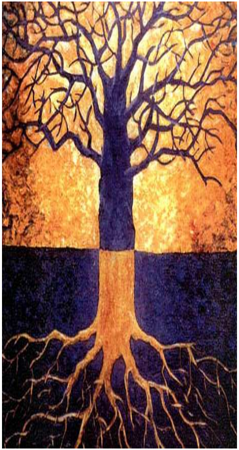


Satisfiability of a Spatial Logic with Tree Variables

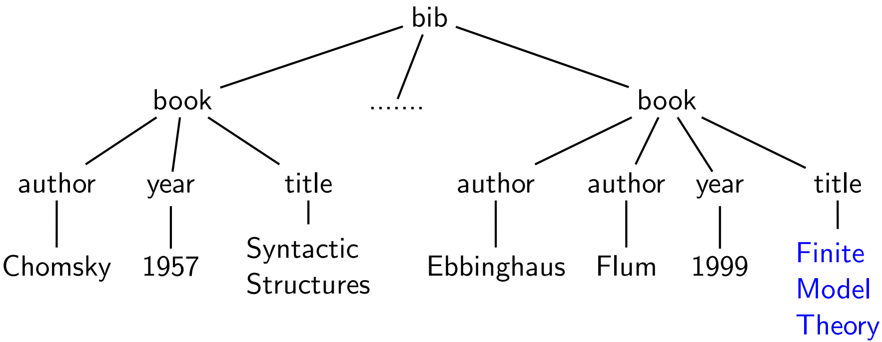
Emmanuel Filiot INRIA Futurs, Lille, Mostrare Project
Jean-Marc Talbot University of Provence, LIF, Marseille
Sophie Tison University of Lille1, LIFL, Mostrare Project

Lausanne, 2007

The Tree Query Logic (TQL)

- introduced by Cardelli and Ghelli (ICALP'02)
- adapted to ordered unranked trees
- to query XML documents
- boolean operations, recursion, tree variables
- extends CDuce pattern-matching language (non-linearity, ...)
- variable-free fragment studied for unordered trees (Boneva, Talbot, Tison, LICS'05)

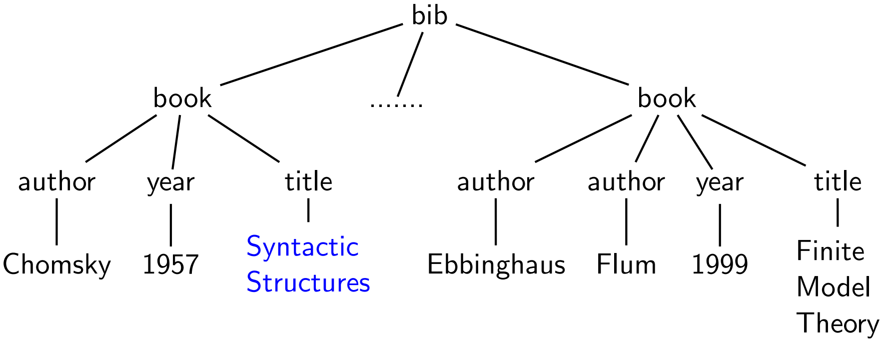
Example



Select titles

$$\mathrm{bib}[\ \_ \fatsemi \mathrm{book}[\ \_ \fatsemi \mathrm{title}[X]] \fatsemi \_\ ]$$

