Interrogation flexible de données semi-structurées : une approche basée sur les ensembles flous

Martine De Calmès, Henri Prade, Florence Sèdes

Interrogation flexible de données semi-structurées :

une approche basée sur les ensembles flous [1]

Martine De Calmès, Henri Prade et Florence Sèdes

IRIT, CNRS et Université Paul Sabatier,
118 Route de Narbonne, 31062 Toulouse cedex 09, France
{decalmes, prade, sedes}@irit.fr

Résumé

Cet article fournit une discussion générale sur la mise en oeuvre de requêtes flexibles sur des collections de Données Semi-Structurées (DSSs). L'objectif est d'adapter les principes des requêtes flexibles, utilisés dans le cadre des bases de données « classiques », c'est-à-dire structurées, à l'interrogation de DSSs, et d'évaluer la pertinence et l'intérêt d'une telle adaptation par rapport à des techniques plus classiques (relaxation, distance d'édition, etc.). Une telle approche permet de gérer les priorités et les préférences de l'utilisateur, mais aussi de prendre en compte des problèmes liés à l'irrégularité de la structure sous-jacente des DSSs. En effet, la notion de requêtes flexibles semble être encore plus prometteuse pour les DSSs que pour les bases de données classiques, en raison de l'hétérogénéité structurale potentielle de celles-ci. La logique floue permet classiquement l'expression de conditions flexibles sur les valeurs d'attribut, mais aussi d'estimer la « proximité » ou similarité structurelle des DSSs, à partir de différents items tels que le degré de similarité des balises, ou des étiquettes d'attribut, au regard des éléments présents dans la requête. Les priorités de l'utilisateur peuvent aussi être introduites dans les requêtes pour indiquer

[1] Cet article constitue une version française remaniée, et étendue de l'article « Flexible Querying of Semi-Structured Data: a Fuzzy Set-Based Approach » (Int. Journal of Intelligent Systems, vol. 22, n. 7, 2007) par les mêmes auteurs.

l'importance relative des conditions élémentaires en terme de contenu, mais aussi ses préférences sur la localisation de l'information dans la structure. L'évaluation des requêtes emploie une échelle qualitative avec un nombre fini de niveaux. Les DSSs retrouvées en réponse à une requête sont alors ordonnées, en utilisant une procédure lexicographique de classement. La procédure est illustrée par des exemples obtenus dans le cadre de l'évaluation d'une plate-forme développée au sein du laboratoire IRIT.

***Mots-clés** : Requête flexible, donnée semi-structurée, XML.*

Abstract

This paper provides a general discussion about how flexible querying can be applied to Semi-Structured Data (SSD). We adapt flexible querying ideas, already used for classically structured databases, to XQuery-like querying of SSD for managing user's priority and preferences, but also for tackling with the variability of SSD underlying structures. Indeed flexible querying seems to be still more useful for SSD than for classical databases, because of the potential structural heterogeneity of the former. Fuzzy sets are useful for expressing flexible requirements on attribute values, and for estimating the degree of similarity of tags, or attribute labels, with elements present in the request. Priorities are introduced in the request for specifying the relative importance of elementary requirements in terms of their contents, but also preferences about the location of information in the structure. The evaluation of the queries uses a qualitative scale with a finite number of levels, for rank-ordering the retrieved pieces of SSD. The procedure is illustrated by examples running on an implemented platform.

***Keywords** : Flexible query, semi-structured data, XML.*

1. INTRODUCTION

Les modèles basés sur les Données Semi-Structurées (DSSs) sont différents des modèles de bases de données classiques, relationnel et/ou objet. Les DSSs sont auto-descriptives, et caractérisées par une structure irrégulière, partielle, adaptative, semi-explicitée et un typage irrégulier [1]. Les collections de documents web, balisés en HTML, XHTML ou XML, constituent donc des prototypes de documents dits eux-mêmes « semi-structurés » car illustrant le concept de DSS : ce sont en effet des

sources de données « dynamiques », dont les données et l'organisation évoluent fréquemment ; leur structure, éventuellement incomplète, n'obéit à aucun schéma prédéfini (c'est le cas par exemple des documents XML bien formés mais non valides, ou valides selon une structure dite « large »). Cette structure peut être partiellement ou complètement inconnue, a priori, de l'utilisateur. Les documents XML syntaxiquement corrects, dits alors *bien formés*, ne sont *valides* que s'ils respectent une DTD (Document Type Définition), structure générique connue a priori, correspondant au schéma d'une base de données.

La base de notre réflexion, dans ce qui suit, prend donc en compte tout type de document XML *bien formé*, mais pas nécessairement validé par une DTD. La structure logique de tels documents - chacun constituant donc une DSS -, différencie les balises, les attributs, leurs valeurs et leur contenu. Les balises et attributs utilisés ne sont pas forcément déclarés a priori dans une DTD, donc a fortiori pas forcément connus de l'utilisateur qui formule les requêtes.

L'évaluation de requêtes portant sur de tels documents est donc d'autant plus difficile que cette structure est irrégulière : même si deux documents semblent « visuellement » similaires dans leur présentation et leur contenu, leurs instances peuvent être différentes. Les termes d'une requête doivent être alors comparés aux balises, identifiants d'attributs, valeurs d'attributs ou contenus, à tout niveau de la structure sous-jacente, afin de vérifier si le document répond ou non à la requête.

C'est à partir de ce constat que nous nous sommes posé la question d'évaluer comment les requêtes flexibles, initialement développées pour les SGBD classiques, permettraient une meilleure expression des préférences de l'utilisateur, une fois adaptées à ces nouvelles structures de données. La réponse à de telles requêtes sera constituée d'une liste de documents, ordonnés selon un niveau de pertinence par rapport à la requête (et non sur une simple évaluation booléenne, considérant que la donnée répond ou non à la requête). Cette pertinence doit être évaluée, par exemple, à partir de niveaux de satisfaction ou de similarité, ce qui implique d'introduire plus de flexibilité dans le processus d'interrogation.

Cet article est organisé en trois parties. La première rappelle des travaux autour de la comparaison et de l'interrogation de documents semi-structurés. La deuxième considère les différentes formes que peuvent prendre les requêtes flexibles adressées à différents types de documents XML. En particulier, nous discutons comment manipuler les préférences de l'utilisateur qui portent sur le contenu de DSSs et la manière dont l'information peut apparaître dans les DSSs à rechercher. La dernière

section décrit une approche pour mettre en œuvre pratiquement des requêtes flexibles adressées aux collections de DSSs.

2. COMPARAISON ET INTERROGATION DE DOCUMENTS XML

La gestion de bases documentaires suppose la mise en oeuvre de fonctionnalités telles que l'interrogation et la surveillance de l'évolution des contenus et de leur organisation. Ainsi, la détection des évolutions d'un document modifié se base sur la comparaison de sa version initiale avec les suivantes, en évaluant leur (dis)similarité [30]. Les structures à comparer sont, dans ce cas, connues a priori, et leurs éléments peuvent être « adressés » par un utilisateur à qui cette structure aura été communiquée.

Interroger des DSSs peut également conduire à comparer deux structures : celle de la requête et celle du document-cible, ou, du moins, les éléments de structures qui auront pu être élicités à partir de son contenu. Mais l'hétérogénéité structurelle étant inhérente aux DSSs issues du web, tels les documents XML, la comparaison de schémas – a priori incomplets - ne peut dès lors reposer sur une comparaison stricte, qui risquerait de conduire à un ensemble vide de réponses. De plus, il est souhaitable de permettre à l'utilisateur d'interroger les sources d'information en n'exigeant qu'un minimum de pré-requis sur les structures sous-jacentes aux collections interrogées ; ceci est envisageable en privilégiant le seul élément dont dispose de manière certaine l'utilisateur, à savoir l'apparence présumée des documents, puisque celui-ci ne peut avoir, dans ce cas, une connaissance complète de la structure.

On examine maintenant brièvement les travaux existants en matière de DSSs tant du point de vue de la détection de différences que de l'interrogation.

