Semistructured Data and Hybrid Multimodal Logic

Nicole Bidoit
LRI-Université Paris XI, Orsay
bidoit@lri.fr

Guidelines

........................what has been done ....

- initial / naive motivation
- what is hybrid modal logic all about ?
- generalizing DTD to capture reference typing ?
- extended DTD and HML

........................work in progress ....

- a tableau system for finite model checking
- expressing Xpath queries and optimization
- HML and automata ...
- other "modal" dimensions : time, ...

initial / naive motivation

- semistructured data is a labelled graph
- a Kripke model is a labelled graph $+$
  (multimodal interpretation for multimodal logic)

- semistructured data is a Kripke model $=$

........................Let us investigate modeling and reasoning over semistructured data by using modal logic ....

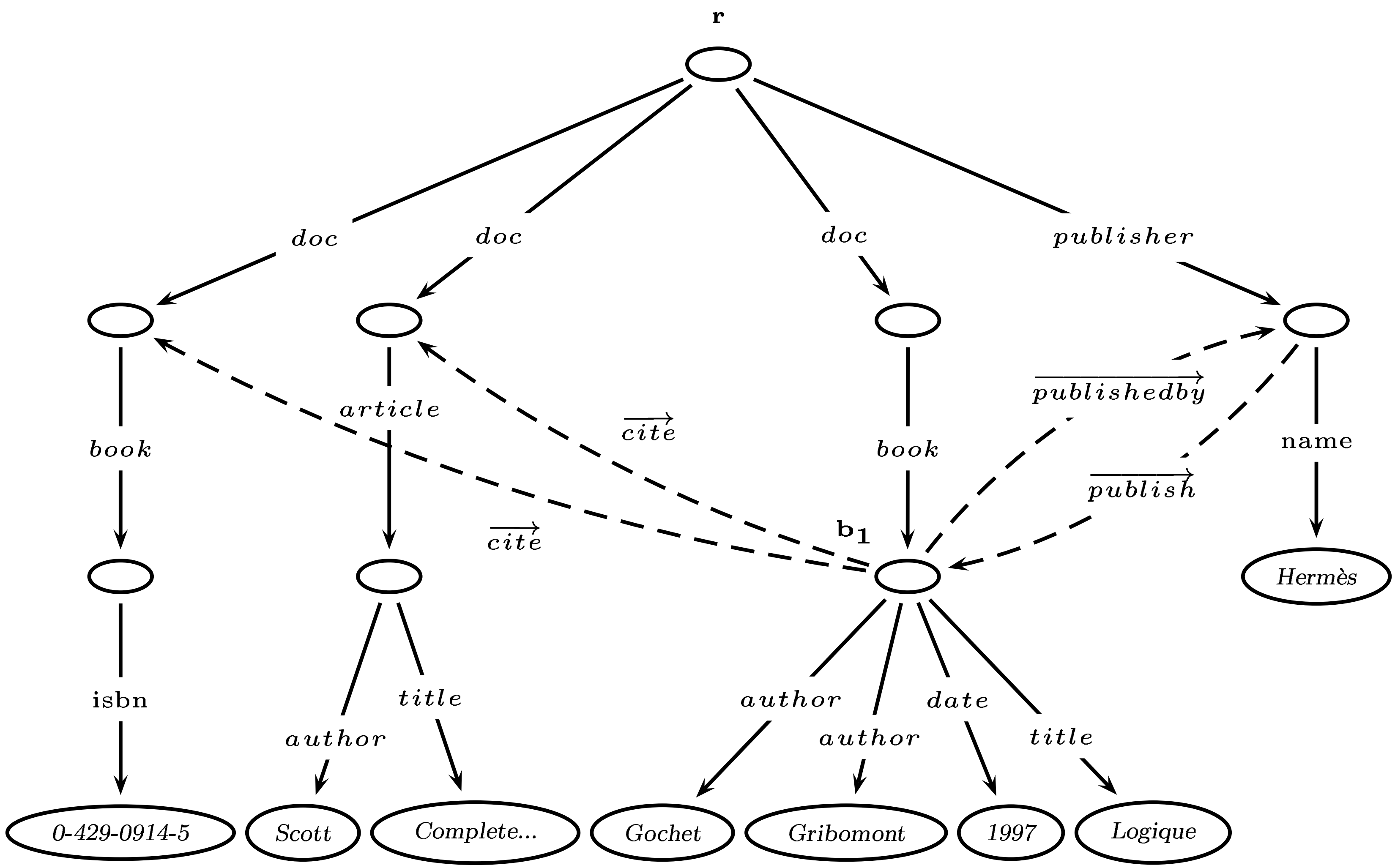

Figure 1: A library database

From modal to hybrid multimodal logic

- Modal propositional logics

  simple languages for talking about any kind of graphs

  tree-structures, transition networks, parse trees,

  networks of properties, ontologies, flows of time, ...

  possible worlds

- Usefuf in a wide range of applications

  (simple syntax, often decidable)

  logics of time, computation, parsing, ... linguistics

- relational structures are ubiquitous

- relational structures are models of classical model theory

  Modal logic is a (decidable) fragment of classical logic

From modal to hybrid multimodal logic

- Syntax

  a set of propositional symbols $p, q, ...,$

  conjunction $\wedge$, negation $\neg$,

  the modalities $[e]$ where $e \in \mathcal{E}$ (finite set of labels),

- Semantics : an internal and local perspective

  To evaluate a formula (satisfaisability)

  one places it inside the model (graph) $\mathfrak{M}$ at some node s

  one is allowed to "scan" nodes **but only those that are accessible** from the current one :

  $\mathfrak{M}, g, s \models [e]\psi$ iff $\forall s'$ such that $(s, s') \in r_e$ we have $\mathfrak{M}, g, s' \models \psi$