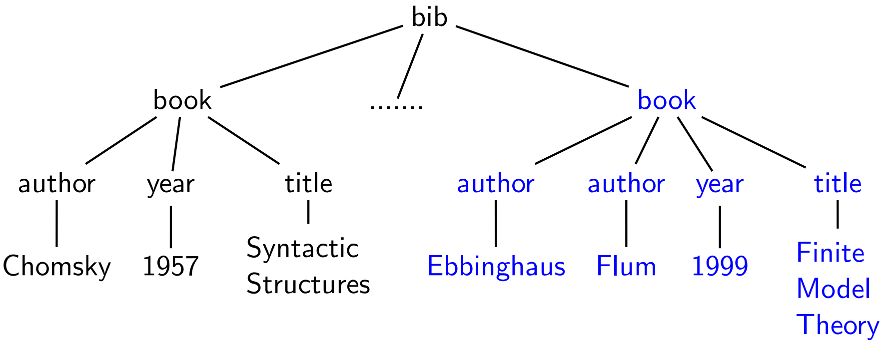
Example



Select titles

$$\mathrm{bib}[\,\_ \fatsemi \mathrm{book}[\,\_ \fatsemi \mathrm{title}[X]] \fatsemi \_\,]$$

Example

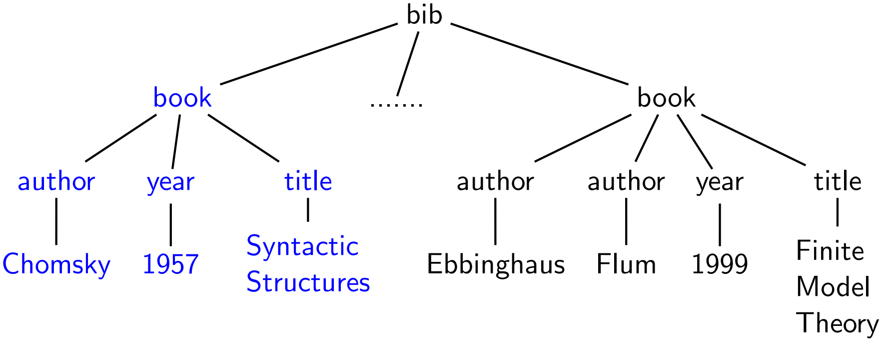


Select books published in 1999

$$\mathrm{bib}[\,\_ \fatsemi \mathrm{book}[\_ \fatsemi \mathrm{year}[1999] \fatsemi \_] \wedge X \fatsemi \_]$$

$\longrightarrow$ Model data-values by an infinite alphabet

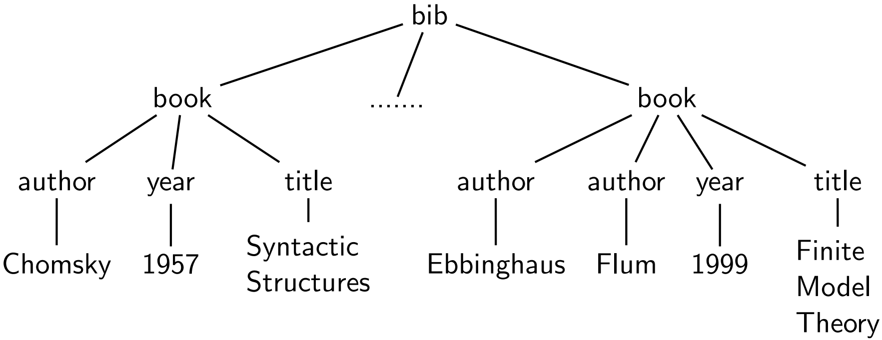
Example



Select books not published in 1999

$$\mathrm{bib}[\,\_ \fatsemi \mathrm{book}[\_ \fatsemi \mathrm{year}[\neg 1999] \fatsemi \_] \wedge X \fatsemi \_]$$