2.1. Détection de différences

La comparaison de structures de documents est un problème classique. La plupart des travaux ne traitent que de la détection de changements dans des fichiers textes [25], des fichiers classiques ou des tables relationnelles [27]. Ils sont basés sur des algorithmes de comparaison d'ensembles d'enregistrements ou tuples identifiés par des clés. *LaDiff* [14], par

exemple, implémente la détection de balises et la visualisation de différences entre documents structurés, par comparaison des structures hiérarchiques des documents en entrée. Les travaux de [15], [14], [41] autour des systèmes XMLTreeDiff, XyDiff, et Xdiff attestent de la pertinence de telles approches d'évaluation des différences entre structures arborescentes, dans l'objectif de détecter les changements.

Des études se sont plus particulièrement intéressées aux documents structurés, par exemple avec SGML [10], [31]. Ces documents sont valides, puisqu'associés à une DTD. Ainsi, leur comparaison ne pose pas de problème particulier, puisqu'il s'agit dès lors de comparer la (les) DTD(s) qu'ils respectent, qui jouent le rôle de structure générique commune. De manière générale, deux documents peuvent être strictement comparés par rapport à leur structure, grâce aux techniques de comparaison de graphes classiques [37], [44]. La relation d'inclusion (« sous-arbre ») entre deux structures ne peut être détectée qu'en considérant les relations d'ordre entre éléments constitutifs de ces structures [26].

De manière générale, l'importance du problème s'illustre à travers la richesse et l'actualité des contributions, depuis les approches de recherche de régularités structurelles [39] [40], de découverte de structures communes approximatives [38], de découverte de sous-structures fréquentes [19], de recherche d'arbres dans une forêt [43] jusqu'à l'analyse de tendances via l'évolution des structures fréquentes au cours du temps [28].

La complexité du problème avec les DSSs est liée à l'hétérogénéité qui les caractérise. Les systèmes, tels Xylème [14], ou [7], détectent des différences en se basant sur l'identification de régularités, qui permet de regrouper les documents selon leur structure, par comparaison directe de structures arborescentes. Cette comparaison de structures s'avère inadaptée pour des comparaisons autres qu'une recherche de correspondance exacte. En effet, dans le contexte des DSSs, la recherche d'arbres identiques n'est pas suffisante.

Le processus de comparaison doit permettre de retrouver des arbres « approximativement égaux », selon un degré de similarité [30]. Des travaux basés sur le nombre de modifications minimales pour passer d'un arbre à un autre vont dans ce sens [29]. La « distance d'édition » est un des principaux indices pour évaluer les similarités entre arbres de documents XML. En effet, le calcul de la distance d'édition consiste à déterminer la séquence d'opérations élémentaires (telles que l'insertion, la suppression ou le renommage de nœud) minimale pour transformer un

arbre en un autre [44], [11]. La distance d'édition est la somme des coûts de ces opérations. Plus la distance d'édition sera faible, plus la similarité entre deux arbres sera forte. On peut aussi mesurer les similarités grâce au calcul de la distance d'alignement des arbres qui est en quelque sorte un cas particulier de la distance d'édition. D'autres travaux sont basés sur la comparaison de sous arbres [24] : ainsi, un algorithme pour résoudre les problèmes d'inclusion d'arbres est donné dans [24].

L'approche basée sur la fermeture floue a été proposée la première fois par Damiani, Tanca et Oliboni [16], [17], [18]. Cette approche est basée sur la notion de graphes flous [9]. Ces derniers sont obtenus à partir des arbres de documents en évaluant l'importance des informations associées aux nœuds de l'arbre. De nouveaux arcs sont ajoutés aux graphes de départ en appliquant la notion de fermeture floue. En effet, le poids associé aux arcs de fermeture est calculé par la conjonction des poids importants des arcs sous-jacents dans le graphe original, par l'intermédiaire d'un connecteur de conjonction (norme triangulaire). L'objectif principal de cette approche est de passer d'un problème d'appariement d'arbres à un problème d'appariement inexact de graphes ce qui par conséquent augmente la flexibilité de l'interrogation. Les réponses sont des sous-graphes du graphe flou fermé, fournies sous forme d'une liste classée selon le degré d'appariement à la requête.

L'implémentation de ces méthodes souffre de certains défauts et limites comme par exemple : le manque de systèmes de tri efficaces pour la méthode par relaxation, l'incapacité de fournir plusieurs réponses par document pour les méthodes basées sur la notion de distance d'édition et la complexité très élevée des autres méthodes. Pour résoudre ces problèmes et franchir les limites des méthodes déjà proposées, nous proposerons dans le chapitre suivant une nouvelle approche pour l'interrogation des données semi-structurées.

2.2. Interrogation de DSS

La plupart des techniques d'interrogation de documents semi-structurés supposent que leur efficacité peut être améliorée à l'aide du concept de similarité [23], [33], [42], [45]. Le calcul de celle-ci nécessite de définir une mesure qui permet de qualifier le degré de ressemblance ou de dissemblance entre deux documents ou encore entre fragments de documents. En effet, une similarité est en fait toute application à valeurs numériques (ou éventuellement sur une échelle discrète totalement ordonnée) qui permet de mesurer le lien entre les individus d'un même

ensemble. Pour une similarité, le lien est d'autant plus fort que sa valeur est grande.

Les requêtes flexibles sont un sujet de recherche qui a rencontré un intérêt croissant dans l'interrogation de DSSs cette dernière décennie et pour lequel plusieurs travaux ont été menés jusqu'ici. Plusieurs techniques ont donc été proposées pour traiter ce problème. On y trouve notamment : la relaxation des requêtes [29] [3], les techniques basées sur la corrélation et les techniques basées sur l'appariement approximatif des arbres [35] [36].

Extraire l'information à partir d'une DSS se fonde en effet sur l'analyse du contenu et par-là même de la structure. Cependant, la structure hiérarchique recherchée peut être incomplète en raison du manque possible d'information explicite dans les DSSs. Pour cette raison, et à cause de l'hétérogénéité des données et de l'absence de règles de syntaxe stricte dans les langages de balisage, le recours à l'appariement exact de structure n'est plus un procédé satisfaisant pour traiter les requêtes sur des DSSs. Une taxonomie des langages d'interrogation de documents XML et des techniques d'appariement approximatif est donnée dans [4], s'étendant de la relaxation d'arbre ou des graphes flous jusqu'à la Recherche d'Information (RI) pour les documents XML tels que XIRQL [22].

Des études se sont plus particulièrement intéressées aux documents structurés, par exemple avec SGML [10], [31]. Ces documents sont valides, puisqu'associés à une DTD. Ainsi, leur comparaison ne pose pas de problème particulier, puisqu'il s'agit dès lors de comparer la (les) DTD(s) qu'ils respectent, qui jouent le rôle de structure générique commune. De manière générale, deux documents peuvent être strictement comparés par rapport à leur structure, grâce aux techniques de comparaison de graphes classiques [37], [44]. La relation d'inclusion (« sous-arbre ») entre deux structures ne peut être détectée qu'en considérant les relations d'ordre entre éléments constitutifs de ces structures [26].

2.2.1. Les langages orientés RI

Les langages orientés RI proposent d'ajouter des nouvelles fonctionnalités concernant la recherche sur le contenu, comme l'utilisation du prédicat "about" au lieu du "contains" de XQuery. En effet, ces langages assignent des valeurs de pertinence à chaque unité d'information, ce qui permet de renvoyer à l'utilisateur des listes triées de résultats. Cependant, la plupart de ces approches utilisent des mesures de similarité basées sur le tf*idf. Le langage de requêtes XIRQL proposé par Fuhr et Grobjohann est un déploiement de XQL pour la RI [22]. Il est basé sur le modèle probabiliste inférentiel de RI [34]. Ce langage étend les opérateurs du langage XPath avec des opérateurs pour la recherche "orientée pertinence". D'autres opérateurs permettent d'effectuer des recherches vagues sur le contenu non-textuel.

