

Towards a Declarative Query and Transformation Language for XML and Semistructured Data: Simulation Unification

François Bry and Sebastian Schaffert

<http://www.pms.informatik.uni-muenchen.de>

Institut für Informatik, LMU München

Introduction

State-of-the-art for XML Query languages (XQuery, XSLT):

- path-oriented navigation through the data
(e.g. `/descendant::a[following-sibling::c]/descendant::b`)
- patterns serve to assemble path-selections

⇒ strong intertwining of query and construction parts

⇒ queries tend to be complicated

Introduction – Example

```
<books-with-prices>
{
  for $b in document("www.bn.com/bib.xml")//book,
    $a in document("www.amazon.com/reviews.xml")//entry
  where $b/title = $a/title
  return
    <book-with-prices>
      { $b/title }
      <price-amazon>{ $a/price/text() }</price-amazon>
      <price-bn>{ $b/price/text() }</price-bn>
    </book-with-prices>
}
</books-with-prices>
```

This XQuery expression creates a summary of prices for books in two bookstores.

Introduction – Xcerpt

- rule-based transformations and queries
- pattern-based (i.e. “context conscious”) instead of path-based selection
- templates for creating results
- strict separation of query and construction parts
- variables relate query and construction parts

Introduction – Example

```
rule { cons {
  books {
    all book {
      TITLE, price-a { PRICEA }, price-b { PRICEB } }
    }
  },
  and {
    query {
      in { "http://bn.com" },
      bib {{
        book {{ TITLE ~> title, price { PRICEA } }}
      }} },
    query {
      in { "http://amazon.com" },
      reviews {{
        entry {{ TITLE ~> title, price { PRICEB } }}
      }} }
  }
}
```

Contents

- Introduction
- Basic Concepts of Xcerpt
- Simulation Unification
 - Simulation
 - Answers
 - Algorithm
- Summary

Xcerpt Concepts

Xcerpt programs build upon the following concepts:

- Database Terms represent semistructured databases
- Query Terms are patterns for querying data
- Construct Terms are templates for the result of a query

Construct-Query Rules relate Query Terms with Construct Terms.

Database Terms

```

bib {
  book {
    title { "TCP/IP Illustrated" },
    authors [ author { last { "Stevens" }, first { "W." } } ],
    publisher { "Addison-Wesley" },
    price { "65.95" }
  },
  book {
    title { "Data on the Web" },
    authors [ author { last { "Abiteboul" }, first { "Serge" } },
              author { last { "Buneman" }, first { "Peter" } },
              author { last { "Suciu" }, first { "Dan" } } ],
    publisher { "Morgan Kaufmann Publishers" },
    price { "39.95" }
  },
  ..
}

```

A database term representing a bibliography. [] denote ordered subterms, { } denote unordered subterms.

Query Terms

Query Terms ...

- specify (incomplete) patterns for the data
- are very similar to goals in logic programming

Query Terms ...

- specify (incomplete) patterns for the data
- are very similar to goals in logic programming

Query Terms may contain the following constructs:

- $l\{t_1, \dots, t_n\}$ and $l[t_1, \dots, t_n]$ denote a total term specification
- $l\{\{t_1, \dots, t_n\}\}$ and $l[[t_1, \dots, t_n]]$ denote a partial specification
- X and $X \rightsquigarrow t$ (read “ X as t ”) are variables, possibly with restriction to t
- $X \rightsquigarrow desc\ t$ denotes a term at indefinite depth

Construct Terms

Construct Terms are templates that ...

- serve to reassemble variables in a new structure (i.e. database term)
- are *templates* that are filled in with values gathered in a query
- contain variables (but no \rightsquigarrow restrictions and no *desc*)

Construct-Query Rules

```
rule { cons {
  books {
    all book {
      TITLE, price-a { PRICEA }, price-b { PRICEB } }
    }
  },
  and {
    query {
      in { "http://bn.com" },
      bib {{
        book {{ TITLE ~> title, price { PRICEA } }}
      }} },
    query {
      in { "http://amazon.com" },
      reviews {{
        entry {{ TITLE ~> title, price { PRICEB } }}
      }} }
  }
}
```

Return a summary of book prices in two bookstores.

