Un premier pas vers la modélisation des données semi-structurées par la logique multi-modale hybride

Nicole Bidoit* — **Serenella Cerrito**** — **Virginie Thion***

**LRI UMR CNRS 8623, Université Paris 11, Centre d'Orsay.*
{bidoit,thion}@lri.fr

***LaMI UMR CNRS 8042, Université d'Evry Val d'Essonne.*
{serena}@lami.univ-evry.fr

RÉSUMÉ. *Les documents XML et les données semi-structurées peuvent être représentées par des graphes étiquetés. Dans cet article, nous établissons un lien direct entre ces graphes de données et les modèles d'un langage de la logique multi-modale hybride. Ce lien est ensuite exploré dans deux directions. Tout d'abord, nous nous intéressons à l'expression de contraintes d'intégrité sur des données semi-structurées en revisitant certaines classes de contraintes étudiées dans la littérature. Puis nous abordons le problème de la définition d'un schéma pour les données semi-structurées. Notre approche conduit à définir un schéma sous forme d'une formule de la logique hybride, formule devant être satisfaite par les instances (graphes de données) du schéma. La contribution de cet article se situe à deux niveaux : 1) la modélisation des données semi-structurées, des contraintes, et la définition de schéma dans un formalisme unique : la logique multi-modale hybride ; 2) la généralisation de la notion de schéma, puisque nous considérons des schémas permettant de manipuler des références "bien typées".*

ABSTRACT. *XML documents and semistructured data, can be seen as labelled graphs. In this paper we set a correspondence between such graphs and the models of a language of hybrid multi-modal logic. This allows us to characterize a schema for semistructured data as a formula of hybrid multi-modal logic, and instances of the schema (data graphs) as models of this formula. We also investigate how to express in such a logic integrity constraints on semistructured data, in particular some classes of constraints widely considered in the literature. The contribution of this work is twofold: 1) We generalize the (DTD) notion of schema, by proposing a definition of schema were references are "well typed" (contrarily to what happens with DTDs). 2) We formalize such a schema, integrity constraints and semistructured data data in a unique framework, namely hybrid multi-modal logic.*

MOTS-CLÉS : *données semi-structurées, logique multi-modale hybride, schema, contraintes*

KEYWORDS: *semistructured data, multi-modal logic, hybrid logic, schema, constraints*

Introduction

Dans la communauté des bases de données, l'intérêt pour les donnéees dites *semi-structurées* est grandissant. L'origine historique de cette notion est liée à l'utilisation de plus en plus intensive de XML comme standard permettant la *publication de données*.

Un document XML peut en fait être vu comme un ensemble de données dont l'organisation est "plus flexible" que celle, par exemple, d'une base de données relationnelle ou objet. En quel sens "plus flexible" ? Tout d'abord, en absence d'une grammaire contraignant l'organisation des données "valides", par exemple en absence de DTD [RAY 01], un document XML peut être vu comme une organisation logique de données qui ne respecte aucune contrainte fixée *à priori*. Même en présence d'une DTD, qui oblige les données dits "valides" à respecter certaines contraintes, on a quand même de larges marges de liberté, car, par exemple, la présence de l'opérateur `?` dans une DTD permet la spécification d'éléments optionnels, la présence de la syntaxe `ANY` permet de ne pas contraindre du tout la structure d'un élément donné, des attributs d'éléments (au sens XML) peuvent être déclarés comme optionnels, etc.

Ceci a mené à une notion générale de *données semi-structurées*, qui va au délà de XML. On trouve dans la littérature sur les données semi-structurées plusieurs propositions de modélisations de cette notion [ABI 00]. En général, une instance de bases de données semi-structurées est vue comme un graphe de données étiqueté. La figure 1 est un exemple classique de données semi-structurées représentant des articles et livres contenus dans une bibliothèque.

Or, il existe une famille de logiques, les logiques dites *modales* dont la sémantique est définie par des graphes (sémantique de *Kripke*) ; ces logiques permettent donc d'exprimer de façon naturelle des propriétés de graphes. Dans ce travail, nous étudions le problème de la modélisation des données semi-structurées, et en particulier de la formalisation des contraintes et du schéma en utilisant une variante de logique modale dite *logique multi-modale hybride* [BLA 00]. Nous proposons une modélisation des données semi-structurées et des contraintes sur celles-ci, ainsi qu'une définition de schéma dans un formalisme unique, celui de la logique multi-modale hybride. La notion de schéma que nous proposons est plus expressive que, par exemple, celle de DTD, car elle permet de "bien typer" la cible d'un lien de référence, chose qu'une DTD ne permet pas de faire.

Nous montrons le bien fondé de notre approche en établissant un certain nombre de propriétés de notre formalisation : notre langage logique d'expression de contraintes est strictement plus expressif que le langage P étudié dans [BUN 98] ; un schéma $\mathcal{G}$ de données peut être représenté fidèlement par une formule $TRAD(\mathcal{G})$ de la logique multi-modale hybride, au sens où les instances de $\mathcal{G}$ coïncident avec le modèle $TRAD(\mathcal{G})$.

Le plan de l'article est le suivant : dans la première section, nous présentons la logique multi-modale hybride ; dans la deuxième section, nous montrons comment exprimer certaines contraintes d'intégrité avec la logique multi-modale hybride ; la section sui-

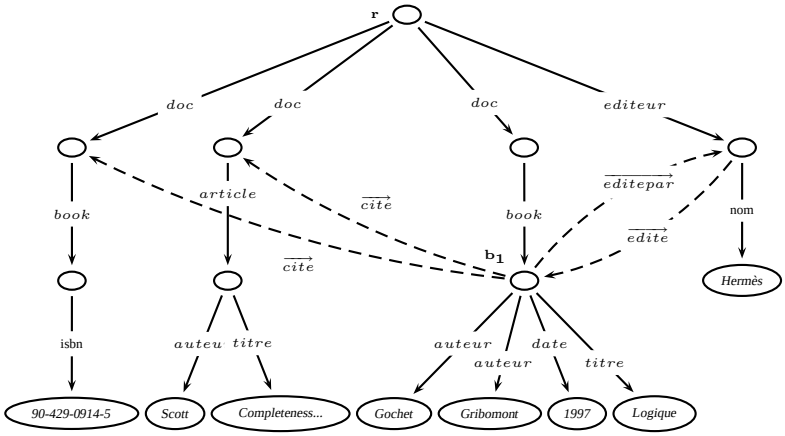

Figure 1. *Représentation d'une base de données de bibliothèque*

vante est consacrée à définir et étudier toujours du point de la logique multi-modale hybride une notion de schéma pour les données semi-structurées intégrant le typage de références ; la conclusion de l'article expose quelques comparaisons avec d'autres travaux et quelques perspectives.

Par manque de place, les preuves des résultats de cet article ne sont pas données. Elles sont disponibles en rapport interne à paraître.

1. La logique multi-modale hybride

Dans la suite, nous travaillons dans le cadre propositionnel. Nous présentons d'abord la logique multi-modale (la logique multi-modale hybride en est une extension).

1.1. *La logique multi-modale*

L'alphabet de la logique multi-modale est constitué de :

– un ensemble de symboles propositionnels $PROP$ notés $p, q, ...$,

– le connecteur logique binaire $\wedge$,

– l'opérateur logique unaire $\neg$,

– un ensemble fini d'étiquettes $\mathcal{E}$,

– les opérateurs modaux $[e]$ où e est une étiquette.

Les formules bien formées FBF de la logique multi-modale sont définies par :
FBF : := $p \mid \neg\psi \mid \psi_1 \wedge \psi_2 \mid [e]\psi$
où ψ, ψ_1 et ψ_2 sont des formules bien formées et $p \in PROP$.

Un modèle fini $\mathfrak{M}$ de la logique multi-modale est un quadruplet (S, s_0, R, V) tel que

– S est un ensemble fini d'états contenant l'état s_0 ;

– $R = \{r_e | e \in \mathcal{E}\}$ est un ensemble de relations binaires sur S, dites *relations d'accessibilité* (R contient exactement une relation r_e pour chaque étiquette e) ;

– V est une fonction $PROP \rightarrow Pow(S)$, qui assigne à chaque proposition p, l'ensemble des états dans lesquels p est satisfaite.