Il est ainsi possible d'effectuer des recherches probabilistes sur le contenu numérique d'un élément. Le tri des documents est effectué selon la probabilité décroissante que le contenu est bien celui spécifié par l'utilisateur dans sa requête. XIRQL utilise les fonctionnalités de correspondance du langage XPath. Cependant, les utilisateurs ne connaissent pas nécessairement la structure des documents de la collection qu'ils interrogent.

Un autre exemple de langage orienté RI est Elixir (« an Expressive and efficient Language for Xml Information Retrieval »). Ce langage proposé par Chinenyanga et Kushmerick en 2001 [12] est une extension de XML-QL. Il est aussi basé sur le modèle vectoriel et utilise une mesure de similarité basée sur le cosinus.

Le langage XFIRM proposé par Sauvagnat [34] est basé sur un modèle de données générique permettant l'implémentation de nombreux modèle de RI et le traitement des collections hétérogènes, c'est-à-dire contenant des documents ne suivant pas la même DTD.

2.3. Bilan

En général, nous pouvons distinguer les tendances suivantes dans le traitement des requêtes XML approximatives :

- manipuler des requêtes « aveugles » (*blind queries*) : comme l'utilisateur peut avoir des connaissances parfaites ou n'avoir aucune connaissance sur la structure des DSSs, une requête peut se référer à

une structure-exemple, ou à une description de la structure recherchée (par exemple, une structure préexistante, une précédente version, une idée qu'il se fait, …),

- faire face à l'ambiguïté et à l'hétérogénéité, comme dans le cas des DSSs qui ont un contenu « sémantique » équivalent, et qui peuvent être représentées de différentes façons,
- traiter des requêtes basées sur le contenu dans les documents multimédia, approche qui ne sera pas abordée ici.

De la première tendance relèvent les approches qui ont choisi le point de vue "structure". Par exemple, Dubois et al. [21] suggèrent une méthode de comparaison graduelle. Elle tolère la suppression d'un niveau de structure si celui-ci ne s'avère pas pertinent, ou la prise en compte de l'importance de la balise/de l'attribut, son statut, son niveau dans l'arborescence, sa valeur, l'importance de ses descendants, etc.

Alilaouar [2] propose un modèle d'interrogation flexible prenant en compte le contenu et la structure sous-jacente des DSSs. La logique floue est utilisée pour représenter les préférences de l'utilisateur sur le contenu et la structure des DSSs. A la fin du processus d'interrogation, chaque réponse est associée à un degré de pertinence calculé en utilisant des mesures de similarité basées sur l'arbre recouvrant minimal pour la structure.

De la même manière, Damiani [16] propose un modèle flexible d'interrogation, basé sur une représentation sous forme de graphes étiquetés des documents XML, enrichis par une évaluation floue de l'importance des informations, au niveau des balises XML. L'originalité se situe, comme nous l'avons explicité plus tôt, dans l'extension de la structure arborescente du document, et la pondération des arcs sur la base de l'importance des noeuds qu'ils associent, afin d'augmenter la mesure du rappel. Des techniques de pondération et d'appariement flou des sous-graphes des données XML sont détaillées dans [17] et [18] dans le même but.

La deuxième tendance regroupe les approches de manipulation des documents XML structurés au moyen des techniques de RI classique. Bordogna [5] et Braga [6] traitent la flexibilité en tenant compte de la différence entre le vocabulaire de la structure et celui des balises. Ces approches fonctionnent sur la base de la connaissance de la sémantique des éléments (noeuds) ou des associations (arcs) à comparer de manière approximative. Cette connaissance peut être extraite à partir des noms d'éléments (balises), des attributs, de leurs valeurs, ou encore à partir des spécifications de n'importe quel utilisateur externe, ensuite utilisées dans

la comparaison approximative des graphes. Ciaccia [13] propose une approche qui exploite une autre forme d'approximations dans les structures de données, et fournit une méthode de classification selon la pertinence qui prend en compte le degré d'exactitude et de complétude des résultats.

L'approche basée sur la logique floue proposée ici diffère des travaux examinés ci-dessus en ce sens qu'aucune technique d'appariement de graphe n'est employée. Nous envisageons l'intégration de la flexibilité dans les requêtes comme un moyen pour prendre en compte les préférences de l'utilisateur dans l'expression des critères pour le DSSs à rechercher. Nous nous servons également des relations de similarité pour l'appariement approximatif des étiquettes telles que les balises, noms des attributs, etc. Cependant, si l'utilisateur pense avoir des connaissances sur la(les) structure(s) possible(s) des documents recherchés, il peut les utiliser pour formuler sa requête.

3. INTERROGATION FLOUE EN CAS DE CONNAISSANCE DE LA STRUCTURE

Les requêtes adressées à des (ensembles de) DSSs peuvent servir différents objectifs :

- rechercher des fragments d'information structurés de manière prédéfinie comme discuté en section 2,
- rendre explicite une partie de la structure de la DSS interrogée qui présente un intérêt pour l'utilisateur,
- retrouver des fragments d'information répondant à des critères de sélection qui se réfèrent au contenu plutôt qu'à la structure.

Dans ce dernier cas, insistons sur le fait qu'une DSS peut se révéler pertinente pour l'utilisateur indépendamment de sa structure interne. C'est sur ce dernier type de requêtes que nous allons nous focaliser dans ce qui suit. Il existe différentes formes d'information semi-structurée. Nous les examinons d'abord brièvement, avant d'aborder la manipulation des requêtes flexibles adressées aux sources contenant de telles informations.

3.1. Documents orientés texte vs. orientés données

La notion d'information semi-structurée peut en effet se référer à différents types de situations.

D'une part, existent de nombreuses collections de documents composés de fragments de textes balisés, les balises pouvant indiquer la présence d'un titre, d'auteur(s), d'une introduction, etc. Dans un tel cas, le *document* semi-structuré est basiquement une séquence de balises encadrant des fragments de texte libre. Les balises peuvent être imbriquées ; en effet, une balise peut identifier un sous-ensemble de balises (elles-mêmes « élémentaires » ou non, i.e. associées à un contenu ou non), identifiant ainsi différents niveaux de hiérarchie entre ces balises. De tels documents sont dits *orientés texte* : ils sont habituellement conçus pour être utilisés par des humains (supports pédagogiques en ligne, ouvrages, doc techniques, messages électroniques, annonces, etc.). Ils sont caractérisés par une structure « peu » régulière ou même quasiment irrégulière, des données qui présentent une granularité plus grande et beaucoup de contenus mixtes. L'ordre dans lequel les éléments enfants d'un même parent apparaissent est important.

A l'opposé, on peut rencontrer des documents semi-structurés décrivant des items en termes d'attributs et de valeurs d'attributs, organisés de manière structurée. Les balises sont utilisées pour étiqueter des sous-ensembles d'attributs ou de balises plus « élémentaires ». Un exemple de ce deuxième type de DSSs est illustré en Annexe. Il est donc par exemple possible de ne rencontrer aucun texte « libre » dans la description. Dans le cas contraire, celui-ci sera vu comme un type spécial de valeur d'attribut (cf. l'élément 'DETAIL' dans l'exemple en Annexe). Ces derniers, dits documents *orientés données,* utilisent XML en tant que vecteur de données. Ils sont conçus pour être exploités par des applications et le fait que XML soit utilisé est généralement accessoire. Autrement dit, il n'est pas primordial pour le fonctionnement de l'application que les données soient stockées en tant que document XML. L'intérêt est souvent d'utiliser XML comme "format d'échange" de données avec d'autres applications facile à utiliser et à interpréter. Ces documents sont caractérisés par une structure assez régulière, des données qui présentent une granularité très fine et l'absence de contenus mixtes. L'ordre dans lequel les éléments enfants d'un même parent apparaissent n'est en général pas significatif.