Construct-Query Rules

```
rule { cons {
  books {
    all book {
      TITLE, price-a { PRICEA }, price-b { PRICEB } }
    }
  },
  and {
    query {
      in { "http://bn.com" },
      bib {{
        book {{ TITLE ~> title, price { PRICEA } }}
      }} },
    query {
      in { "http://amazon.com" },
      reviews {{
        entry {{ TITLE ~> title, price { PRICEB } }}
      }} }
  }
}
```

Query Term

Query Term

Return a summary of book prices in two bookstores.

Construct-Query Rules

```
rule { cons {  
  books {  
    all book {  
      TITLE, price-a { PRICEA }, price-b { PRICEB } }  
    }  
  },  
  and {
```

Construct Term

```
    query {  
      in { "http://bn.com" },  
      bib {{  
        book {{ TITLE ~> title, price { PRICEA } }}  
      }} },  
    query {  
      in { "http://amazon.com" },  
      reviews {{  
        entry {{ TITLE ~> title, price { PRICEB } }}  
      }} }  
  }  
}
```

Return a summary of book prices in two bookstores.

Simulation Unification

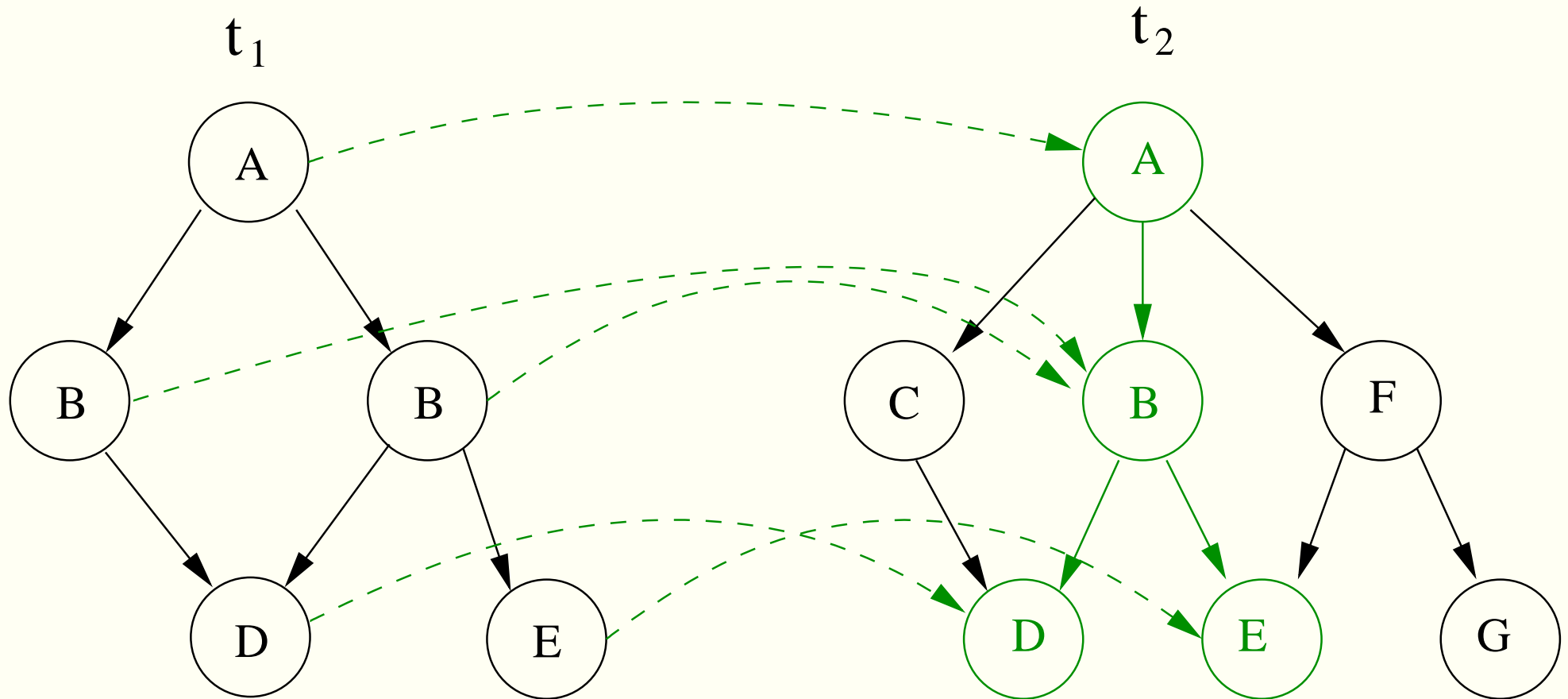
XML terms require a non-standard form of unification:

- ordered/unordered term specification has to be respected
- partial/total term specification suggests using an order instead of an equality between two terms that are unified
- the descendant construct needs a special treatment

Simulation Unification is a non-standard unification which fulfills these requirements.

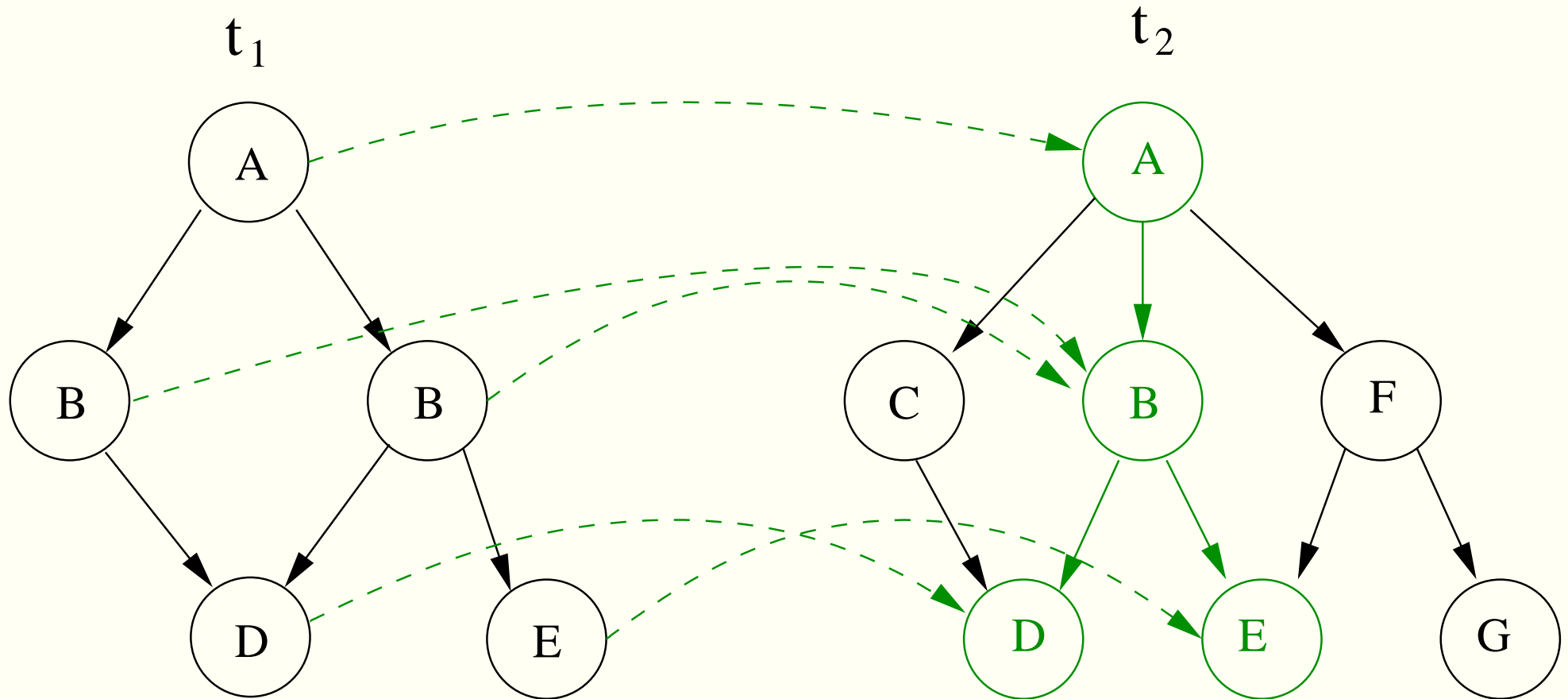
Simulation Unification is based on the notion of a *Simulation* of one term in another.

