Réécriture sémantique de bio-requêtes[1] centrée sur les préférences de l'utilisateur.

Kunalè KUDAGBA [1], Omar El BEQQALI [2], Hassan BADIR [3]

[1,2] *Equipe de recherches GR2SMI*
Université Sidi Mohammed Ben Abdallah
Faculté des Sciences de Fès
B P : 1796, Fès – Atlas - MAROC
[1] kkunale@ fsdmfes.ac.ma , [2] oelbeqqali@fsdmfes.ac.ma

[3] *Ecole Nationale des Sciences Appliquées, ENSA - Tanger*
BP: 1818 – Tanger – MAROC
hbadir@ensat.ac.ma

RÉSUMÉ *Dans cet article, nous proposons une caractérisation du problème de réécriture de requêtes sur la base des correspondances sémantiques comme étant un Hypergraphe. Par conséquent, la génération des réécritures candidates est formulée comme la découverte des Transversaux minimaux associés à l'hypergraphe. Nous exploitons et adaptons les algorithmes disponibles en Théorie des Graphes pour retrouver toutes les réécritures possibles. Ensuite, dans des travaux futurs, des critères pertinents nous permettront de déterminer les réécritures qualitatives et optimales, en accord avec les préférences de l'utilisateur et les performances des sources de données protéomiques.*

ABSTRACT. *In this paper, we propose a characterization of query rewriting problem using semantic mappings as an associated hypergraph. Hence, the generation of candidates rewritings can be formulated as the discovery of minimals Transversals of a given hypergraph. We exploit and adapt algorithms available in Hypergraph Theory to find all candidates rewritings from a query answering problem. Then, in future work, some relevant criteria should help to determine optimal and qualitative rewritings, according to user needs, and proteomics sources performances.*

MOTS-CLÉS : *Protéomique, Ontologie, XML, Arbres, Web Sémantique, Réécriture de requêtes, Transversaux minimaux.*

KEYWORDS: *Proteomics, Ontology, XML, Trees, Semantic Web, Query Rewriting, minimal Transversals.*

[1] Requêtes globales portant sur des sources de données biologiques, protéomiques en particulier.

1. Introduction

1.1. *Contexte général et Objectifs*

Les récentes avancées technologiques et les multiples projets de séquençage du génome [2, 8] produisent en permanence des quantités colossales de données biologiques, accessibles sur le web [1]. Elles renferment des informations sur les gènes, les protéines, les processus métaboliques, les structures 3D, les propriétés physico - chimiques, les maladies,…

L'exploitation biomédicale et pharmaceutique [8] de ces informations nécessite souvent d'effectuer des requêtes complexes et multiples à travers des sources de données autonomes, hétérogènes et distribuées. Les différences rencontrées dans les formats de données, les problèmes de nomenclature, la diversité des protocoles de rapatriement et la diversité des thématiques abordées constituent des obstacles réels pour l'intégration des données recueillies. Dans le contexte du Web, les *médiateurs* peuvent jouer un rôle d'interface de requêtes entre le biologiste et les sources de données cibles. Ils donnent à l'utilisateur l'illusion d'interroger un système homogène et centralisé en lui évitant d'avoir à trouver les sources de données pertinentes pour sa requête, de les interroger une à une dans leur langage spécifique, selon leur schéma particulier et de combiner lui-même les informations obtenues [35].

La *réécriture de requêtes* [12] constitue une tâche délicate dans le processus d'intégration de données. En effet, pour un usager (le biologiste dans notre cas) posant ses requêtes globales en fonction des termes spécifiés dans le Schéma Global (celui du médiateur), une reformulation quasi-équivalente dans les termes utilisés dans les schémas locaux est nécessaire. En général, plusieurs sources de données peuvent répondre à la même portion de la requête globale, nous proposons un cadre formalisé et sémantique (logiques de représentation des connaissances et algorithmes de graphes) pour la génération des réécritures candidates. Par conséquent, dans des travaux futurs, nous proposerons une sélection et un ordonnancement de ces réécritures suivant des critères déterminés par les préférences de l'utilisateur et la qualité de service des sources protéomiques.

Nous associerons au problème de réécriture un Hypergraphe H (V, E) et par conséquent la génération des réécritures candidates se résumerait en la découverte des Transversals minimaux dans un Hypergraphe H. Le but est d'utiliser les résultats connus et solides en Théorie des Graphes pour donner des bases mathématiques fortes à notre Travail. L'algorithme proposé exploite les connaissances protéomiques à savoir une Ontologie de domaine (dénommé O'proteomics) et les Correspondances sémantiques (Mappings en anglais) existant entre les concepts et relations inter-conceptuelles de l'Ontologie et les éléments/attributs spécifiés dans les Schémas XML [19] locaux. Les Mappings sont exprimés suivant une approche L.A.V[2]. Dans une telle situation, les concepts locaux des sources ciblées sont exprimés en fonction des concepts du schéma global, l'ontologie de domaine.

[2] L.A.V: Local As View.

1.2. *Exemple de motivation*

Considérons à titre d'illustration, deux sources de données protéomiques disponibles au format XML, et décrites suivant des Schémas XML [19] hétérogènes :

- **Document XML 1**

```
<PROTEIN_SET>
<PROTEIN>
 <ACCESSION>P26954</ACCESSION>
 <ENTRY_NAME>IL3B_MOUSE</ENTRY_NAME>
<PROTEIN_NAME>Interleukin-3receptorclass II beta chain [Precursor]
</PROTEIN_NAME>
 <GENE_NAME>CSF2RB2</GENE_NAME>
  <ORGANISM taxonomy_id="10090"> Mus musculus</ORGANISM>
     <COMMENT>SUBCELLULAR LOCATION: Type I membrane protein.
</COMMENT>
 <COMMENT>similarity: belongs to the cytokine family of receptors </COMMENT>
 <KEYWORD> Receptor </KEYWORD>
 <KEYWORD> Glycoprotein </KEYWORD>
 </PROTEIN>
 </PROTEIN_SET>
```

