(Co)algebraic specification and its application in XML-based modelling

Jens Lechner

Lehrstuhl 1
Fakultät für Informatik
TU Dortmund

03. April 2016

CMCS 2016

Overview

1. Algebraic Specification of DTDs and Modelling of XML-Documents
2. Algebraic Specification and Modelling of XPath-Queries
3. Difficulty of certain decision problems concerning XPath

DTDs as signatures, CT_Σ as a model of documents

- A *document type definition (DTD)* appears to be a special kind of *context-free grammar (CFG)* (cf. [3]) and therefore it gives raise to a *constructive signature* $\Sigma = (S, \mathcal{I}, C)$ in the sense of [1, 2]. Here S is a finite set of sorts, $\mathcal{I}$ is a countable set of countable sets of indices and C a set of constructors $c : e \to s$ where e a type built up from S, $\mathcal{I}$ and $\mathcal{I}$-indexed sums and products.
- Documents satisfying the DTD are modelled as elements of the set CT_Σ of *finite and infinite ground* Σ*-terms*.

The carrier of CT_Σ

- The carrier of CT_Σ is a set of certain partial functions

$$X^* \multimap \bigcup \mathcal{I} \cup C \cup \{tup\},$$

viewed as edge- and node-labelled trees, where $X = \bigcup \mathcal{I} \cup \{sel\}$.

- Therefore, as set, CT_Σ is isomorphic to a subset of (total) functions

$$\mathbb{N} \longrightarrow \mathbb{N}.$$

Example I

- Consider the DTD

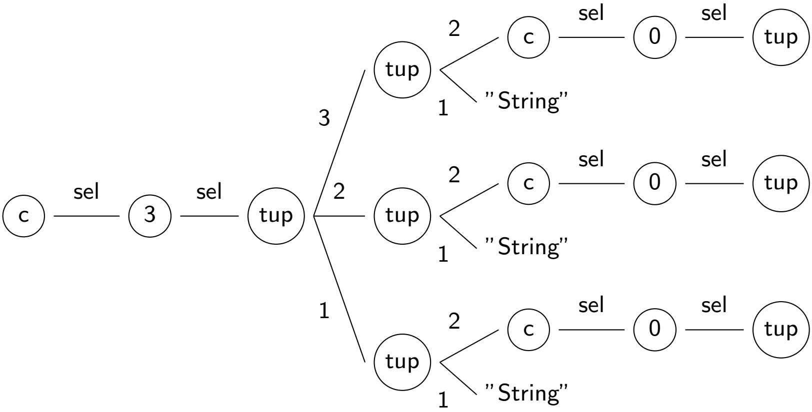
$$\texttt{<!ELEMENT s (String, s)}^{*}\texttt{>}.$$

- This yields the signature

$$\begin{aligned}\Sigma &= (S = \{s\},\ \mathcal{I} = \{\text{String}\},\\ &\quad C = \{c : \coprod_{n\in\mathbb{N}} (\text{String} \times s)^n \to s\}).\end{aligned}$$

Example II

- A document satisfying the DTD, viewed as ground Σ-term, looks for example like:



The (abstract) syntax of XPath I

- The syntax of (regular) XPath presents itself as a CFG (cf. [4]):

$$\begin{aligned}
locpath &::= axis \mid locpath\,[fexpr] \mid /\,locpath \mid locpath\,/\,locpath \mid \\
&\quad\; locpath\ \texttt{union}\ locpath \\
fexpr &::= locpath \mid \texttt{not}\ fexpr \mid fexpr\ \texttt{and}\ fexpr \mid fexpr\ \texttt{or}\ fexpr \\
axis &::= primaxis \mid ?fexpr \mid axis\,;axis \mid axis \cup axis \mid axis^* \\
primaxis &::= \mathsf{Self} \mid \Downarrow \mid \Uparrow \mid \Rightarrow \mid \Leftarrow
\end{aligned}$$