Graph Simulation (1)



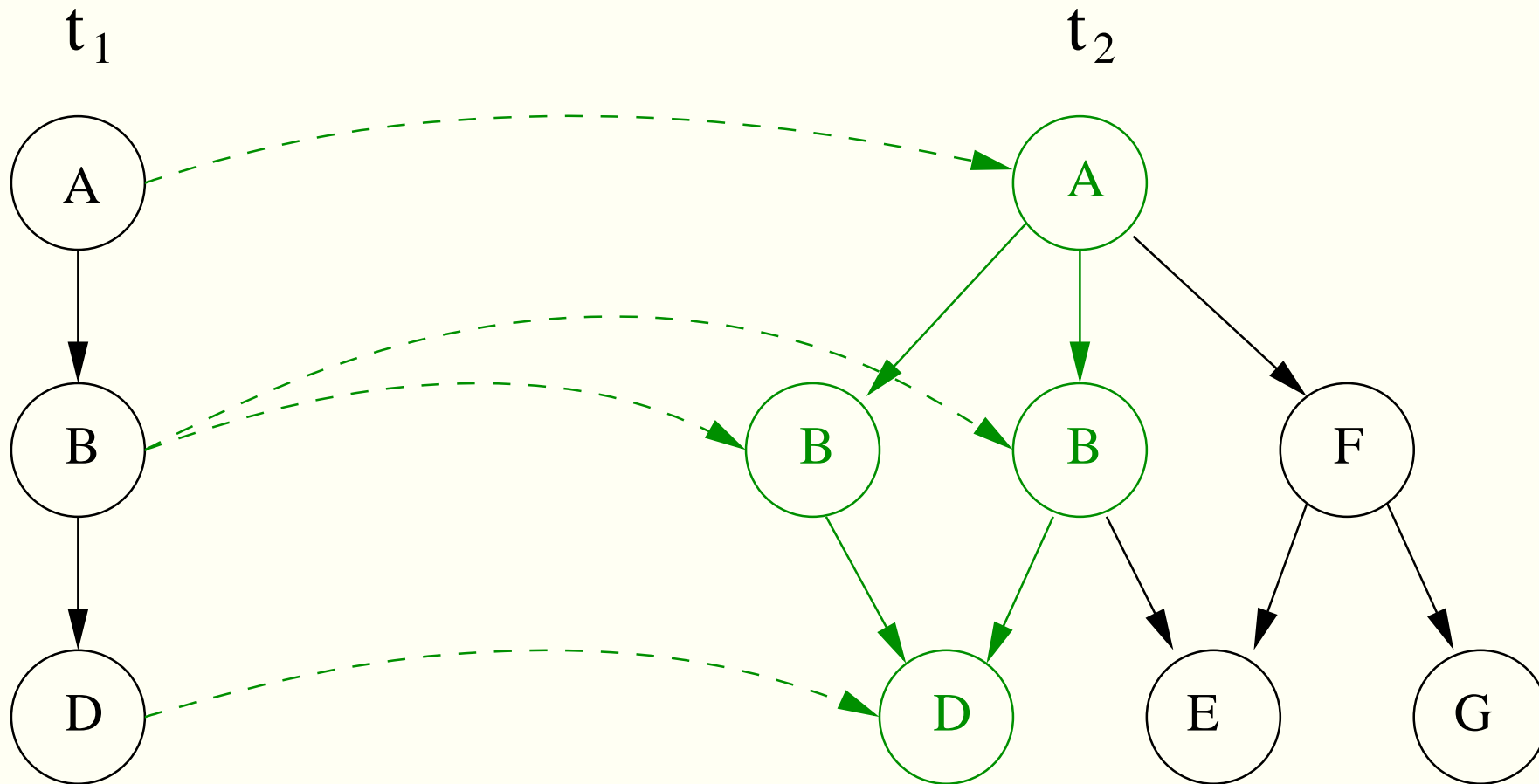
There exists a simulation from t_1 into t_2 : $t_1 \preceq t_2$.

Graph Simulation (1)



There exists a simulation from t_1 into t_2 : $t_1 \preceq t_2$.

Graph Simulation (2)



A non-minimal simulation from t_1 into t_2 .