Il est ici immédiat de voir qu'un modèle de la logique multi-modale est un graphe orienté, étiqueté sur les arcs : les sommets du graphe sont les états du modèle et chaque relation r_e de R définit les arcs du graphe et leur étiquetage. Or, classiquement, les données semi-structurées sont représentées par un graphe orienté et étiqueté, et parfois même de façon plus restreinte par un arbre étiqueté. Il paraît donc naturel de faire la correspondance entre modèle de la logique multi-modale et données semi-structurées. Ce lien a d'ailleurs déjà été exploré par [ALE 01].

Exemple 1 La figure 1 représente à la fois un exemple classique de données semi-structurées et un modèle de la logique multi-modale. En effet, on peut considérer que

– le sommet r correspond à l'état s_0,

– l'ensemble des sommets du graphe correspond à l'ensemble des états S,

– à chaque étiquette e du graphe correspond une relation binaire r_e de $R = \{r_{doc}, r_{editeur}, r_{book}, r_{\overrightarrow{cite}}, \cdots\}$. Chaque arc d'origine s_1, de destination s_2 et étiqueté par e dans le graphe de données correspond à un couple (s_1, s_2) dans la relation r_e.

– les valeurs "étiquetant" les feuilles correspondent aux propositions du langage et par exemple la présence de Scott "sur" la $2^{ème}$ feuille f à partir de la gauche correspond dans le modèle à $f \in V(Scott)$; il convient de prendre garde ici à ne pas confondre les noms (r et b_1) donnés à certains sommets pour les besoins de la présentation, avec des propositions. Il est d'ailleurs à noter que si $\mathfrak{M}$ correspond à des données semi-structurées alors la fonction V de $\mathfrak{M}$ associe à chaque proposition p des états "feuilles" du graphe (i.e. sommets de degré sortant nul).

Soit $\mathfrak{M}$ un modèle, s un état et ψ une formule bien formée. $\mathfrak{M}$ satisfait ψ à l'état s, noté $\mathfrak{M}, s \models_{mm} \psi$, est défini par induction sur ψ comme suit :

$\mathfrak{M}, s \models_{mm} p$	ssi $s \in V(p)$, où $p \in PROP$
$\mathfrak{M}, s \models_{mm} \psi_1 \wedge \psi_2$	ssi $\mathfrak{M}, s \models_{mm} \psi_1$ et $\mathfrak{M}, s \models_{mm} \psi_2$
$\mathfrak{M}, s \models_{mm} \neg\psi$	ssi $\mathfrak{M}, s \not\models_{mm} \psi$
$\mathfrak{M}, s \models_{mm} [e]\psi$	ssi pour tout $s' \in S$, $(s, s') \in r_e$ implique $\mathfrak{M}, s' \models_{mm} \psi$

Exemple 1 (suite) Pour le modèle de la figure 1, la formule $[auteur]\neg Scott$ évaluée en b_1 est satisfaite car aucun des états accessibles de b_1 par un arc étiqueté par $auteur$ ne satisfait $Scott$. Pour le même modèle, la formule $[editeur][nom]Hermès$ évaluée à la racine r est satisfaite.

Dans la suite, nous utilisons de façon classique les opérateurs $\vee$, $\Rightarrow$ et $\langle e \rangle$ définis par : $\psi_1 \vee \psi_2 =_{def} \neg(\neg\psi_1 \wedge \neg\psi_2)$, $\psi_1 \Rightarrow \psi_2 =_{def} \neg\psi_1 \vee \psi_2$ et $\langle e \rangle\psi =_{def} \neg[e]\neg\psi$. Nous utilisons aussi le symbole $\top$ pour désigner une tautologie.

Exemple 1 (suite) Pour le modèle de la figure 1, la formule $\langle titre \rangle \top$ évaluée en b_1 est satisfaite car il existe un successeur de b_1 accessible par un arc étiqueté par $titre$. Sur le même principe, la formule $[doc](\langle book \rangle \top \vee \langle article \rangle \top)$ évaluée à la racine est satisfaite dans ce modèle.

Le langage peut aussi être étendu avec l'opérateur G dont la sémantique est :
$\mathfrak{M}, s \models_{mm} G\psi$ ssi quelque soit $s' \in S$ tel que $(s, s') \in R^+$ on a $\mathfrak{M}, s' \models_{mm} \psi$ où R^+ est la fermeture transitive de la relation $R = \cup_{e \in \mathcal{E}} r_e$.

De la même manière que $\langle e \rangle$ a été défini à partir de $[e]$, l'opérateur F est défini à partir de G par $F\psi =_{def} \neg G \neg \psi$. La formule $F\psi$ exprime la propriété "il existe un état, accessible à partir de l'état courant par un chemin de longueur quelconque, tel que ψ". Et, de façon duale, la formule $G\psi$ exprime la propriété "pour tous les états accessibles à partir de l'état courant par un chemin de longueur quelconque, ψ est vérifiée".

Exemple 1 (suite) Toujours en se référant à la figure 1, la formule $F\ Scott$ évaluée en la racine r est satisfaite car un état accessible de la racine vérifie $Scott$. Par contre, la formule $G[\overrightarrow{cite}]\langle book \rangle \top$ évaluée en la racine r n'est pas satisfaite car b_1 est un état accessible de la racine pour lequel $[\overrightarrow{cite}]\langle book \rangle \top$ n'est pas satisfaite.

A ce point de la présentation, le lecteur peut entrevoir la possibilité d'utiliser la logique multi-modale pour exprimer des contraintes d'intégrité sur des données semi-structurées. Par exemple, la contrainte "tout article a au moins un auteur" s'exprime par la formule $[doc][article]\langle auteur \rangle \top$.

Toutefois, il apparaît que l'expressivité de la logique multi-modale est insuffisante pour exprimer certaines propriétés intéressantes telles que les contraintes introduites dans [BUN 98] et que nous étudierons dans la section 2. Cette remarque nous amène à présenter une extension de la logique multi-modale appelée logique multi-modale hybride.

1.2. *La logique multi-modale hybride*

Afin de motiver l'introduction de la logique modale hybride, nous allons reprendre ici à notre compte les arguments de [BLA 00]. La logique modale est un formalisme simple pour travailler/manipuler des structures relationnelles telles que des graphes. Toutefois, la logique modale n'offre pas de mécanismes permettant de faire référence à des sommets particuliers d'un graphe. Ceci restreint le pouvoir de représentation et de raisonnement de la logique modale. Ainsi les logiques hybrides [ARE 99], dans leur forme les plus simples, sont des extensions des logiques modales permettant justement de faire référence explicitement aux sommets des structures sous jacentes i.e. des graphes. L'objet de cet article n'étant pas de présenter les logiques modales hybrides, nous proposons au lecteur intéressé un ensemble d'articles sur le sujet, disponibles sur le site `http ://www.hylo.net`, parmi lesquels certains montrent que les logiques

modales hydrides “subsument” les logiques de description, la logique terminologique (feature logic), la logique temporelle...

Le langage de la logique hybride permet donc de nommer les états et d’exprimer qu’une formule doit être satisfaite dans un état nommé. La logique hybride introduit pour ce faire quatre outils fondamentaux :

les nominaux sont des symboles propositionnels (donc des formules) spéciaux : chaque nominal “est vrai” dans un et un seul état du modèle. Un nominal “nomme” l’unique état dans lequel il est vrai ;

les variables d’état sont aussi des formules qui permettent de désigner un état ;

les opérateurs de satisfaction sont de la forme $@_u$ où u est une variable d’état ou un nominal. Une formule de la forme $@_u\psi$, où ψ est une formule, signifie que ψ doit être satisfaite à l’état désigné ou nommé par u ;

l’opérateur “binder” $\downarrow$: une formule de la forme $\downarrow x\ \psi$, où x est une variable d’état et ψ est une formule, lie toutes les occurrences de x dans ψ à l’état courant de l’évaluation.

L’alphabet de la logique multi-modale hybride étend l’alphabet de la logique multi-modale par : un ensemble de nominaux $NOM = \{a, b, \cdots\}$, un ensemble de variables d’états $SVAR = \{x, y, z, \cdots\}$, l’opérateur $\downarrow x$ où x est une variable d’état, les opérateurs $@_u$ où u est une variable d’état ou un nominal ($PROP$, NOM et $SVAR$ disjoints deux à deux).

Les formules bien formées FBF de cette logique sont définies par :
FBF : := $p \mid \top \mid \neg\psi \mid \psi_1 \vee \psi_2 \mid \psi_1 \wedge \psi_2 \mid [e]\psi \mid \langle e\rangle\psi \mid G\psi \mid F\psi \mid u \mid \downarrow x\ \psi \mid @_u\psi$
où ψ, ψ_1 et ψ_2 sont des formules bien formées, $p \in PROP$, $x \in SVAR$ et $u \in NOM \cup SVAR$.

