Vers une Interopérabilité Multi-Niveaux des Métadonnées

Samir Amir— Ioan Marius Bilasco — Thierry Urruty — Chabane Djeraba

LIFL UMR CNRS 8022, Université de Lille1, Télécom-Lille1, 50 avenue Halley Parc scientifique de la Haute-Borne, 59655, Villeneuve d'Ascq, France.

{samir.amir, marius.bilasco, thierry.urruty, chabane.djeraba}@lifl.fr

RÉSUMÉ. *Plusieurs techniques de matching ont été proposées ces dernières décennies permettant la réalisation de l'interopérabilité des métadonnées. Toutefois, la plupart de ces techniques se concentrent sur le matching au bas niveau (niveau schéma) en prenant en compte les schémas provenant d'un seul langage de description, ce qui résout partiellement le problème d'hétérogénéité. Dans ce contexte, nous proposons, une nouvelle approche de matching, baptisée MuMIe (Multi-level Metadata Integration). Elle a pour but de réaliser l'interopérabilité sur les deux niveaux (langage et schéma de description). La technique proposée transforme les schémas provenant de différents langages en graphes, en capturant uniquement quelques concepts basiques. Une méthodologie de matching est ensuite effectuée dans le bas niveau, permettant de trouver les correspondances entre les nœuds des graphes, via l'utilisation de plusieurs informations sémantiques et structurelles. Les expérimentations effectuées montrent les bonnes performances du système proposé.*

ABSTRACT. *Several matching techniques have been proposed in recent decades to achieve metadata interoperability. However, most of the techniques focus on matching in the lowest level (schema level) taking into account only schemas defined using one description language which partially solve the problem of heterogeneity. In this context, we propose in this paper, a new matching approach, named MuMIe (Multi-level Metadata Integration), it aims at achieving interoperability on both levels (schema and description language). The proposed technique transforms schemas from different description languages in directed labeled graphs capturing only some basic concepts. A methodology for matching is then performed in the lower level to find the mapping between graph nodes. This is done by using several structural and semantic information. Our experimental results demonstrate the performances the proposed system.*

MOTS-CLÉS : *Interopérabilité, intégration des métadonnées, langages de description.*

KEYWORDS: *Interoperability, metadata integration, description languages*

1. Introduction

L'omniprésence des ressources numériques et non numériques dans notre vie pose le problème de leur gestion efficace et a généré un intérêt croissant pour l'utilisation des métadonnées destinées à améliorer la performance et l'efficacité des procédures de recherche, de filtrage et d'accès à ces types de contenus. Les métadonnées peuvent véhiculer divers types d'information décrivant les types des contenus multimédia (ex : vidéo, image, audio, etc.), les informations sémantiques de ces contenus (ex : image conceptuelle, contexte d'une vidéo, etc.), les caractéristiques des appareils recevant ou transmettant ces contenus (ex : réseaux, TV, mobile, etc.) et les caractéristiques des utilisateurs (ex : profil, préférences, etc.).

En anticipant la croissance des métadonnées, nous prévoyons qu'il sera de plus en plus difficile d'obtenir un accès uniforme aux objets multimédia en raison du nombre grandissant de communautés indépendantes de métadonnées. Chaque communauté combine les termes à partir de plusieurs vocabulaires spécifiques, et utilise différentes structures pour organiser les métadonnées. L'interopérabilité des métadonnées devient donc un enjeu crucial.

Ces dix dernières années, une grande variété de techniques d'interopérabilité ont été proposées. Une de ces techniques est l'introduction d'un schéma de métadonnées standardisé (ex : Dublin Core, CIDOC/CRM, MPEG-7, MPEG-21, FGDC, IMS, etc.) [HAU] et de langages de définition de schémas standardisés (ex : le Schéma XML, UML, OWL, le Schéma RDF, etc.) [HAS 10]. Mais pour des raisons stratégiques ou politiques, il est impossible d'adhérer à un standard ou à un langage unique. A ce stade, plusieurs solutions d'intégration des métadonnées ont donc été proposées et sont analysées dans [HAS 10]. Pour la plupart, ces travaux s'intéressent à la création d'une ontologie de haut niveau contenant les informations communes sur les métadonnées à intégrer. Ensuite, un mapping manuel est effectué entre cette dernière et les autres formats de métadonnées [AMI 09].

Les travaux mentionnés ci-dessus ne se sont pas avérés très concluants car le processus d'intégration est effectué manuellement, ce qui entraîne un coût temporel assez important. Par ailleurs, l'intégration doit être mise à jour à chaque fois qu'un nouveau format de métadonnées apparaît. Ainsi, plusieurs techniques d'intégration automatique ont été développées pour faciliter le processus d'intégration. Les techniques de matching jouent un rôle central dans le cadre de ces approches [EUZ 07].

Dans la section suivante, nous discutons quelques travaux de l'état de l'art et soulignons auxquels notre travail apporte des réponses. Ensuite, nous présentons en détail notre solution en insistant sur l'homogénéisation des langages de description et sur les mesures de similarité linguistique et structurelle que nous mettons en oeuvre. Nous analysons les résultats obtenus dans des expérimentations réalisées sur des standards issus du monde multimédia tels que MPEG-7, MPEG-21, DIG35, EXIF, MIX, etc. Nous concluons le travail en rappelant les avancées réalisées par ce travail par rapport à l'état de l'art et nous donnons quelques pistes pour les travaux futurs.

2. L'état de l'art

Étant donné l'existence de l'hétérogénéité des métadonnées aux niveaux des schémas et de leur langages de définition, il est nécessaire de réaliser l'interopérabilité pour chaque niveau. Dans cette section, nous discutons quelques travaux existants dans l'état de l'art, organisés selon le niveau de traitement de l'interopérabilité.