Simulation Unifier (1)

A simulation unification is initialised with an inequation between a query term and a construct term:

$$t_q \preceq t_c$$

Simulation Unifier (1)

A simulation unification is initialised with an inequation between a query term and a construct term:

$$t_q \preceq t_c$$

A unifier is a substitution σ such that for all grounding substitutions τ of t_c :

- $t_q \sigma \tau \preceq t_c \sigma \tau$
- for all subterms $X \rightsquigarrow t$ of t_q : $t \sigma \tau \preceq X \sigma \tau$

Simulation Unifier (2)

Example:

$$\begin{array}{l} f \{ \{ \\ \quad X \rightsquigarrow g \{ \{ b \} \}, \\ \quad X \rightsquigarrow g \{ \{ c \} \} \\ \} \} \end{array}$$
$$\begin{array}{l} f \{ \\ \quad g \{ a, b, c \}, \\ \quad h \\ \} \end{array}$$

Unifiers for these two terms are: $X = g\{a, b, c\}$ and $X = g\{b, c\}$

Answers

An answer in a database $D \subseteq \mathcal{T}_{db}$ to a query term t_q is a database term $t \in D$ such that there exists a simulation unifier σ with $t_q\sigma \preceq t$.

Example:

$$f \left\{ \left\{ \begin{array}{l} x \rightsquigarrow g \left\{ \left\{ b \right\} \right\}, \\ x \rightsquigarrow g \left\{ \left\{ c \right\} \right\} \end{array} \right\} \right\}$$

$$D = \left\{ \begin{array}{l} f \left\{ g \{ a, b, c \}, h \right\}, \\ f \left\{ g \{ b \}, g \{ c \} \right\} \end{array} \right\}$$

The only answer to the left query term in the database D is $f\{g\{a, b, c\}, h\}$.

- 1. Initialisation:** $C := t^q \preceq t^c$
(with t^q query term, t^c construct term and t^q, t^c variable disjoint).
- 2. Term Decomposition:**
Until C can no longer be modified, repeat performing one of:
 - Apply a (applicable) Term Decomposition rule to C
 - Put C in disjunctive normal form
- 3. Variable Binding:**
Replace each $X \preceq t$ in C with $X = t$.
- 4. Consistency Verification:**
For each disjunct D of C and for each variable X occurring in D do:
Replace in D the equations $X = t_1, \dots, X = t_n$ by $X = t$ where t is a (commutative) unifier of $\{t_1, \dots, t_n\}$.

Term Decomposition

Root Elimination:

$$l\{\{t_1^1, \dots, t_n^1\}\} \preceq l\{t_1^2, \dots, t_m^2\} \Leftrightarrow \bigvee_{\pi \in \Pi} \bigwedge_{1 \leq i \leq n} t_i^1 \preceq \pi(t_i^1) \\ \text{if } n, m \geq 1$$

where Π is the set of total functions $\{t_1^1, \dots, t_n^1\} \rightarrow \{t_1^2, \dots, t_m^2\}$.

Term Decomposition

Root Elimination:

$$l\{\{t_1^1, \dots, t_n^1\}\} \preceq l\{t_1^2, \dots, t_m^2\} \Leftrightarrow \bigvee_{\pi \in \Pi} \bigwedge_{1 \leq i \leq n} t_i^1 \preceq \pi(t_i^1) \text{ if } n, m \geq 1$$

\rightsquigarrow Elimination:

$$X \rightsquigarrow t^1 \preceq t^2 \Leftrightarrow t^1 \preceq t^2 \wedge t^1 \preceq X \wedge X \preceq t^2$$

where Π is the set of total functions $\{t_1^1, \dots, t_n^1\} \rightarrow \{t_1^2, \dots, t_m^2\}$.