3.2. Niveau d'importance et degré de similarité dans les requêtes floues

La logique floue a été introduite au cours des deux dernières décennies dans les requêtes adressées aux bases de données usuelles, ou pour la recherche de documents (non-structurés) pertinents, à partir de mots-clés, en RI. Les préférences modélisent le degré d'association des mots clés avec les niveaux d'importance. Nous pouvons également employer les thésaurus, qui fournissent des degrés d'interchangeabilité entre les mots clés. L'approche basée ensembles flous des requêtes flexibles dans les bases de données relationnelles s'avère utile pour ordonner des ensembles de réponses sur la base des préférences des utilisateurs exprimées sous la forme de contraintes floues exprimant des besoins flexibles, et tenir compte de l'imprécision ou de l'incertitude des données stockées si nécessaire.

Dans le cas de bases de données, la flexibilité peut revêtir différentes formes. Elle peut tout d'abord traduire les préférences sur des valeurs d'attributs (par ex., « chercher un appartement pas trop cher, proche du centre »), représentées au moyen de fonctions d'appartenance d'ensembles flous. Il est également possible d'associer des niveaux de priorité ou d'importance à chaque condition élémentaire portant sur un attribut [19], [20].

Un autre type de flexibilité est basé sur l'utilisation de relations de similarité définies sur les domaines des attributs, qui permettent au système d'élargir les requêtes aux valeurs proches de ce qui a été explicitement requis.

Ainsi, l'évaluation d'une question conjonctive impliquant un ensemble de *n* critères P_i sur leur attribut respectif a_i, où $w_i \in [0,1]$ est le niveau d'importance du critère P_i, peut être effectuée par une expression de la forme :

$$E(x) = \min_i \max(1 - w_i, P_i(a_i(x))) \quad (1)$$

où $a_i(x)$ est la valeur de l'attribut i pour l'item x. Quand $w_i = 1$, l'importance de l'attribut i est maximale. D'ailleurs on suppose que $\max_i w_i = 1$, c.-à-d. que les conditions les plus prioritaires ont un niveau maximal d'importance et leur violation mène à un score égal à 0. Pour $0 < w_i < 1$, un mauvais score de $P_i(a_i(x))$, c.-à-d. le fait que $P_i(a_i(x))$ est proche de 0, ne peut pas être pris en compte pour l'évaluation globale qui pour ce critère ne peut pas descendre en dessous de $1 - w_i$. Ainsi l'effet de mauvais score de $P_i(a_i(x))$ est limité et contrôlé.

D'autres formes, plus générales, de requêtes flexibles mettent en jeu des prédicats flous non-unaires, tels que, par exemple, des relations floues de comparaison entre les valeurs des attributs. Un autre type de flexibilité est basé sur l'utilisation des relations de similarité définies sur les domaines d'attribut, permettant au système d'interrogation d'étendre les requêtes à des valeurs proches de ce qui est explicitement exigé. Ceci se traduit par une formule de la forme suivante :

$$E(x) = \min_i \max_t \min(S_i(a_i(x),t),P_i(t)) \quad (2)$$

où S_i est une relation floue de similarité sur le domaine d'attribut de a_i (S_i est supposée symétrique, réflexive), et $S_i(a_i, .)$ définit l'ensemble flou de valeurs proches de $a_i(x)$ dans le sens de S_i.

Il est possible de raffiner l'ordre fourni par les évaluations (1) et (2), qui peuvent mener assez souvent à des évaluations égales. Pour faire cela, nous employons un ordre lexicographique [32] des vecteurs qui sont construits à partir des composants apparaissant dans l'agrégation définie par $\min_i$. Ceci peut être particulièrement utile dans le cas où nous emploierions une échelle d'évaluation finie avec un nombre restreint d'éléments dans l'intervalle [0, 1]. En effet, dans le cas fini, il est encore plus probable d'avoir des égalités puisque le nombre de classes d'items ayant les mêmes évaluations, en utilisant (1) ou (2), dépend du nombre de niveaux dans l'échelle.

Dans ce qui suit, en raison de la nature qualitative des évaluations, des échelles finies avec un nombre restreint de niveaux seront utilisées, ainsi qu'une procédure de classement lexicographique.

3.3. Evaluation qualitative des interrogations

Une DSS étant constituée de balises, noms d'attributs, valeurs d'attributs et texte libre, une requête peut rechercher une information apparaissant à n'importe quel niveau de la structure de la DSS. L'utilisateur qui formule la requête peut ou non connaître l'organisation sous-jacente en terme d'attributs et de balises dans les sources qu'il interroge. S'il ne connaît pas cette information, la requête peut consister à découvrir la manière dont la ressource est organisée, aussi bien qu'à trouver l'(es) élément(s) recherché(s).

Dans cette sous-section, on suppose que l'utilisateur « a une idée » a priori de la structure en terme de balises/attributs, puisque nous nous concentrons sur le processus d'évaluation.

La source interrogée contenant de l'information semi-structurée, les balises peuvent intervenir dans l'expression des priorités. C'est le cas, par exemple, dans la requête suivante (en supposant qu'il existe des balises délimitant un titre et un résumé) : « trouver le document dans lequel les mots-clés 'flou' et 'base' apparaissent dans le résumé (repéré par une balise 'RESUME') et *de préférence* dans le titre (repéré par une balise 'TITRE') ». Cet exemple suggère plus généralement que dans ce cas, il faut manipuler conjointement deux types de priorités i) entre mots-clés comme en RI classique, et ii) au niveau des balises.

La requête pourra être étendue, comme il est habituel en RI, en considérant également les mots-clés similaires, ou, ici, des étiquettes synonymes pour les balises considérées, par exemple 'TITLE' pour 'TITRE'. Des attributs peuvent être impliqués, comme dans la requête « trouver les documents dont la catégorie est 'article' ou *peut-être* 'thèse', et où les mots-clés 'flou' (ou *peut-être* 'soft') et 'base' apparaissent dans le résumé et *de préférence* dans le titre ».

L'évaluation d'un prédicat de la forme 'A et *de préférence* B' est réalisée via une expression de la forme $\min(A(x), \max(1-\rho, B(x)))$, où $A(x)$ et $B(x)$ sont les degrés auxquels l'item x satisfait A et B, et ρ est d'autant plus grand que la préférence pour B est forte. En particulier, dans le cas de propriétés binaires A et B, l'évaluation retourne 0 dès que A n'est pas satisfait, 1 quand A et B sont tous les deux satisfaits, et $0 < 1 - \rho < 1$ quand A est satisfait et pas B. L'évaluation d'un prédicat de la forme 'A ou *peut-être* B' est obtenue par $\max(A(x), \min(\lambda, B(x)))$. Cette expression égale 1 quand A est satisfait, λ quand A est faux et B vrai, et 0 quand les deux sont faux, en cas de propriétés binaires.

Pour la requête donnée en exemple ci-dessus, un vecteur d'évaluation à 3 composantes peut être calculé. Il associe une évaluation élémentaire d'appartenance à l'attribut 'catégorie', et deux indicateurs de présence de mots-clés dans les sections identifiées par des balises. Les termes *peut-être* et de *préférence* sont pris en compte en pondérant les max et les min des expressions :

$$
\begin{aligned}
&(\mu_{cat.}(x),\\
&\quad \min(\chi_{résumé}(base), \max(1-\rho, \chi_{titre}(base))),\\
&\quad \min(\max(\chi_{résumé}(flou), \min(\lambda, \chi_{résumé}(soft))),\\
&\qquad \max(1-\rho, \chi_{titre}(flou), \min(\lambda', \chi_{titre}(soft))))),
\end{aligned}
$$

où $\mu_{cat.}(x)$ égale 1 si x = 'article', τ (<1) si x = 'thèse', $\chi_{<section>}(<mot>)$ égale 1 ou 0 selon que <mot> apparaît dans <section> ou non, et où $\tau < \lambda < 1-\rho < 1$, en supposant que la tolérance pour le remplacement de « flou » par « soft » est plus grande que pour le remplacement de « article »

par « thèse », et que la permission de ne pas avoir « base » dans le « titre » est plus importante que ces tolérances.