2.1. *L'interopérabilité des langages de description*

Plusieurs langages de description existent dans la littérature (ex: le Schéma XML, UML, OWL, schéma RDF, etc.). Cependant, afin que les systèmes d'information puissent communiquer les uns avec les autres, il doit exister un accord sur la signification des primitives du langage utilisé. Pour cela, une variété de méthodes d'homogénéisation visant la conversion de ces langages hétérogènes vers le même langage ont été proposées. Parmi ces méthodes, on peut citer les travaux présentés dans [GAR 05] et [YAN 07] où les auteurs proposent des règles de conversion de Schémas XML en OWL. Ces règles permettent de capturer une partie de la sémantique implicite des Schémas XML. Dans [FON 97], les auteurs proposent une méthodologie permettant de traduire les schémas et les convertir en base de données relationnelles. La spécification du métamodèle pour la définition des ontologies [LEE 08] offre un ensemble de métamodèles et de correspondances permettant de traduire les métamodèles UML et les langages d'ontologie comme les Schémas RDF et OWL. Dans [MIL 07], les auteurs présentent une approche permettant de générer automatiquement des Schémas RDF à partir de spécifications de Schémas XML. D'autres approches pour la conversion des langages de description ont été également proposées, certaines d'entre elles sont analysées dans [HAS 10].

En raison d'incohérences structurelles et sémantiques substantielles parmi les langages de définition des schémas, les travaux précédemment mentionnés n'ont pas été particulièrement probants car la traduction *intégrale* d'un langage à un autre entraîne la perte d'informations sémantiques et structurelles précieuses.

2.2. *L'interopérabilité des schémas*

Concernant le problème d'hétérogénéité au niveau des schémas, des outils et mécanismes sont nécessaires pour réaliser l'interopérabilité. Ces outils doivent résoudre les problèmes d'hétérogénéité des schémas et aligner les termes entre les métadonnées. Pour ce faire, plusieurs techniques de matching des schémas ont été proposées ces dix dernières années mais, à ce jour, peu de systèmes supportent des schémas issues de différents langages. On trouve notamment les systèmes suivants : Similarity Flooding [MEL 02] (SQL, XML et RDF), Cupid [MAD 01] (SQL et XML), Clio [MIL 01], [POP 02] (SQL et XML) et S-Match [GIU 04] (SQL, XML et Ontologies). Ces approches n'utilisent pas la majorité des informations structurelles et sémantiques (ex : propriétés d'équivalence, caractéristiques de généralisation, etc.). De plus, le gros in-

convénient de ces méthodes est leur façon d'utiliser les informations structurelles. Par exemple, les auteurs de [MAD 01] considèrent que la plus grande partie du contenu des informations est représentée par des feuilles, et que ces feuilles sont soumises à un plus petit nombre de variations entre les schémas que les structures internes. Ainsi, la similarité des intra-nœuds s'appuie sur la similarité de l'ensemble de leurs feuilles. Cela ne s'applique pas toujours puisque l'on peut trouver des concepts équivalents apparaîssant dans des structures totalement différentes, et des concepts totalement indépendants appartenant à des structures isomorphiques. La méthode développée dans [MEL 02], basée sur l'idée de la propagation de la similarité, possède un inconvénient majeur. Le concept de base de l'algorithme est que l'adjacence contribue à la propagation de la similarité. Ainsi, l'algorithme réagira de manière inattendue lorsque les informations d'adjacence ne seront pas préservées. L'approche proposée dans [GIU 04] a uniquement recours à la relation " parent-enfant " pour calculer les contextes de similarité structurelle. Le travail dans [GIU 04] est limité uniquement aux structures arborescentes des schémas et ignore les relation inter-nœuds (propriétés).

2.3. *Discussion*

Suite à l'étude de l'état de l'art, nous avons remarqué qu'il est impossible de trouver un seul langage de description supportant toutes les caractéristiques sémantiques et structurelles des schémas hétérogènes. Nous avons également remarqué que peu de systèmes de matching supportent des schémas provenant des différents langages de description. En plus, ces systèmes sont limités uniquement à deux ou trois langages et présentent certaines limitations, notamment dans l'utilisation de l'information structurelle. Pour contribuer à résoudre ces problèmes, nous proposons une méthodologie de matching sur les deux niveaux. Cette méthodologie vise à obtenir une interopérabilité au niveau des langages et des schémas par un mapping entre les concepts basiques des différents langages. Ces concepts sont transformés ensuite en graphes étiquetés qui jouent le role d'un espace sur lequel toutes les ontologies sont projetables. Nous calculons par la suite la similarité syntaxique, sémantique et structurelle entre les nœuds des deux graphes afin de trouver les mappings entre les différents éléments des schémas hétérogènes. De plus, notre approche résout le problème du calcul structurel relatif aux systèmes de matching mentionnés plus haut, cela grâce à l'utilisation de plusieurs contextes de similarité [LEE 02].

3. L'approche MuMIe (Multi-level Metadata Integration)

Cette section montre les deux principales étapes du système proposé. La première est celle de la construction des graphes qui consiste à obtenir une interopérabilité au niveau des langages de description, obtenue grâce à la transformation de tous les schémas dans un espace commun de représentation. La seconde étape est celle du matching qui sert à trouver les alignements entre métadonnées, par le biais de plusieurs types de mesures de similarité (linguistique, structurelle, etc.).

3.1. *Construction des graphes*

En raison de l'hétérogénéité des langages de définition des schémas, il est impossible de trouver un modèle de représentation unique supportant toutes les caractéristiques des schémas. Dans notre approche, nous modélisons ces schémas en tant que graphes orientés et étiquetés représentant uniquement les concepts basiques des langages de description, qui sont les classes, les attributs et les éléments les propriétés reliant ces entités (Figure 1). Ces concepts sont des les informations de base pour tous les langages descriptifs. Autrement dit, ces entités forment un espace de présentation sur lequel toute ontologie est projetable. Le Tableau 1 montre les concepts basiques des trois langages (XSD, RDFS et OWL). Ce tableau représente un ensemble de règles que nous avons définies et dans lesquelles nous considérons que les éléments, les types complexes et les classes sont représentés par des nœuds de type *nœud normal* au niveau du graphe. Quant aux propriétés inter nœuds, elles sont représentées par des nœuds de type *nœud propriété*.

