Simplifying XML Schema:
Single-type approximations of regular tree languages

Wenbo Lian

National Engineering Research Center of Fundemental Software,
Institute of Software, Chinese Academy of Sciences

May 25, 2013
Outline

1 Motivation

2 Preliminaries

3 Upper XSD-approximations
   • Single-type up approximations of EDTDs
   • Unions of XSDs
   • Intersection of XSDs
   • Complements of XSDs
Motivation

- XML documents can be exactly modeled by XML Schema Definition
- XML Schema Definition can be as strong as EDTD
- For practical use, constraints are imposed on XML Schema Definitions (XSD), which may break the closure of boolean operators
  - UPA (Unique Particle Attribution): In G.J Bex et al.
  - EDC (Element Declaration Consistent): In this paper
- Element Declarations Consistents
  - Elements with the same name in the same content model must have the same type
  - Advantage: Facilitates a simple one-pass top-down validation algorithm
  - Disadvantage: Break the closure of XSD under union and set difference
- Approximations is needed in two flavours
  - Upper XSD-approximation: Union of two XSD
  - Lower XSD-approximation: Description of interface in web
The DTD example

```xml
<!DOCTYPE CONFERENCE [  
  <!ELEMENT conference (track+|(session, break?)+)>  
  <!ELEMENT track (session, break?)+>  
  <!ELEMENT session (chair, talk+)>  
  <!ELEMENT talk ((title, authors)| (title, speaker))>  
  <!ELEMENT chair (#PCDATA)>  
  <!ELEMENT break (#PCDATA)>  
  <!ELEMENT title (#PCDATA)>  
]> 
```
The XML Schema example

```xml
<xsd:complexType name="track">
    <xsd:sequence minOccurs="1" maxOccurs="unbounded">
        <xsd:choice>
            <xsd:element name="invSession" type="invSession"
            minOccurs="1" maxOccurs="1"/>
            <xsd:element name="conSession" type="conSession"
            minOccurs="1" maxOccurs="1"/>
        </xsd:choice>
        <xsd:element name="break" type="xsd:string"
        minOccurs="0" maxOccurs="1"/>
    </xsd:sequence>
</xsd:complexType>
```
Contributions and Related work

**Contributions**

- Every EDTD has a unique upper XSD-approximation
- The approximation of two XSDs union and set difference can be determined in polynomial time
- Deciding whether \( S \) is the minimal upper XSD-approximation of \( D \) is complete for PSPACE where \( S \) is a single-type EDTD, \( D \) is an EDTD

**Related work**

- Murata et al. establish a *taxonomy* of XML Schema in terms of tree language
- Martens et al. characterized ST-REG as the subclass of the regular tree language closed under ancestor-guarded subtree exchange
Outline

1. Motivation

2. Preliminaries

3. Upper XSD-approximations
   - Single-type up approximations of EDTDs
   - Unions of XSDs
   - Intersection of XSDs
   - Complements of XSDs
Strings, trees, and contexts

**Definition.** *State-labeled automata* \( N(Q, \Sigma, \delta, S, F) \): \( \forall q \in Q \), the set \( \{a| \exists p \in Q \text{ such that } q \in \delta(p, a)\} \) is a *singleton*

\( N(w) \): The resulting state set of \( N \) after reading \( w \) from some state \( s \in I \)

**Definition.** \( \Sigma \)-Tree: \( \text{Dom}(t) = \{\varepsilon\} \cup \{iu: 1 \leq i \leq n, u \in \text{Dom}(t)\} \)

\( \Sigma \)-label: Denoted by \( \text{lab}^t(v) \)

**Definition.** \( \text{ch-str}^t(v) \): The child string of node \( v \), i.e., the string \( \text{lab}^t(v_1) \ldots \text{lab}^t(v_n) \)

**Definition.** \( \text{anc-str}^t(v) \) where node \( v \) is \( i_1 \ldots i_k \): \( \text{lab}^t(\varepsilon)\text{lab}^t(i_1) \ldots \text{lab}^t(i_1 \ldots i_{k-1}) \text{lab}^t(v) \)

**Definition.** *Context*: A tree with a “hole” marker •
Definition 2.1. \( DTD \): A tuple \((\Sigma, d, S_d)\), where

- \( \Sigma \): Finite alphabet
- \( d: \Sigma \rightarrow \Sigma^* \)
- \( S_d \subseteq \Sigma \) is the set of start symbols
- the size of DTD: \(|\Sigma| + |S_d| + |d|\)
- A tree \( t \) accepted by \( L(D) \) (or \( L(d) \)) if \( \forall v \in Dom(t), \ ch-str^t(v) \in d(lab^t(v)) \)
Definition 2.1. *DTD*: A tuple \((\Sigma, d, S_d)\), where

- \(\Sigma\): Finite alphabet
- \(d: \Sigma \rightarrow \Sigma^*\)
- \(S_d \subseteq \Sigma\) is the set of start symbols
- the size of DTD: \(|\Sigma| + |S_d| + |d|\)

Definition 2.2. *EDTD*: A tuple \((\Sigma, \Delta, d, S_d, \mu)\)

- \(\Delta\): A finite type set
- \((\Delta, d, S_d)\): A DTD
- \(\mu: \Delta \rightarrow \Sigma\)
- A tree accepted by D if \(\exists t' \in L(d)\) such that \(\mu(t') = t\)
Definition 2.1. DTDD: A tuple $(\Sigma, d, S_d)$, where
- $\Sigma$: Finite alphabet
- $d: \Sigma \rightarrow \Sigma^*$
- $S_d \subseteq \Sigma$ is the set of start symbols
- the size of DTD: $|\Sigma| + |S_d| + |d|