  $\mathfrak{M}, g, s \models \langle e \rangle\psi$ iff $\exists s'$ such that $(s, s') \in r_e$ with $\mathfrak{M}, g, s' \models \psi$

From modal to hybrid multimodal logic

- Modal Logics : What exactly is missing ?
  1. Nodes (states) are at the heart of modal logic
  2. Nothing to grips with them

Example : No e-labelled edge from the node s to itself

$\neg\langle e \rangle$??

- Hybrid Modal Logics : What do we need ?

  to deal with nodes **explicitly**

- Syntax

  nominals = names for nodes

  state variables = variables capturing nodes

  move to operator $@_x$ = moves to the node x

  binder $\downarrow x$ = binds x to the current node

  nominals and state variables are formulas

  the move to operator and the binder are new "modalities"

From modal to hybrid multimodal logic

Example : No e-labelled edge from the node s to itself

$$\downarrow x \ \ \neg\langle e\rangle x$$

- Semantics

$\mathfrak{M}, g, s \models a$ iff $I_{nom}(a) = s$ (a is a nominal)

$\mathfrak{M}, g, s \models x$ iff $g(x) = s$ (x is a state variable)

$\mathfrak{M}, g, s \models \downarrow x\ \psi$ iff $\mathfrak{M}, g', s \models \psi$ with $g \overset{x}{\sim} g'$ and $g'(x) = s$

$\mathfrak{M}, g, s \models @_x\psi$ iff $\mathfrak{M}, g, g(x) \models \psi$

$\mathfrak{M}$ is a Kripke structure (a labelled graph), g is a valuation of state variables and s is a node in $\mathfrak{M}$.

Constraints over semistructured data and Hybrid modal Logic

Examples :

$[author]\neg Scott$

$[doc](\langle book\rangle\top \vee \langle article\rangle\top)$

$@_{root}[doc][article]\langle author\rangle\top$

$[publisher][name]Hermès$

$\downarrow x\ \langle\overrightarrow{publishedby}\rangle\langle\overrightarrow{publish}\rangle x$

• First Result: Hybrid multimodal logic subsumes the language $\mathcal{P}$ devised to define **forward and backward constraints**.

Examples : ``given any book x, if x is published by y then y publishes x''.