- **Document XML 2**

```
<PROTEIN_BASE>
<PROTEIN  ACC_NUMBER="P26954 >
<ENTRY>IL3B_MOUSE</ENTRY>
 <PROTEIN_NAME>Interleukin-3receptor class II beta chain
Precursor]</PROTEIN_NAME>
 <GENE>CSF2RB2</GENE>
 <ORGANISM >
<TAX_ID >10090</TAX_ID >
<NAME>  Mus musculus</NAME>
</ORGANISM>
<COMMENT>SUBCELLULAR LOCATION: Type I membrane  protein. </COMMENT>
<KEYWORDS> Receptor  </KEYWORDS>
 <KEYWORDS> Signal </KEYWORDS>
 </PROTEIN>….
  </PROTEIN_BASE>
```

Code 1. *Sources de données protéomiques hétérogènes*

On suppose ensuite que pour ses activités de recherches, un protéomicien veut extraire via le Web, toutes les informations relatives à la protéine appelée **Interleukin-3 receptor class II beta chain [Precursor]**, présente chez une certaine espèce de Souris appelée **Mus musculus** (Documents XML 1 et 2).

On remarque aisément que pour désigner le gène codant pour cette protéine, la source 1 utilise un Elément GENE_NAME alors que la source 2 utilise un Elément

GENE. Dans la source 1, l'identification de l'organisme exprimant la protéine est un attribut taxonomy_id alors dans la source 2, il s'agit d'un Elément imbriqué TAX_ID.

L'intégration et l'interrogation de ces données protéomiques sont donc confrontées à une série de problèmes tels que :

- le caractère hétérogène et autonome des sources de données protéomiques qui engendre souvent des conflits, d'ordre syntaxique et sémantique.
- lors de la formulation de sa requête via à un médiateur Web, les mécanismes actuels de réponse aux requêtes ne prennent pas en compte les préférences du protéomicien.

Les mécanismes actuels de réponse aux requêtes peuvent être améliorés si les préférences de l'utilisateur, les performances et la fiabilité des sources de données sont modélisées et associées au processus de réécriture. L'objectif de ce travail de recherches est de proposer une réécriture de requêtes guidée par la sémantique et centrée sur les préférences du protéomicien.

La seconde section donne un bref état de l'art concernant les systèmes de médiation de données semi-structurées sur le Web. La section 3 présente la représentation des connaissances protéomiques. La section 4 décrit la formalisation du problème de réécriture, l'algorithme proposé et son évaluation. Enfin, dans la section 5, la conclusion rappelle essentiellement les apports de ce travail de recherche et projette les travaux futurs à court terme concernant notre problématique.

2. Etat de l'art sur les médiateurs Web et la réécriture de requêtes

L'un des premiers systèmes basés sur une approche L.A.V et consacrés à l'intégration XML est le Framework AGORA [12], qui réalise en fait une extension du modèle relationnel. Bien qu'il offre une vue XML pour les données relationnelles et semi-structurées, cette vue est traduite en un schéma relationnel générique, les ressources XML sont décrites comme des vues relationnelles portant sur le schéma global et les expressions XQuery sont traduites en des requêtes SQL standard, qui sont ensuite décomposées et évaluées.

Information Manifold [27] utilise également une approche L.A.V. Dans ce système, le Schéma Global est un Schéma relationnel, et les Logiques de description sont utilisées pour représenter les hiérarchies de classes. Les sources sont exprimées comme des vues relationnelles à travers le schéma. La réécriture de requêtes se base sur l'algorithme de Bucket et reformule une requête conjonctive exprimée dans les termes du Schéma global en exploitant les vues-sources. Il examine indépendamment chacune des sous-requêtes et essaye de retrouver les réécritures possibles mais en perdant certaines, compte tenu des sous-requêtes isolées.

La médiation sémantique dans C-Web [22] est basée sur des thesauri. Amann et ses collègues discutent d'un système d'intégration au sein duquel les mappings de nature L.A.V sont établis avec une Ontologie supportant l'héritage et les rôles.

Le prototype résultant STyX [28] exploite une ontologie comme Schéma Global et traduit des requêtes exprimées en OQL en expressions XQuery sur des sources de données XML hétérogènes. STyX réalise des mappings entre les expressions XQuery et les concepts de l'ontologie.

Lehti et ses collègues [21] ont pour leur part réalisé des mappings entre les constructeurs de XML Schema et les concepts d'une Ontologie de domaine OWL. La principale différence avec le système STyX demeure l'expressivité des mappings sémantiques. Bien que leur approche ne soit pas flexible ni puissant du fait de l'utilisation de mappings XPath, il est en principe possible de détecter des incohérences présentes dans les mappings avec l'aide d'un moteur d'inférence.

Le problème de réécriture de requêtes a longtemps été étudié et un large survol portant sur les données structurées été proposé par Alon Halevy [15, 41]. Le lecteur pourra également se référer aux papiers [42,17] qui présentent exclusivement la réécriture de requêtes portant sur les données semi-structurées.

Benattallah [23] et ses associés ont proposé un algorithme de réécriture BestQRC qui permet de déterminer les meilleures réécritures suivant des critères précisés par l'internaute. Ils ont construit un Framework WS-CatalogNet scalable et guidé par la sémantique pour le partage et la sélection de Catalogues Commerciaux sur le Web par des Communautés d'affaires diverses et relatives à l'hôtellerie, la location de voitures, les réservations de vols, ... WS-CatalogNet repose également sur une infrastructure P2P dans laquelle la requête de l'utlisateur est décomposée entre une partie devant s'exécuter sur le pair questionné et l'autre devant être traitée sur des pairs annexes. Nous en proposons une réadaptation au contexte de l'intégration de sources XML de données protéomiques sur le Web.

3. Représentation des connaissances protéomiques

Devant le caractère hétérogène des sources de données, l'autre difficulté à surmonter dans le processus d'intégration de données demeure le choix des formalismes et langages devant servir à spécifier les schémas locaux des sources de données, les mappings, le schéma global du médiateur, les requêtes posées au médiateur et les requêtes d'évaluation sur les sources.