Definition 2.2. EDTDD: A tuple $(\Sigma, \Delta, d, S_d, \mu)$
- $\Delta$: A finite type set
- $(\Delta, d, S_d)$: A DTD
- $\mu$: $\Delta \rightarrow \Sigma$

Proviso 2.3. All EDTDs are reduced.
- Reduced: for any type $\tau \in \Delta$, there exists a tree $t' \in L(d)$ and a node $u$ such that $lab^{t'}(u) = \tau$
- Any EDTD has an equivalent reduced EDTD and can be computed from a given EDTD in polynomial time
- Similar to CFG, see [J. Albert et.al 2001, W. Martens et.al 2009]
**Definition 2.2.** \(EDTD\): A tuple \((\Sigma, \Delta, d, S_d, \mu)\)

- \(\Delta\): A finite type set
- \((\Delta, d, S_d)\): A DTD
- \(\mu: \Delta \rightarrow \Sigma\)

**Definition 2.4.** \(Single-type EDTD\): An EDTD \((\Sigma, \Delta, d, S_d, \mu)\) with property that no two types \(\tau_1\) and \(\tau_2\) exists with \(\mu(\tau_1) = \mu(\tau_2)\) such that

- \(\tau_1, \tau_2 \in S_d\)
- there is a type \(\tau\) such that \(w_1\tau_1v_1 \in d(\tau)\) and \(w_2\tau_2v_2 \in d(\tau)\) for some strings \(w_1, v_1, w_2\) and \(v_2\).
- \(ST-REG\) is the class of regular tree language can be definable by single-type EDTDs.
**Definition 2.2.** *EDTD*: A tuple $(\Sigma, \Delta, d, S_d, \mu)$

- $\Delta$: A finite type set
- $(\Delta, d, S_d)$: A DTD
- $\mu$: $\Delta \rightarrow \Sigma$

**Definition 2.4.** *Single-type EDTD*: An EDTD $(\Sigma, \Delta, d, S_d, \mu)$ with property that no two types $\tau_1$ and $\tau_2$ exists with $\mu(\tau_1) = \mu(\tau_2)$ such that

- $\tau_1, \tau_2 \in S_d$
- there is a type $\tau$ such that $w_1 \tau v_1 \in d(\tau)$ and $w_2 \tau v_2 \in d(\tau)$ for some strings $w_1, v_1, w_2$ and $v_2$.
- ST-REG is the class of regular tree language can be definable by single-type EDTDs.

Intuitively use an automaton to help assign type for a tree which may be accepted by EDTD

**Definition 2.5.** *type automaton* of an EDTD $D = (\Sigma, \Delta, d, S_d, \mu)$ a *state-labeled* NFA without final states such that $Q = \Delta \cup \{ q_{init} \}$ and foreach $q \in Q$

- if $q = q_{init}$, then $\delta(q, a) = \{ \tau | \mu(\tau) = a \text{ and } \tau \in S_d \}$ and
- otherwise, $\delta(q, a) = \{ \tau | \mu(\tau) = a \text{ and } \tau \text{ occurs in some word in } d(q) \}$
Example 2.6. Given an EDTD $D = (\Sigma, \Delta, \delta, S, \mu)$, with

$\Delta = \{ \tau_a, \tau_b, \tau_b^1, \tau_b^2 \}$, $S = \{ \tau_a \}$ and $\mu(\tau_a) = a$, $\mu(\tau_b^1) = \mu(\tau_b^2) = b$:

- $\tau_a \rightarrow \tau_a + \tau_b^1$, $\tau_b^1 \rightarrow \tau_b^2 + \varepsilon$,
- $\tau_b^2 \rightarrow \tau_a + \tau_b^2 + \varepsilon$

Construct the type automaton for the EDTD.
Example 2.6. Given an EDTD $D = (\Sigma, \Delta, d, S_d, \mu)$, with $\Delta = \{\tau_a, \tau_b^1, \tau_b^2\}$, $S_d = \{\tau_a\}$ and $\mu(\tau_a) = a$, $\mu(\tau_b^1) = \mu(\tau_b^2) = b$:

- $\tau_a \rightarrow \tau_a + \tau_b^1$, $\tau_b^1 \rightarrow \tau_b^2 + \varepsilon$,
- $\tau_b^2 \rightarrow \tau_a + \tau_b^2 + \varepsilon$

Construct the type automaton for the EDTD.

Observation 2.7

- Given an EDTD, its type automation can be construct in linear time.
- For each EDTD, the state $q_{init}$ of its type automaton has no incoming transitions.
- The type automaton of an EDTD is a DFA iff $D$ is a single-type EDTD.
Definition 2.8. A DFA-based XSD is a pair $D=(\Sigma, A, d, S_d)$, where $A=(Q, \Sigma, \delta, \{q_{init}\}, \emptyset)$ is a state-labeled DFA with:

- initial state $q_{init}$ and without final states
- $d$ is a function from $Q\backslash \{q_{init}\}$ to regular languages over $\Sigma$
- $S_d \subseteq \Sigma$ is the set of start symbols
- A tree $t$ satisfies $D$ if $\text{lab}^t(\varepsilon) \in S_d$ and for every node $u$ where $A(\text{anc-str}^t(u)) = \{q\} \land \text{ch-str}^t(u) \in d(q)$

Proposition 2.9. DFA-based XSDs are expressively equivalent to single-type EDTDs and one can translate between DFA-based XSDs and single-type EDTDs in linear time.
**Definition 2.8.** A DFA-based XSD is a pair $D= (\Sigma, A, d, S_d)$, where $A= (Q, \Sigma, \delta, \{q_{\text{init}}\}, \emptyset)$ is a state-labeled DFA with:

**Proposition 2.9.** DFA-based XSDs are *expressively equivalent* to single-type EDTDs and one can translate between DFA-based XSDs and single-type EDTDs in linear time.