$\forall xy(\exists z(r_{doc}(root, z) \wedge r_{book}(z, x)) \wedge r_{\overrightarrow{publishedby}}(x, y) \Rightarrow r_{\overrightarrow{publish}}(y, x))$

$@_{root}[doc][book]\downarrow x\ ([\overrightarrow{publishedby}]\langle\overrightarrow{publish}\rangle x)$

Constraints over semistructured data and Hybrid modal Logic

- Second Result: Hybrid multimodal logic is strictly more expressive than the language $\mathcal{P}$ devised to define **forward and backward constraints**.

Examples : `a book has exactly one isbn number`.

$@_{root}[doc][book]{\downarrow}x\ (\langle isbn\rangle{\downarrow}y\ (@_x[isbn]y))$

- Other modalities (behond first order):
  G : accessibility via all path **F** : accessibility via one path

$G\psi \equiv$ at any s accessible via a path from the current state, ψ holds.
$F\psi \equiv$ there is a s accessible from the current state via a path where ψ is satisfied.

Generalizing DTDs

- Main Goal : Typing references

  A schema is specified by a **pattern grammar**

  A datagraph is an instance of a schema if

  Forgetting about the references leads to a tree

  The pattern grammar strictly matches the data graph

  Marks are valids (marking functions are defined during matching)

- Pattern grammar (by example)

$$
\begin{array}{rcl}
Root & ::= & (doc\ Doc)^*, (publisher\ Publisher)^* \\
Publisher & ::= & (name\ Name)^!, (\overrightarrow{publish}\ Book)^* \\
Doc & ::= & (article\ Art)^! \mid (book\ Book)^! \\
Art & ::= & (author\ Name)^+, (title\ Name)^!, (date\ Dat)^?, (\overrightarrow{cite}\ Doc)^* \\
Book & ::= & (isbn\ Isb)^!, (\overrightarrow{cite}\ Doc)^* \mid\ (author\ Name)^+, (date\ Dat)^!, \\
 & & (title\ Name)^!, (\overrightarrow{cite}\ Doc)^*, (\overrightarrow{publisedby}\ Publisher)^! \\
Name & ::= & \Lambda \qquad Dat ::= \Lambda \qquad Isb ::= \Lambda\ \}
\end{array}
$$

Generalizing DTDs

- Pattern grammar (Limitations)

  Root is the start symbol, and

  No pattern $(e\ Root)^{op}$ occurs in the right hand side of rules

  For each couple of patterns $(e_1\ N_1)^{op_1}$ and $(e_2\ N_2)^{op_2}$,

  if $e_1 = e_2$ then $N_1 = N_2$

- What does it implies ?

  Each node of an instance has a unique "type".

- An example of a recursive schema

Root	::=	$(tree\ Tr)^+$
Tr	::=	$(lt\ Tr)!, (rt\ Tr)!\ \mid (leaf\ L)!$

Generalizing DTDs and Hybrid Multimodal logic

- Result :