Voici, intuitivement, la signification des opérateurs hybrides $@_u$ et $\downarrow$: $@_u\psi$ signifie “aller à l’état nommé (ou désigné) par u (qui est l’unique état où u est vrai)”, et évaluer ψ en u.” $\downarrow x\ \psi$ signifie “x désigne maintenant l’état courant dans ψ”.

Il est nécessaire maintenant d’enrichir légèrement la notion de modèle modal pour intégrer l’introduction des nominaux, des variables d’états, des formules de satisfaction et de “binding”.

Une structure $\mathfrak{M}$=$(S, s_0, R, V, \mathcal{I}_{nom})$ est un modèle de la logique multi-modale hybride si (1) (S, s_0, R, V) est un modèle de la logique multi-modale, et (2) $\mathcal{I}_{nom}$ est une fonction $NOM \rightarrow S$ qui assigne un état à chaque nominal.

Une valuation est une fonction $g : SVAR \rightarrow S$ qui assigne un état à chaque variable d’état. La notation $g \overset{x}{\sim} g'$ (g' est un x-variant de g) signifie que g' et g sont des valuations (pour un modèle $\mathfrak{M}$) telles que g' est identique à g sauf éventuellement pour l’argument x.

Nous sommes donc prêts à donner la sémantique des formules “hybrides”.

$\mathfrak{M}$ satisfait ψ à l’état s respectivement à une valuation g, noté $\mathfrak{M}, g, s \models_{hm} \psi$, est défini par induction sur ψ comme suit (nous ne définissons ici que la sémantique des

formules “hybrides”) :

$\mathfrak{M}, g, s \models_{hm} a$ ssi $I_{nom}(a) = s$, où $a \in NOM$
$\mathfrak{M}, g, s \models_{hm} x$ ssi $g(x) = s$, où $x \in SVAR$
$\mathfrak{M}, g, s \models_{hm} \downarrow x\ \psi$ ssi $\mathfrak{M}, g', s \models_{hm} \psi$ avec $g \overset{x}{\sim} g'$ et $g'(x) = s$, où $x \in SVAR$
$\mathfrak{M}, g, s \models_{hm} @_x\psi$ ssi $\mathfrak{M}, g, g(x) \models_{hm} \psi$ où $x \in SVAR$
$\mathfrak{M}, g, s \models_{hm} @_a\psi$ ssi $\mathfrak{M}, g, I_{nom}(a) \models_{hm} \psi$ où $a \in NOM$

Dans la suite, nous avons besoin des notions sémantiques suivantes :

– $\mathfrak{M}$ satisfait globalement ψ respectivement à une valuation g, noté $\mathfrak{M}, g \models_{hm} \psi$, ssi pour tout état $s \in S$, $\mathfrak{M}, g, s \models_{hm} \psi$;

– $\mathfrak{M}$ satisfait globalement ψ, noté $\mathfrak{M} \models_{hm} \psi$, ssi pour toute valuation g, $\mathfrak{M}, g \models_{hm} \psi$.

Exemple 1 (suite) Nous continuons l’exemple 1 en ajoutant à notre langage le nominal $root$. La formule $\downarrow x\ \langle\overrightarrow{editepar}\rangle\langle\overrightarrow{edite}\rangle x$ évaluée en b_1 est satisfaite car, de b_1, il existe un état accessible par un arc étiqueté par $\overrightarrow{editepar}$ pour lequel il existe un arc sortant étiqueté par $\overrightarrow{edite}$ et revenant en b_1. Nous remarquons l’utilisation du binder “$\downarrow x\ \cdots$” pour assigner à la variable d’état x l’état courant de l’évaluation, en l’occurrence l’état b_1. Nous attirons l’attention du lecteur sur l’utilisation de cette variable en tant que formule dans “$\cdots \langle\overrightarrow{edite}\rangle x$”. La formule $@_{root}[doc][article]\langle auteur\rangle\top$ évaluée en b_1 est également satisfaite, à condition bien sûr que le nominal $root$ nomme la racine r du graphe de données.

Nous avons déja remarqué que le graphe de données de la figure 1 peut être vu comme un modèle de la logique multi-modale. Il est facile de le voir comme un modèle de la logique multi-modale hybride en choisissant l’ensemble NOM des nominaux du langage comme étant le singleton $\{root\}$, interprété par r i.e. $I_{nom}(root) = r$.

[BLA 99] exhibe un certain nombre de formules hybrides dites pures (ne contenant pas de symbole propositionnel) dont l’intérêt est d’exprimer des contraintes de structure sur le graphe d’un modèle. Par exemple :
• la réflexivité de la relation d’accessibilité r_e peut s’exprimer soit par $\mathfrak{M} \models_{hm} \downarrow x\ \langle e\rangle x$ soit par $\mathfrak{M} \models_{hm} @_x\langle e\rangle x$,
• il est possible d’exprimer que tous les états de S sont accessibles à partir de $root$ par $\mathfrak{M} \models_{hm} \downarrow x\ @_{root}Fx$.

2. Données semi-structurées et contraintes d’intégrité

Partant du principe que les données semi-structurées sont représentées par un graphe étiqueté enraciné, et sachant que cette structure peut être vue comme un modèle de la logique multi-modale hybride, il est possible d’exprimer une contrainte d’intégrité sur ces données sous la forme d’une formule de cette logique.

Des contraintes sur les données semi-structurées sont souvent exprimées en termes de navigation dans le graphe de données, c’est-à-dire sous la forme de propriétés des

chemins du graphe : ces contraintes sont dites *de chemin* et ont été étudiées, entre autre, dans [ABI 97, BUN 98, ALE 01, CAL 99, CAL 98, DEU 01, CAL 02].

Nous nous intéressons ici aux contraintes étudiées dans [BUN 98], formalisées par un sous-langage de la logique classique du premier ordre appelé P ; cette classe de contraintes est une généralisation naturelle des contraintes de [ABI 97].

Dans cette section, nous présentons tout d'abord le langage P. Nous donnons ensuite quelques contraintes exprimables dans P et proposons de les exprimer par une formule de la logique multi-modale hybride. Puis nous établissons un résultat d'expressivité, à savoir que la logique multi-modale hybride est plus expressive que P.

Un chemin peut être représenté par une formule de la logique du premier ordre à deux variables libres. Ainsi, un chemin est une formule $\alpha(x, y)$ de l'une des formes suivantes :
- $x = y$, écrit $\varepsilon(x, y)$ et appelé *chemin vide*,
- $K(x, y)$, K étant un symbole de relation binaire,
- $\exists z(K(x, z) \wedge \beta(z, y))$, K étant un symbole de relation binaire et $\beta(z, y)$ un chemin.

Une formule de P est une formule soit de la forme $\forall xy(\alpha(root, x) \wedge \beta(x, y) \Rightarrow \gamma(x, y))$ soit de la forme $\forall xy(\alpha(root, x) \wedge \beta(x, y) \Rightarrow \gamma(y, x))$ où α, β et γ sont des chemins. Appelons *contraintes de chemin*, les contraintes exprimées par une formule de P.

Dans la suite, les exemples se réfèrent aux données de la figure 1. Un exemple de contrainte de chemin est "`Tout document` x `cité par un livre est accessible à partir de la racine de la base`", exprimable dans P par la formule (1) $\forall x(\exists y(r_{doc}(root, y) \wedge (\exists z(r_{book}(y, z) \wedge r_{cite}(z, x)))) \Rightarrow r_{doc}(root, x))$. Ce type de contrainte est appelé *extent constraint* dans [BUN 98].

Un autre type de contraintes de chemin sont les contraintes dites *inverse constraint*. Ces contraintes stipulent que si y est accessible de x par un chemin α alors x est accessible de y par un chemin β. Un exemple est la contrainte "`pour tout` x`, si` x `est édité par` y `alors` y `édite` x", exprimable dans P par la formule (2) $\forall xy(\exists z(r_{doc}(root, z) \wedge r_{book}(z, x)) \wedge r_{\overrightarrow{editepar}}(x, y) \Rightarrow r_{\overrightarrow{edite}}(y, x))$.

Ces contraintes sont exprimables par des formules de la logique multi-modale hybride. Par exemple, (1) se traduit par $@_{root}[doc][book][\overrightarrow{cite}]{\downarrow}x\ (@_{root}\langle doc\rangle x)$ et (2) se traduit par $@_{root}[doc][book]{\downarrow}x\ ([\overrightarrow{editepar}]\langle\overrightarrow{edite}\rangle x)$.