**Proof.**

From XSD to Single-type EDTD, intuitively as the reverse of construction of type automaton.

From DFA-based XSD $D= (\Sigma, A, d, S_d)$, where $A= (Q, \Sigma, \delta, \{q_{\text{init}}\}, \emptyset)$ to single-type EDTD $E= (\Sigma, \Delta, d', S'_d, \mu)$:

- $\Delta = \{(a, q) \in \Sigma \times Q | \exists p: \delta(p, a) = q \in A\}$
- $S'_d = \{(a, q) | a \in S_d \text{ and } \delta(q_{\text{init}}, a) = q \in A\}$
- $\mu((a, q)) = a$ for every $(a, q) \in \Delta$, and
- for each $(a, q) \in \Delta$, we define $d'((a, q))$ to be the language $\{(a_1, q_1) \cdots (a_n, q_n) \in \Delta^* | a_1 \cdots a_n \in d(q)\}$ and, for each $a_i, \delta(q, a_i) = q_i$ in $A$.
Definition 2.8. A DFA-based XSD is a pair $D=(\Sigma, A, d, S_d)$, where $A=(Q, \Sigma, \delta, \{q_{init}\}, \emptyset)$ is a state-labeled DFA with:

Proposition 2.9. DFA-based XSDs are expressively equivalent to single-type EDTDs and one can translate between DFA-based XSDs and single-type EDTDs in linear time.

Proof.

From single-type EDTD $E=(\Sigma, \Delta, d, S_d, \mu)$

- $S_d' = \{\mu(\tau) | \tau \in S_d\}$
- $A=(Q, \Sigma, \delta, \{q_{init}\})$ is the type automaton of $E$
- for each $\tau \in \Delta$, we define $d'(\tau) = \mu(d(\tau))$, where $\mu(d(\tau))$ denotes the homomorphic extension of $\mu$ to string languages
Definition 2.10. A tree language $T$ is *closed under ancestor-guarded subtree exchange* if the following property holds. Whenever for two $t_1, t_2 \in T$ with nodes $v_1, v_2$ respectively, $\text{anc-str}^{t_1}(v_1) = \text{anc-str}^{t_2}(v_2)$ then
\[
t_1[v_1 \leftarrow \text{subtree}^{t_2}(v_2)] \in T
\]
**Definition 2.10.** A tree language $T$ is **closed under ancestor-guarded subtree exchange** if the following property holds. Whenever for two $t_1, t_2 \in T$ with nodes $v_1, v_2$ respectively, $\text{anc-str}^{t_1}(v_1) = \text{anc-str}^{t_2}(v_2)$ then

$$ t_1[v_1 \leftarrow \text{subtree}^{t_2}(v_2)] \in T $$

**Theorem 2.11.** A regular language $T$ is definable by a single-type EDTD iff it is **closed under ancestor-guarded subtree exchange**.
Definition 2.10. A tree language $T$ is closed under ancestor-guarded subtree exchange if the following property holds. Whenever for two $t_1, t_2 \in T$ with nodes $v_1, v_2$ respectively, $\text{anc-str}^{t_1}(v_1) = \text{anc-str}^{t_2}(v_2)$ then

$$t_1[v_1 \leftarrow \text{subtree}^{t_2}(v_2)] \in T$$

Theorem 2.11. A regular language $T$ is definable by a single-type EDTD iff it is closed under ancestor-guarded subtree exchange.

Definition 2.12. minimal upper XSD-approximation of an EDTD $D$: A Single-type EDTD $D_1$ where $L(D) \subseteq L(D_1)$ and no $D'$ exists such that $L(D) \subseteq L(D') \subset L(D_1)$
Definition 2.10. A tree language $T$ is \textit{closed under ancestor-guarded subtree exchange} if the following property holds. Whenever for two $t_1, t_2 \in T$ with nodes $v_1, v_2$ respectively, $\text{anc-str}_{t_1}(v_1) = \text{anc-str}_{t_2}(v_2)$ then

$$t_1[v_1 \leftarrow \text{subtree}^{t_2}(v_2)] \in T$$

Theorem 2.11. A regular language $T$ is definable by a single-type EDTD iff it is \textit{closed under ancestor-guarded subtree exchange}.

Definition 2.12. \textit{minimal upper XSD-approximation} of an EDTD $D$: A single-type EDTD $D_1$ where $L(D) \subseteq L(D_1)$ and no $D'$ exists such that $L(D) \subseteq L(D') \subset L(D_1)$

Theorem 2.13. The universality problem for EDTDs, i.e., deciding whether $\mathcal{T}_\Sigma \subseteq L(D)$ for an EDTD $D$, is EXPTIME-complete.
Prove some basic properties about languages definable by single-type EDTDs and their closure properties

**Definition 2.14.** Let $T$ be a tree language. $\text{closure}(T)$ means the smallest tree language which contains $T$ and which is closed under ancestor-guarded subtree exchange.

**Lemma 2.15.** Let $(X_i)_{i \in I}$ be an arbitrary family of tree languages where each $X_i$ is closed under ancestor-guarded subtree exchange. Then the intersection $\bigcap_{i \in I} X_i$ is also closed under ancestor-guarded subtree exchange.
Prove some basic properties about languages definable by single-type EDTDs and their closure properties

**Definition 2.14.** Let $T$ be a tree language. $\text{closure}(T)$ means the smallest tree language which contains $T$ and which is closed under ancestor-guarded subtree exchange.

**Lemma 2.15.** Let $(X_i)_{i \in I}$ be an arbitrary family of tree languages where each $X_i$ is closed under ancestor-guarded subtree exchange. Then the intersection $\bigcap_{i \in I} X_i$ is also closed under ancestor-guarded subtree exchange.

