XML Schemas Admitting 1-Pass Preorder Typing

Wim Martens       Frank Neven       Thomas Schwentick
Outline

- XML Schema Languages
- Single-Type SDTDs
- 1-Pass Preorder Typing
- Restrained Competition SDTDs
- Unique Particle Attribution vs 1PPT
- Conclusion
XML Schema Languages: DTDs

- DTDs (Document Type Definitions):

  - store → whisky whisky*
  - whisky → name price
XML Schema Languages: DTDs

- DTDs (Document Type Definitions):

```
store → whisky whisky*  
whisky → name price
```

```
store
   └── whisky
        └── name
            └── "Ardbeg"
        └── price
            └── 45
whisky
   └── name
        └── "Glenfarclas"
   └── price
        └── 31
```
XML Schema Languages: DTDs

Regular expressions should be deterministic

Backward compatibility with SGML:
“It is an error if an element in the document can match more than one occurrence of an element type in the content model [without looking ahead].”

Example:
\( bc + bd \). Where do we match \( b \) in the string \( bd \)? \( b(c + d) \) is deterministic.

Purpose: facilitate validation!
XML Schema Languages: DTDs

Regular expressions should be deterministic

- Backward compatibility with SGML:
  “It is an error if an element in the document can match more than one occurrence of an element type in the content model [without looking ahead].”

- Example:
  \(bc + bd\). Where do we match \(b\) in the string \(bd\)?
  \(b(c + d)\) is deterministic.

Purpose: facilitate validation!

In which way does this constrain the schemas?
Can you recognize deterministic regular expressions?

What are the properties of deterministic regular expressions?

Is it decidable whether a regexp is equivalent to a deterministic one?
[Brüggemann-Klein, Wood 1998]:

- Can you recognize deterministic regular expressions?
  
  A regular expression is **deterministic** (one-unambiguous) iff its Glushkov automaton is deterministic (PTIME).

- What are the properties of deterministic regular expressions?
  - *Not every regular language can be denoted by a deterministic regular expression.* E.g., \((a + b)^*a(a + b)\).
  
  - *Semantical characterization in terms of orbits*

- Is it decidable whether a regexp is equivalent to a deterministic one?
  
  Yes
Specialized DTDs (SDTDs) [Papakonstantinou, Vianu, 2000]:

≡ tree automata on unranked trees

- store → (whisky$^1$)* whisky$^2$ (whisky$^2$)*
- whisky$^1$ → name price
- whisky$^2$ → name price discount
Specialized DTDs (SDTDs) [Papakonstantinou, Vianu, 2000]:
≡ tree automata on unranked trees

store → (whisky$^1$)$^*$ whisky$^2$ (whisky$^2$)$^*$
whisky$^1$ → name price
whisky$^2$ → name price discount

store
  └── whisky
      └── name
        └── "Ardbeg"
  └── whisky
      └── name
        └── "Glenfarclas"
  └── whisky
      └── name
        └── "Glenkinchie"
  └── whisky
      └── price
        └── 45
  └── whisky
      └── price
        └── 31
  └── whisky
      └── price
        └── 32
      └── discount
        └── "10%"
Specialized DTDs (SDTDs) [Papakonstantinou, Vianu, 2000]:

\[ \equiv \text{tree automata on unranked trees} \]

\[
\begin{align*}
\text{store} & \rightarrow (\text{whisky}^1)^* \text{whisky}^2 (\text{whisky}^2)^* \\
\text{whisky}^1 & \rightarrow \text{name price} \\
\text{whisky}^2 & \rightarrow \text{name price discount}
\end{align*}
\]

Typing: associating the right types to nodes
XML Schema

To facilitate validation/typing:

Element Declarations Consistent Rule (EDC):

“It is illegal to have two elements of the same name [...] but different types in a content model”

[XML Schema Part 0: Primer]

XML Schemas are SDTDs with a single-type restriction

[Murata,Lee,Mani 2001]
EDC in SDTDs

Single-type SDTDs:

Different types for a label in the same rhs are not allowed!