Soit $\mathcal{H}$ le sous langage de la logique multi-modale hybride ne contenant pas les opérateurs F et G. En général, *toute* formule de P a une traduction τ dans $\mathcal{H}$.

Théorème 1 Soit $\mathfrak{M}$ un modèle de logique multi-modale hybride et $\mathcal{I}$ l'interprétation du premier ordre correspondante[1]. Soit ψ une formule quelconque de P. Il existe une formule φ de $\mathcal{H}$ telle que : $\mathcal{I} \models_{fo} \psi\ ssi\ \mathfrak{M} \models_{hm} \varphi$

1. Nous ne détaillons la correspondance entre $\mathfrak{M}$ et $\mathcal{I}$: essentiellement les relations d'accessibilité sont traduites par des prédicats binaires et leur interprétation.

Ce résultat montre que la logique multi-modale hybride est au moins aussi expressive que P. Cette propriété peut être aussi déduite à partir du résultat [ARE 99] qui dit que $\mathcal{H}$ a exactement le même pouvoir expressif que le fragment de la logique du premier ordre dit "borné", qui inclue P.

On remarque aussi que certaines contraintes ne sont pas exprimables dans P, par exemple : `un livre a exactement un numéro isbn`. Par contre, cette contrainte se traduit dans $\mathcal{H}$ par $@_{root}[doc][book]{\downarrow}x\ (\langle isbn\rangle{\downarrow}y\ (@_x[isbn]y))$.

De plus, il est bien connu qu'aucune contrainte faisant référence à la fermeture transitive d'une relation n'est exprimable dans la logique du premier ordre. Ainsi, par exemple, la contrainte **"`Dans la base, on trouve un document écrit par Scott`"** (en sachant que les documents peuvent apparaître à plusieurs niveaux du graphe de données) n'est pas exprimable dans P. Par contre, cette contrainte est exprimée par la formule $\langle auteur\rangle Scott \vee (F\langle auteur\rangle Scott)$ du langage de la logique multi-modale hybride. Ceci montre que :

Théorème 2 La logique multi-modale hybride est strictement plus expressive que P.

Le problème de l'implication pour P étant indécidable, les auteurs de [BUN 98] étudient ensuite des restrictions de P pour lesquelles ce problème est décidable. Selon le même principe, puisque le problème de l'implication pour la logique multi-modale hybride a été prouvé indécidable, il conviendrait d'exhiber des sous-langages de la logique multi-modale hybride pour lesquels ce problème serait décidable. Par exemple, puisque c'est la présence de l'opérateur $\downarrow$ qui rend $\mathcal{H}$ indécidable [ARE 99], il serait intéressant d'étudier le fragment de la logique multi-modale hybride contenant F et G mais ne contenant pas $\downarrow$. Ceci fera l'objet d'un futur travail.

3. Données semi-structurées et schéma

L'objectif de cette section est d'étudier la notion de schéma de données semi-structurées. Dans un premier temps, nous proposons une notion de schéma plus générale que celles rencontrées dans la littérature standard sur XML et en particulier plus générale que la notion de DTD [RAY 01]. L'apport essentiel de notre notion de schéma est la gestion de références garantissant un bon "typage" des données référencées. L'outil utilisé pour définir un schéma est une extension des techniques usuellement exploitées et est développé sous la forme d'une grammaire de graphe très simple. Dans un deuxième temps, nous nous intéressons à la spécification d'un schéma par une formule de la logique multi-modale hybride et donc nous nous attaquons au problème classique suivant : étant donné un schéma $\mathcal{G}$, et un graphe $\mathfrak{M}$ de données semi-structurées, existe-t-il une formule ψ de la logique multi-modale hybride telle que $\mathfrak{M}$ est une instance du schéma $\mathcal{G}$ si et seulement si $\mathfrak{M}$ satisfait la formule ψ ?

3.1. *Schéma de données semi-structurées*

Un schéma étant une grammaire particulière, nous définissons ici ce que nous entendons par *grammaire*. Pour cela, nous définissons la notion de *motif* puis celle de *règle*.

Pour ce faire, nous considérons dans la suite : un ensemble fini $\mathcal{V}$ de symboles dits *non terminaux* contenant au moins le symbole $Root$; un ensemble fini d'étiquettes $\mathcal{E}$ disjoint de $\mathcal{V}$ et partitionné en deux sous-ensembles E et $\overrightarrow{E}$ (les étiquettes de $\overrightarrow{E}$ sont appelées *références*) ; un symbole spécial Λ.

Par convention, un symbole non terminal commence par une lettre majuscule, et une étiquette par une lettre minuscule. Les références sont surlignées par le symbole $\rightarrow$. La première notion nécessaire à la définition d'une grammaire de motifs est celle de motif :

Définition 1 (Motif) Un *motif* $expr$ est une expression de l'une des formes suivantes :

1) Λ ; on dit alors que $expr$ est le *motif vide* ;

2) $(e\ N)^{op}$ tel que $e \in \mathcal{E}$, $N \in \mathcal{V}$ et $op \in \{*, +, !, ?\}$; on dit alors que $expr$ est un *motif élémentaire* ;

3) $m_1, \cdots, m_k$ tel que pour tout $j \in [1..k]$, m_j est un motif élémentaire ; on dit alors que $expr$ est un *motif conjonctif* ;

4) $m_1 \mid \cdots \mid m_k$ tel que pour tout $j \in [1..k]$, m_j est un motif conjonctif ; on dit alors que $expr$ est un *motif disjonctif* ;

Exemple 2 Soit un ensemble d'étiquettes $\mathcal{E}_1 = E_1 \cup \overrightarrow{E}_1$ où :
$E1 = \{doc, editeur, nom, article, book, auteur, titre, date, isbn\}$, et
$\overrightarrow{E}1 = \{\overrightarrow{edite}, \overrightarrow{cite}, \overrightarrow{editepar}\}$
Soit un ensemble $\mathcal{V}1 = \{Root, Editeur, Doc, Art, Book, Nom, Dat, Isb\}$ de symboles non terminaux. Voici quelques exemples de motifs :

1) $(doc\ Doc)^*$ est un motif élémentaire qui va servir à exprimer, lorsqu'il "filtre" un graphe en un état s, que zéro ou plusieurs arcs étiquetés par doc doivent sortir de cet état s. Les motifs $(doc\ Doc)^+$, $(doc\ Doc)^!$ et $(doc\ Doc)^?$ sont des variantes du motif $(doc\ Doc)^*$ qui permettent d'exprimer des contraintes de cardinalité sur le nombre d'arcs sortant de s (au moins 1, exactement 1, zéro ou 1).

2) $(auteur\ Nom)^+, (date\ Dat)^!$ est un motif conjonctif et sert à exprimer qu'au moins un arc étiqueté par $auteur$ et exactement un arc étiqueté par $date$ doivent sortir de l'état s.

3) $(article\ Art)^! \mid (book\ Book)^!$ est un motif disjonctif et permet d'exprimer qu'ou bien exactement un arc étiqueté par $article$ ou bien exactement un arc étiqueté par $book$ doit sortir de l'état s.

4) $((auteur\ \ Nom)^+ \mid (isbn\ \ Isb)^!), (\overrightarrow{cite}\ \ Doc)^*$ n'est pas un motif valide mais ce qu'il exprime (potentiellement) peut être formulé par $(auteur\ Nom)^+, (\overrightarrow{cite}\ Doc)^* \mid (isbn\ Isb)^!, (\overrightarrow{cite}\ Doc)^*$.

Une grammaire de motifs va être définie de façon classique comme un ensemble de règles qui sont définies par :

Définition 2 (Règle) Une règle est une expression de la forme $N ::= expr$ où N est un symbole non terminal et $expr$ est un motif.