The (abstract) syntax of XPath II

- Like every CFG, it gives raise to its abstract syntax, that is a constructive signature

$$
\begin{aligned}
\Pi = (P &= \{\ locpath, fexpr, primaxis, axis\ \},\\
\mathcal{J} &= \emptyset,\\
F &= \{\ \begin{array}{llll}
f_{axis} & : axis \to locpath, & f_{select} & : fexpr \to axis,\\
f_{filter} & : locpath \times fexpr \to locpath, & f_{compax} & : axis \times axis \to axis,\\
f_{root} & : locpath \to locpath, & f_{unionax} & : axis \times axis \to axis,\\
f_{comp} & : locpath \times locpath \to locpath, & f_{trans} & : axis \to axis,\\
f_{union} & : locpath \times locpath \to locpath, & f_{\mathsf{Self}} & : 1 \to primaxis,\\
f_{path} & : locpath \to fexpr, & f_{\Downarrow} & : 1 \to primaxis,\\
f_{not} & : fexpr \to fexpr, & f_{\Uparrow} & : 1 \to primaxis,\\
f_{and} & : fexpr \times fexpr \to fexpr, & f_{\Rightarrow} & : 1 \to primaxis,\\
f_{or} & : fexpr \times fexpr \to fexpr, & f_{\Leftarrow} & : 1 \to primaxis,\\
f_{id} & : primaxis \to axis\ \})
\end{array}
\end{aligned}
$$

The semantic of XPath

- The semantic of a XPath-Expression $\alpha \in T_\Pi$ is now a *fold* into the following Π-Algebra *Sem* (cf. [4]):

$$\begin{aligned} Sem_{locpath} = Sem_{axis} = Sem_{primaxis} &= CT_\Sigma \longrightarrow X^* \longrightarrow \mathcal{P}(X^*) \\ Sem_{fexpr} &= CT_\Sigma \longrightarrow \mathcal{P}(X^*) \end{aligned}$$

$$\begin{aligned}
f_{axis} &= \text{id} \\
f_{filter}(f,e)(D)(w) &= f(D)(w) \cap e(D) \\
f_{root}(f)(D)(w) &= \begin{cases} f(D)(\varepsilon) & w = \varepsilon \\ \emptyset & \text{sonst} \end{cases} \\
f_{comp}(f,g)(D)(w) &= g(D)(f(D)(w)) \\
f_{union}(f,g)(D)(w) &= f(D)(w) \cup g(D)(w) \\
f_{path}(f)(D) &= \{x \mid f(D)(x) \neq \emptyset\} \\
f_{not}(e)(D) &= X^* \setminus e(D) \\
f_{and}(e,e')(D) &= e(D) \cap e'(D) \\
f_{or}(e,e')(D) &= e(D) \cup e'(D) \\
f_{id} &= \text{id}
\end{aligned}$$

$$\begin{aligned}
f_{select}(e)(D)(w) &= e(D) \cap \{w\} \\
f_{compax}(f,g)(D)(w) &= g(D)(f(D)(w)) \\
f_{unionax}(f,g)(D)(w) &= f(D)(w) \cup g(D)(w) \\
f_{trans}(f)(D)(w) &= \{w\} \cup \textstyle\bigcup_{n\in\mathbb{N}} M_n, \\
&\text{with } M_0 = f(D)(w) \text{ and } M_{n+1} = f(D)(M_n) \\
f_{\text{Self}}(D)(w) &= \{\text{w}\} \\
f_{\Downarrow}(D)(w) &= \{x \mid (w,x) \in R_{\Downarrow}\} \\
f_{\Uparrow}(D)(w) &= \{x \mid (x,w) \in R_{\Downarrow}\} \\
f_{\Rightarrow}(D)(w) &= \{x \mid (w,x) \in R_{\Rightarrow}\} \\
f_{\Leftarrow}(D)(w) &= \{x \mid (x,w) \in R_{\Rightarrow}\}
\end{aligned}$$