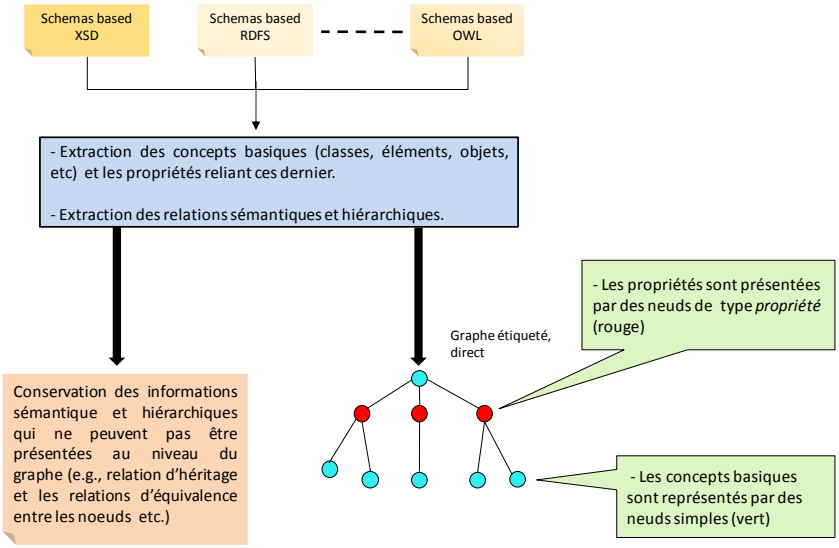

Figure 1. *Illustration de l'étape de la construction du graphe.*

Tableau 1. *Les concepts basiques des langages XSD, RDFS et OWL*

Type du nœud	XSD	RDFS	OWL
Normal	xsd:element, xsd:attribute, xsd:attributeGroup xsd:group	rdfs:class	owl:class
Propriété	le nom du nœud parent + celui du fils	rdf:property	owl:objectProperty

Les caractéristiques sémantiques et hiérarchiques présentées dans le Tableau 2 ne sont pas représentées par des nœuds dans le graphe mais comme des propriétés de

ces derniers. La Section 3.2.2 montre des exemples d'utilisation de ces règles pour détecter les mappings entre les nœuds.

Nous classifions les nœuds en *nœud simple* et *nœud propriété*. Les nœuds simples correspondent aux concepts (ex: rdfs:class, xsd:complexType, xsd:element, etc.), et les nœuds de propriété correspondent aux propriétés reliant les concepts de schémas (ex : rdf:property, owl:ObjectProperty, etc.).

Tableau 2. *Les caractéristiques sémantiques et hiérarchiques*

XSD	xsd:restriction, xsd:abstraction, xsd:extension, xsd:substitutionGroup
RDFS	rdfs:seeAlso, rdfs:isDefinedBy, rdfs:subClassOf
OWL	owl:unionOf, owl:complementOf, owl:intersectionOf, owl:TransitiveProperty, owl:someValuesFrom owl:equivalentClass, owl:disjointWith, rdfs:subClassOf, owl:distinctMembers, owl:differentFrom owl:AllDifferent, owl:sameAs, owl:equivalentProperty

3.2. *Le système de matching*

Une fois que les schémas sont transformés en graphes, nous calculons les similarités entre les nœuds de ces graphes en utilisant plusieurs types d'information linguistiques et structurelles.

3.2.1. *Calcul de la similarité linguistique*

Après avoir appliqué les méthodes classiques de filtrage linguistique (segmentation en unités, lemmatisation, etc), nous calculons la similarité linguistique entre toutes les paires des nœuds dans les deux schémas à aligner. La similarité linguistique est calculée à partir de la similarité des noms et la similarité des commentaires.

a) Similarité des noms

L'objectif de cette étape est de trouver un alignement initial par le calcul de similarité entre les noms des nœuds dans les deux schémas à aligner. Chaque nœud est représenté par un ensemble d'unités linguistiques (*tokens*). Nous commençons d'abord par l'explicitation du sens des unités lexicales par l'utilisation d'une source linguistique. En utilisant WordNet ([FEL 98]), on peut trouver les synonymes d'un terme donné. Cela contribue à résoudre les problèmes d'hétérogénéité survenant lorsque les communautés développant les métadonnées utilisent des termes différents pour décrire la même information. Par exemple, quelques communautés ([GRO 08]) utilisent le terme *type* pour décrire le type d'un contenu multimédia. Certaines autres ([LEE 08]) utilisent le terme *format* ou *genre* pour décrire la même information. Après l'étape d'explicitation, chaque nœud n_i représenté par un ensemble d'unités lexicales M_i aura des ensembles de synonymes *synsets* pour chaque unité lexicale m_i. $M_i^{'}$ est le résultat final qui regroupe tous les *synsets* résultant de l'explicitation de M_i.

$$M_i^{'} = M_i \bigcup \{m_k | \exists m_j \in M_i \bigcap m_k \in synset(m_j)\}$$

La similarité des noms S_{nom} entre deux nœuds (n_1, n_2) appartenant à deux graphes est calculée en utilisant la distance de Jaro-Winkler (JW) ([BIL 03]) entre chaque unité lexicale $m_i \in M_1^{'}$ et toutes les unité lexicales $m_j \in M_2$ (et vice versa) ([LIN 08]). Le choix de cette distance s'appuie sur l'étude comparative faite dans [COH 03]. Le score (MJW) maximum est retenu.