Exemple 2 (suite) Voici un ensemble de règles dont certaines utilisent les motifs présentés dans l'exemple 2 :

$$\begin{array}{rrl}
\mathcal{R}_1 = \{ & Root ::= & (doc\ Doc)^*, (editeur\ Editeur)^* \\
& Editeur ::= & (nom\ Nom)^!, (\overrightarrow{edite}\ Book)^* \\
& Doc ::= & (article\ Art)^! \mid (book\ Book)^! \\
& Art ::= & (auteur\ Nom)^+, (titre\ Nom)^!, (date\ Dat)^?, (\overrightarrow{cite}\ Doc)^* \\
& Book ::= & (isbn\ Isb)^!, (\overrightarrow{cite}\ Doc)^* \mid (auteur\ Nom)^+, (date\ Dat)^!, \\
& & (titre\ Nom)^!, (\overrightarrow{cite}\ Doc)^*, (\overrightarrow{editepar}\ Editeur)^! \\
& Nom ::= & \Lambda \quad Dat ::= \Lambda \quad Isb ::= \Lambda\ \}
\end{array}$$

Une grammaire de motifs est un ensemble de règles, sous certaines conditions que voici :

Définition 3 (Grammaire de motifs)
Une *grammaire de motifs* $\mathcal{G}$ est un quadruplet $(\mathcal{V}, Root, \mathcal{E}, \mathcal{R})$ où

– $\mathcal{R}$ est un ensemble fini de règles constitué d'une et une seule règle pour chaque symbole non terminal de $\mathcal{V}$;

– pour tout couple de motif élémentaires $(e_1\ N_1)^{op_1}$ et $(e_2\ N_2)^{op_2}$ ayant une occurrence dans $\mathcal{R}$, $e_1 = e_2$[2] implique $N_1 = N_2$;

– il n'existe pas dans $\mathcal{R}$ de motif élémentaire $(e\ Root)^{op}$ (e et op quelconques) et $Root$ est le symbole d'entrée de la grammaire.

Exemple 2 (suite) L'ensemble $\mathcal{R}_1$ de règles constitue, avec l'ensemble de symboles non terminaux $\mathcal{V}_1$ et l'ensemble d'étiquettes $\mathcal{E} = E_1 \cup \overrightarrow{E}_1$, une grammaire de motifs $(\mathcal{V}_1, Root, \mathcal{E}_1, \mathcal{R}_1)$ appelée $\mathcal{G}_1$ dans la suite. Remarquez qu'il existe bien une règle associée à chaque symbole de $\mathcal{V}_1$. Notez que l'étiquette $date$ est utilisée dans deux motifs ayant des opérateurs de cardinalité différents : $(date\ Dat)^!$ et $(date\ Dat)^?$. Notez également que différentes étiquettes sont associées à un même symbole non terminal, par exemple, $(auteur\ Nom)^+$ et $(titre\ Nom)^!$.
Si la règle associée à Art était $Art ::= (titre\ Nom)^!, (\overrightarrow{cite}\ Book)^*$ alors $\mathcal{R}_1$ contiendrait à la fois un motif $(\overrightarrow{cite}\ Book)^*$ dans la règle de Art et un motif $(\overrightarrow{cite}\ Doc)^*$ dans la règle de $Book$, ce qui n'est pas autorisé par la condition (3) de la définition de grammaire de motifs (définition 3).

De façon plus générale, notez que le fait d'interdire la présence de deux règles $N ::= expr_1$ et $N ::= expr_2$ ayant même partie gauche dans une grammaire n'est pas restrictif car ces deux règles peuvent être exprimées par l'unique règle $N ::= expr_1 \mid expr_2$.

Notations : Etant donné $e \in \mathcal{E}$, $Symb(e)$ est l'unique symbole non terminal $N \in \mathcal{V}$ tel que $(e\ N)^{op}$ (*op* quelconque) est un motif apparaissant dans $\mathcal{R}$. Etant donné un

2. Ici e_1 et e_2 sont des étiquettes ou des références.

symbole non terminal $N \in \mathcal{V}$, $Label(N)$ est l'ensemble des étiquettes e de $\mathcal{E}$ telles que $Symb(e) = N$. Par exemple pour $\mathcal{G}_1$, $Symb(\overrightarrow{cite})$=$Doc$, $Symb(date)$=Dat, $Label(Dat) = \{date\}$ et $Label(Nom) = \{auteur, titre, nom\}$. $Expr(N)$ dénote la partie droite de la règle associée au symbole non terminal N dans la grammaire. Par exemple pour $\mathcal{G}_1$, $Expr(Root)$ est le motif $(doc\ Doc)^*, (editeur\ Editeur)^*$ et $Expr(Nom)$ est le motif vide Λ.

On peut associer de manière très simple à une grammaire de motifs $\mathcal{G}$ un graphe de dépendance (entre symboles non terminaux). Les sommets de ce graphe sont les symboles non terminaux et il existe un arc de N à M si M à une occurrence dans $Expr(N)$ dans un motif élémentaire $(e\ M)^{op}$ où $e \in E$. Nous insistons ici sur le fait que le graphe de dépendance est construit à partir de la gammaire en faisant abstraction des motifs élémentaires utilisant une référence (une étiquette de $\overrightarrow{E}$).

Exemple 2 (suite) Le graphe de dépendance associé à la grammaire $\mathcal{G}_1$ de l'exemple 2 est dessiné en figure 2.a. Dans ce graphe, on remarque que le sommet associé à Doc n'est pas accessible du sommet associé à $Book$ car il n'existe aucun motifs élémentaire $(e\ M)^{op}$ où $e \in E$ dans $Expr(Book)$. L'expression $Expr(Book)$ contient bien le motif élémentaire $(\overrightarrow{cite}\ Doc)^*$ mais $\overrightarrow{cite}$ appartenant à $\overrightarrow{E}$, ce motif ne permet pas "d'avoir un arc de $Book$ à Doc" dans le graphe de dépendance associé à $\mathcal{G}_1$.

Nous pouvons maintenant définir un schéma de données semi-structurées comme une grammaire de motifs particulière :

Définition 4 (Schéma) Un schéma de données semi-structurées est une grammaire de motifs $\mathcal{G} = (\mathcal{V}, Root, E \cup \overrightarrow{E}, \mathcal{R})$ telle que pour tout symbole non terminal N de $\mathcal{V}$ il existe un chemin de $Root$ à N dans le graphe de dépendance de la grammaire $\mathcal{G}$.

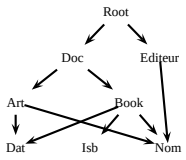

Figure 2.a
Graphe de dépendance associé à la grammaire $\mathcal{G}_1$

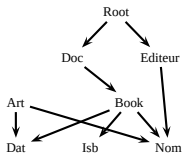

Figure 2.b
Graphe de dépendance associé à la grammaire $\mathcal{G}_2$

Figure 2. *Graphe de dépendance associé aux la grammaires $\mathcal{G}_1$ et $\mathcal{G}_2$*

Exemple 2 (suite) La grammaire de motifs $\mathcal{G}_1$ est un schéma de données : tous les sommets de son graphe de dépendance (représenté par la figure 2.a) sont accessibles à partir du sommet associé à $Root$. Par contre, considérons la grammaire $\mathcal{G}_2 = (\mathcal{V}_1, Root, \mathcal{E}_1, \mathcal{R}_2)$ telle que les règles de $\mathcal{R}_2$ sont les mêmes que celles de $\mathcal{R}_1$ sauf pour Doc dont la règle associée est Doc : :=$(book\ Book)^!$. Le graphe de dépendance de $\mathcal{G}_2$

se trouve en figure 2.b. Le sommet associé au non terminal Art n'est pas accessible à partir du sommet associé à $Root$ donc la grammaire $\mathcal{G}_2$ n'est pas un schéma.

Dans la suite, par abus, nous utilisons indifféremment les termes grammaire de motifs et schéma (en supposant évidemment que la grammaire considérée est un schéma).

Pour définir une instance d'un schéma, nous allons procéder en deux étapes : la première détermine globalement si un graphe de données est filtré par la grammaire de motifs et prépare la vérification ultérieure du "bon typage" des références ; la seconde étape effectue essentiellement la vérification du bon typage des références.

Dans la suite,

– le schéma considéré est la grammaire de motifs $\mathcal{G} = (\mathcal{V}, Root, E \cup \overrightarrow{E}, \mathcal{R})$;

– le graphe de données considéré $\mathcal{S}$ est un graphe (S, r, R, V) enraciné, dont les arcs et les sommets sont étiquetés, S est un ensemble d'états, $R = \{r_e | e \in E \cup \overrightarrow{E}\}$ est un ensemble de relations binaires sur S, r est la racine de $\mathcal{S}$ et V est une fonction de $PROP \rightarrow Pow(S)$; $Out_e(s)$ est l'ensemble des états s_1 tels que $(s, s_1) \in r_e$;

– si U est un ensemble d'états, alors l'expression $|U| \cong card(op)$ signifie : $|U| \geq 0$ si $op = *$, $|U| > 0$ si $op = +$, $|U| = 1$ si $op =!$, $0 \leq |U| \leq 1$ si $op =?$;

– le sous-graphe de données obtenu en supprimant de $\mathcal{S}$ toutes les références est noté $Pre(\mathcal{S})$ et il est défini par : $(S, r, Pre(R), V)$ de $\mathcal{S}$, où $Pre(R) = \{r_e | r_e \in R \; et \; e \in E\}$.