Example

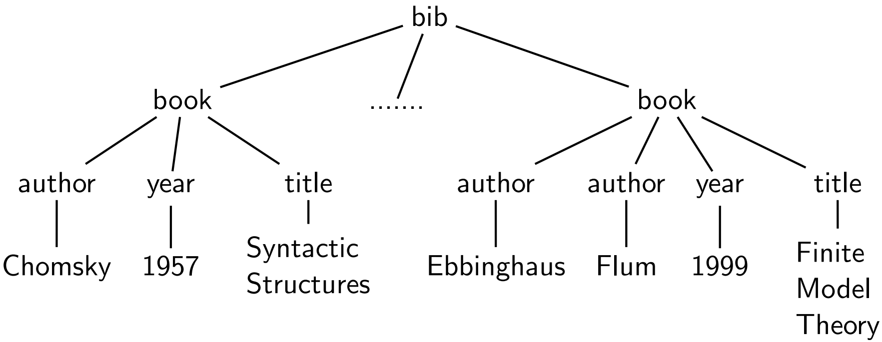


Select books which occur at least twice

$$\mathrm{bib}[\ \_ \fatsemi X \wedge \mathrm{book}[\_] \fatsemi \_ \fatsemi X \wedge \mathrm{book}[\_] \fatsemi \_\ ]$$

$\longrightarrow$ Use non-linearity to check tree equalities

Example



Check whether every book has at least one author

$$\mathrm{bib}[\ (\mathrm{book}[\mathrm{author}[\_] \fatsemi \_])^* \ ]$$

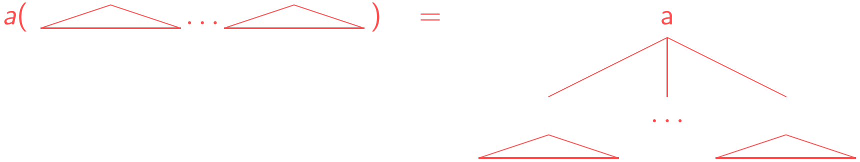
$\longrightarrow$ Use iteration to navigate by width

Outline

1. **Hedges, Automata and TQL**
2. Towards a Decidable Fragment of TQL
3. Tree Automata with Global Equalities and Disequalities (TAGED)
4. MSO with Tree Isomorphisms Tests

Hedge Signature

- $\Sigma = \{a, b, f, \dots\}$: countable set of labels
- constant 0: empty hedge
- unary symbols $a \in \Sigma$:



- binary symbol $⨾$

Hedges

Definition (Hedges)

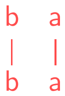
A **hedge** h is a term over the signature $\{0, \fatsemi, (a)_{a \in \Sigma}\}$. Equality relation satisfies:

$$0 \fatsemi h = h \qquad h \fatsemi 0 = h \qquad h_1 \fatsemi (h_2 \fatsemi h_3) = (h_1 \fatsemi h_2) \fatsemi h_3$$

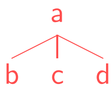
A **tree** is a rooted hedge.

Example

$b(b(0)) \fatsemi a(a(0)) \quad \longrightarrow$



$a(b(0) \fatsemi c(0) \fatsemi d(0)) \quad \longrightarrow$

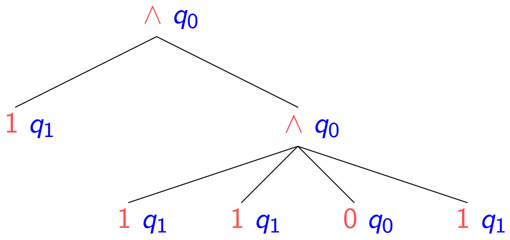


Hedge Automata (Murata, 99)

- Q: set of states
- $F \subseteq Q$: set of final states
- $\Delta \subseteq 2^{\Sigma} \times \mathrm{REG}(Q) \times Q$: set of rules, denoted $\alpha(L) \rightarrow q$
- α finite or cofinite set of labels
- $a(L) \rightarrow q$ stands for $\{a\}(L) \rightarrow q$