La première composante de l'expression correspond à $\max(\chi_{\text{article}}(x), \min(\tau, \chi_{\text{thèse}}(x)))$, où $\chi_{\text{article}}(x) = 1$ si x = 'article' et 0 sinon, $\chi_{\text{thèse}}(x) = 1$ si x = 'thèse' et 0 sinon. La deuxième composante vaut 1 si 'base' est dans 'titre' et dans 'résumé', $1-\rho$ si 'base' est seulement dans 'résumé', 0 sinon. De plus, si on a seulement 'soft' au lieu de 'flou', le terme correspondant vaudra λ au lieu de 1. Ces vecteurs peuvent être ordonnés en utilisant l'ordre lexicographique *leximin* [32]. Les composantes des vecteurs à comparer sont préalablement réordonnées selon leurs valeurs croissantes. Ensuite, les vecteurs ainsi obtenus - ici des triplets de valeurs entre 0 et 1 - sont vus comme les mots ordonnés d'un dictionnaire, l'ordre numérique décroissant jouant le rôle de l'ordre alphabétique.

Ainsi, les 'articles' contenant à la fois les mots 'base' et 'flou' dans le 'résumé' et le 'titre' seront classés en premier : ils correspondent au vecteur (1, 1, 1) ; viennent ensuite ceux qui contiennent à la fois 'base' et 'flou' dans le résumé, et 'base' et 'soft' dans le titre, qui correspondent au vecteur (λ, 1, 1), et ainsi de suite. Par exemple le vecteur (τ, λ, $1-\rho$) correspond à un document qui est une 'thèse' où 'base' et 'soft' n'apparaissent que dans le 'résumé'.

Dans ce cas, la procédure de classement doit reposer sur des échelles « qualitatives », car il devient difficile d'utiliser une échelle à trop de niveaux de manière significative. Nous proposons des échelles à 5 niveaux, à la fois pour évaluer l'importance, et les niveaux de satisfaction d'un document pour un prédicat élémentaire. Par exemple, l'importance peut être donnée par une échelle allant de « impératif » (α), « très important » (β), « assez important » (γ), « peu important » (δ), jusqu'à « pas important du tout » (ε), avec $\alpha > \beta > \gamma > \delta > \varepsilon$, l'ordre inverse $1 - (\cdot)$ étant défini par $1 - \alpha = \varepsilon$, $1 - \beta = \delta$, $1 - \gamma = \gamma$, $1 - \delta = \beta$, $1 - \varepsilon = \alpha$.

Pour les degrés de satisfaction, l'échelle linéaire ordonnée utilisée va de "satisfaisant" (α), "susceptible d'être satisfaisant" (β), "peut-être satisfaisant" (γ), "peu susceptible d'être satisfaisant" (δ), jusqu'à "non satisfaisant" (ε), et est encodée de la même manière.

Des requêtes similaires à celle donnée ci-dessus peuvent être adressées à une collection de documents "moins" textuels, comme suggéré par cet exemple : « Cherche maison à louer pour au moins 4 ou *peut-être* 3 personnes, telle que les caractéristiques "réfrigérateur" (ou *peut-être* "congélateur") et "télévision" apparaissent dans la description détaillée, ou *de préférence* dans la description du confort », qui correspond à l'expression suivante :

$(\max(\mu_{aumoins4}(x), \min(\beta, \mu_{aumoins3}(x)),$
$\min(\chi_{détail}(télévision), \max(1 - \delta, \chi_{confort}(télévision))),$
$\min(\max(\chi_{détail}(réfrigérateur), \min(\gamma, \chi_{détail}(congélateur))),$
$\max(1 - \delta, \chi_{confort}(réfrigérateur),$
$\min(\gamma, \chi_{confor}t(congélateur)))))),$

où $\mu_{aumoinsN}(X)$ est égal à 1 si x peut accueillir au moins N personnes (et 0 sinon). $\chi_{<description>}(<mot>)$ est égal à 1 ou 0 selon que <mot> apparaît dans <description> ou non.

4. INTERROGATION DANS LE CAS D'UNE STRUCTURE INCONNUE

En l'absence de connaissance a priori de la structure, l'utilisateur est amené à formuler des requêtes se référant à des éléments pouvant avoir des noms, rôles, niveaux différents selon les documents. Une manière possible d'évaluer une requête consiste alors à établir une correspondance éventuelle entre les éléments de la requête et les attributs/balises sources. Une alternative, que nous ne considérerons pas ici, consisterait à faire proposer par le système une reformulation de la requête en utilisant le langage de la source en terme de balises et d'attributs.

4.1. Principes généraux de l'approche

La collection à interroger étant composée de DSSs ayant des structures différentes (hétérogènes), un guide de données, fournissant mots-clés ou expressions-clés du domaine d'application, est supposé mis à disposition pour aider à la formulation dans une première phase. Ces «étiquettes»-clés peuvent être des noms d'attributs ou des balises de DSSs, ou des expressions plus ou moins équivalentes. Ce sera particulièrement le cas si le guide de données est obtenu à partir d'une analyse des régularités des diverses structures de DSSs. Dans ce cas, les balises ou étiquettes d'attribut identifiées à partir de la régularité des descriptions peuvent être affichées, y compris selon leur niveau d'imbrication s'il y a lieu. Le guide de données pourrait être également basé sur des ontologies décrivant le domaine de connaissances considéré. Ce guide de données est un outil visant à aider l'utilisateur à exprimer ses requêtes : celui-ci est seulement supposé avoir des connaissances générales du domaine (il comprend les

mots-clés), mais pas concernant les structures fondamentales des documents.

Une requête conjonctive est composée par l'utilisateur en choisissant quelques étiquettes du guide de données. Pour ces étiquettes, l'information associée sera visualisée dans les DSSs sélectionnées qui satisfont les critères d'interrogation. De plus, l'importance de chaque critère élémentaire dans la requête peut être calculée. L'évaluation d'une requête conduit à la recherche dans chaque DSS des balises et des attributs pouvant être considérés comme suffisamment similaires avec les étiquettes choisies. A titre d'exemple, les thésaurus fournissent des étiquettes qui sont (plus ou moins) synonymes. Une fois ceci établi, un processus doit identifier si des valeurs liées peuvent correspondre à ce qui est recherché. Si la valeur ne peut pas être identifiée, cela signifie que la requête ne peut pas être évaluée selon l'attribut considéré.

Formellement, une requête est une liste (conjonctive) de conditions (avec leurs niveaux d'importance respectifs λ_1,... λ_i,... λ_n) de la forme A_i, θ_i [v_i] où A_i est un élément qui apparaît dans le guide de données, θ_i un opérateur utilisé dans ce contexte et qui peut se référer à la valeur v_i. Plusieurs situations peuvent être rencontrées dans une DSS :

1. A_i est un nom d'une balise,
2. A_i est similaire au nom d'une balise,
3. A_i un nom d'attribut (ou similaire à),
4. A_i apparaît comme une valeur d'attribut (ou similaire à),
5. A_i apparaît dans un fragment de texte libre lié à une balise (ou similaire à).

Dans les trois premiers cas, le texte libre ou la valeur associée à une balise (cas 1, 2), ou un attribut (cas 3) peut être positivement ou négativement interprétable, ou non-interprétable (cas par exemple des balises "vides"), selon la condition θ_i [v_i]. Par exemple, si la condition à évaluer est "existe un jardin", <jardin>oui</>, <jardin>non</>, et <jardin>ombragé</> illustrent ces trois cas (le système pouvant interpréter les mots "oui" et "non", mais pas "ombragé"). Si elle est positivement interprétée, l'évaluation renvoie la valeur maximale dans l'échelle, à savoir α. Cependant, si nous devons employer les étiquettes similaires afin d'obtenir une interprétation positive, l'évaluation est diminuée du niveau de similarité de l'étiquette de substitution (l'échelle à 5 niveaux étant également employée pour la similarité). Par exemple, pour la recherche « d'une maison équipée d'un congélateur », une maison ayant un réfrigérateur (seulement) peut recevoir une évaluation égale à S(réfrigérateur, congélateur), selon l'équipement. En cas d'impossibilité à interpréter, une évaluation, s'étendant de la valeur « moyenne » γ à la plus

petite, ε, peut être retournée selon le contexte. En cas d'interprétation négative, la plus petite valeur, ε, est retournée (ou δ, si l'évaluation conduit à une certaine incertitude). Les deux derniers cas 4 et 5 sont un peu différents. Dans le cas 4, où A_i est une valeur d'attribut, le document peut être jugé comme potentiellement pertinent selon la condition ; une évaluation selon la valeur « moyenne » peut être pertinente. Dans le cas 5, l'évaluation de la pertinence potentielle dépendrait de la capacité à interpréter le texte libre et les structures sous-jacentes.