Définition 5 (Filtrage et marquage) On suppose que $Pre(\mathcal{S})$ est acyclique.

Un motif $expr$ **filtre strictement** $\mathcal{S}$ en l'état s ssi le motif $expr$ filtre $\mathcal{S}$ en s et, pour tout $l \in \mathcal{E} - E_{expr}$, où E_{expr} est l'ensemble des étiquettes ayant une occurrence dans $expr$, $Out_l(s) = \emptyset$.

Simultanément au filtrage, nous définissons deux **fonctions de marquage** $marq$ et $requis$ associant à un état de S un ensemble de symboles non terminaux. Initialement, pour tout état s, on pose $marq(s) = \emptyset$, sauf pour $marq(r) = \{Root\}$, et, pour tout état s, on pose $requis(s) = \emptyset$.

Le motif $expr$ **filtre** $\mathcal{S}$ en un sommet s de $\mathcal{S}$ est défini par :

1) $(e \; N)^{op}$ filtre $\mathcal{S}$ en s ssi

- si $e \in E$ alors $|Out_e(s)| \cong card(op)$ et, pour tout $s_1 \in Out_e(s)$, le motif $Expr(N)$ filtre strictement $\mathcal{S}$ en s_1. Pour tout $s_1 \in Out_e(s)$, on définit la nouvelle valeur de $marq(s_1)$ par l'ajout de N à l'ancienne valeur de $marq(s_1)$.

- si $e \in \overrightarrow{E}$ alors $|Out_e(s)| \cong card(op)$ et pour tout $s_1 \in Out_e(s)$, on définit la nouvelle valeur de $requis(s_1)$ par l'ajout de N à l'ancienne valeur de $requis(s_1)$.

2) Λ filtre $\mathcal{S}$ en s ssi quelque soit $e \in \mathcal{E}$, $Out_e(s) = \emptyset$.

3) $(e_1 \; N_1)^{op_1}, \cdots, (e_j \; N_j)^{op_j}$ filtre $\mathcal{S}$ en s ssi pour tout k de $[1..j]$, le motif $(e_k \; N_k)^{op_k}$ filtre $\mathcal{S}$ en s.

4) $expr_1|\cdots|expr_n$ filtre $\mathcal{S}$ en s ssi il existe $i \in [1..n]$ tel que le motif $expr_i$ filtre strictement $\mathcal{S}$ en s.

La condition d'acyclicité sur $Pre(\mathcal{S})$ permet d'assurer que le filtrage est défini correctement. La définition de filtrage pourrait être généralisée pour nous dispenser de cette condition d'acyclicité sur $Pre(\mathcal{S})$, ceci au prix de quelques modifications techniques. Toutefois, étant donné que la définition d'une instance (ci-dessous) impose à $Pre(\mathcal{S})$ d'être un arbre, nous avons choisi de définir le filtrage dans ce cas particulier.

Définition 6 (Instance d'un schéma) Le graphe de données $\mathcal{S}$ est une instance du schéma $\mathcal{G}$ si les conditions ci-dessous sont vérifiées : (i) $Pre(\mathcal{S})$ est un arbre orienté de racine r, (ii) $Expr(Root)$ filtre strictement le graphe de données $\mathcal{S}$ en la racine r, et (iii) Pour tout état s de S, si $requis(s) \neq \emptyset$ alors $requis(s) = marq(s)$.

Exemple 1 (suite) Le graphe de données de la figure 1 est une instance de la grammaire $\mathcal{G}_1$. Ici, $Pre(\mathcal{S})$ est le graphe obtenu à partir de la figure en retirant les arcs pointillés (les références). La figure 3 représente les valeurs des fonctions $marq$ et $requis$ pour ce graphe de données par rapport à la grammaire $\mathcal{G}_1$.

Les valeurs de la fonction $marq$ se trouvent dans les sommets correspondant aux états de la figure 1. Comme le graphe de données la figure 1 est une instance de la grammaire $\mathcal{G}_1$, la valeur de $requis$ est égale à la valeur de $marq$ pour les sommets dans lesquels $requis$ est non vide. On remarque que ces sommets sont les sommets pointés par des références, ceci correspondant bien à la définition de la fonction $requis$.

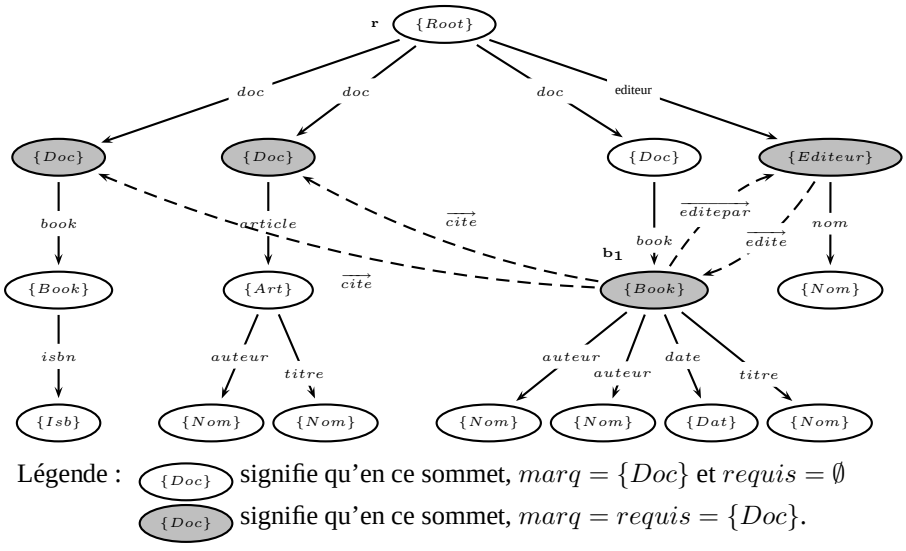

Figure 3. *Valeurs des fonctions $marq$ et $requis$ du graphe de données de la fi gure 1 en fonction de la grammaire $\mathcal{G}_1$*

Les deux propriétés suivantes éclaircissent la signification des deux fonctions $marq$ et $requis$ dont le rôle intuitif est de permettre le "typage" les états référencés.
La première propriété est liée à la structure d'arbre imposée sur $Pre(\mathcal{S})$.

Propriété 1 Si $Pre(\mathcal{S})$ est un arbre orienté de racine r et si $Expr(Root)$ filtre strictement le graphe de données $\mathcal{S}$ en la racine r, alors pour tout état s, $marq(s)$ est un singleton, i.e. l'état s est marqué par un et un seul symbole non terminal.

La deuxième propriété résulte de la restriction (ii) posée sur les règles d'une grammaire de motifs qui nous a permis de définir $Symb(e)$ comme étant l'unique symbole non terminal apparaissant dans des motifs élémentaires associés à l'étiquette e.

Propriété 2 Si $Pre(\mathcal{S})$ est un arbre orienté de racine r et si $Expr(Root)$ filtre strictement le graphe de données $\mathcal{S}$ en la racine r, alors pour tout état s de $\bigcup_{s_1 \in S} out_e(s_1)$, on a $marq(s) = \{Symb(e)\}$.

Cette seconde propriété nous permet de déduire que si $Pre(\mathcal{S})$ est un arbre orienté de racine r et que $Expr(Root)$ filtre strictement le graphe de données $\mathcal{S}$ en la racine r alors un arc entrant dans un sommet s et étiqueté par e détermine la valeur de la fonction $marq$ de ce sommet, et cette valeur est $\{Symb(e)\}$. Ainsi, $Symb(e)$ correspond à un "type" du sommet s.

Exemple 1 (suite) Considérons la figure 3. Un arc étiqueté par $book$ a pour destination b_1, on peut donc en déduire que b_1 "est de type" $Symb(book)$ c'est-à-dire $Book$. Pour vérifier que b_1 est de type $Book$, on vérifie que ses arcs sortant vérifient la règle associée à $Book$ dans $\mathcal{G}$.