Example:

\[
\text{store} \rightarrow (\text{whisky}^1)^* \text{whisky}^2 (\text{whisky}^2)^* \\
\text{whisky}^1 \rightarrow \text{name}^2 \text{price}^3 \\
\text{not allowed} \\
\text{allowed}
\]
EDC in SDTDs

Single-type SDTDs:

- store → regulars* discounts discounts*
- regulars → whisky¹
- discounts → whisky²
- whisky¹ → name price
- whisky² → name price discount
EDC in SDTDs

Single-type SDTDs:

- store → regulars* discounts discounts*
- regulars → whisky¹
- discounts → whisky²
- whisky¹ → name price
- whisky² → name price discount

```
store
  └── regulars
      │              └── discounts
      │                  │                      └── whisky
      │                  └── whisky
      └── whisky
             └── name price
              └── whisky
                     └── name price discount
                     │                     └── discount
                     └── whisky
                          └── name price
                               └── whisky
                                    └── name price discount
                                    │                     └── discount
                                    └── whisky
                                         └── name price discount
                                             └── discount
```

"Ardbeg" 45  "Glenfarclas" 31  "Glenkinchie" 32  "10%"
EDC in SDTDs

Single-type SDTDs:

store → regulars* discounts discounts*
regulars → whisky
discounts → whisky
whisky → name price
whisky → name price discount

Note: DTD ⊆ single-type SDTD ⊆ SDTD
Questions

- Can you recognize single-type SDTDs?  
  \textit{Trivial}

- What kind of languages can be defined by single-type SDTDs?  
  ???

- Is it decidable whether an SDTD is equivalent to a single-type SDTD?  
  ???
The Ancestor-String

α
Ancestor-Guarded Subtree Exchange

$T$ a regular tree language
The Equivalence

Let $T$ be a regular tree language

**THEOREM:** The following are equivalent:
- $T$ is definable by a **single-type** SDTD
- $T$ is closed under **ancestor-guarded subtree exchange**
Ancestor-Guarded DTDs

Ancestor-guarded DTD consists of triples \((r, a) \rightarrow s\)

\[ \in L(r) \]

\[ \in L(s) \]
Ancestor-Guarded DTDs

Ancestor-guarded DTD consists of triples \((r, a) \rightarrow s\)

Example:

\[
\begin{align*}
(\epsilon, \text{store}) & \rightarrow \text{regulars}^* \text{ discounts discounts}^* \\
(\text{store, regulars}) & \rightarrow \text{whisky} \\
(\text{store, discounts}) & \rightarrow \text{whisky} \\
(\text{store regulars, whisky}) & \rightarrow \text{name price} \\
(\text{store discounts, whisky}) & \rightarrow \text{name price discount}
\end{align*}
\]
The Equivalence

Let $T$ be a regular tree language

**THEOREM:** The following are equivalent:

- $T$ is definable by a single-type SDTD
- $T$ is closed under ancestor-guarded subtree exchange
- $T$ is definable by an ancestor-guarded DTD
Questions

- Can you recognize single-type SDTDs?
  \textit{Trivial}

- What kind of languages can be defined by single-type SDTDs?
  \begin{itemize}
    \item \textit{Semantical characterizations:}
      \begin{itemize}
        \item ancestor-guarded subtree exchange
        \item \ldots
      \end{itemize}
    \item \textit{Syntactical characterizations:}
      \begin{itemize}
        \item ancestor-guarded DTDs
        \item \ldots
      \end{itemize}
  \end{itemize}

- Is it decidable whether an SDTD is equivalent to a single-type SDTD?
Questions

Can you recognize single-type SDTDs?
Trivial

What kind of languages can be defined by single-type SDTDs?

Semantical characterizations:
ancestor-guarded subtree exchange

... 

Syntactical characterizations:
ancestor-guarded DTDs

...

Is it decidable whether an SDTD is equivalent to a single-type SDTD?
Yes, EXPTIME-complete
Outline

- XML Schema Languages
- Single-Type SDTDs
- 1-Pass Preorder Typing
- Restrained Competition SDTDs
- Unique Particle Attribution vs 1PPT
- Conclusion
Optimize EDC?