$$MJW(m_i, M_k^{'}) = max_{m_j \in M_k^{'}} JW(m_i, m_j)$$

Finalement, la moyenne des meilleures similarités est calculée :

$$S_{nom}(n_1, n_2) = \frac{\sum_{m_i \in M_1} MJW(m_i, M_2^{'}) + \sum_{m_j \in M_2} MJW(m_j, M_1^{'})}{|M_1| + |M_2|}$$

b) Similarité des commentaires

En raison de la complexité du vocabulaire technique, l'information portée par les noms des nœuds est parfois insuffisante. En conséquence, plusieurs nœuds peuvent avoir une faible valeur de similarité S_{nom} même s'ils décrivent la même information. Afin de faire face à ce manque d'information, les commentaires correspondant aux nœuds (e.g., *rdfs:comment*, *xsd:documentation*, etc.) sont utilisés comme une deuxième source d'information sémantique. Nous appliquons la technique TF/IDF ([RIJ 79]) utilisée dans le domaine de la recherche d'information pour calculer la similarité entres les commentaires. Pour ce faire, tous les commentaires dans les deux schémas à intégrer sont considérés comme des documents. Chaque nœud sera représenté par un vecteur dont les coordonnées sont les résultats de TF/IDF. Par conséquent, la similarité entre deux nœuds est la distance entre les deux vecteurs correspondant à leurs commentaires. Afin d'illustrer le calcul de ces vecteurs, nous supposons que *v* = (w_1, w_2,...., w_P) est le vecteur représentant un nœud donné *n*. *P* = $|U|$ est le nombre des mots distincts dans tous les commentaires appartenant aux deux schémas. Le i_{th} élément w_i de *v*, qui représente le nœud *n* dans un schéma, est calculé comme suit:

$$w_i = tf_i * idf_i \qquad idf_i = \log_2 \frac{N}{b_i} \qquad [1]$$

où tf_i est la fréquence du terme. tf_i représente le nombre de fois que le i_{th} mot dans *U* apparaît dans le commentaire correspondant à n_i. idf_i est la fréquence inverse de document. Elle sert à calculer le logarithme de l'inverse de la proportion de documents du corpus qui contiennent le mot w_i. *N* est le nombre de commentaires dans *U*, dans les deux schémas. b_i est le nombre de commentaires qui contiennent le mot w_i au moins une fois. Comme nous l'avons mentionné précédemment, la similarité entre deux nœuds n_i et n_j est la distance entre les vecteurs correspondants à leurs commentaires v_i et v_j. Cette distance est une similarité cosinus, elle est calculée comme suit:

$$S_{commentaire}(v_i, v_j) = \frac{\sum_{k=1}^{P} w_{ik} w_{jk}}{\sqrt{\sum_{k=1}^{P} (w_{ik})^2 * \sum_{k=1}^{P} (w_{jk})^2}} \qquad [2]$$

Le résultat des calculs effectués précédemment est une matrice de similarité linguistique *lSim*:

$$lSim(n_i, n_j) = \mu_1 * S_{name}(n_i, n_j) + \mu_2 * S_{comment}(n_i, n_j) \quad [3]$$

$\mu_1 + \mu_2 = 1$ et $(\mu_1, \mu_2) \geq 0$

3.2.2. *Utilisation des informations hiérarchiques et sémantiques*

Les informations sémantiques et structurelles retenues à l'étape de construction des graphes sont utilisées par notre système pour détecter spécialement les mappings complexes (n:m). Dans la suite de cette section, nous montrons quelques exemples d'utilisation des ces informations:

– **Relations de généralisation :** La relation de généralisation entre deux types indique que l'un est un sous-ensemble de l'autre (e.g., *xsd:extension*, *rdfs:subClassOf*, *xsd:abstraction*, etc.). Cette information nous aide dans la détection des mappings complexes (n:m). Par exemple, si on considère que *ms:Agent* est un attribut défini dans un schéma qui sera aligné à MPEG-7, cet attribut aura une valeur importante *lSim* avec l'attribut *mpeg7:AgentType*. A ce stade, l'union des deux éléments $mpeg7$:$PersonType$ et $mpeg7$:$OrganizationType$ sera également considérée comme candidate pour le matching avec l'attribut ms:$Agent$ car *mpeg7:OrganizationType* et *mpeg7:PersonType* sont des extensions de type *mpeg7:AgentType*. D'autres informations structurelles sont également utilisées (e.g., *owl:disjointWith*, *owl:unionOf*, etc.).

– **Relations sémantiques :** Si on considère que n_i et n_j sont deux nœuds ayant un niveau de similarité important, et n_k est un autre nœud correspondant à une classe ayant une propriété d'équivalence avec celle correspondant à n_i (e.g., *owl:equivalentProperty*, *owl:equivalentClass*, *xsd:substitutionGroup*, etc.) ([HAU]), cette information nous permet de déduire que n_k est un autre nœud candidat pour n_j. Notre approche utilise aussi d'autres relations dans la détection des matchings complexes (e.g., *owl:sameAs*, *owl:differentFrom*, etc.).

Les mappings obtenus suite à l'analyse linguistique et sémantique sont souvent nombreux et inadéquats à cause des *faux amis* et de la largesse consentie notamment en ce qui concerne l'utilisation des synonymes. Dans notre vision de l'intégration, nous souhaitons élargir les possibilités des mappings au niveau linguistique et éliminer ensuite les faux mappings en appliquant un filtrage qui s'appuie sur des notions structurelles et sémantiques tel que décrit dans la section suivante.