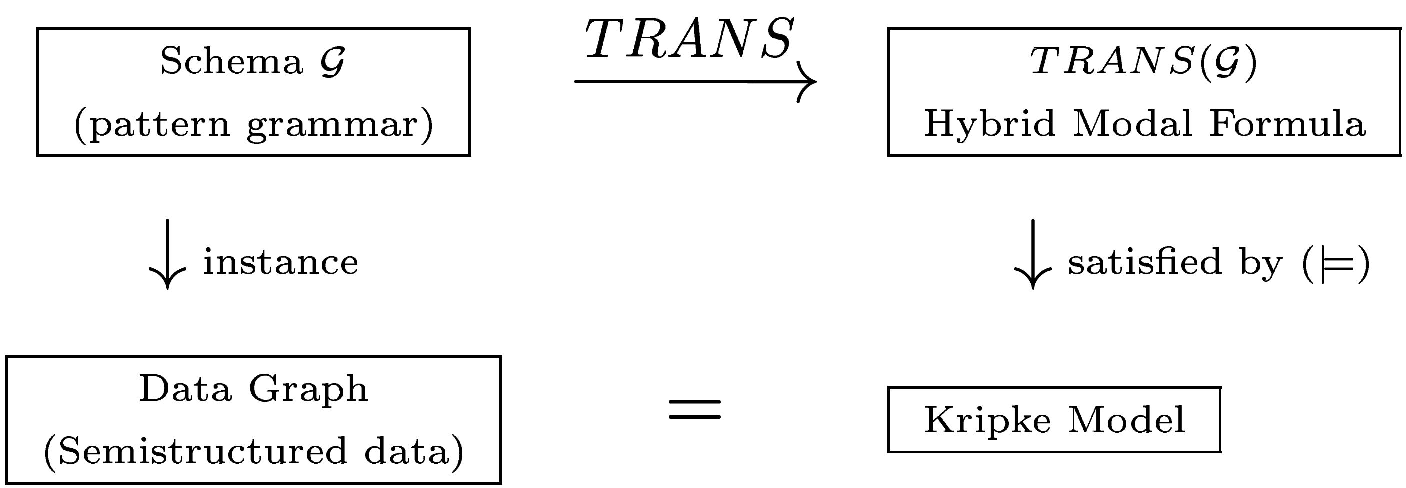

Figure 2: Schema Translation

Generalizing DTDs and Hybrid Multimodal logic

- Translation of Pattern Grammar

  The data graph without references is a tree

  The pattern grammar matches the data graph

$$@_{root}\left(\varphi_{Root} \wedge \bigwedge_{e \in E} G^*[e]\varphi_{Symb(e)}\right)$$

Marks (for reference typing) are valid

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

Generalizing DTDs and Hybrid Multimodal logic — The Example

$$@_{root} \quad (\varphi_{Root} \wedge G^*[doc]\varphi_{Doc} \wedge G^*[publisher]\varphi_{Publisher} \wedge G^*[Name]\varphi_{Name} \wedge G^*[article]\varphi_{Art} \wedge G^*[book]\varphi_{Book} \wedge G^*[auteur]\varphi_{Name} \wedge G^*[title]\varphi_{Name} \wedge G^*[date]\varphi_{Dat} \wedge G^*[isbn]\varphi_{Isb})$$

$$\varphi_{Root} \equiv_{def} \bigwedge_{e \in \mathcal{E} - \{doc, publisher\}} \neg\langle e \rangle \top$$

$$\varphi_{Publisher} \equiv_{def} \downarrow x \ \langle Name \rangle \downarrow y \ (@_x[Name]y) \wedge \bigwedge_{e \in \mathcal{E} - \{Name, \overrightarrow{publish}\}} \neg\langle e \rangle \top$$

$$\varphi_{Doc} \equiv_{def} \downarrow x \ \langle book \rangle \downarrow y \ (@_x[book]y) \wedge \downarrow x \ \langle article \rangle \downarrow y \ (@_x[article]y) \wedge \bigwedge_{e \in \mathcal{E} - \{book, article\}} \neg\langle e \rangle \top$$

$$@_{root} \quad (G^*[\overrightarrow{cite}]\downarrow x \ (@_{root}F^*\langle doc \rangle x) \wedge G^*[\overrightarrow{publish}]\downarrow x \ (@_{root}F^*\langle book \rangle x) \wedge G^*[\overrightarrow{publishedby}]\downarrow x \ (@_{root}F^*\langle publisher \rangle x))$$

A Tableau System for model checking

- Problem

  Given a pattern grammar $\mathcal{G}$ and constraints $\mathcal{C}$

  Find a (finite) instance of $\mathcal{G}$ satisfying $\mathcal{C}$ (if it exists)

- Prefixed Tableau System

  Numerator and denominator of rules are prefixed with
  a "frame" (a pre-instance)

  The pattern grammar is embedded in the proof system

- Limitations

  Non recursice pattern grammar
  $\rightarrow$ enforces the building of finitely deep graphs

  Restriction on the imbrication of $@_x$ and $\downarrow x$
  $\rightarrow$ enforces the building of finitely large graphs