Les sources de données protéomiques ciblées sont des Documents XML instanciant leurs Schémas XML respectifs. XML [16] est de nos jours la recommandation W3C[3] pour l'échange de données sur la Toile. Les langages inspirés du vocabulaire XML proposés et utilisés dans le domaine biologique sont nombreux

[3] World Wide Web Consortium

et variés. Nous citerons à titre d'exemples les langages CellML[4] pour représenter les connaissances relatives à la cellule et SMBL[5] pour la représentation des processus biochimiques.

3.1. *Modèle d'abstraction de documents XML: Trees - Model*

Les Schémas XML [19] sont plus appropriés que les DTDs (Document Type Definition) pour exprimer la syntaxe, la structure, les contraintes de cardinalité et de typage exigées par les données protéomiques.

Les Schémas XML constituent eux-mêmes des documents particuliers respectant la syntaxe XML. Des modèles d'abstraction variés ont été proposés pour représenter ou formaliser la structure des documents XML. Le W3C a proposé un modèle de données générique dénommé le Modèle Objet de Document (Document Object Model – DOM en anglais) [18] qui est doté d'une structure de données (les arbres) et d'un ensemble de fonctions (API, Application Programming Interface) dédiées à leur manipulation.

Dans le cadre du projet Xylème [25], une abstraction sur la base des arbres a été proposée pour la construire des schémas globaux matérialisés à partir de schémas locaux spécifiés sous forme de DTDs.

3.2. *Descriptions logiques de chemins et d'arbres XML*

3.2.1. *XML et les langages logiques*

Formaliser une connaissance consiste à la rendre compréhensible par la machine. La sémantique joue alors un rôle très important dans ce processus d'où l'usage de formalismes et de langages logiques. Les langages déclaratifs et les formalismes logiques ont été utilisés pour intégrer les données semi-structurées depuis les premières tentatives, comme à travers MSL/WSL dans le Projet Tsimmis [32], STRUDEL [33], ou F-Logic dans le Projet FLORID [34]. Dans [20], Gotlob et associés ont étudié la complexité et le pouvoir expressif des requêtes conjonctives à travers les arbres. Leur utilisation en Intégration de données n'est pas récente et a porté des fruits. Deutsch et Tannen [13] ont exprimé des requêtes XQuery sous forme de requêtes conjonctives sur des arbres dans un processus d'intégration de données.

Dans [10, 11] Torsten Schlieder a développé un langage ApproXQL, qui exploite entre autres des opérateurs logiques pour formaliser des requêtes riches et beaucoup plus expressives. Ces requêtes, exprimées sous forme conjonctive, peuvent

[4] Cellular Markup Language, www.cellml.org/public/about/what_is_cellml.html
[5] Systems Biology Markup Language, http://smbl.org/index.psp

également s'interpréter comme des arbres. Wolfang May [29] propose et utilise XPathlog, une extension logique de XPath avec des règles Datalog pour intégrer des sources XML dont les contenus sont complémentaires ou chevauchants sur le Web. En définissant une structure de données appelée X-Structure, l'auteur propose des stratégies pour détecter ces chevauchements et appliquer des opérations appropriées dans un Framework d'intégration semi-matérialisé [31]. Les mises à jour de la base de données interne sont assurées grâce à une spécification déclarative exploitant la syntaxe et la sémantique du XPath étendu. XPathlog est implémenté à travers le Framework LoPiX [30]. Mary Fernandez et ses associés [39] proposent une algèbre basée sur les Types et qui capture l'essence du XML Schema sans pertes d'informations. L'algèbre proposée offre également un cadre formel de mise en correspondance avec le modèle de données des requêtes XML. L'implémentation est réalisée en OCaml et peut détecter les erreurs sur les types et les schémas. Nous portons un nouveau regard sur la formalisation logique des arbres en nous basant sur la logique des $\psi - termes$ [36].

3.2.2. Vers un langage de représentation inspiré des $\psi - termes$

Afin de fournir une formalisation sémantique, nécessaire pour une caractérisation rigoureuse et précise des requêtes globales, nous proposons un langage de description de structures arborescentes et hiérarchisées, basé sur le formalisme des $\psi - termes$ .Les $\psi - termes$ [36] sont issus des O.S.F (Order Sorted Features) ou Termes à Traits Ordonnés en français, qui fournissent une représentation adéquate des enregistrements flexibles. Ce sont des structures typées, dotées d'attributs, qui peuvent être imbriquées, et qui sont ordonnées grâce à un ordre de sous-sortes. Les définitions de sortes correspondent à des déclarations de classes, imposant ainsi des contraintes sur la structure des Objets. Ces contraintes consistent en sortes de variables et des équations entre les chemins d'accès de Traits. Formellement, les définitions de sortes peuvent être vues comme des axiomes [37]. Les $\psi - termes$ sont un formalisme comparable aux logiques de description [14], mais conçus comme une extension de Prolog. Ils sont également comparables aux graphes conceptuels en ce sens qu'ils constituent des graphes enracinés.

Nous pensons que les structures arborescentes peuvent être vues comme des $\psi - termes$ et nous les utilisons ici cette représentation pour décrire les Schémas XML des sources de données protéomiques.

Dans la banque de données protéomiques d'origine Suisse dénommée Swissprot[6], les protéines sont décrites comme des objets biologiques ayant un numéro d'accession, une entrée, un nom avec des synonymes, un nom de gène codant, une taille, et un organisme. La structure primaire des protéines est une séquence ordonnée d'acides aminés, conformément à l'information prescrite dans le gène codant. Les protéines sont donc assemblées à partir des acides aminés qui présentent