3.2.3. *Calcul de la similarité structurelle*

Nous adoptons un calcul de la similarité structurelle pour éliminer les faux candidats détectés lors du calcul de la similarité linguistique. La similarité structurelle d'un nœud donné représente le contexte dans lequel se trouve ce nœud. Ce contexte est donné par les ancêtres de ce nœud, les fils immédiats et les feuilles [LEE 02]. La similarité des nœuds est obtenue en combinant ces trois similarités.

a) Similarité des ancêtres

Les ancêtres d'un nœud n_i sont définis par le chemin p_i s'étendant de n_i jusqu'à la racine du graphe. Par conséquent, afin de calculer la similarité entre deux contextes correspondant aux nœuds (n_i, n_j), la similarité entre les chemins (p_i, p_j) doit être calculée. Pour ce faire, plusieurs méthodes de similarité existent. Dans ce papier nous utilisons l'approche proposée par [D.C 03]. Ils proposent une méthode flexible permettant de calculer les similarités entre les chemins des nœuds en tenant compte de quatre paramètres: $lcs_n(p_i, p_j)$ est la plus longue sous-séquence commune entre p_i et p_j normalisée par la longueur de p_i. $pos(p_i, p_j)$ considère que le matching idéal entre deux chemins (p_i, p_j) est celui qui commence au premier nœud de p_i sans discontinuité. $gap(p_i, p_j)$ est utilisé pour s'assurer que les occurrences de p_i et p_j soient proches les unes des autres. $ld(p_i, p_j)$ donne des valeurs importantes aux chemins (p_i, p_i) dont la valeur de la longueur est proche. La combinaison des paramètres mentionnés précédemment donne la similarité *ps* entre deux chemins (p_i, p_i):

$$ps(p_i, p_j) = \delta lcn_n(p_i, p_j) + \varphi pos(p_i, p_j) - \theta gap(p_i, p_j) - \lambda ld(p_i, p_j) \quad [4]$$

où δ, φ, θ et λ sont déterminées sur la base des expérimentations effectuées dans ([D.C 03]). δ = 0.75; φ = 0.25; θ=0.25; λ=0.2. Sur la base de ces quatre critères, nous introduisons une nouvelle adaptation et relaxation des contraintes définies dans ([D.C 03]) comme suit:

– Pour les quatre paramètres, la plus longue sous-séquence commune (lcs) est calculée en fonction de la matrice de similarité linguistique *lSim* calculée dans la section précédente. Cela signifie que deux nœuds (n_i, n_j) sont considérés identiques si la valeur de *lSim*(n_i, n_j) est supérieure à un seuil donné e.g. 0.8. C'est-à-dire, la comparaison entre les nœuds tient compte de l'information syntaxique et sémantique au lieu de faire une comparaison classique entre les chaînes de caractères.

– Les paramètres dans ([D.C 03]) ont été définis pour les schémas XML qui sont basés sur une classification taxonomique. Pour cette raison, l'utilisation de ces paramètres dans un graphe étiqueté nécessite une autre relaxation pour le calcul de lcs. Par exemple, si on considère les deux triplets RDF *(Object, ContentOn, URL)* et *(Object, HasReference, URL)*, on peut remarquer que le nom de la propriété dans le premier triplet (*Content*) correspond à *Object*. Par contre, le nom de la propriété dans le deuxième triplet (*Reference*) correspond à *URL*. A ce stade, nous introduisons une nouvelle relaxation en considérant les nœuds de type propriété comme des nœuds qui peuvent être permutés avec leur fils ou parent immédiat. Pour ce faire, chaque chemin est divisé en un ensemble de segments, chaque segment est composé de deux nœuds adjacents, l'un des deux est de type propriété (Figure 2).

Si on considère les deux chemins montrés dans la Figure 2 comme deux contextes, la valeur de *lcs* sera de 5 au lieu de 0 après les deux types de relaxation introduits précédemment. Finalement, la similarité entre deux contextes des ancêtres est la similarité entre les chemins, pondérée par la linguistique similarité des nœuds correspondant (n_i, n_j):

$$ancSim(n_i, n_j) = ps(n_i, n_j) * lSim(n_i, n_j) \quad [5]$$

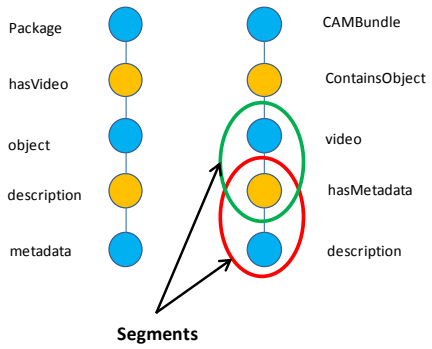

Figure 2. *Exemple de chemin et segment*

b) Similarité des fils immédiats

Afin d'obtenir la similarité des fils immédiats *immSim* entre deux nœuds (n_i, n_j), on compare les deux sous-ensembles des fils immédiats $S = \{s_1, s_2, \cdots, s_n\}$ et $S^{'} = \{s_1^{'}, s_2^{'}, \cdots, s_m^{'}\}$. Les fils immédiats désignent uniquement les nœuds simples. Les nœuds propriétés (reliant deux nœuds simples) sont inclus uniquement dans le calcul de la similarité entre chemins par l'utilisation de la similarité *SimI* entre chaque paire de nœuds appartenant aux deux ensembles, où:

$$SimI(s_i^{'}, s_j) = ps(s_i^{'}, s_j) * lSim(s_i^{'}, s_j) \quad [6]$$

$ps(s_i^{'}, s_j)$ est la similarité entre les deux chemins allant de $(s_i^{'}, s_j)$ à (n_i, n_j). Puis les paires ayant une valeur maximale de similarité sont sélectionnées.