**Proof.**
Let $X = \bigcap_{i \in I} X_i$.
- Let $t_1, t_2$ from $X$ with nodes $v_1, v_2$ where $\text{anc-str}^{t_1}(v_1) = \text{anc-str}^{t_2}(v_2)$
- With $t = t_1[v_1 \leftarrow \text{subtree}^{t_2}(v_2)] \in X$
Single-type closure and derivation trees

Prove some basic properties about languages definable by single-type EDTDs and their closure properties

**Definition 2.14.** Let $T$ be a tree language. $\text{closure}(T)$ means the smallest tree language which contains $T$ and which is closed under ancestor-guarded subtree exchange.

**Definition 2.16.** Let $X$ be a tree language and $t$ a tree from $\text{closure}(X)$. A *derivation tree* of $t$ w.r.t $X$ is a (finite) binary tree $\vartheta$ labeled with tree from $\text{closure}(X)$ such that:

- The root of $\vartheta$ is labeled with $t$:
  \[ \text{lab}^\vartheta(\varepsilon) = t \]

- For each leaf $v \in \text{Dom}(\vartheta)$, we have $\text{lab}^\vartheta(v) \in X$.

- For each internal node $v \in \text{Dom}(\vartheta)$ and $i \in \{1, 2\}$, let $t_i = \text{lab}^\vartheta(v_i)$. Then there are nodes $u_i \in \text{Dom}(t_i)$ such that $\text{anc-str}^{t_i}(u_1) = \text{anc-str}^{t_2}(u_2)$.
Lemma 2.17. Let $X$ be a tree language and $t$ a tree. Then $t \in \text{closure}(X)$ iff $t$ has a derivation tree w.r.t $X$.

Proof.

For if part. Whenever $t$ has a derivation tree w.r.t. $X$, then $t \in \text{closure}(X)$. Immediately

The only if part.

- $T_i$ the set of trees from $\text{closure}(X)$ which have a derivation tree of height $i$
- $T_0$ is $X$. Suppose $t \in T_i$ and $\vartheta$ is the derivation tree of $t$, then $t(\vartheta, \vartheta) \in T_{i+1}$, so $T_i \subseteq T_{i+1}$
- $T = \bigcup_{i \in \mathbb{N}} T_i$
- For $t_1, t_2 \in T$, there exist $n_1, n_2$ such that $t_1 \in T_{n_1}, t_2 \in T_{n_2}$, any tree $t$ obtained by applying ancestor-guarded subtree exchange to $t_1, t_2$ is in $T_{\max(n_1, n_2)+1} \subseteq T$, so $T$ is closed under ancestor-guarded subtree exchange and contains $X$.

- $\text{closure}(X)$ is the smallest set closed under ancestor-subtree exchange which contains $X$, $\text{closure}(X) \subseteq T$. 

Lian (nfs)
Simplifying XML Schema
May 25, 2013 9 / 19
Outline

1 Motivation

2 Preliminaries

3 Upper XSD-approximations
   - Single-type up approximations of EDTDs
   - Unions of XSDs
   - Intersection of XSDs
   - Complements of XSDs
Outline

1 Motivation

2 Preliminaries

3 Upper XSD-approximations
   - Single-type up approximations of EDTDs
   - Unions of XSDs
   - Intersection of XSDs
   - Complements of XSDs
Construction 3.1. (*Minimal upper approximation of an EDTD*). Let $D = (\Sigma, \Delta, d, S_d, \mu)$ be an EDTD. Let $N = (Q_N, \Sigma, \delta_N, \{q_{init}\})$ be the type automaton of $D$, and let $A_N = (Q, \Sigma, \delta, \{\{q_{init}\}\})$ be the DFA obtained from $N$ by performing the standard subset construction.

- $Q \subseteq 2^{Q_N}$ is the *smallest* set such that $\{q_{init}\} \in Q$ and whenever $S \in Q$ then for every $a \in \Sigma$ then for every $a \in \Sigma$ we have $\bigcup_{q \in S} \delta_N(q, a) \in Q$
- each non-initial state consists of a set of types $S$ of $D$ in which, for every $\tau, \tau' \in S$
- DFA-based XSD $(\Sigma, A_N, d', S_d')$ where
  - $S_d' = \{a \in \Sigma|\tau \in S_d, \mu(\tau) = a\}$
  - $d'(S) := \bigcup_{\tau \in S} \mu(d(\tau))$ for every $S \in Q$
- $\mu$ canonically extended to languages
Theorem 3.2. The minimal upper XSD-approximation of an EDTD is unique and can be computed in exponential time. There is a family of EDTDs \((D_n)_{n \geq 2}\), such that the size of every \(D_n\) is \(O(n)\) but the type-size of the minimal upper XSD-approximation is \(\Omega(2^n)\).

Proof.

First prove that an EDTD \(D=(\Sigma, \Delta, d, S_d, \mu)\), determinizing its type automaton result a DFA-based XSD \(D'=(\Sigma, A, d', S'_d)\) which is the unique minimal upper XSD-approximation:

- \(L(D) \subseteq L(D')\)
**Theorem 3.2.** The minimal upper XSD-approximation of an EDTD is *unique* and can be computed in *exponential* time. There is a family of EDTDs $(D_n)_{n \geq 2}$, such that the size of every $D_n$ is $O(n)$ but the type-size of the minimal upper XSD-approximation is $\Omega(2^n)$

**Proof.**

First prove that an EDTD $D= (\Sigma, \Delta, d, S_d, \mu)$, determinizing its type automaton result a DFA-based XSD $D' = (\Sigma, A, d', S_d')$ which is the unique minimal upper XSD-approximation

- $L(D) \subseteq L(D')$
- Suppose $t \in L(D)$, according to the definition of EDTD $\exists t' \in L(d)$ and $\mu(t') = t$, such that $\forall v \in Dom(t'), ch-str^{t'}(v) \in d(lab^{t'}(v))$.
- Let $v \in Dom(t)$ and $S = A(anc-str^t(v))$.
- According to Construction of $D'$, $lab^t(v) \in S$ and $ch-str^t(v) \in d'(S)$.
- As this holds for all nodes of $t$, $t \in L(D')$
Theorem 3.2. The minimal upper XSD-approximation of an EDTD is unique and can be computed in exponential time. There is a family of EDTDs \((D_n)_{n \geq 2}\) such that the size of every \(D_n\) is \(O(n)\) but the type-size of the minimal upper XSD-approximation is \(\Omega(2^n)\).