[6] http://www.swiwwprot.ch

des caractéristiques telles que leur nom, leur codage, leur masse molaire, leur point isoélectrique, et leur constante de dissociation.

```
< XSD: GROUP NAME="PROTEIN_SET">
<XSD: ELEMENT NAME="PROTEINE">
 <XSD: COMPLEXTYPE >
      <XSD: ELEMENT NAME="ACCESSION" TYPE="XSD:STRING">
      <XSD: ELEMENT NAME="ENTRYNAME" TYPE="XSD:STRING">
      <XSD: ELEMENT NAME="NAME" TYPE="XSD:STRING">
      <XSD: ELEMENT NAME="SYNONYME TYPE="XSD:STRING" MINOCCURS="0"
                                                    MAXOCCURS="UNBOUNDED" >
      <XSD: ELEMENT NAME="GENENAME" TYPE="XSD:STRING">
      <XSD: ELEMENT NAME="LENGTH" TYPE="XSD:INTEGER">
     <XSD: ELEMENT NAME="ORGANISM" TYPE="XSD:STRING">
    < XSD: GROUP REF="AMINOACIDS" MINOCCURS="0"  MAXOCCURS="UNBOUNDED" / >
 </XSD: COMPLEXTYPE >
</XSD: ELEMENT>
</XSD: GROUP>
…
< XSD: GROUP NAME="AMINOACIDS">
<XSD: ELEMENT NAME="AMINOACID">
 <XSD: COMPLEXTYPE >
      <XSD: ELEMENT NAME="AMINONAME" TYPE="XSD:STRING">
      <XSD: ELEMENT NAME="CODONE" TYPE="XSD:STRING">
      <XSD: ELEMENT NAME="CODTHREE" TYPE="XSD:STRING">
      <XSD: ELEMENT NAME="MASSE" TYPE="XSD:INTEGER">
     <XSD: ELEMENT NAME="POINTISOELEC" TYPE="XSD:STRING">
      <XSD: ELEMENT NAME="CSTENH2" TYPE="XSD:INTEGER">
      <XSD: ELEMENT NAME="CSTERAD" TYPE="XSD:INTEGER">
      <XSD: ELEMENT NAME="CSTECOOH" TYPE="XSD:INTEGER">
 </XSD: COMPLEXTYPE >
</XSD: ELEMENT>
</XSD: GROUP>
```

Code 2. *Schéma XML inspiré de la source de données Swissprot/UniprotKB.*

Dans le **Code 3**, nous construisons un $\psi - terme$ associé au schéma de description XML d'un acide aminé (Elément de type complexe : AMINOACID):

```
A : aminoacide ( aminoname => String,
                 codage( codone => String,
                         codethree => String ),
                 masse => Integer;
                 pointisoelec => Integer;
                 constdissoc( cstenh2 => integer,
                              csterad => integer,
                              cstecooh => integer),
).
```

Code 3. *Expression d'arbres XML à l'aide de* $\psi - trees$ **.**

La norme XML Schema [19] étant très expressive, nous tiendrons compte des spécificités liées à la déclaration des Séquences (Element Sequences), des Choix (Element Choices), … et à l'expression des cardinalités. Nous donnerons leurs interprétations syntaxique et sémantique et montrerons que les principales règles de subsomption et de généralisation des Termes OSF sont conservées et restent toujours valides. Ainsi, le **Code 4** montre la translation en $\psi - trees$ des cardinalités minimales et maximales portant sur l'Elément SYNONYME :

$$
\begin{aligned}
&P : proteine(\quad ... \\
&\qquad\qquad synonyme => String[0,*], \\
&\qquad\qquad ...).
\end{aligned}
$$

Code 4. *Expression des cardinalités maximales et minimales à l'aide de* $\psi - trees$ **.**

Dans notre approche, la requête de l'utilisateur et les schémas de description des sources de données sont d'abord abstraites et formalisées comme des arbres étiquetés et ordonnés (Trees-Model). Leur spécification est réalisée grâce au langage de description que nous venons de définir : les $\psi - trees$. En effet, toutes les connaissances de notre domaine, à savoir l'ontologie de domaine, les correspondances sémantiques (mappings, en anglais) et la requête globale, seront toutes représentées et formalisées selon le modèle Trees-Model puis spécifiées grâce au langage des $\psi - trees$.

4. Réécriture sémantique de bio-requêtes

La réécriture de requêtes [35] consiste à reformuler une requête globale exprimée dans les termes du Schéma Global en fonction des termes utilisés dans les Schémas locaux. Elle doit également permettre une décomposition de la requête globale en une combinaison de requêtes locales, quand une seule source de données ne permet pas de répondre complètement à la requête réécrite.

La réécriture de requêtes constitue également une stratégie pour uniformiser le langage de spécification des requêtes globales au langage d'évaluation de requêtes adaptées aux sources de données. La réécriture suppose de réaliser un choix judicieux sur le formalisme de représentation des connaissances (Graphes conceptuels, Logiques de description, Psi - Termes,…) afin de favoriser le passage à l'échelle du Web et la décomposition de la requête de l'utilisateur en une conjonction de sous-requêtes. Etant donné que les sources de données sont conçues indépendamment, les termes utilisés pour décrire les concepts de base (au sein des modèles de données en présence) sont fréquemment différents.

4.1. *Formalisation du problème de réécriture*

Une initiative primordiale à prendre consiste à épurer la requête globale en déterminant la sous-requête qui n'est pas résolue par l'infrastructure de sources protéomiques. Etant donné une requête globale Q et le couple $Sch'O$, notre démarche de réécriture consiste à déterminer deux sous - requêtes Q_{valide} et $Q_{invalide}$ telles que :

$$Q = Q_{valide} \wedge Q_{invalide}$$

De manière explicite, il s'agira de déterminer :

- $Q' = Q_{valide}$ est la partie pouvant être résolue par les sources enregistrées. Q' est la partie de Q qui va subir l'opération de réécriture sur la base des connaissances exprimées au sein du modèle de données $Sch'O$. Notre but est de proposer ensuite une subdivision de la réécriture de $Q' = Q_{valide}$ en sous requêtes $Q'_1, Q'_2, \ldots, Q'_m$ avec $1 \leq m \leq n$ et n est le nombre total de correspondances sémantiques précalculées et enregistrées dans l'infrastructure d'intégration alors que m désigne le nombre de sources sollicitées pour fournir une réponse à Q.

- $Q'' = Q_{invalide}$ a une taille la plus petite possible. La sous-requête Q'' est la partie qui ne peut être résolue par les sources disponibles dans l'infrastructure d'intégration au moment de la formulation des requêtes globales Q. Il s'agit là d'une étape d'épuration de Q car l'utilisateur, formulant sa requête sur la base des concepts de l'ontologie de domaine, pourrait solliciter des ressources non encore disponibles dans les sources enregistrées. Aucun traitement ne sera donc réalisé sur Q'' dans la suite.