EDC: type of a node only depends on ancestor-string!
Optimize EDC?

EDC: type of a node only depends on ancestor-string!

XML streaming: determine the type of a node when its opening tag is met

We call this 1-Pass Preorder Typing
Optimize EDC?

EDC: type of a node *only depends on ancestor-string!*

XML streaming: determine the type of a node when its opening tag is met

We call this **1-Pass Preorder Typing**

\[
\begin{align*}
    a & \rightarrow b^1 + b^2 \\
    b^1 & \rightarrow c \\
    b^2 & \rightarrow d
\end{align*}
\]
defines trees \(a\) and \(b^1\) and \(b^2\) and \(d\)

XML Schemas Admitting 1-Pass Preorder Typing – p.18/31
Questions

- Can you recognize 1PPT SDTDs?
  
- What kind of languages can be defined by 1PPT SDTDs?
  
- Is it decidable whether an SDTD is equivalent to a 1PPT SDTD?
Outline

- XML Schema Languages
- Single-Type SDTDs
- 1-Pass Preorder Typing
- Restrained Competition SDTDs
- Unique Particle Attribution vs 1PPT
- Conclusion
Restrained Competition SDTDs

[Murata et al., 2001]: A regular expression $r$ over types restrains competition iff there are no strings $wa^i v$ and $wa^j v'$ in $L(r)$ with $i \neq j$

I.e. the type of $a$ is determined by its left siblings

An SDTD is restrained competition iff every regular expression restrains competition

\[
\begin{align*}
\text{store} & \rightarrow (\text{whisky}^1)^* \text{ discounts} (\text{whisky}^2)^+ \\
\text{discounts} & \rightarrow \varepsilon \\
\text{whisky}^1 & \rightarrow \text{name price} \\
\text{whisky}^2 & \rightarrow \text{name price discount}
\end{align*}
\]

Note: $\text{DTD} \subseteq \text{stSDTD} \subseteq \text{rcSDTD} \subseteq \text{SDTD}$
The Ancestor-Sibling-String
Anc-Sib-Guarded Subtree Exchange

$T$ a regular tree language
Anc-Sib guarded DTD consists of triples \((r, a) \rightarrow s\)
Ancestor-Sibling-Guarded DTDs

Anc-Sib guarded DTD consists of triples \((r, a) \rightarrow s\)

Example:

- \((\varepsilon, \text{store}) \rightarrow \text{whisky} \ast \text{discounts whisky}^+\)
- \((\text{store} \# \text{whisky}^*, \text{discounts}) \rightarrow \varepsilon\)
- \((\text{store} \# \text{whisky}^*, \text{whisky}) \rightarrow \text{name price}\)
- \((\text{store} \# \text{whisky}^* \text{discounts whisky}^*, \text{whisky}) \rightarrow \text{name price discount}\)
The Equivalence

Let $T$ be a regular tree language.

**THEOREM:** The following are equivalent:

- $T$ is definable by a **restrained competition SDTD**
- $T$ is closed under **ancestor-sibling-guarded subtree exchange**
- $T$ is definable by an **ancestor-sibling-guarded DTD**
Let $T$ be a regular tree language.

**THEOREM:** The following are equivalent:
- $T$ is definable by a restrained competition SDTD
- $T$ is closed under ancestor-sibling-guarded subtree exchange
- $T$ is definable by an ancestor-sibling-guarded DTD
- $T$ allows 1-Pass Preorder Typing

1-Pass Preorder Typeable SDTDs are exactly the rcSDTDs!
The Equivalence: 1PPT vs rcSDTD

Intuition: not rcSDTD implies not 1PPT
Suppose we have $x \rightarrow r$

where $w_ai v$ and $w_aj v'$ in $L(r)$ and $i \neq j$
**The Equivalence: 1PPT vs rcSDTD**

Intuition: not rcSDTD implies not 1PPT

Suppose we have \( x \rightarrow r \)

where \( wa^i v \) and \( wa^j v' \) in \( L(r) \) and \( i \neq j \)

Then we can make trees
The Equivalence: 1PPT vs rcSDTD

Intuition: not rcSDTD implies not 1PPT
Suppose we have $x \rightarrow r$
where $wa^i v$ and $wa^j v'$ in $L(r)$ and $i \neq j$

Then we can make trees
Questions

Can you recognize 1PPT SDTDs?