Enfin, une fois qu'une condition de sélection i est évaluée par un niveau ρ_i de l'échelle, le niveau d'importance λ_i est pris en compte, en retournant $\max(1 - \lambda_i, \rho_i)$ comme évaluation.

4.2. Exemple

Ces idées ont fait l'objet de développements qui ont été implantés sur la plate-forme PRETI (http://www.irit.fr/PRETI). Voir [19] pour une représentation de différentes fonctionnalités en relation avec la mise en œuvre de requêtes flexibles. Le point de départ est une collection au format XML qui contient les descriptions de maisons à louer et autres gîtes. Chaque item élémentaire correspond à une information de type simple (chaîne, entier, double, booléen, réel,…). Tout descripteur peut inclure un champ donnant des informations (éventuellement supplémentaires) sous forme d'un court texte en langue naturelle. Cette collection regroupe donc des DSSs ayant des structures sous-jacentes éventuellement différentes.

Compte tenu de la multiplicité des problèmes posés par la formulation de requêtes sur des collections de DSSs dont la structure est inconnue a priori, nous avons exploré la mise en oeuvre de différentes stratégies.

En pratique, une aide pour la formulation de la requête sous forme de structure-exemple peut être apportée en proposant à l'utilisateur de visualiser la structure (arborescente) sous-jacente (partielle ou totale) de la collection interrogée sous forme de guide des données, c'est-à-dire en affichant les étiquettes des balises (et éventuellement des attributs) selon leur niveau d'imbrication. A partir de la structure qui lui est proposée, l'utilisateur peut formuler un premier type de requête en choisissant les éléments à afficher. L'utilisateur peut aussi formuler un autre type de requête à partir des éléments qui lui sont proposés, - avec des prédicats plus ou moins complexes -, associés par des opérateurs éventuellement flexibles tels que 'est-environ', 'est-très-proche'. Ces opérateurs peuvent être typés de manière à ce qu'ils ne puissent être associés qu'avec les

éléments auxquels ils s'appliquent. De la même façon que pour des requêtes flexibles sur une base de données, des niveaux d'importance peuvent être spécifiés afin de pondérer la conjonction des contraintes élémentaires posées par la requête.

La figure 1 illustre la requête « cherche gîte à louer près de la mer ayant environ 3 couchages et *peut-être* une télévision » formulée par l'utilisateur via l'interface PRETI.

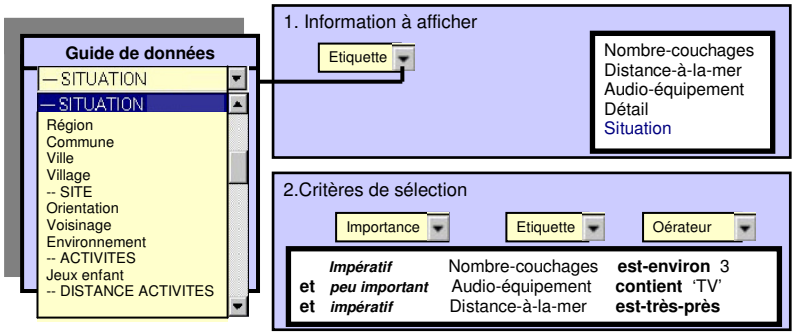

FIG. 1 – ***Exemple de requête flexible***

La requête comporte deux parties (boîtes 1 et 2 de la figure 1), l'une explicitant les informations à afficher dans les DSSs sélectionnées, l'autre les conditions de sélection. Les conditions sont spécifiées sur les étiquettes du guide de données à travers la boîte 2, via des opérateurs comme « existe », « contient », ou « est environ », « près de »,… Les opérateurs sont associés aux étiquettes pour lesquelles ils sont adaptés : par exemple., « est-environ » ne peut s'appliquer qu'à des étiquettes supposées avoir des valeurs numériques. De même, les opérateurs disponibles pour des étiquettes liées à la distance, comme l'item « distance à la mer », sont 'est-très-proche', 'est-assez-proche' et 'est-très-loin', qui sont supposés être modélisés par des ensembles flous sur les domaines numériques. L'évaluation de cette requête implique la recherche de balises ou attributs dans chaque DSS qui sont considérés comme équivalents aux étiquettes de la requête. Il faut noter que l'utilisation de ces opérateurs, dont la fonction d'appartenance doit être élicitée via l'utilisateur, suppose des connaissances sur les documents, comme, dans ce cas, le fait que la distance est exprimée en km.

La figure 2 montre un résultat partiel de la requête ci-dessus. Les descriptions pertinentes des maisons à louer ont été complétées par le

degré de satisfaction correspondant, et elles sont lexicographiquement ordonnées comme expliqué dans la section 3.2.

Notons que le traitement de l'exemple ci-dessus n'introduit pas la similarité entre les étiquettes. Comme expliqué dans 4.1, nous pourrions étendre la procédure d'évaluation en tenant compte de la similarité approximative entre les étiquettes dans la requête et les noms des balises et les attributs dans le document. Ceci s'applique également aux valeurs d'attribut ou de balise. Par exemple, si la condition est "audio-équipement contient TV", les DSSs qui intègrent <EQUIPEMENT> télévision, téléphone </>, peuvent être évaluées positivement.

<MAISON-A-LOUER ID=398-11>
<EVALUATION> α,α,α </> …
<NBCOUCHAGE> 3 </> **<DISTANCE** ID ='**mer**' > 0.5 </>
<AUDIO > TV </> **<AUDIO** > phone </> **<DETAIL>** Séjour-cuisine, 1 chambre (1 lit double - 1 lit simple), salle d'eau et WC. Chauffage électrique. Petit gîte de plain-pied au coeur du village avec cour attenante et fleurie de 30m2. Télévision. Situé à 1km de l'étang et à 500m de la mer. Salon de jardin. </> </>
…
<MAISON-A-LOUER ID=262-11>
< EVALUATION >γ,α,α</> …
<CAPACITE **NBCOUCHAGE='3'** NBCHAMBRE='2'> … </>
<DISTANCE-MER> 0.7 </> **<EQUIPEMENT-AUDIO** > null </>
<DETAIL> Rez-de-chaussée: séjour-cuisine avec cheminée. Lave-linge. 1er étage: 2 chambres (1 lit 2 pers. - 1 lit 1 pers.). Salle de bain, WC. Maison à la ferme avec salon de jardin et barbecue, à 900m du village. </>
</>
…
<MAISON-A-LOUER ID=522-11>
< EVALUATION > ε,γ,α </> …
<COUCHAGES > 4 </>
<DISTANCE TO ='**mer'** > 1.0 </>
<AUDIO> </>
<DETAIL> Rez-de-chaussée: cuisine, salle à manger. 1 chambre (1 lit double. - 2 lits simples), salle de bain, WC. Prise TV. Cottage situé dans un village typé et animé du bord de mer. Jardin ombragé commun aux deux cottages, salon de jardin. </> </>
…
<MAISON-A-LOUER ID=208-11>
< EVALUATION > ε,δ,α </> …
<COUCHAGES > 4 </>
<DISTANCE ID ='**mer**' > 4.0 </>

<DETAIL> Rez-de-chaussée et étage : cuisine, séjour, 2 chambres (1 lit 2 pers. - 2 lits 1 pers.), salle d'eau, WC, chauffage électrique. Maison mitoyenne dans le village, avec cour fermée et abri couvert. **</>**
</>

FIG. 2 –*Extrait du résultat de la requête*

4.3. Quelques autres améliorations

Dans le cas où l'utilisateur ne connaît pas a priori la structure de DSSs, les réponses aux requêtes peuvent l'aider à découvrir la manière dont les structures sous-jacentes de DSSs sont présentées dans la collection. Ceci peut lui permettre de formuler ensuite ses requêtes avec plus de précision. Donc, l'évaluation d'une requête pourrait également employer les techniques d'appariement flou des modèles "fuzzy pattern matching" en utilisant des variables libres [8], ce qui nous permettrait de rassembler des valeurs ou des étiquettes dans un contexte prescrit et de les réutiliser dans la recherche.