Le problème consiste donc à trouver la réécriture de Q' qui est l'ensemble $Q'_{reecriture} = \{(Q'_j, m_j)\}$ des couples (Q'_j, m_j) tel que Q'_j soit la subdivision de la requête Q' à laquelle il sera possible de répondre par le mapping m_j. On peut se rendre compte aisément que plusieurs réécritures sont possibles. En effet, la même subdivision Q'_j de la requête Q' peut être fournie par diverses sources de données, et donc par plusieurs mappings.

L'algorithme que nous proposons, *Q-Candidates'Finder,* prend comme entrées une requête globale Q et le schéma $Sch'O$, et fournit en sortie la génération des réécritures candidates $r(Q)$, c'est-à-dire l'ensemble de toutes les réécritures possibles $Q'_{reecriture}$.

4.2. *Algorithme de réécriture inspiré des Hypergraphes*

Les requêtes sont spécifiées grâce au langage $\psi - trees$ et par conséquent il est possible d'en fournir une forme conjonctive de contraintes atomiques à satisfaire

pour répondre à la requête globale posée. Ces contraintes traduisent les buts et les conditions sur les ressources contenues dans les Sources de données.

4.2.1. Variante de la caractérisation mathématique

Afin de fournir une caractérisation mathématique de notre problème de réécriture, nous présentons une variante à la formalisation du problème. Nous nommons la réécriture de $Q_{valide} = Q'$ la requête valide, par $Q'_{reecriture}$ dans la suite de notre réflexion. On veut donc calculer :

- toutes les réécritures candidates de la forme suivante :

$$Q'_{réécriture} = Q'_1 \wedge Q'_2 \wedge ..... \wedge Q'_m$$

- ainsi que les requêtes partielles Q'_j composant ces réécritures et répondues par une Source S_j de la forme suivante :

$$Q'_j = \wedge_{j=1}^{k} C_{ij}$$

Sachant que k désigne le nombre de contraintes C_i satisfaites par la requête partielle Q'_j de Q' et $l \leq m$ désigne le nombre de contraintes atomiques dans Q .

Vu sous cet angle, le problème de réécriture consiste d'abord en la génération des réécritures candidates, qui peut se caractériser comme un problème courant, assez connu en Théorie des Hypergraphes, consistant à retrouver les *Transversals minimaux* d'un Hypergraphe.

Définition 6 Hypergraphe, Transversal, Transversal minimal [7].

Un **Hypergraphe** est la donnée d'un couple $H = (V, E)$ tels que $V = \{v_1, v_2, ..., v_n\}$ est un ensemble fini d'éléments et $E = \{E_1, E_2, ..., E_m\}$ est une famille de sous-ensembles vérifiant les conditions suivantes:

i. $E_i \neq \emptyset, (i = 1, ..., m)$

ii. $\bigcup_{i=1}^{m} E_i = V$

Les éléments de V sont appelés *sommets* alors que les éléments de E sont appelés les *hyperarêtes* de l'Hypergraphe H . Un Hypergraphe peut être interprété comme la généralisation d'un Graphe au sein duquel la restriction selon laquelle une arête doit avoir seulement deux sommets est écartée.

Etant donné un hypergraphe $H = (V, E)$, un ensemble $T \subseteq V$ est appelé **Transversal** de H , si son intersection avec chacune des hyperarêtes n'est pas vide, c'est à dire $T \cap E_i \neq \emptyset, \forall E_i \in E$. Un Transversal est dit **minimal** si aucun sous-ensemble T' de T ne constitue un Transversal de H .

4.2.2. Construction de l' hypergraphe associé et algorithmes proposés

Etant donné un problème de réécriture de Q sur la base des mappings M , nous construisons l'hypergraphe $H_{Q,M}(V,E)$ associé comme suit :

- A chaque Mapping m_i appartenant à M et décrivant une ressource locale (élément, attribut, …) en fonction des concepts et relations de l'ontologie de domaine, nous associons un Sommet V_{m_i}, c'est-à-dire que nous construisons au final un ensemble de sommets $V = \{V_{m_i}, i \in [1, n]\}$.

- A chaque contrainte C_i exprimée dans la requête conjonctive $Q_{valide} = Q'$, nous associons une Hyperarête E_{C_i}, c'est-à-dire qu'au final nous aurons construit un ensemble d'hyperarêtes $E = \{E_{C_i}, i \in [1, l]\}$. Les hyperarêtes E_{C_i} constituent des sous-ensembles de V constitués seulement des mappings qui permettent de résoudre la contrainte considérée.

Nous présentons maintenant notre algorithme de réécriture de requêtes **Q-Candidates'Finder** qui doit intégrer une version améliorée de l'algorithme classique de génération de Transversaux minimaux:

Entrées: Une requête globale Q and $Sch'O = (O'\ proteomics, M\ mappings)$

Sorties: L'ensemble $Q_{candidates} = \{(Q_{valide}, Q_{invalide})\}$ des réécritures candidates, telles que :

- $Q_{valide} = r_i(Q) = Q' = \{(Q'_j, m_j)\}$
- $Q_{invalide}$ est la partie invalide de Q.

1: Construire l'Hypergraphe associé $H_{Q,M}(V,E)$

2 : Calculer $Q_{invalide} = \wedge_{i=1}^{k} C_i$ telle que C_i ne soit répondue par aucun Mapping dans M .

3: Construire l'Hypergraphe associé $H^*_{Q,M}(V,E^*)$

4 : $Q_{candidates} = \varnothing$

5: Générer l'Hypergraphe Transversal de $H^*_{Q,M}(V,E^*)$ noté *HypTransv*.

en utilisant l'algorithme Classique – **Figure 3**

6: **Pour toutes les arêtes** $X = \{V_{m_1}, V_{m_2}, ..., V_{m_p}\} \in$ HypTransv **faire**

7 : $Q_{valide} = r(Q) = Q' = \{(Q'_j, m_j), j \in [1, p]\}$

telle que Q'_j soit une subdivision de Q_{valide} répondue par le mapping m_j.