Query Evaluation

- As seen before, CT_Σ is isomorphic to a subset of $\mathbb{N} \longrightarrow \mathbb{N}$, furthermore X^* is isomorphic to $\mathbb{N}$, because X is countable, so we can regard the carrier of *Sem* as

$$\mathbb{N}^{\mathbb{N}} \longrightarrow \mathbb{N} \longrightarrow 2^{\mathbb{N}} \qquad \text{instead of} \qquad CT_\Sigma \longrightarrow X^* \longrightarrow \mathcal{P}(X^*).$$

- Now one can ask for the difficulty of *query evaluation*. That is, given a XPath expression $\alpha \in T_\Pi$, a document $D \in CT_\Sigma$ and a context node $w \in X^*$, how difficult is $fold^{Sem}(\alpha)(D)(w)$, which is a subset of $\mathbb{N}$?
- The answer has to be figured out.

Satisfiability and Containment Problem

- For a XPath expression $\alpha \in T_\Pi$, a document $D \in CT_\Sigma$ and a context node $w \in X^*$ write

$$(D, w) \models \alpha \quad \Longleftrightarrow \quad \exists v \in X^* \;\; v \in \mathit{fold}^{Sem}(\alpha)(D)(w).$$

  Furthermore write $\mathrm{Mod}(\alpha) = \{(D, w) \in CT_\Sigma \times X^* \mid (D, w) \models \alpha\}$.
- Given a XPath expression $\alpha \in T_\Pi$, the *satisfiability problem (SAT)* asks whether $\mathrm{Mod}(\alpha) = \emptyset$, the *containment problem (CON)*, given two XPath expressions $\alpha, \beta \in T_\Pi$, whether $\mathrm{Mod}(\alpha) \subseteq \mathrm{Mod}(\beta)$.
- These are problems well studied for T_Σ, i.e. finite documents (see for example [4, 7, 8, 9]).

A first Approach using Effective Descriptive Set Theory

- Viewing T_Π as a subset of $\mathbb{N} \longrightarrow \mathbb{N}$ the difficulty of SAT is the difficulty of

$$Sat = \{\alpha \in T_\Pi \mid \mathrm{Mod}(\alpha) = \emptyset\} \in \mathcal{P}\left(\mathbb{N}^{\mathbb{N}}\right).$$

- Consider the formula

$$\varphi(\alpha, D) = \forall w\ \alpha \in T_\Pi \wedge D \in CT_\Sigma \Rightarrow \forall v\ v \notin fold^{Sem}(\alpha)(D)(w).$$

  If $\{(\alpha, D) \mid \varphi(\alpha, D)\}$ would be arithmetic (what has to be shown) it would follow that $Sat = \{\alpha \mid \forall D\ \varphi(\alpha, D)\}$ is Π^1_1 (cf. [5]).

- As a next step it could be studied, if this approach works for finite documents, classifying the problem in the polynomial hierarchy.

Bibliography I

- P. Padawitz, *Übersetzerbau (Algebraic compiler construction)*, course notes, TU Dortmund (2016)
- P. Padawitz, *Fixpoints, Categories, and (Co)Algebraic Modeling*, course notes, TU Dortmund (2016)
- J. Berstel, L. Boasson, *Formal properties of XML grammars and languages*, Acta Informatica 38, 649-671 (2002)
- M. Marx, *XPath with Conditional Axis Relations*, LNCS 2992, pp. 477-494 (2004)
- D. Marker, *Descriptive Set Theory*, lecture notes, University of Illinois at Chicago (2002)
- Y. Moschovakis, *Descriptive Set Theory*, American Mathematical Society, 2nd edition, ISBN : 978-0821848135 (2009)

Bibliography II

- G. Gottlob, C. Koch, and R. Pichler, *Efficient Algorithms for Processing XPath Queries*, Proceedings of the 28th VLDB Conference, Hong Kong, China (2002)
- J. Hidders, *Satisfiability of XPath Expressions*, Proceedings of the 9th International Conference on Data Base Programming Languages (DBPL), Potsdam, Germany (2003)
- G. Miklau, D. Suciu, *Containment and Equivalence for a Fragment of XPath*, Journal of the ACM, Vol. 51, No. 1, pp. 2?45. (2004)