Term Decomposition

Root Elimination:

$$l\{\{t_1^1, \dots, t_n^1\}\} \preceq l\{t_1^2, \dots, t_m^2\} \Leftrightarrow \bigvee_{\pi \in \Pi} \bigwedge_{1 \leq i \leq n} t_i^1 \preceq \pi(t_i^2) \\ \text{if } n, m \geq 1$$

\rightsquigarrow Elimination:

$$X \rightsquigarrow t^1 \preceq t^2 \Leftrightarrow t^1 \preceq t^2 \wedge t^1 \preceq X \wedge X \preceq t^2$$

Descendant Elimination:

$$\text{desc } t^1 \preceq l_2\{t_1^2, \dots, t_m^2\} \Leftrightarrow t^1 \preceq l_2\{t_1^2, \dots, t_m^2\} \vee \bigvee_{1 \leq i \leq m} \text{desc } t^1 \preceq t_i^2 \\ \text{if } m \geq 0$$

where Π is the set of total functions $\{t_1^1, \dots, t_n^1\} \rightarrow \{t_1^2, \dots, t_m^2\}$.

Summary

This talk presented the language Xcerpt:

- Xcerpt is a rule-based query and transformation language for XML and SSD
- Evaluation is based on ideas from logic programming
- XML data can be represented as terms but need a quite different treatment
- Simulation Unification is a non-standard unification algorithm that meets the requirements of XML data

End

Thank you!

Xcerpt is an ongoing research project. For more information,
please visit <http://www.xcerpt.org>

Root Elimination:

$$(1) \quad l \preceq l\{t_1^2, \dots, t_m^2\} \Leftrightarrow \text{true} \quad \text{if } m \geq 1$$

$$l \preceq l\{\} \Leftrightarrow \text{true}$$

$$l\{\} \preceq l\{t_1^2, \dots, t_m^2\} \Leftrightarrow \text{false} \quad \text{if } m \geq 1$$

$$l\{\} \preceq l \Leftrightarrow \text{true}$$

$$l\{\} \preceq l\{\} \Leftrightarrow \text{true}$$

$$(2) \quad l\{\{t_1^1, \dots, t_n^1\}\} \preceq l \Leftrightarrow \text{false} \quad \text{if } n \geq 1$$

$$l\{\{t_1^1, \dots, t_n^1\}\} \preceq l\{\} \Leftrightarrow \text{false} \quad \text{if } n \geq 1$$

$$l\{t_1^1, \dots, t_n^1\} \preceq l \Leftrightarrow \text{false} \quad \text{if } n \geq 1$$

$$l\{t_1^1, \dots, t_n^1\} \preceq l\{\} \Leftrightarrow \text{false} \quad \text{if } n \geq 1$$

Term Decomposition (2)

$$(3) \quad l\{\{t_1^1, \dots, t_n^1\}\} \preceq l\{t_1^2, \dots, t_m^2\} \Leftrightarrow \bigvee_{\pi \in \Pi} \bigwedge_{1 \leq i \leq n} t_i^1 \preceq \pi(t_i^1) \quad \text{if } n, m \geq 1$$

$$l\{t_1^1, \dots, t_n^1\} \preceq l\{t_1^2, \dots, t_m^2\} \Leftrightarrow \bigvee_{\pi \in \Phi} \bigwedge_{1 \leq i \leq n} t_i^1 \preceq \pi(t_i^1) \quad \text{if } n, m \geq 1$$

$$(4) \quad l_1\{\{t_1^1, \dots, t_n^1\}\} \preceq l_2\{t_1^2, \dots, t_m^2\} \Leftrightarrow \text{false} \quad \text{if } l_1 \neq l_2 \ (n, m \geq 0)$$

$$l_1\{t_1^1, \dots, t_n^1\} \preceq l_2\{t_1^2, \dots, t_m^2\} \Leftrightarrow \text{false} \quad \text{if } l_1 \neq l_2 \ (n, m \geq 0)$$

where Π (Φ , resp.) is the set of total (total surjective, resp.) functions $\{t_1^1, \dots, t_n^1\} \rightarrow \{t_1^2, \dots, t_m^2\}$.