What kind of languages can be defined by 1PPT SDTDs?

- **Semantical characterizations:**
  - restrained competition SDTDs
  - ancestor-sibling guarded subtree-exchange
  - ...

- **Syntactical characterizations:**
  - ancestor-sibling guarded DTD
  - ...

Is it decidable whether an SDTD is equivalent to a 1PPT SDTD?
Questions

- Can you recognize 1PPT SDTDs?
  Yes, in NLOGSPACE

- What kind of languages can be defined by 1PPT SDTDs?
  - **Semantical characterizations:**
    - restrained competition SDTDs
    - ancestor-sibling guarded subtree-exchange
    - ...
  - **Syntactical characterizations:**
    - ancestor-sibling guarded DTD
    - ...

- Is it decidable whether an SDTD is equivalent to a 1PPT SDTD?
  Yes, EXPTIME-complete
Outline

- XML Schema Languages
- Single-Type SDTDs
- 1-Pass Preorder Typing
- Restrained Competition SDTDs
- Unique Particle Attribution vs 1PPT
- Conclusion
Unique Particle Attribution vs 1PPT

Unique Particle Attribution

New name for one-unambiguous or determinism constraint

**Unique Particle Attribution** \(\Rightarrow\) **1PPT**

**Intuition:**

\[
a^1 \cdot b^1 \cdot (b^2 + c^1)^* \cdot a^2 \cdot c^1 \\
\]

\[
a_1 \cdot b_2 \cdot (b_3 + c_4)^* \cdot a_5 \cdot c_6 \\
\]

**rc:** \( wa_i v \in L(r), wa_j v' \in L(r) \Rightarrow i = j \)

**deterministic:** \( wa_i v \in L(r), wa_j v' \in L(r) \Rightarrow i = j \)
Unique Particle Attribution vs 1PPT

Unique Particle Attribution

New name for one-unambiguous or determinism constraint

- Unique Particle Attribution \( \Rightarrow \) 1PPT
  
  Intuition:
  
  \[
  a^1 \? \quad b^1 \quad (b^2 + c^1)^* \quad a^2 \quad c^1 \\
  a_1 \? \quad b_2 \quad (b_3 + c_4)^* \quad a_5 \quad c_6
  \]

- 1PPT \( \not\Rightarrow \) Unique Particle Attribution
  
  Example: \((a^1 + b^1)^* a^1 (a^1 + b^1)\)
Outline

- XML Schema Languages
- Single-Type SDTDs
- 1-Pass Preorder Typing
- Restrained Competition SDTDs
- Unique Particle Attribution vs 1PPT
- Conclusion
Conclusions

- EDC, 1PPT have elegant semantical characterizations
- Characterizations provide a toolbox for proofs
Conclusions

- 1PPT is strictly larger than EDC
- 1PPT is a robust notion
- 1PPT has a syntactical counterpart
- When content models are deterministic:
  - validation against 1PPT SDTD is in linear time, using space linear in depth of the document
  - inclusion/equivalence of 1PPT SDTDs in \( \text{PTIME} \)
  - minimizing is in \( \text{PTIME} \), unique minimal 1PPT SDTD

These are the same complexities as for EDC SDTDs!
Conclusions

- 1PPT is strictly larger than EDC
- 1PPT is a robust notion
- 1PPT has a syntactical counterpart
- When content models are deterministic:
  - validation against 1PPT SDTD is in linear time, using space linear in depth of the document
  - inclusion/equivalence of 1PPT SDTDs in \textit{PTIME}
  - minimizing is in \textit{PTIME}, unique minimal 1PPT SDTD

These are the same complexities as for EDC SDTDs!

- EDC/UPA are currently in XML Schema specification!
- UPA alone already implies 1PPT!