8 : $Q_{candidates} = Q_{candidates} \cup Q_{valide}$

9 : **Fin - Pour**

Figure 3. *Algorithme de réécriture de requêtes Q-Candidates' Finder*

De nombreux travaux [3, 6] ont porté sur les algorithmes de génération d'un Hypergraphe Transversal, qui est un ensemble de Transversaux minimaux. L'un des premiers résultats demeure l'Algorithme de Berge [3], mais de nouvelles variantes ont été élaborées afin d'améliorer sa complexité algorithmique [7, 11].

4.3. *Illustration concrète de Q-Candidates'Finder*

Afin d'illustrer l'approche de réécriture proposée, considérons les données suivantes:

- Des Correspondances sémantiques de cardinalité 1:1, exprimées suivant une approche LAV entre les concepts locaux (colonne LHS, Left Hand Side) et les concepts globaux d'un fragment de l'Ontologie de domaine (colonne RHS, Right Hand Side). Nous traitons des mappings formalisés en $\psi - trees$, de type Equivalence ou Subsomption entre les chemins de localisation des ressources (et non des sous-arbres).

	LHS	**Type**	**RHS**
O'proteomics Ontology :	**Gene (Genes, Proteins, Species, Organisms)** **Proteine (IdProteins, Peptides, DevStadiums)** …		
Mapping M1: Description de la Source S1	**S1_Gene (** S1_GeneName, S1_ProteinName; S1_species, S1_organisms)	**Equivalence**	**Gene (** Genes, Proteins, Species, Organisms)
Mapping M2: Description de La Source S 2	**S2_Gene (** S2_NomGenes, S2_NomProteines, S2_Especes, S2_Organismes)	**Equivalence**	**Gene (** Genes Proteins Species Organisms)
Mapping M3: Description Of Source S3	**S3_TreeLife (** S3_Species, S3_Genus)	**Subsomption**	**Gene (**… Species, Organisms, …)

Figure 4. *Fragment d'Ontologie et Mappings pertinents pour notre illustration*

- Une requête biologique exprimée en fonction des termes exprimant les concepts et relations de l'Ontologie de domaine et traduisant la sémantique suivante:

Quels sont les Genes dont les Proteines peuvent avoir un Signal peptidique et pour lesquels, il est établi qu'elles s'expriment à un stade Shrizont Tardif pour le Plasmodium falciparum?

Intuitivement, la requête globale peut s'écrire comme la conjonction de contraintes atomiques suivantes:

$$Q = C_{Genes} \wedge \mathrm{C}_{\mathrm{Proteins}} \wedge C_{Peptides} \wedge C_{DevStadiums} \wedge C_{Organisms} \wedge C_{Species}$$

L'Hypergraphe associé $H_{Q,M}(V,E)$ au problème concret donné consiste en les ensembles de sommets et d'hyperarêtes suivants:

$$V = \{V_{S2\_Gene}, V_{S1\_Gene}, V_{S3-TreeLife}\}$$

$$E = \begin{Bmatrix} E_{C\_Genes}, E_{C\_Proteins}, E_{C\_Peptides}, \\ E_{C\_DevStadiums}, E_{C\_organisms}, E_{C\_Species} \end{Bmatrix}$$

Nous montrons ainsi une sorte de diagramme de Venn qui traduit les relations particulières. En effet, nous matérialisons chacune des contraintes de la requête globale dans un Ensemble et établissons une relation de type "est répondue par" vers un Ensemble contenant tous les Mappings. Ainsi, nous schématisons dans la **Figure 5** que la contrainte $E_{C\_Proteins}$ peut être satisfaite par les mappings V_{S2_Gene} et V_{S1_Gene} alors que pour la contrainte $E_{C\_DevStadiums}$ il n'y a aucun Mapping, d'où le sous-ensemble vide.

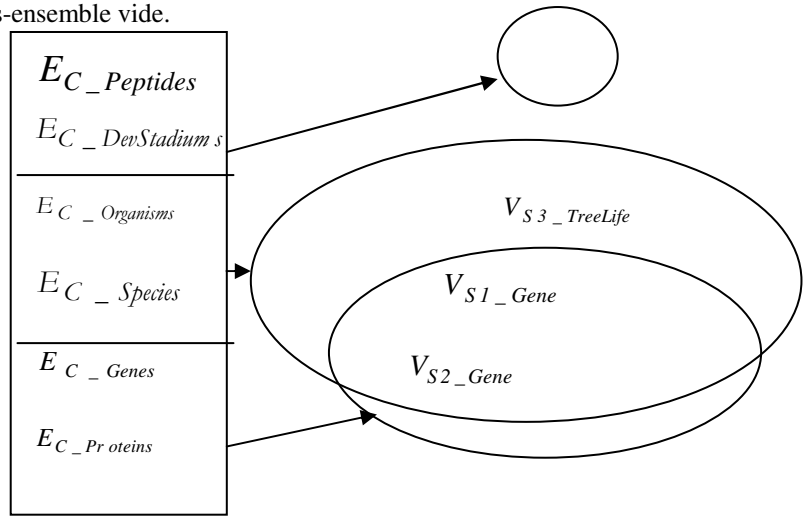

Figure 5. *Illustration de l'Hypergraphe associé*

- Détermination de $Q_{invalide}$

$Q_{invalide}$ est la partie de Q qui ne peut être résolue par les Sources courantes. Nous constituerons $Q_{invalide}$ comme la conjonction de toutes les contraintes de Q' caractérisées par des hyperarêtes vides dans l'Hypergraphe $H_{Q,M}(V,E)$. D'après notre schéma, les hyperarêtes vides sont celles en relation "est répondue par" avec un sous – ensemble vide de sommets. En scrutant la **Figure 5**, on peut déduire aisément qu'aucun mapping ne fournit les ressources relatives à DevStadiums (stade de développement) and Peptides (Type de signal) conformément aux exigences à l'algorithme Q-Candidates'Finder: Ligne de Code 2). Par consequent,

$$Q_{invalide} = C_{Peptides} \wedge C_{DevStadiums}$$

- Détermination Q_{valide}

Q_{valide} est la partie de Q qui peut être résolue par les sources disponibles. Ce qui signifie que nous construirons Q_{valide} avec toutes les contraintes de Q' qui sont caractérisées par des hyperarêtes non vides dans l'Hypergraphe $H_{Q,M}(V,E)$. Nous obtenons alors :

$$Q_{valide} = C_{Genes} \wedge C_{Proteins} \wedge C_{Organisms} \wedge C_{Species}$$

Il sera donc en partie répondu à la requête globale. En effet, Q_{valide} signifie que nous pourrions juste répondre à la requête suivante, étant donné les mappings sémantiques dont nous disposons dans le Framework d'intégration:

Donnez les gènes codant pour les protéines présentes chez le Plasmodium falciparum?