Nous pourrions envisager une approche un peu différente, le processus d'interrogation étant mis en oeuvre par une stratégie en deux phases. La requête est d'abord évaluée par une procédure de recherche de motifs restreinte au niveau des balises et des noms d'attributs. Afin de rendre la comparaison plus flexible, une ontologie conceptuelle pourra être utilisée pour élargir la signification de la requête, et fournir un contexte à celle-ci. Cette ontologie sera supposée regrouper des étiquettes de balises ou d'attributs. Ces étiquettes identifient des items dont le concept associé dans l'ontologie est pertinent. Ceci peut en effet aider l'utilisateur à se focaliser sur des documents l'intéressant. Pour des sources identifiées comme pertinentes, la deuxième phase d'évaluation peut consister soit à exploiter directement le contenu des documents retrouvés (dans la mesure du possible), soit à préciser la requête sous forme d'exemples exprimés dans le format des DSS obtenues à l'issue de la première phase.

En outre, bien que les balises ne soient pas des attributs proprement dits, nous pouvons imaginer avoir des balises jouant le rôle de "super-attributs", c'est-à-dire qu'une fonction est alors associée à une balise. Cette fonction indique comment les valeurs d'étiquette liées aux balises ou aux attributs, dont l'union constitue le sous-ensemble étiqueté par la balise, doivent être combinées afin de calculer la valeur associée à l'étiquette (vue comme super-attribut). Elle rend possible de déduire une valeur plus globale d'attribut des différentes valeurs associées à des balises plus élémentaires, apparaissant à un niveau de la hiérarchie. Par

exemple, la balise "nombre-de-lits" peut être associée à la somme des valeurs numériques de lits "DOUBLE", lits "SIMPLE" et lits "APPOINT" (cf. exemple en Annexe).

5. CONCLUSION

Nous avons exposé dans cet article les éléments d'une discussion sur l'interrogation flexible pour la manipulation d'information semi-structurée. Selon qu'on manipule des documents textuels semi-structurés ou des données semi-structurées, on peut être plus près des techniques RI floues, ou des méthodes floues appliquées aux bases de données.

Chercher dans un document ce qui est un attribut et ce qui est sa valeur, ce qui peut être plutôt facile si l'attribut est rencontré dans la DSS, pourrait être aussi difficile que la compréhension de langage naturel dans le pire des cas. Ceci crée une incertitude dans le processus d'évaluation. C'est pourquoi ce papier a souligné l'utilisation d'échelles qualitatives pour évaluer les préférences, la similarité ou le niveau de satisfaction, dans le processus de d'évaluation des requêtes.

Nous avons proposé une méthode d'interrogation de DSSs basée sur un guide de données. Le processus d'interrogation distingue l'affichage de l'information pertinente d'une part, l'évaluation des DSSs sélectionnées (quand cela est possible), et leur ordonnancement d'autre part. L'information est retrouvée principalement au moyen d'appariement d'étiquettes et en tenant compte des synonymies connues (approximatives).

La façon d'ordonner les DSSs doit refléter les niveaux de satisfaction des préférences exprimées dans les requêtes comme dans l'interrogation flexible de base de données. On peut également formuler la manière d'afficher les documents retrouvés en tenant compte de la plausibilité de l'adéquation des DSSs retrouvées avec la requête. Cette plausibilité reflète alors le fait que nous sommes d'autant moins certains qu'une DSS est pertinente que le processus d'appariement a été plus approximatif.

6. ANNEXE - EXTRAIT DE LA COLLECTION

```
<MAISON-A-LOUER>
<NUMERO>20-11</>
```

<CAPACITE>
<NB-CHAMBRES>2</>
<LITS><ADULTES>
<SIMPLE>1</> <DOUBLE>2</> <APPOINT>0</> </>
<ENFANT>0</> <BEBE>0</> </> </>
<ACTIVITES>
<JEU-ENFANT>balençoire</> <DIST ID="mer">9.5</>
<DIST ID="tennis">0.5</> <DIST ID="équitation"> 6.0</>
<DIST ID="piscine">1.1</> </>
<CARACTERISTIQUE> <NB-ETOILES>1</>
< STYLE>maison</>
<CONFORT> <EQUIPEMENT>
<ELECTRO> réfrigérateur, congélateur, lave-linge </>
<AUDIO> TV</> </>
<DIST ID="gare">14.0</> <DIST ID="commerce">0.5</> </>
<TARIF> <PERIOD ID="bas-saison">210</>
<PERIOD ID="juillet">270</> <PERIOD ID="aout">270</> **</>** </>
<DETAIL> Maison à 2 étages – cuisine avec lave-linge, séjour, 2 chambres (2 lits 2pers. – 1 lit 1pers.), salle de bain. Jardin avec barbecue. </>
</>

<MAISON-A-LOUER>
<NUMERO>30-11</>
<CAPACITE>
<NB-CHAMBRES>4</> <NB-COUCHAGES>4</>
<BEDS>
<SIMPLE>2</> <DOUBLE>2</> </></>
<ACTIVITES> <DISTANCES>
<SKI>0.5</> <PATINAGE>0.5</> <GARE>20.0</> </></>
<CARACTERISTIQUE>
<NB-ETOILES>*</> < STYLE>appartement</> </>
<EQUIPEMENT>
<CHAUFFAGE>électrique</><ELECTRO> frigidaire </>
<ELECTRO> congélateur </> </>
<TARIF>
<BAS-SAISON> 230</> <JANVIER>300</> <FEVRIER>300</> </>
<DETAIL> Appartement au 2ème étage – cuisine avec congélateur, séjour avec cheminée ; 2 chambres 2 lits doubles et 2 lits simples. Prise TV. Casier à ski. Situé au pied des pistes. </> </>

<MAISON-A-LOUER>
<NUMERO>35-11</>
<CAPACITE> <NB-CHAMBRES>2</>
<LITS><ADULTES>

<SIMPLEE>1</> <DOUBLE>2</> <APPOINT>0</></>
<ENFANT>0</><BEBE>0</> </> </>
<ACTIVITES>
<JEU-ENFANT></> <DISTANCE ID=”mer">10.</>
<DISTANCE ID="tennis ">0.4</><DISTANCE ID="équitation">1.</> </>
<CARACTERISTIQUE>
<NB-ETOILES>1</> < STYLE ID=”maison”></></>
<CONFORT><EQUIPEMENT>
<ELECTRO ID=”réfrigérateur”>oui</><ELECTRO ID=”congélateur”>oui</>
<AUDIO ID=”TV”>oui</> </>
<DISTANCE ID="gare">14.0</><DISTANCE ID="commerce">0.5</> </>
<DETAIL> Charmante maison située au cœur d’un village avec jardin attenant fermé. Barbecue et salon de jardin.</>
<TARIF>
<PERIOD ID="bas-saison">210</> <PERIOD ID="juillet">270</>
<PERIOD ID="aout">270</> </>
</>

7. REFERENCES

[1] S. Abiteboul, Semi-structured information, *Proc. ICDT’97, Int. Conf. Database Theory*, 1-20.

[2] A. Alilaouar, Contribution à l’interrogation flexible de données semi-structurées, *Doctorat Université Paul Sabatier*, Toulouse 3, Octobre 2007.