On peut remarquer que la notion de schéma proposée permet d'exprimer que, dans toute instance du schéma, tout arc de référence doit être "bien typé" : le sommet destination d'une référence a une valeur de la fonction $requis$ déterminée par le schéma. Formellement, ceci correspond aux conditions (ii) et (iii) de la définition 6. *Cette contrainte de typage de référence n'est pas exprimable par d'autres notions de schéma pour les données semi-structurées présentes dans la littérature, typiquement les DTD.* Par exemple, dans une DTD, on ne pourrait pas indiquer qu'un lien de référence $\overrightarrow{editepar}$ doit pointer vers un éditeur (plutôt que vers un livre, un article, etc).

Pour vérifier que $\mathcal{S}$ est une instance de $\mathcal{G}$ il suffit de vérifier que (1) $Pre(\mathcal{S})$ est un arbre orienté de racine r et (2) pour tout état s de $\mathcal{S}$, si un arc étiqueté par e a pour destination s alors les arcs sortant de s vérifient la règle associée à $Symb(e)$.

Cette dernière remarque peut aider le lecteur à comprendre les intuitions sous-jacentes à la formalisation d'un schéma par une formule de la logique logique multi-modale hybride proposée dans la section suivante.

3.2. *Schéma de données semi-structurées et logique multi-modale hybride*

Dans cette section, nous allons montrer qu'un schéma de données semi-structurées (tel que défini précédemment) peut être exprimé par une formule de la logique multi-modale hybride, comme les contraintes le sont.

Notre objectif est de définir une fonction de traduction $TRAD$ qui, à un schéma $\mathcal{G}$, associe une formule $TRAD(\mathcal{G})$ de la logique multi-modale hybride telle que : pour tout graphe $\mathfrak{M}$ de données semi-structurées, $\mathfrak{M}$ est une instance de $\mathcal{G}$ si et seulement si $\mathfrak{M}$ satisfait la formule $TRAD(\mathcal{G})$. Cette démarche peut être visualisée par le schéma de la figure 4.

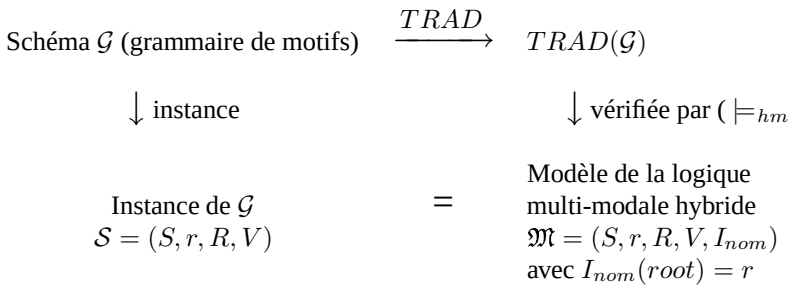

Figure 4. *Principe de la traduction du schéma*

Les deux structures $\mathfrak{M}$ et $\mathcal{S}$ étant très similaires, nous nous permettons de les identifier dans la suite et de les noter $\mathfrak{M}$.

La définition d'une instance d'un schéma fait intervenir une première condition ($Pre(\mathfrak{M})$ est un arbre) qui est en fait indépendante de la grammaire de motifs spécifiant le schéma. Nous allons donc commencer par traduire cette condition par une formule, notée $arbre$, de la logique multi-modale hybride.

Le fait que $Pre(\mathfrak{M})$ est le sous-graphe de $\mathfrak{M}$ obtenu en supprimant les arcs étiquetés par des références est traduit (dans la formule $arbre$) par l'utilisation des opérateurs modaux $\langle e \rangle$ et $[e]$ avec $e \in \mathcal{E}$ exclusivement, et une restriction des opérateurs F et G, dans le même esprit :

Soient $R^E = \bigcup_{e \in E} r_e$ (l'ensemble des relations associées à des étiquettes qui ne sont pas des références) et R^{E^+} la fermeture transitive de R^E. Soit F^E l'opérateur défini par : $\mathfrak{M}, g, s \models_{hm} F^E\varphi$ ssi il existe $s' \in S$ tel que $((s, s') \in R^{E^+}$ et $\mathfrak{M}, g, s' \models_{hm} \varphi)$ De la même façon que G peut être défini par rapport à F, l'opérateur G^E est défini par $G^E\psi =_{def} \neg F^E \neg\psi$.

Pour des raisons de lisibilité, nous utilisons dans la suite les abréviations F^{E*} et G^{E*} définies respectivement par $F^{E*}\psi =_{def} \psi \vee F^E\psi$ et $G^{E*}\psi =_{def} \psi \wedge G^E\psi$.

$Pre(\mathfrak{M})$ est un arbre ssi les trois propriétés ci-dessous sont vérifiées :
(1) tous les états de $Pre(\mathfrak{M})$ sont accessibles à partir de la racine $root$, ce que nous

traduisons en logique multi-modale hybride par la formule $\psi_1 =_{def} \downarrow x\ @_{root} F^{E*} x$;
(2) $Pre(\mathfrak{M})$ ne contient pas de cycle, ce que nous traduisons en logique multi-modale hybride par la formule $\psi_2 =_{def} \downarrow x\ \neg F^{E} x$;
(3) tout état de $Pre(\mathfrak{M})$ a au plus un prédécesseur, ce que nous traduisons en logique multi-modale hybride par la formule $\psi_3 =_{def}$
$@_{root} G^{E*} \downarrow x\ @_{root} G^{E*} \downarrow y\ @_{root} G^{E*} \downarrow z\ (\bigvee_{e \in E} @_y \langle e \rangle x \wedge \bigvee_{e \in E} @_z \langle e \rangle x \rightarrow @_y z)$.

Nous pouvons établir le résultat intermédiaire suivant :

Lemme 1 $Pre(\mathfrak{M})$ est un arbre ssi $\mathfrak{M} \models_{hm} arbre$ où $arbre =_{def} \psi_1 \wedge \psi_2 \wedge \psi_3$.

Sachant que $Pre(\mathfrak{M})$ est un arbre, il convient maintenant d'exprimer les conditions (ii) et (iii) de la définition d'une instance (définition 6). Les deux conditions sont traduites simultanément par la formule $\varphi(\mathcal{G})$

$$@_{root}\left(\varphi_{Root} \wedge \bigwedge_{e \in E} G[e]\varphi_{Symb(e)} \wedge \bigwedge_{\overrightarrow{e} \in \overrightarrow{E}} G[\overrightarrow{e}]\downarrow x \left(\bigvee_{e \in Label(Symb(\overrightarrow{e})) \cap E} @_{root} F\langle e \rangle x\right)\right)$$

où φ_{expr} est définie de la façon suivante :

1) Si $expr$ est Λ alors $\varphi_{expr} = \bigwedge_{e \in \mathcal{E}} \neg\langle e \rangle \top$

2) Si $expr$ est un motif disjonctif $expr_1 | \cdots | expr_j$ alors $\varphi_{expr} = \bigvee_{i \in [1..j]} \varphi_{expr_i}$

3) Si $expr$ est un motif conjonctif[3] $m_1, \cdots, m_j$ alors
$\varphi_{expr} = \bigwedge_{e \in \mathcal{E}_+} \langle e \rangle \top \ \wedge \ \downarrow x \bigwedge_{e \in \mathcal{E}_!} (\langle e \rangle \downarrow y\ (@_x [e] y)) \ \wedge$
$\bigwedge_{e \in \mathcal{E}_?} (\neg\langle e \rangle \top \vee (\downarrow x\ \langle e \rangle \downarrow y\ (@_x [e] y))) \ \wedge \ \bigwedge_{e \in \mathcal{E} - E_{expr}} \neg\langle e \rangle \top$

où $\mathcal{E}_+ = \{e \mid (e\ N)^+\ apparait\ dans\ expr\}$, $\mathcal{E}_! = \{e \mid (e\ N)!\ apparait\ dans\ expr\}$, $\mathcal{E}_? = \{e \mid (e\ N)?\ apparait\ dans\ expr\}$, et E_{expr} l'ensemble des étiquettes apparaissant dans $expr$.

Avant d'énoncer le résultat principal de cette section, nous illustrons la traduction d'un schéma semi-structuré en une formule de la logique multi-modale hybride pour notre exemple courant.

