applications (CORIA'04), pages 161–177, Toulouse, France, March 2004. Hermès.

[ZE98] Oren Zamir and Oren Etzioni. Web document clustering : A feasibility demonstration. In SIGIR '98 : Proceedings of the 21st Annual International ACM SIGIR Conference on Research and Development in Information Retrieval, August 24-28 1998, Melbourne, Australia, pages 46–54. ACM, 1998.

[ZL01] Chengxiang Zhai and John Lafferty. Model-based feedback in the language modeling approach to information retrieval. In CIKM '01 : Proceedings of the tenth international conference on Information and knowledge management, pages 403–410, New York, NY, USA, 2001. ACM.

[ZMWSD95] Justin Zobel, Alistair Moffat, Ross Wilkinson, and Ron Sacks-Davis. Efficient retrieval of partial documents. Information Processing and Management, 31(3) :361–377, 1995.

[Zse] The Zettair search engine. http ://www.seg.rmit.edu.au/zettair/.

Résumé

La recherche d'information sur des documents structurés tente de répondre de manière ciblée à une requête utilisateur en ne fournissant que des éléments de documents (doxels) pour satisfaire ce besoin d'information. Ce travail de thèse étudie l'apport de la caractérisation des relations (structurelles et non structurelles) entre parties de documents structurés dans ce contexte. Nous modélisons l'indexation des documents structurés en utilisant la structure et les relations entre doxels et nous caractérisons ces relations par des valeurs d'exhaustivité et de spécificité relatives. Le processus de recherche basé sur ces documents structurés génère des documents virtuels résultats, en spécifiant les liens pertinents entre les doxels. Le modèle est validé par des expérimentations sur la campagne d'évaluation INEX 2007 (660 000 documents Wikipedia, 100 requêtes) et les résultats obtenus montrent une amélioration de 24% en précision moyenne avec le modèle vectoriel.

Mots-clés: recherche d'information, documents structurés, documents virtuels, exhaustivité relative, spécificité relative, relations non-structurelles.

Abstract

Information retrieval on structured documents attempts to answer in a precise way to a user request by providing only elements of documents (doxels) that satisfies this need for information. This thesis investigates the characterization of relations (structural and non-structural) between parts of structured documents in this context. We model structured documents indexing using the structure and relations between doxels and we characterize these relations by relative exhaustivity and specificity values. The querying process based on these structured documents generates virtual documents as results, indicating the relevant links between doxels. The model is validated through the evaluation campaign INEX 2007 data (660 000 documents Wikipedia, 100 requests) and the results show an improvement of 24% in average precision with the vector space model.

Keywords: information retrieval, structured documents, virtual documents, relative exhaustivity, relative specificity, non-structural relations.