Proof.

First prove that an EDTD \(D = (\Sigma, \Delta, d, S_d, \mu)\), determinizing its type automaton result a DFA-based XSD \(D' = (\Sigma, A, d', S_d')\) which is the unique minimal upper XSD-approximation

- \(L(D') \subseteq \text{closure}(L(D))\)
- iterate over the nodes of \(t \in L(D')\) in a breadth first manner, such that when we reach a node \(v\), construct a tree \(t_v\) which satisfies
  - \(t_v \in \text{closure}(L(D))\)
  - the parts of \(t\) and \(t_v\) up to \(v\) (breadth first manner) and their children are isomorphic.
Minimal upper approximation of EDTD

**Theorem 3.2.** The minimal upper XSD-approximation of an EDTD is *unique* and can be compute in *exponential* time. There is a family of EDTDs \((D_n)_{n\geq 2}\), such that the size of every \(D_n\) is \(O(n)\) but the type-size of the minimal upper XSD-approximation is \(\Omega(2^n)\)

**Proof.**

First prove that an EDTD \(D=(\Sigma, \Delta, d, S_d, \mu)\), determinizing its type automaton result a DFA-based XSD \(D'=(\Sigma, A, d', S'_d)\) which is the unique minimal upper XSD-approximation

- \(L(D') \subseteq \text{closure}(L(D))\)
- construct the sequence of trees \(t_v\)
  - \(\forall v \in \text{Dom}(t)\) assign a type \(\tau_v\) such that \(ch-\text{str}^t(v) \in \mu(d(\tau_v))\)
  - Iterate \(v \in \text{Dom}(t)\) in *breadth first manner*
  - When \(v\) is root node, \(t_v \in L(D)\) can be construct as \(L(D)\) is reduced therefore \(t_v \in \text{closure}(L(D))\)
  - \(t_u \in \text{closure}(L(D))\) was constructed, \(v\) next node to iterate, \(\exists t'_v \in L(D)\) by assigning \(\tau_v\) to \(t'_v\) where \(\text{anc-}\text{str}^{t'_v}(v) = \text{anc-}\text{str}^t(v)\) and \(ch-\text{str}^{t'_v}(v) = ch-\text{str}^t(v)\) as \(D\) is *reduced* and the construction of \(D'\)
  - \(t_u[v \leftarrow \text{subtree}^{t'_v}(v)] \in \text{closure}(L(D))\) can be the \(t_v\) in the sequence
Theorem 3.2. The minimal upper XSD-approximation of an EDTD is unique and can be computed in exponential time. There is a family of EDTDs $(D_n)_{n \geq 2}$, such that the size of every $D_n$ is $O(n)$ but the type-size of the minimal upper XSD-approximation is $\Omega(2^n)$.

Proof.

Now prove that the exponential type size cannot be avoided.

Can consider the unary tree, each node has at most one child, such a tree can be viewed as a regular word.

EDTDs and stEDTDs can intuitively correspond to NFAs and DFAs.

Need to translate between stEDTDs and DFAs.
Theorem 3.2. The minimal upper XSD-approximation of an EDTD is unique and can be computed in exponential time. There is a family of EDTDs \((D_n)_{n \geq 2}\), such that the size of every \(D_n\) is \(O(n)\) but the type-size of the minimal upper XSD-approximation is \(\Omega(2^n)\).

Proof.

Prove that exists a family of EDTDs.

\[ L_n = (a + b)^* a(a + b)^n \]

- Property: the unique node at distance \(n\) of the leaf node is \(a\)
- let EDTD \(D_n\) accepts \(L_n, D_n\) can be easily construct of size linear in \(n\)
- \(D'_n = (\Sigma, A, d, S_d)\) a DFA-based XSD such that \(L(D_n) = L(D'_n)\)
- \(A_n\) is a DFA obtained from \(A\) by
  - \(\forall q \in A\) where \(\varepsilon \in L(d(q))\), mark \(q\) final
  - remove all transitions \((q, \sigma, q')\) where \(\sigma \notin L(d(q))\)
- \(L(A_n) = L_n\) and the DFA which accepts \(L_n\) is of size exponential in \(n\)
Lemma 3.3 Let \( D_1 \) be an EDTD and let \( D_2 \) be a single-type EDTD. Testing whether \( L(D_1) \subseteq L(D_2) \) is in PTIME

Proof.

Let \( D_1 = (\Sigma, \Delta_1, d_1, S_{d_1}, \mu_1) \) and \( D_2 = (\Sigma, \Delta_2, d_2, S_{d_2}, \mu_2) \) let for each \( i \in \{1, 2\}, A_i = (Q_i, \Sigma, \delta_i, I_i) \).

\( L(D_1) \not\subseteq L(D_1) \) iff there exitsts a type \( \tau_2 \in \Delta_2 \) for which there exists a string \( w \) which

- \( A_2(w) = \{\tau_2\} \), \( A_1(w) = S_1 \), and
- there exists a \( \tau_1 \in S_1 \) and a string \( v \in d_1(\tau_1) \) such that \( \mu_1(v) \not\in \mu_2(d_2(\tau_2)) \)
Lemma 3.3 Let $D_1$ be an EDTD and let $D_2$ be a single-type EDTD. Testing whether $L(D_1) \subseteq L(D_2)$ is in PTIME.