[3] S. Amer-Yahia, S. Cho, D. Srivastava, Tree Pattern Relaxation, In *EDBT’02*, Prague, Czech Republic, pp. 496-513, 2002.

[4] S. Amer-Yahia, N. Koudas, D. Srivastava, Approximate Matching in XML, *ICDE'03 Tutoria*l. http://www.research.att.com/~sihem/publications/

[5] G. Bordogna, G. Pasi, Flexible querying of Web documents, *Proc. ACM Symp. Appl. Comp. 2002*, Madrid (Spain), 675-680.

[6] D. Braga, A. Campi, E. Damiani, P.L. Lanzi., G. Pasi, FXPath: Flexible querying of XML documents, *Proc. EuroFuse 2002, Workshop on Information Systems*, Varenna (Italy), B. De Baets, J. Fodor, G. Pasi eds., 115-120.

[7] A. Broder, R. Kumar., F. Maghoul, P. Raghavan, S. Rajagopalan, R. Stata, A. Tomkins, J. Wiener, Graph structure in the web, WWW 9th Conf. *Proc., http://www9.org/w9cdrom/160*.

[8] M. Cayrol, H. Farreny, H. Prade, Fuzzy pattern matching. *Kybernetes*, 11, 103-116, 1982.

[9] K. Chan, Y. Cheung, Fuzzy Attribute Graph with Applications to Character Recognition, *IEEE Trans. Systems, Man and Cybernetics*, 1992.

[10] S. Chawathe, H. Garcia-Molina, Meaningful change detection in structured data, *Proc. ACM SIGMOD Int. Conf. Manag. of Data*, 1997, 22-32.

[11] S. Chawathe, Comparing Hierarchical Data in External Memory. In *proc of VLDB'99*, Edinburgh, Scotland, U.K, pp.90-101, 1999.

[12] T. Chinenyanga, N. Kushmerick, An expressive and efficient language for XML information retrieval, *Journal of the American Society for Information Science and Technology*, v.53 n.6, p.438-453, 2002.

[13] P. Ciaccia, W. Pezzo, Adding flexibility to structure similarity queries on XML data, *FQAS'2002*, T. Andreasen et col. Eds., Springer Verlag, LNAI 2522, 124-139.

[14] G. Cobéna, S. Abiteboul, A. Marian, Detecting changes in XML documents, *Actes BDA'2001*, Agadir (Maroc), Cepaduès, Toulouse.

[15] D. Curbera, A. Epstein, Fast difference and Update of XML Documents, *Xtech'99*, San Jose, March 99.

[16] E. Damiani, L. Tanca, Blind queries to XML data, *Proc. 11th DEXA*. Springer Verlag, LNCS 1873, Eds M. Ibrahim, J. Küng, N. Revell, 345-356, 2000.

[17] E. Damiani, L. Tanca, F. Arcelli Fontana, Fuzzy XML queries via context-based choice of aggregations. *Kybernetika*, 36(6): 635-655, 2000.

[18] E. Damiani, B. Oliboni, L. Tanca, Fuzzy techniques for XML Data Smushing, *LNCS* n° 2206, pp. 637-689, 2001.

[19] M. De Calmès, D. Dubois, E. Hüllermeier, H. Prade, F. Sèdes, Flexibility and fuzzy case-based evaluation in querying: an illustration in a experimental setting, *International Journal of Uncertainty, Fuzziness and Knowledge-Based Systems*, 11(1):43-66, 2003.

[20] D. Dubois, M. Nakata, H. Prade, Find the items which certainly have (most of) important characteristics to a sufficient degree. *Proc. 7th IFSA World Cong.*, Prague, Academia publ., 243-248, 1997.

[21] D. Dubois, H. Prade, F. Sedes, The use of some fuzzy logic techniques in multimedia databases querying, *IEEE Transactions on Data and Knowledge Engineering*, 13 (3): 383-393, 2001.

[22] N. Fuhr, K. Großjohann**,** XIRQL: "A query language for information retrieval in XML documents", *ACM SIGIR 2001***,** New Orleans, USA, pp. 172 – 180, 2001.

[23] D. Hawking, P. Thistlewaite. Proximity operators - so near and yet so far, Proceedings of the 4th Text Retrieval Conference (TREC-4.), pp. 131-143, 1995.

[24] S. Khanna, R. Motwani, et F. F. Yao. Approximation algorithms for the largest common subtree problem. Technical report, Stanford University, 1995.

[25] M. Kifer, EDIFF- a comprehensive interface to diff for Emacs 19. Available through anonymous ftp at *ftp.cs.sunysb.edu*, 1995.

[26] P. Kilpelaïnen, Tree matching problems with application to structured text databases. *Tech. Report, Dept. Computer Science, Univ. Helsinky*, Finland, 1992.

[27] W. Labio, H. Garcia-Molina, Efficient algorithms to compare snapshots. *Manuscript, available by anonymous ftp from db.stanford.edu* in pub/labio/1995/, 1995.

[28] P.A. Laur, M. Teisseire, P. Poncelet, Données semi-structurées - Découverte, maintenance et analyse de tendances, *RSTI série ISI*, 8 (5-6), 49-78, 2003.

[29] D.W. Lee, Query relaxation for XML model, *Ph.D Dissertation, Univ. California*, L.A., 2002.

[30] D. Lin, An Information-Theoretic Definition of Similarity, in 15th *Int. Conf. on Machine Learning*, 1998, pp. 296-304.

[31] T. Milo, S. Zohar, Using schema matching to simplify heterogeneous data translation, *VLDB'98*, 122-133, New York, 1998.

[32] Moulin H., *Axioms of Cooperative Decision Making*, Cambridge University Press, Cambridge, MA, 1988.

[33] G. Salton, C. Buckley, Automatic text structuring and retrieval: Experiments in automatic encyclopedia searching, In *ACM/SIGIR Conference*, 1991, pp. 21-31.

[34] K. Sauvagnat, Modèle flexible pour la recherche dans des corpus de documents semi-structurés, Thèse de doctorat, Toulouse 3, 2005.

[35] M. Selkow, The Tree-to-Tree Editing Problem, *Information processing letters* (6)6: 1977, pp.184-186.

[36] D. Shasha, K. Zhang, *Approximate tree pattern matching in pattern matching string, trees and arrays*, chapter 14. Oxford University Press, 1997.

[37] J.T.L. Wang, K. Zhang, K. Jeong, D. Sasha, A system for approximate tree matching. *IEEE Transactions on Knowledge and Data Engineering*, 6 (4): 559-571, 1994.

[38] J.T.L. Wang, B.A. Shapiro, D. Shasha, K. Zhang, C.Y. Chang, Automated Discovery Structures, *Int. Conf. on Knowledge Discovery and Data Mining (KDD'96)*, 1996, pp. 70-75.

[39] K. Wang, H. Liu, Schema discovery for semi-structured data, *Int. Conf. on Knowledge Discovery and Data Mining (KDD'97)*, Newport Beach, USA, August 1997, pp. 271-274.

[40] K. Wang, H. Liu, Discovering structural association of semi-structured data, *IEEE Trans. on Knowledge and Data Engineering*, Jan. 1999, pp. 353-371.

[41] Y. Wang, D. DeWitt, J.Y. Cai, XDiff. an effective change detection algorithm for XML documents, *ICDE* 2003.

[42] R. Wilkinson. Effective retrieval of structured documents, In *SIGIR'94*, 1994, pp. 311-317.

[43] M. Zaki, Efficiently Mining Frequent Trees in a Forest, *Int. Conf. on Knowledge Discovery and Data Mining (SIGKDD'02)*, Edmonton, Canada, July 2002.

[44].K. Zhang, D. Shasha, Simple fast algorithms for the editing distance between trees and related problems. *SIAM Journal of Computing*, 18 (6): 1245-1262, 1989.

[45] J. Zobel, A. Moffat, R. Wilkinson, Efficient retrieval of partial documents, *Information Processing and Management*, 31(3), 1995, p. 361-377.