- Détermination de réécritures candidates $Q_{candidates}$

A partir de Q_{valide} calculée, nous générons un nouvel hypergraphe associé $H^{*}_{Q,M}(V,E^{*})$, (Ligne de code 3).

Trouver toutes les réécritures candidates suppose d'abord de construire le Produit Cartésien de tous les ensembles de mappings associés aux contraintes de la requête biologique globale. Dans notre situation, nous obtiendrons donc des quadruplets (4-uplets), ayant éliminé les hyperarêtes vides dans $H^{*}_{Q,M}(V,E^{*})$. Nous générons ensuite, pour chaque 4-uplet, un Ensemble de cardinalité maximale égale à 4. Ces ensembles correspondent en fait aux Transversals de l'Hypergraphe d'où l'initiative de la caractérisation proposée. En pratique, il serait bénéfique d'utiliser un nombre minimal de Sources protéomiques pour répondre à la requête Q_{valide}. En conséquence, un privilège devra être donné aux transversals qui mobilisent autant que possible moins de mappings provenant de Sources différentes. Cette condition est garantie si nous considérons les Transversals minimaux c'est-à-dire les Transversals qui ne contiennent pas eux–mêmes de sous–ensembles qui soient des Transversals de l'hypergraphe. Pour mieux percevoir cette nouvelle

situation, nous pourrions associer à chaque Transversal un autre ensemble constitué des sources dont sont originaires les mappings qui les composent.

De ce point de vue, il apparaît clairement que n'importe quel ensemble obtenu contenant les mappings V_{S1_Gene} and V_{S2_Gene} est un Transversal, et constitue donc une réécriture possible de Q. Par contre, les transversals minimaux sont $\{ V_{S1\_Gene} \}$ et $\{ V_{S2\_Gene} \}$ qui vont nous permettre de construire deux réécritures candidates. Dans notre cas, nous trouvons 36 quadruplets qui correspondent en tout à 6 Transversals et seulement à 2 Transversals minimaux. A partir des Transversals minimaux obtenus, notre algorithme peut reformuler les réécritures candidates en $\psi - trees$ (voir Q-Candates'Finder: Ligne de code 6 à 9).

5. Conclusion et perspectives

Notre travail de recherche s'inscrit dans le contexte général de l'intégration sémantique de données protéomiques sur le Web. Dans ce papier, nous nous concentrons sur la réécriture de requêtes globales. Notre approche consiste à déterminer des réécritures candidates en s'appuyant sur les connaissances sémantiques du domaine, à savoir l'ontologie de domaine et les mappings sémantiques précalculés, validés et stockés dans le Framework d'intégration.

Comme notre but final est de déterminer une réécriture unique de meilleure qualité, nous définirons un critère de classement des réécritures candidates servant de base au choix de la réécriture optimale. Nous présentons également les prémices d'un formalisme logique pour décrire les connaissances protéomiques et exprimer les requêtes globales du protéomicien. Ce formalisme dénommé $\psi - trees$ s'inspire d'un langage de description de Traits appelé $\psi - termes$ dont il reprend essentiellement les constructeurs.

Dans nos travaux futurs, nous traiterons de plus près l'optimisation et la complexité de l'algorithme de réécriture de requêtes proposé. Nous comptons valider dans un avenir proche notre approche en développant un prototype sur des schémas de sources de données protéomiques comme celles dont la scrutation nous a permis de construire l'ontologie de domaine.

Nous améliorerons $\psi - trees$ pour prendre en compte la richesse des spécifications de XML Schéma et XQuery afin de répondre ainsi aux exigences de scalabilité et de flexibilité d'un médiateur Web.

6. Bibliographie

[1] Jacob Kohler, Integration of Life Science databases. Drug Discovery Today, BIOSILICO Vol.2, No.2, March 2004.

[2] S.B. Davidson, C. Overton and P. Buneman: Challenges in integrating biological data sources. Journal of Computational Biology, 2(4):557–572, 1995.

[3] Claude Berge. Hypergraphs. North Holland, Amsterdam, 1989. ISBN 0 444 874895; QA166.23.B4813 1989.

[4] A. Deutsch, V. Tannen, XML queries and constraints, containment and reformulation, Elsevier's Journal of Theoretical Computer Science, 336 (2005), Pages: 57-87

[6] T. Eiter and G. Gottlob. Identifying the minimal transversals of a hypergraph and related problems. SIAM Journal on Computing, 24(6):1278{1304, 1995.

[7] Dimitris J. Kavvadias and Elias C. Stavropoulos. An Efficient Algorithm for The Transversal Hypergraph Generation, Journal of Graph Algorithms and Applications, Vol.9, No.2, Pges.239-264, 2005.

[8] Martin Bishop, Genetics Databases, Academic Press, 1999.

[9] A. Deutsch and V. Tannen. Reformulation of XML Queries and Constraints, In Proceedings of the 9th International Conference on Database Theory (ICDT).Pges 225-241

[10]H. Mannila and K-J Raiha. The Design of Relational Databases. Addison - Wesley, Wokingham, England, 1994.

[11] C. REY, F. TOUMANI, M.-S. HACID, A. LÉGER, An algorithm and a prototype for the dynamic discovery of e-services, Report, 2003, LIMOS, Clemont-Ferrand, France.

[12] I.Manolescu, D. Florescu, and D. K. Kossmann. Answering XML queries over heterogeneous data sources. In Proceedings of the 27th International Conference on Very Large Data Bases(VLDB '01), pages 241–250, Orlando, Sept. 2001. Morgan Kaufmann.

[13] Torsten Schlieder, ApproXQL: Design and Implementation of an Approximate Pattern Matching Language for XML, Technical Report, Freie Universitat Berlin, May 2001.