↷ **Elimination:**

$$X \rightsquigarrow t^1 \preceq t^2 \quad \Leftrightarrow \quad t^1 \preceq t^2 \wedge t^1 \preceq X \wedge X \preceq t^2$$

Descendant Elimination:

$$\text{desc } t^1 \preceq l_2\{t_1^2, \dots, t_m^2\} \Leftrightarrow t^1 \preceq l_2\{t_1^2, \dots, t_m^2\} \vee \bigvee_{1 \leq i \leq m} \text{desc } t^1 \preceq t_i^2$$

if $m \geq 0$

Initialisation:

$$C = f\{\{X \rightsquigarrow g\{\{b\}\}, X \rightsquigarrow g\{\{c\}\}\}\} \preceq f\{g\{a, b, c\}, h\}$$

Root Elimination (3):

$$\begin{aligned} C = & (X \rightsquigarrow g\{\{b\}\} \preceq g\{a, b, c\} \wedge X \rightsquigarrow g\{\{c\}\} \preceq g\{a, b, c\}) \vee \\ & (X \rightsquigarrow g\{\{b\}\} \preceq g\{a, b, c\} \wedge X \rightsquigarrow g\{\{c\}\} \preceq h) \vee \\ & (X \rightsquigarrow g\{\{b\}\} \preceq h \wedge X \rightsquigarrow g\{\{c\}\} \preceq g\{a, b, c\}) \vee \\ & (X \rightsquigarrow g\{\{b\}\} \preceq h \wedge X \rightsquigarrow g\{\{c\}\} \preceq h) \end{aligned}$$

\rightsquigarrow -Elimination

$$\begin{aligned}
C = & (g\{\{b\}\} \preceq g\{a, b, c\} \wedge g\{\{b\}\} \preceq X \wedge X \preceq g\{a, b, c\} \wedge \\
& g\{\{c\}\} \preceq g\{a, b, c\} \wedge g\{\{c\}\} \preceq X \wedge X \preceq g\{a, b, c\}) \vee \\
& (g\{\{b\}\} \preceq g\{a, b, c\} \wedge g\{\{b\}\} \preceq X \wedge X \preceq g\{a, b, c\} \wedge \\
& g\{\{c\}\} \preceq h \wedge g\{\{c\}\} \preceq X \wedge X \preceq h) \vee \\
& (g\{\{b\}\} \preceq h \wedge g\{\{b\}\} \preceq X \wedge X \preceq h \wedge \\
& g\{\{c\}\} \preceq g\{a, b, c\} \wedge g\{\{c\}\} \preceq X \wedge X \preceq g\{a, b, c\}) \vee \\
& (g\{\{b\}\} \preceq h \wedge g\{\{b\}\} \preceq X \wedge X \preceq h \wedge \\
& g\{\{c\}\} \preceq h \wedge g\{\{c\}\} \preceq X \wedge X \preceq h)
\end{aligned}$$

Decomposition

$$\begin{aligned}
C = & (true \wedge g\{b\} \preceq X \wedge X \preceq g\{a, b, c\} \wedge \\
& true \wedge g\{c\} \preceq X \wedge X \preceq g\{a, b, c\}) \vee \\
& (true \wedge g\{b\} \preceq X \wedge X \preceq g\{a, b, c\} \wedge \\
& false \wedge g\{c\} \preceq X \wedge X \preceq h) \vee \\
& (false \wedge g\{b\} \preceq X \wedge X \preceq h \wedge \\
& true \wedge g\{c\} \preceq X \wedge X \preceq g\{a, b, c\}) \vee \\
& (false \wedge g\{b\} \preceq X \wedge X \preceq h \wedge \\
& false \wedge g\{c\} \preceq X \wedge X \preceq h)
\end{aligned}$$

Simplification

$$\begin{aligned}
C &= (g\{\{b\}\} \preceq X \wedge X \preceq g\{a, b, c\} \wedge \\
&\quad g\{\{c\}\} \preceq X \wedge X \preceq g\{a, b, c\}) \vee \\
&\quad \textit{false} \vee \\
&\quad \textit{false} \vee \\
&\quad \textit{false}
\end{aligned}$$

Simplification

$$\begin{aligned}
C &= (g\{\{b\}\} \preceq X \wedge X \preceq g\{a, b, c\} \wedge \\
&\quad g\{\{c\}\} \preceq X \wedge X \preceq g\{a, b, c\})
\end{aligned}$$

Variable Binding

$$C = (g\{\{b\}\} \preceq X \wedge X = g\{a, b, c\} \wedge g\{\{c\}\} \preceq X \wedge X = g\{a, b, c\})$$

Simulation Unifier

$$\sigma = \{ X = g\{a, b, c\} \}$$