Exemple 1 (suite) La formule associée à la grammaire $\mathcal{G}_1$ définie dans l'exemple 2 est $TRAD(\mathcal{G}_1) = arbre \wedge \varphi(\mathcal{G}_1)$ où

$\varphi(\mathcal{G}_1)$ est $@_{root}$ $(\varphi_{Root} \wedge G[doc]\varphi_{Doc} \wedge G[editeur]\varphi_{Editeur} \wedge G[nom]\varphi_{Nom} \wedge G[article]\varphi_{Art} \wedge G[book]\varphi_{Book} \wedge G[auteur]\varphi_{Nom} \wedge G[titre]\varphi_{Nom} \wedge G[date]\varphi_{Dat} \wedge G[isbn]\varphi_{Isb} \wedge$
$G[\overrightarrow{cite}]\downarrow x\ (@_{root} F\langle doc \rangle x) \wedge G[\overrightarrow{edite}]\downarrow x\ (@_{root} F\langle book \rangle x) \wedge$
$G[\overrightarrow{editepar}]\downarrow x\ (@_{root} F\langle editeur \rangle x))$

avec (de façon non exhaustive)

3. Un motif élémentaire est considéré ici comme étant le cas particulier de motif conjonctif à un élément

...
φ_{Doc} : $\downarrow a\ \langle book\rangle \downarrow b\ (@_a[book]b)$
$\wedge\ \downarrow a\ \langle article\rangle \downarrow b\ (@_a[article]b)$
$\wedge\ \bigwedge_{e\in\mathcal{E}\setminus\{book,article\}} \neg\langle e\rangle\ \top$
...
φ_{Nom} : $\bigwedge_{e\in\mathcal{E}} \neg\langle e\rangle\ \top$

L'exemple principal associé au graphe de données présenté dans la figure 1 est un modèle de $TRAD(\mathcal{G}_1)$ donc ce graphe est une instance du schéma $\mathcal{G}_1$.

Nous énonçons maintenant le résultat principal de cette section :

Théorème 3 $\mathfrak{M}$ est une instance de $\mathcal{G}$ ssi $\mathfrak{M} \models_{hm} arbre \wedge \varphi(\mathcal{G})$.

Voici un second exemple illustrant la traduction d'un schéma *récursif*. La grammaire que nous utilisons pour cet exemple décrit un arbre binaire.

Exemple 2 (schéma récursif) Soit la grammaire $\mathcal{G}_3 = (\mathcal{V}_3, Root, \mathcal{E}_3, \mathcal{R}_3)$
où $\mathcal{V}_3 = \{Root, Abr, Feuil\}$, $\mathcal{E}_3 = \{arbre, ag, ad, feuille\}$ et
$R_3 = \{$ Root ::= $(arbre\ Abr)^+$
Abr ::= $(ag\ Abr)!, (ad\ Abr)!$
$\mid (feuille\ Feuil)!\ \}$

La formule associée à la grammaire $\mathcal{G}_3$ est $TRAD(\mathcal{G}_3) = arbre \wedge \varphi(\mathcal{G}_3)$ où
$\varphi(\mathcal{G}_3)$ est $@_{root}(\varphi_{Root} \wedge G[ag]\varphi_{Abr} \wedge G[ad]\varphi_{Abr} \wedge G[feuille]\varphi_{Feuil})$
avec φ_{Root} : $\langle arbre\rangle\top \wedge \bigwedge_{e\in\mathcal{E}\setminus\{arbre\}} \neg\langle e\rangle\ \top$

φ_{Abr} : $\downarrow a\ \langle ag\rangle \downarrow b\ (@_a[ag]b) \wedge \downarrow a\ \langle ad\rangle \downarrow b\ (@_a[ad]b) \wedge \bigwedge_{e\in\mathcal{E}\setminus\{ag,ad\}} \neg\langle e\rangle\ \top$
$\vee\ \downarrow a\ \langle feuille\rangle \downarrow b\ (@_a[feuille]b) \wedge \bigwedge_{e\in\mathcal{E}\setminus\{feuille\}} \neg\langle e\rangle\ \top$

φ_{Feuil} : $\bigwedge_{e\in\mathcal{E}} \neg\langle e\rangle\ \top$

Autres approches et perspectives

Dans la section 2, nous avons déjà explicité le lien entre notre approche et le travail de [BUN 98].

Dans [ALE 01], une logique propositionnelle dynamique avec nominaux, notée PDL^{PATH}, est proposée afin d'étudier la formalisation de contraintes sur les chemins pour les données semistructurées. La sémantique de cette logique est définie par des graphes orientés, non ordonnés, enracinés et connexes. D'une part, les auteurs montrent que cette logique permet de formaliser les contraintes d'inclusion mais pas celles dites *lollipop*. D'autre part, ils montrent que le problème de l'implication pour ces contraintes est décidable en temps exponentiel.

– Le langage de la logique logique multi-modale hybride que nous explorons dans le présent article permet d'exprimer des propriétes de graphes que PDL^{PATH} ne peut pas exprimer : ceci grâce à la présence de variables d'état et du binder $\downarrow$. En effet, comme montré dans [ARE 99], la logique multi-modale hybride sans les opérateurs de clôture F et G, sans variables d'états et sans binder $\downarrow$ est décidable. Par contre l'ajout du binder $\downarrow$ rend le problème de la satisfiabilité indécidable : la logique obtenue est équivalente au fragment de la logique classique du premier ordre dit "borné".

– PDL^{PATH} permet d'exprimer la clôture réflexive et transitive de n'importe quelle relation de chemin, et non pas seulement de l'union des relations d'accessibilité comme c'est le cas pour notre langage.

Parmi les perspectives de notre travail, il serait intéressant d'étudier des fragments de la logique multi-modale hybride, interprétés sur des graphes connexes enracinés, et ceci en relation avec PDL^{PATH}. Le fragment contenant F et G mais ne contenant pas le binder $\downarrow$ est un premier candidat. Cette étude pourrait produire en général des informations sur la décidabilité de fragments de logique multi-modale hybride.

En connection avec ce qui précède, nous avons aussi pour objectif d'identifier des fragments intéressants dans le cadre de l'optimisation et du problème de l'inclusion de requêtes XPATH dans la lignée de [DEU 01, CAL 02].

A notre connaissance, notre travail est pionier en ce qui concerne la notion de schéma, notion assez générale pour permettre le typage des références. Les liens entre notre approche et celles de [CAL 98] méritent d'être explorés.

4. Bibliographie

[ABI 97] ABITEBOUL S., VIANU V., « Regular path queries with constraints », *PODS*, 1997, p. 122–133.

[ABI 00] ABITEBOUL S., BUNEMMANN P., SUCIU D., *Data on the web*, Morgan Kauffman, 2000.

[ALE 01] ALECHINA N., DEMRI S., DE RIJKE M., « Path Constraints from a Modal Logic Point of View (extended abstract) », LENZERINI M., NARDI D., NUTT W., SUCIU D., Eds., *Proc. 8th Int. Workshop on Knowledge Representation meets Databases (KRDB'01), Roma, Italy, Sep. 2001*, vol. 45, 2001.

[ARE 99] ARECES C., BLACKBURN P., MARX M., « A Road-map on Complexity for Hybrid Logics », FLUM J., RODRÍGUEZ-ARTALEJO M., Eds., *Computer Science Logic*, n° 1683 LNCS, Springer, 1999, p. 307–321, Proceedings of the 8th Annual Conference of the EACSL, Madrid, September 1999.

[BLA 99] BLACKBURN P., TZAKOVA M., « Hybrid Languages and Temporal Logic », *Logic Journal of the IGPL*, vol. 7, n° 1, 1999, p. 27–54.

[BLA 00] BLACKBURN P., « Representation, Reasoning, and Relational Structures : a Hybrid Logic Manifesto », *Logic Journal of the IGPL*, vol. 8, n° 3, 2000, p. 339–365.

[BUN 98] BUNEMAN P., FAN W., WEINSTEIN S., « Path Constraints on Semistructured and Structured Data », *PODS*, Jun 1998, p. 129–138.

[CAL 98] CALVANESE D., GIACOMO G. D., LENZERINI M., « Semi-structured Data with Constraints and Incomplete Information », *International Workshop on Description Logics (DL'98)*, 1998.

[CAL 99] CALVANESE D., GIACOMO G. D., LENZERINI M., « Queries and Constraints on Semi-structured Data », *Lecture Notes in Computer Science*, vol. 1626, 1999.

[CAL 02] CALVANESE D., LENZERINI M., MOTWANI R., Eds., *XPath containment in the presence of disjunction, DTDs, and variables*, vol. 2572 de *Lecture Notes in Computer Science*, Springer, 2002.

[DEU 01] DEUTSCH A., TANNEN V., « Containment of Regular Path Expressions under Integrity Constraints », *Knowledge Representation Meets Databases*, 2001.

[RAY 01] RAY E. T., *Introduction à XML*, 0'Reilly, 2001.