Proof.

Provide a PTIME algorithm for the complement of the problem

1. Compute the binary relation $R = \{(\tau_1, \tau_2) | \exists w \text{ such that } \tau_1 \in A_1(w) \text{ and } A_2(w) = \{\tau_2\}\}$

2. Test whether exists a pair $(\tau_1, \tau_2)$ in $R$ for which $\mu_1(d_1(\tau_1)) \not\equiv \mu_2(d_2(\tau_2))$

   the step(1) can be computed in polynomial time by considering product automation $A_1 \times A_2$

   the step(2) is in PTIME since both $\mu_1(d_1(\tau_1))$ and $\mu_2(d_2(\tau_2))$ can be represented by polynomial-size DFAs
Theorem 3.4 (See [25].) The complexity of the language inclusion problem $L(X) \subseteq L(Y)$ is PSPACE-complete when $X$ and $Y$ are given as regular expressions or NFAs.
**Theorem 3.5** Deciding whether a single-type EDTD is a minimal upper XSD-approximation of a given EDTD is PSPACE-complete.

**Proof.**

For the upper bound, let $D_1 = (\Sigma, \Delta_1, d_1, S_{d_1}, \mu_1)$ a single-type EDTD, $D$ an EDTD.

- First, test whether $L(D) \subseteq L(D_1)$
- Let $D_2$ be the minimal upper XSD-approximation of $D$ according to Theorem 3.2, claim that
  - $D_1$ is the minimal upper XSD-approximation of $D$ iff $L(D_1) \subseteq L(D_2)$: Easy to proof
  - Can test whether $L(D_1) \subseteq L(D_2)$ in PSPACE without fully constructing $D_2$
Theorem 3.5 Deciding whether a single-type EDTD is a minimal upper XSD-approximation of a given EDTD is PSPACE-complete.

Proof.

For the upper bound, let $D_1 = (\Sigma, \Delta_1, d_1, S_{d_1}, \mu_1)$ a single-type EDTD, $D$ an EDTD.

- According to the proof of [W. Martens et.al 2007], testing $L(D_1) \subseteq L(D_2)$ reduces to
  - Computing a correspondence relation $R \subseteq \Delta_1 \times \Delta_2$ between their types
  - For each pair $(\tau_1, \tau_2) \in R$, testing the inclusion $\mu_1(d_1(\tau_1)) \subseteq \mu_2(d_1(\tau_2))$

- In other words, $L(D_1) \not\subseteq L(D_2)$ iff
  
  there is a $(\tau_1, \tau_2) \in R$ such that $\mu_1(d_1(\tau_1)) \not\subseteq \mu_2(d_1(\tau_2))$
Minimal upper approximation of EDTD

**Theorem 3.5** Deciding whether a single-type EDTD is a minimal upper XSD-approximation of a given EDTD is PSPACE-complete.

**Proof.**

For the upper bound, let $D_1 = (\Sigma, \Delta_1, d_1, S_{d_1}, \mu_1)$ a single-type EDTD, $D$ an EDTD.

- PSPACE procedure consist of following steps:
  - Guess $w$ and keep track of $(A_1(w), A_2(w))$ **without constructing $A_2$ itself**. $A_1, A_2$ the type automaton corresponding to $D_1, D_2$
  - Whether $\mu_1(d_1(\tau_1)) \not\equiv \mu_2(d_1(\tau_2))$ is the same as $\mu_1(d_1(\tau)) \not\equiv \mu(d(\tau_1)) + \cdots + \mu(d(\tau_k))$
  - Intuitively, we **guess** the path instead of constructing all the possible states

\[\square\]
Theorem 3.5 Deciding whether a single-type EDTD is a minimal upper XSD-approximation of a given EDTD is PSPACE-complete.

Proof.

The PSPACE for the lower bound can be obtained from the fact that testing $L(A) \subseteq L(A_1) \cup \cdots \cup L(A_n)$ for DFAs $A, A_1, \ldots, A_n$ is PSPACE-complete.

Construct an EDTD $D$ which takes $\tau^1_r \rightarrow L(A_1), \ldots, \tau^n_r \rightarrow L(A_n)$ as the content model where $\forall 1 \leq i \leq n, \mu_1(\tau^i_r) = r$.

Construct single-type $D_1$ as which takes $\tau_r \rightarrow L(A)$ as the content model where $\tau_r = r$.

The problem for testing $L(A) \subseteq L(A_1) \cup \cdots \cup L(A_n)$ will reduce to test whether $D_1$ is the minimal XSD-approximation of $D$. \qed
Outline

1 Motivation

2 Preliminaries

3 Upper XSD-approximations
   - Single-type up approximations of EDTDs
   - Unions of XSDs
   - Intersection of XSDs
   - Complements of XSDs
Unions of XSDs

**Theorem 3.6.** Let $D_1$ and $D_2$ be two single-type EDTDs.

- The minimal upper XSD-approximation of $L(D_1) \cup L(D_2)$ is unique and can be computed in time $O(|D_1||D_2|)$.
- There exists a family of single-type EDTDs $(D^n_1, D^n_2)_{n \geq 1}$, such that the size of every $D^n_1$ and $D^n_2$ is $O(n)$ but the type-size of the minimal upper XSD-approximation for $L(D^n_1) \cup L(D^n_2)$ is $\Omega(n^2)$

**Proof.**

$$D_1 = (\Sigma, \Delta_1, d_1, S_{d_1}, \mu_1) \text{ and } D_2 = (\Sigma, \Delta_2, d_2, S_{d_2}, \mu_2)$$

$D$:EDTD D where $L(D) = L(D_1) \cup L(D_2)$ obtained by computing the cross-product of $D_1$ and $D_2$

Type automaton of D is product of type automata of $D_1$ and $D_2$

Product of deterministic automata is deterministic. Determinization is trival and perform in time $O(|D_1||D_2|)$

The type-size of the minimal upper XSD-approximation $D'$ for $L(D_1) \cup L(D_2)$ is $O(|D_1||D_2|)$

From the Theorem 3.2, this XSD-approximation is unique minimal XSD-approximation.
Unions of XSDs

Theorem 3.6. Let $D_1$ and $D_2$ be two single-type EDTDs.