[14] F. Baader, D. Calvanese, D. McGuinness, E.D. Nardi, P.Patel-Schneider, The Description Logic Handbook, Theory, Implementation and Applications, Cambridge University Press, Cambridge, 2003.

[15] A.Y. Halevy, Answering queries using views: a survey, VLDB J. 10 (4) (2001) 270–294.

[16] T. Bray, J. Paoli, and C.M. Sperberg-McQueen, "Extensible Markup Language (XML) 1.0," W3C Recommendation, Feb. 1998; available online at http://www.w3.org/TR/REC-xml.

[17] A. Calý., D. Calvanese, GD. Giacomo, M. Lenzerini, View based query answering and query containment over semi-structured data Database Programming Languages (DBPL), Eight International Workshops, 2004.. Pages: 40–61.

[18] T. Pixley, Document object model (DOM) Level 2 Events Specification, Version 1.0 W3C Recommendation, W3C, 13th November, 2000. See also: http://www.w3.org/TR/2000/REC-DOM-Level-2-Events-20001113

[19] H.S. Thompson and al., "XML Schema Part 1: Structures," W3C, work-in-progress, current as of Apr. 2000.

See also: http://www.w3.org/TR/2000/WD-xmlschema-1-20000407/.

[20] G. Gottlob, C. Koch, K.U Schulz, Conjunctive Queries over Trees, Proceedings 23rd ACM SIGMOD-SIGART Symposium on Principles of Database Systems (PODS 2004), Paris, France. ACM Press, New York, USA, pp. 189-200.

[21] Patrick Lehti, Peter Fankhauser, "XML Data Integration with OWL: Experiences and Challenges," saint, p. 160, 2004 Symposium on Applications and the Internet (SAINT'04), 2004.

[22] B. Amann, C. Beeri, I. Fundulaki, and M. Scholl. Ontology-based integration of XML web resources. In Int'l Semantic Web onference '02, pages 117.131, 2002.

[23] B. Benatallah, M-S Hacid, H-y. Paik, C. Rey, F. Toumani, Towards semantic-driven, flexible and scalable framework for peering and querying e-catalog communities, Elsevier's Journal of Information Systems, pages 266-294, 2006.

[24] A. Halevy, Z. Ives, I. Tatarinov, and P. Mork. Piazza: Data management infrastructure for semantic web applications. In Proceedings of the International. WWW Conference, 2003.

[25] C. Delobel, C. Reynaud, M-C. Rousset, J-P. Sirot b, D. Vodislav, Semantic integration in Xyleme: a uniform tree-based approach, Elsevier's Journal of Data & Knowledge Engineering 44 (2003) 267–298

[26] C-WEB Project: http://cweb.inria.fr

[27] A. Levy, A. Rajaraman, J. Ordille, Querying Heterogeneous Information Sources Using Source Descriptions, In Proc. VLDB, pages 251-262, Mumbai (Bombay), India, September 1996.

[28] I. Fundulaki, B. Amann, C. Beeri, and M. Scholl. STYX: Connecting the XML World to the World of Semantics resources. In Proceedings of EDBT, Prague, Czech Republic, March 2002.

[29] Wolfang May, XPathLog: A Declarative, Native XML Data Manipulation Language. International Database Engineering and Applications Workshop (IDEQS401), IEEE Computer Science Press, 2001.

[30] W. May, LoPiX. A System for XML Data Integration and Manipulation, International Conference on Very Large Data Bases (VLDB), Demonstration Track, 2001. Pages: 707-708.

See also http://www.informatik.uni6freiburg.de/~may/lopix/

[31] W. May, Logic-based XML Data Integration: a semi-materializing approach, Elsevier's Journal of Applied Logic, 3 (2005), Pages: 271-307.

[32] H.Garcia-molina, Y.Papakonstantinou, D. Quass, A.Raaraman, Y. Sagiv, J. Ullman, V. Vassalos, J. Widom, The TSIMMIS approach to mediation : Data models and languages, Journal Intelligent Information Systems 8(2) (1997), Pages: 117-132.

[33] M. Fernandez, D. Florescu, A. Levy, D. Suciu, A Query Language for a web-site management system, SIGMOD Record 26 (3) (1997) Pages: 4-11.

[34] B. Ludascher, R. Himmeroder, G. Lausen, W. May, C.Schelpphorst, Managing semi-structured data with FLORID: Adductive object-oriented perspective, Information Systems 23(8) (1998) 589-612.

[35] M.-C.Rousset, A. Bidault, C. Froidevaux, H. Gagliardi, F.Goasdoue, C.Reynaud, B. Safar. Construction de médiateurs pour intégrer des sources d'information multiples et hétérogènes : le projet PICSEL. Revue I3. Vol.2. n°1. Pages: .5 - 59 (2002).

[36] H. Ait - Kaci, A. Podelski, S. C. Goldstein, Order-Sorted Theory Unification. Journal of Logic Programming, 30(2). 99:124, 1997.

[37] H. Ait - Kaci, A. Podelski, Towards the meaning of LIFE. In The Proceedings on 3rd International Symposium on Programming Language Implementation and Logic Programming, Berlin, Pages 255 - 274, 1991

[38] D. Chamberlin, XQuery: An XML Query Language. IBM Systems Journal, Vol. 41, No. 4, 2002.

[39] M. Fernandez, J. Siméon, P. Wadler, An Algebra for XML Query, In Proceedings of FSTTCS 2000, Pages: 11 - 45.

[40] S. Bechofer, F. van Hamelen, J. Hendler, I. Horrocks, D. L. McGuiness, P. F. Patel-Schneider, L., A. Steins. OWL Web Ontology Language, W3C Recommendation, February 2004,

See also http://www.w3.org/TR/2004/REC-owl-ref-20040210/

[41] Jeffrey D. Ullman, Information integration using logical views. Theoretical Computer Science, 239(2):189–210, 2000.

[42] Y. Papakonstantinou, V. Vassalos, Query rewriting for semistructured data, In Proceedings of The Conference on Management of Data, Philadelphia, PA, USA, June 1999, Pages 455 - 466.