$$MaxSimI(s_i, S^{'}) = max_{s_j^{'} \in S^{'}} SimI(s_i, s_j^{'}) \quad [7]$$

$$MaxSimI(s_i^{'}, S) = max_{s_j \in S} SimI(s_i^{'}, s_j) \quad [8]$$

Finalement, la moyenne des meilleures similarités est prise afin da calculer la valeur de similarité correspondant aux fils *immSim*:

$$immSim(n_i, n_j) = \frac{\sum_{i=1}^{|S|} MaxSimI(s_i, S^{'}) + \sum_{i=1}^{|S^{'}|} MaxSimI(s_i^{'}, S)}{|S'| + |S|} \quad [9]$$

c) Similarité des feuilles

Le contexte des feuilles d'un nœud simple n_i est l'ensemble de feuilles reliées à ce nœud. Si on considère que $l_i \in$ *leaves*(n_i) est un nœud de type feuille, alors la contexte de l_i est le chemin p_i allant de n_i à l_i. A ce stade, le contexte des feuilles est donné par:

$$leafSim(l_i, l_j) = ps((p_i, p_j)) * lSim(l_i, l_j) \quad [10]$$

Afin de mesurer la similarité entre deux feuilles $l_i \in$ *leaves*(n_i) et $l_j \in$ *leaves*(n_j), on calcule la similarité des feuilles *leafSim* entre chaque paire des feuilles dans les

deux ensembles des feuilles. Puis on sélectionne la paire ayant la valeur maximale de similarité. La moyenne des meilleures similarités est prise (Section 3.2.3)

d) Similarité des nœuds

La similarité des nœuds est obtenue par la combinaison des trois similarités décrites précédemment : similarité des ancêtres, similarité des fils immédiats et similarité des feuilles.

$$\textit{nodeSim}(n_i, n_j) = \alpha * \textit{ancSim}(n_i, n_j) + \beta * \textit{immSim}(n_i, n_j) + \gamma * \textit{leafSim}(n_i, n_j) \quad [11]$$

où $\alpha + \beta + \gamma = 1$ et $(\alpha, \beta, \gamma) \geq 0$

Une fois que la similarité structurelle est effectuée, le système retourne pour chaque nœud n_i les K nœuds correspondant aux K-plus grandes valeurs de *nodeSim*. Ces nœuds doivent avoir une valeur de *nodeSim* supérieure à un seuil donné. Dans la mesure où la valeur de K est supérieure à 1, c'est l'utilisateur qui sélectionne parmi les propositions. Le dictionnaire externe est mis à jour selon les retours de l'utilisateur.

4. Expérimentation

Dans cette section, nous relatons les expérimentations que nous avons menées pour valider notre solution. Nous commençons tout d'abord pour la description des métadonnées utilisées. Ensuite, nous étudions l'influence des paramètres (α, β et γ) apparaissant dans la mesure de similarité structurelle. L'introduction de ces seuils offre une certaine souplesse à notre solution. Cependant, il contraint l'utilisateur à fixer lui même ces seuils. Afin de pouvoir guider le choix des valeurs adaptées aux divers contextes d'utilisation, nous étudions comment la modification de ces paramètres influe sur les résultats de matching en considérant des standards de métadonnées variés (MPEG-7, DIG35 et EXIF). Finalement, la dernière expérimentation vise à mesurer la qualité du matching en utilisant plusieurs standards de métadonnées hétérogènes.

4.1. *Caractéristiques des métadonnées utilisées*

L'approche MuMIe a été testée sur plusieurs standards de métadonnées (MPEG-7, MPEG-21, EXIF, MIX, DIG35, PeCMan, DIG35 Ontology et CAM4Home) [HAU] [BIL 10]. Ces standards présentent une hétérogénéité significative aux deux niveaux (Tableau 3). Afin de pouvoir comparer nos résultats avec Cupid [MAD 01], nous avons sélectionné dans un premier temps uniquement les standards dont les schémas ont été définis avec le Schéma XML. Pour ce faire, nous avons calculé le matching entre TV-Anytime et tous les standards dont les schémas ont été définis avec le Schéma XML. Ensuite, afin que nos résultats soient comparables avec SF [MEL 02], nous avons pris en compte tous les schémas du Tableau 3. Nous avons calculé le matching entre le métamodèle CAM4Home[1] présenté dans [BIL 10] et le reste des schémas.

1. http://www.cam4home-itea.org/

La définition du métamodèle CAM4home est faite avec RDF Schéma, le choix de CAM4Home comme un schéma médiateur est fait car ce dernier contient plusieurs informations communes avec le reste des standards.

Tableau 3. *Caractéristiques des métadonnées utilisées*

Standard de métadonnées	Profondeur	Nombre de nœuds	Langage
MIX	7	41	XSD
DIG35	9	57	XSD
EXIF	6	63	XSD
MPEG-7	13	115	XSD
MPEG-21	11	143	XSD
PeCMan Ontology	7	46	OWL
DIG35 Ontology	9	57	OWL
CAM4Home	17	153	RDFS
TV-Anytime	8	71	XSD

4.2. *Paramétrage*

Afin de connaître l'influence de chaque contexte sur le matching, nous avons effectué plusieurs expérimentations en utilisant des combinaisons différentes des paramètres α (le poids des ancêtres), β (le poids des fils), γ (le poids des feuilles). La Figure 3 montre les résultats de matching en termes de *F-Mesure* [MAD 01] entre les standards (MPEG-7, DIG35) et (DIG35, EXIF) respectivement. Nos expérimentations ont montré qu'une partie importante de l'information structurelle est contenue dans le contexte des nœuds parents où les valeurs implorantes de F-mesure sont localisées pour $\alpha \in$ [0.45 0.6]; cela explique l'intérêt de quelques stratégies de matching qui considèrent que le contexte d'un nœud dépend uniquement de ces parents [BOU 03], [MIL 01], [STU 01]. En outre, les expérimentations ont montré que les contextes des fils immédiats et feuilles ont un rôle important dans le processus de matching ($\beta \in$ [0.1 0.15] et $\gamma \in$ [0.25 0.4]). Cependant, la sélection des paramètres α, β et γ pour chaque paire de nœuds, en fonction de leur position dans le graphe, pourra sans doute augmenter la qualité de matching. Cela fera partie de nos futures travaux de recherche.