- The minimal upper XSD-approximation of $L(D_1) \cup L(D_2)$ is unique and can be computed in time $O(|D_1||D_2|)$.
- There exists a family of single-type EDTDs $(D^1_n, D^2_n)_{n \geq 1}$, such that the size of every $D^1_n$ and $D^2_n$ is $O(n)$ but the type-size of the minimal upper XSD-approximation for $L(D^1_n) \cup L(D^2_n)$ is $\Omega(n^2)$.

Proof.

Prove the second goal

Fix $n$ and consider the following single-type EDTD $D_1$ with $S_d = \{\tau^0_a, \tau^0_b\}$

$$
\begin{align*}
\tau^i_a &\rightarrow \tau^{i+1}_a + \tau^{i+1}_b + \varepsilon \quad (\text{for all } 0 \leq i < n - 1) \\
\tau^i_b &\rightarrow \tau^i_a + \tau^i_b + \varepsilon \quad (\text{for all } 0 \leq i < n), \\
\tau^{n-1}_a &\rightarrow \tau^n_b + \varepsilon \\
\tau^n_b &\rightarrow \tau^n_b + \varepsilon
\end{align*}
$$

The language $L(D_1)$ consist of unary trees which contains at most $n$ node labeled with $a$. By changing the roles of $a$ and $b$, define $D_2$ such that $L(D_2)$ consists of unary trees which contain at most $n$ nodes labeled with $b$. \qed
Unions of XSDs

**Theorem 3.6.** Let $D_1$ and $D_2$ be two single-type EDTDs.

- The minimal upper XSD-approximation of $L(D_1) \cup L(D_2)$ is unique and can be computed in time $O(|D_1||D_2|)$.
- There exists a family of single-type EDTDs $(D_1^n, D_2^n)_{n \geq 1}$, such that the size of every $D_1^n$ and $D_2^n$ is $O(n)$ but the type-size of the minimal upper XSD-approximation for $L(D_1^n) \cup L(D_2^n)$ is $\Omega(n^2)$.

**Proof.**

Let $D'$ be the minimal upper XSD-approximation of $L(D_1) \cup L(D_2)$. Now show that type-size of $D'$ is $\Omega(n^2)$.

- $N'$ is the type automaton for $D'$. Let $\tau_{k,l} = N'(a^kB^l)$ for $1 \leq k, l \leq n$.
- Consider types for $(k, l) \neq (k', l')$ and assume $\tau_{k,l} = \tau_{k',l'}$, $k > k'$. Both $t = a^{k}b^{2n}a^{n-k}$ and $t' = a^{k'}b^{2n}a^{n-k'}$ are in $L(D')$.
- Applying ancestor-type-guarded subtree exchange to node $v = 1^{k+l-1}$ in $\text{Dom}(t)$ and node $v' = 1^{k'+l'-1}$ get a tree $t'' = t[v \leftarrow \text{subtree}^t(v')] = a^{k}b^{l+2n-l'}a^{n-k}$ also belongs to $L(D')$.
- Since $a^{k}b^{l+2n-l'}a^{n-k} \notin L(D_1) \cup L(D_2)$. A contradiction.
Outline

1 Motivation

2 Preliminaries

3 Upper XSD-approximations
   - Single-type up approximations of EDTDs
   - Unions of XSDs
   - Intersection of XSDs
   - Complements of XSDs
**Proposition 3.7.** Let $D_1$ and $D_2$ be single-type EDTDs. The intersection $L(D_1) \cap L(D_2)$ is definable by a single-type EDTD.

**Proof.**

From Lemma 2.15
Regular languages closed under intersection
Theorem 2.11: stEDTD = regular tree language + ancestor-guarded subtree exchange
Theorem 3.8. Let $D_1$ and $D_2$ be two single-type EDTDs. The minimal upper XSD-approximation of $L(D_1) \cap L(D_2)$ is unique, defines precisely $L(D_1) \cap L(D_2)$ and can be computed in time $O(|D_1||D_2|)$. There is a family of pairs of single-type EDTDs $(D^n_1, D^n_2)_{n \geq 1}$, such that the size of every $D^n_1$ and $D^n_2$ is at least $n$ and the type-size of the minimal upper XSD-approximation for $L(D_1) \cap L(D_2)$ is $\Omega(|D^n_1||D^n_2|)$.

Proof.

The construction of the intersection of $D_1$ and $D_2$ is analogous to the construction in the proof of Theorem for Union of XSDs. It is different that we need to construct the intersection of the two internal DFAs.

Use the standard product construction of DFAs. It’s possible to construct in $O(|D_1||D_2|)$.

To prove the second part of the theorem, take the unary trees as an example:

let $D^n_1$ and $D^n_2$ accept unary trees of the form $a^{k \times p_1}$ and $a^{k \times p_2}$, where $1 \leq k$ and $p_1 \neq p_2$ are two smallest prime numbers larger than $n$. 


Outline

1 Motivation

2 Preliminaries

3 Upper XSD-approximations
   - Single-type up approximations of EDTDs
   - Unions of XSDs
   - Intersection of XSDs
   - Complements of XSDs
Complements of XSDs

**Theorem 3.9.** Let D be a single-type EDT. The minimal upper XSD-approximation for the complement of D is unique and can be computed in time polynomial in |D|.

**Proof.**

Let $D = (\Sigma, \Delta, d, S_d, \mu)$ and let $E = (\Sigma, A, f, S'_d)$ be the DFA-based XSD equivalent to D with $A = (\Delta, \Sigma, \delta, \{q_{init}\})$. Prove in two steps:

**First** construct an EDT $D_c$ for the complement of D

$D_c = (\Sigma, \Delta_c, d_c, S_{d_c}, \mu_c)$, use two set of types: $\Delta$ and $\Sigma$

- $\Delta_c = \Delta \cup \Sigma$
- for every $\tau \in \Delta$, $\mu_c(\tau) = \mu(\tau)$ and for every $a \in \Sigma$, $\mu_c(a) = a$
- $S_{d_c} = S_d \cup (\Sigma \setminus \mu(S_d))$
- for every $\tau \in \Delta$, $d_c(\tau) = (\Sigma^* \setminus f(\tau)) + \Sigma^* \cdot \bigcup_{a \in \Sigma} \delta(\tau, a) \cdot \Sigma^*$
- for every $a \in \Sigma$, $d_c(a) = \Sigma^*$

The EDT $D_c$ accepts $\mathcal{T}_\Sigma \setminus L(D)$ and $|D_c| = O(|\Sigma||D|)$
**Theorem 3.9.** Let $D$ be a single-type EDTD. The minimal upper XSD-approximation for the complement of $D$ is unique and can be computed in time polynomial in $|D|$.

**Proof.**

Let $D=(\Sigma, \Delta, d, S_d, \mu)$ and let $E=(\Sigma, A, f, S'_d)$ be the DFA-based XSD equivalent to $D$ with $A=(\Delta, \Sigma, \delta, \{q_{init}\})$. Prove in two steps:

*Then* the minimal upper approximation of $D_c$ can be constructed in polynomial time.

- Determinizing the type automaton of $D_c$ using subset construction can be done in polynomial time
- Type automaton $N_c$ of $D_c$ contains the type automaton $A$ of $D$ as a sub-automaton
- The subset construction result in an automaton in which every state is a state of $\{\tau, a\}$
Complements of XSDs

**Theorem 3.10.** Let $D_1$ and $D_2$ be single-type EDTDs. The minimal upper approximation of $L(D_1) \setminus L(D_2)$ can be computed in time polynomial in $|D_1| + |D_2|$

**Proof.**

Let, for each $i \in \{1, 2\}, D_i = (\Sigma, \Delta_i, d_i, S_{d_i}, \mu_i)$. Prove the theorem in two steps:

- Construct an EDTD $D_c$ for the language $L(D_1) \setminus L(D_2)$
- Its minimal upper approximation can be constructed in polynomial time

let $A_1 = (\Delta_1 \cup \{q_{init}^1\}, \Sigma, \delta_1, \{q_{init}^1\})$ be the type automaton of $D_1$
let $E_2 = (\Sigma, A_2, f_2, S'_{d_2})$ be the DFA-based XSD equivalent to $D_2$ obtained by the construction in Proposition 2.9. $A_2 = (\Delta_2 \cup \{q_{init}^2\}, \Sigma, \delta_2, \{q_{init}^2\})$ is the type automaton of $E_2$

$L(D_2) = L(E_2)$ if $t \in L(D_1) \setminus L(D_2)$ iff $t \in L(D_1) \setminus L(E_2)$

Given a tree $t$, the EDTD $D_c$ for $L(D_1) \setminus L(E_2)$ tests whether $t \in L(D_2)$ and, in parallel, guesses the path towards such a node $v$ and test whether $ch-str^t(v) \notin f_2(\tau)$
**Complements of XSDs**

**Theorem 3.10.** Let $D_1$ and $D_2$ be single-type EDTDs. The minimal upper approximation of $L(D_1) \setminus L(D_2)$ can be computed in time polynomial in $|D_1| + |D_2|$

**Proof.**

Use two sets of types $\Delta_1$ and $\Delta_1 \times \Delta_2$. Use the types $\Delta_1 \times \Delta_2$ for the path from root to $v$, use $\Delta_1$ to type all other nodes. Let $P = \{ (\tau_1, \tau_2) \in \Delta_1 \times \Delta_2 | \mu_1(\tau_1) = \mu_2(\tau_2) \}$, define

- $\Delta_c = \Delta_1 \uplus P$
- for every $\tau \in \Delta_1$, $\mu_c(\tau) = \mu(\tau)$ and for every $(\tau_1, \tau_2) \in P$, $\mu_c((\tau_1, \tau_2)) = \mu_1(\tau_1)$
- $S_{d_c} = (P \cap (S_{d_1} \times S_{d_2})) \uplus \{ \tau_1 \in S_{d_1} | \nexists \tau_2 \in S_{d_2} \text{ with } \mu(\tau_2) = \mu(\tau_1) \}$
- for every $(\tau_1, \tau_2) \in P$,
  - $d_c((\tau_1, \tau_2)) = \{ w \in d_1(\tau_1) | \mu_1(w) \notin f_2(\tau_2) \} \cup \{ w_1(\tau_1', \tau_2')w_2 | w_1\tau_1'w_2 \in d_1(\tau_1), \mu_1(\tau_1') = \mu_2(\tau_2') = a, \mu_1(w_1\tau_1'w_2) \in f_2(\tau_2), \delta_1(\tau_1, a) = \tau_1' \text{ and } \delta_2(\tau_2, a) = \tau_2' \}$
- for every $\tau \in \Delta_1$, $d_c(\tau) = d_1(\tau)$