4.3. *Qualité de matching*

Notre solution a été proposée en considérant les trois critères d'évaluation (*Précision*, *Rappel* et *F-mesure*) utilisés dans [MAD 01] et [MEL 02]. Les paramètres de l'équation (11) ont été choisis suite aux expériences effectuées dans la Section 4.2 ($\alpha = 0.55, \beta = 0.15, \gamma = 0.30$). Les résultats du matching entre TV-Anytime et les standards définis avec le Schéma XML sont montrés dans le Tableau 4. Le Tableau 5 montre les résultats du matching entre CAM4Home et le reste des standards.

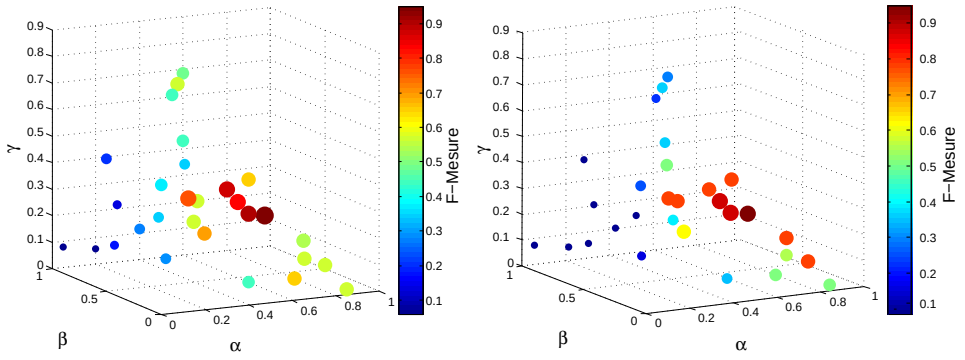

Figure 3. *Influence de α, β et γ sur la qualité du matching*

Tableau 4. *Résultats du matching entre TV-Anytime et les standards basés sur Schéma XML*

standards de métadonnées	Precision	Recall	F-measure
MIX	50%	55%	52%
EXIF	62%	65%	63%
MPEG-7	75%	70%	72%
MPEG-21	43%	47%	45%
DIG35	77%	82%	79%

Tableau 5. *Résultats du matching entre CAM4Home et le reste des standards*

Standards de métadonnées	Precision	Recall	F-measure
MIX	92%	89%	90%
DIG35	87%	90%	88%
EXIF	90%	81%	85%
MPEG-7	77%	64%	70%
MPEG-21	72%	60%	65%
PeCMan Ontology	94%	89%	91%
DIG35 Ontology	85%	90%	87%
TV-Anytime	77%	72%	75%

4.4. *Etude comparative*

Nous avons comparé les performances de notre solution avec celles de Cupid et SF. La comparaison a été faite avec ces deux approches car elles sont basées sur le matching des schémas, elles sont multi-niveaux (SQL et Schéma XML pour Cupid,

SQL, XML et RDF pour SF) et elles utilisent une similarité linguistique et structurelles dans leur processus de matching. Les résultats de cette étude comparative en terme de F-Mesure sont montrés dans les Figures 4 et 5.

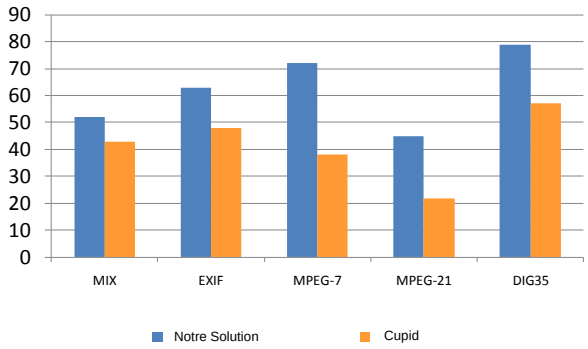

Figure 4. *Etude comparative avec Cupid (F-Mesure).*

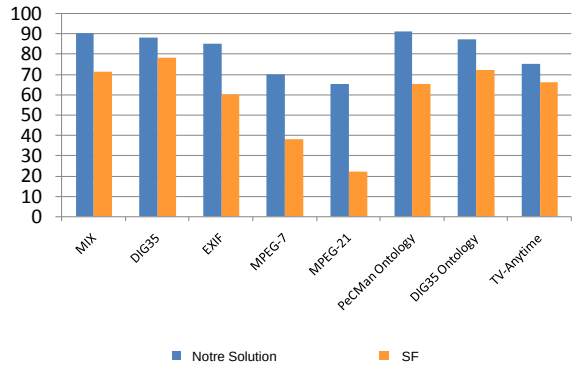

Figure 5. *Etude comparative avec SF (F-Mesure).*

4.5. *Discussion sur la performance de l'approche proposée*

Les résultats expérimentaux ont montré que, sur toutes les standards de métadonnées utilisés, notre solution était meilleure en termes F-Measure. Cela est principalement dû à l'utilisation de l'information linguistique qui est une combinaison des scores obtenus à partir des des commentaires et des noms des nœuds, à l'exploitation des informations hiérarchiques et sémantiques (Section 3.2.2) permettant de détecter d'autres mappings même si les nœuds correspondants ne sont pas linguistiquement identiques. Ces informations servent également dans la détection des matching complexes. Nous avons utilisé dans notre approche trois contextes pour le calcul de la

similarité structurelle, ce qui donne une flexibilité importante avec une meilleure exploitation de l'information structurelle. Alors que Cupid et SF utilisent uniquement un seul contexte de similarité (similarité des feuilles pour Cupid et fils immédiats pour SF).

5. Conclusion

Nous avons présenté dans cet article une méthodologie de matching multi-niveaux. Cette méthodologie, déployée en deux niveaux (langage de description et schéma), prend en considération les schémas de métadonnées quel que soit leur langage de description (Schéma XML, Schéma RDF, OWL), les transforme en graphes étiquetés tout en conservant les informations structurelles et sémantiques, cherche les similarités entre les nœuds en utilisant plusieurs types d'information syntaxique, sémantique et structurelle. Notre expérimentation a montré que la combinaison de ces informations augmente de manière significative la détection des matchings corrects entre les métadonnées. Comme perspective à notre travail, nous prévoyons d'améliorer notre méthodologie par une meilleure exploitation de l'information structurelle. Nous explorons principalement l'utilisation des relations d'adjacence entre les nœuds pour une meilleure détection des mappings.

6. Bibliographie

[AMI 09] AMIR S., « Un système d'intégration de métadonnées dédiées au multimédia », *INFORSID*, 2009, p. 491-492.

[BIL 03] BILENKO M., MOONEY R. J., COHEN W. W., RAVIKUMAR P. D., FIENBERG S. E., « Adaptive Name Matching in Information Integration », *IEEE Intelligent Systems*, vol. 18, n° 5, 2003, p. 16-23.

[BIL 10] BILASCO I. M., AMIR S., BLANDIN P., DJERABA C., LAITAKARI J., MARTINET J., MARTÍNEZ-GRACÍA E., PAKKALA D., RAUTIAINEN M., YLIANTTILA M., ZHOU J., « Semantics for intelligent delivery of multimedia content », *SAC*, 2010, p. 1366-1372.

[BOU 03] BOUQUET P., SERAFINI L., ZANOBINI S., « Semantic Coordination: A New Approach and an Application », *International Semantic Web Conference*, 2003, p. 130-145.

[COH 03] COHEN W. W., RAVIKUMAR P. D., FIENBERG S. E., « A Comparison of String Distance Metrics for Name-Matching Tasks », *IIWeb*, 2003, p. 73-78.

[D.C 03] D.CARMEL, MAAREK Y. S., MANDELBROD M., MASS Y., SOFFER A., « Searching XML documents via XML fragments », *SIGIR*, 2003, p. 151-158.

[EUZ 07] EUZENAT J., SHVAIKO P., *Ontology matching*, Springer-Verlag, Heidelberg (DE), 2007.

[FEL 98] FELLBAUM C., Ed., *WordNet: An Electronic Lexical Database*, MIT Press, Cambridge, MA, 1998.

[FON 97] FONG J., « Converting Relational to Object-Oriented Databases », *SIGMOD Record*, vol. 26, n° 1, 1997, p. 53-58.

[GAR 05] GARCÍA R., CELMA O., « Semantic Integration and Retrieval of Multimedia Metadata », *2nd European Workshop on the Integration of Knowledge, Semantic and Digital Media*, 2005, p. 69-80.

[GIU 04] GIUNCHIGLIA F., SHVAIKO P., YATSKEVICH M., « S-Match: an Algorithm and an Implementation of Semantic Matching », *ESWS*, 2004, p. 61-75.

[GRO 08] WORKING GROUP X., « Extensible Metadata Platform Specification », XMPSpecificationPart2.pdf, février 2008.

[HAS 10] HASLHOFER B., KLAS W., « A survey of techniques for achieving metadata interoperability », *ACM Comput. Surv.*, vol. 42, n° 2, 2010.

[HAU] HAUSENBLAS M., « Multimedia Vocabularies on the Semantic Web », `http://www.w3.org/2005/Incubator/mmsem/`.

[LEE 02] LEE M.-L., YANG L. H., HSU W., YANG X., « XClust: clustering XML schemas for effective integration », *CIKM*, 2002, p. 292-299.

[LEE 08] LEE W., BÜRGER T., SASAKI F., MALAISÉ V., « Use Cases and Requirements for Ontology and API for Media Object », juillet 2008.

[LIN 08] LIN F., SANDKUHL K., « A Survey of Exploiting WordNet in Ontology Matching », *IFIP AI*, 2008, p. 341-350.

[MAD 01] MADHAVAN J., BERNSTEIN P. A., RAHM E., « Generic Schema Matching with Cupid », *VLDB*, 2001, p. 49-58.

[MEL 02] MELNIK S., GARCIA-MOLINA H., RAHM E., « Similarity Flooding: A Versatile Graph Matching Algorithm and Its Application to Schema Matching », *ICDE*, 2002, p. 117-128.

[MIL 01] MILLER R. J., HERNÁNDEZ M. A., HAAS L. M., YAN L.-L., HO C. T. H., FAGIN R., POPA L., « The Clio Project: Managing Heterogeneity », *SIGMOD Record*, vol. 30, n° 1, 2001, p. 78-83.

[MIL 07] MILETIC I., VUJASINOVIC M., IVEZIC N., MARJANOVIC Z., « Enabling Semantic Mediation for Business Applications: XML-RDF, RDF-XML and XSD-RDFS transformations », *IESA*, 2007, p. 483-494.

[POP 02] POPA L., VELEGRAKIS Y., MILLER R. J., HERNÁNDEZ M. A., FAGIN R., « Translating Web Data », *VLDB*, 2002, p. 598-609.

[RIJ 79] VAN RIJSBERGEN C. J., *Information Retrieval*, Butterworths, London, 2 édition, 1979.

[STU 01] STUMME G., MAEDCHE A., « FCA-MERGE: Bottom-Up Merging of Ontologies », *IJCAI*, 2001, p. 225-234.

[YAN 07] YANG K., STEELE R., LO A., « An Ontology for XML Schema to Ontology Mapping Representation », *iiWAS*, 2007, p. 101-111.