Example

tree and run



transitions

$$1(\epsilon) \rightarrow q_1$$
$$0(\epsilon) \rightarrow q_0$$
$$\wedge({q_1}^* q_0 (q_0 + q_1)^*) \rightarrow q_0$$
$$\wedge({q_1}^*) \rightarrow q_1$$

final states

$$F = \{q_1\}$$

TQL formulas

- formulas ϕ interpreted as set of hedges: $[\![\phi]\!] \subseteq \text{Hedges}$
- syntax and semantics:

empty hedge	0
location	$\alpha[\phi]$
concatenation	$\phi \mathbin{;} \phi'$

TQL formulas

- formulas ϕ interpreted as set of hedges: $[\![\phi]\!] \subseteq \text{Hedges}$
- syntax and semantics:

$$
\begin{array}{llcl}
\text{empty hedge} & [\![0]\!] & = & \{0\} \\
\text{location} & [\![\alpha[\phi]]\!] & = & \{a(h) \mid h \in [\![\phi]\!], a \in \alpha\},\ \alpha \subseteq \Sigma \\
\text{concatenation} & [\![\phi \mathbin{;} \phi']\!] & = & \{h \mathbin{;} h' \mid h \in [\![\phi]\!], h' \in [\![\phi']\!]\}
\end{array}
$$

TQL formulas

- formulas ϕ interpreted as set of hedges: $[\![\phi]\!] \subseteq \text{Hedges}$
- syntax and semantics:

$$
\begin{array}{llcl}
\text{empty hedge} & [\![0]\!] & = & \{0\} \\
\text{location} & [\![\alpha[\phi]]\!] & = & \{a(h) \mid h \in [\![\phi]\!], a \in \alpha\}, \;\; \alpha \subseteq \Sigma \\
\text{concatenation} & [\![\phi \fatsemi \phi']\!] & = & \{h \fatsemi h' \mid h \in [\![\phi]\!], h' \in [\![\phi']\!]\} \\
 & & & \\
\text{truth} & [\![\_]\!] & = & \text{Hedges} \\
\text{conjunction} & [\![\phi \wedge \phi']\!] & = & [\![\phi]\!] \cap [\![\phi']\!] \\
\text{negation} & [\![\neg\phi]\!] & = & \text{Hedges} \setminus [\![\phi]\!] \\
 & & & \\
\text{iteration} & [\![\phi^*]\!] & = & 0 \;\cup\; \bigcup_{i>0} \underbrace{[\![\phi]\!] \fatsemi \ldots \fatsemi [\![\phi]\!]}_{i \ \text{times}}
\end{array}
$$

TQL formulas: tree variables and recursion

- **tree variables** $X, Y, \ldots$ may occur: $\rho : \{X, Y, \ldots\} \to \text{Trees}$
- **recursion variables** $\xi, \ldots$ may occur: $\delta : \{\xi, \ldots\} \to 2^{\text{Hedges}}$
- syntax and semantics:

$$\begin{array}{lll} \text{tree variable} & [\![X]\!]_{\rho,\delta} & = \{\rho(X)\} \\ \text{recursion variable} & [\![\xi]\!]_{\rho,\delta} & = \delta(\xi) \\ \text{least fixpoint} & [\![\mu\xi.\phi]\!]_{\rho,\delta} & = \bigcap\{S \subseteq \text{Hedges} \mid [\![\phi]\!]_{\rho,\delta[\xi\mapsto S]} \subseteq S\} \end{array}$$

- all formulas considered in this talk are **recursion-closed**.
  Interpretation over ρ **only**, denoted $[\![\phi]\!]_{\rho}$.

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

- unary trees labeled only by as: $\mu\xi.(a[\xi] \;\vee\; 0)$

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

- unary trees labeled only by as: $\mu\xi.(a[\xi] \ \vee \ 0)$

$$a(\ a(\ 0\ ))\quad \models \quad \mu\xi.(a[\xi] \ \vee \ 0)$$

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

- unary trees labeled only by as: $\mu\xi.(a[\xi] \vee 0)$

$$\begin{array}{lcl} a(\ a(\ 0\ )) & \models & \mu\xi.(a[\xi] \vee 0) \\ a(\ a(\ 0\ )) & \models & a[\xi] \vee 0 \end{array}$$

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

- unary trees labeled only by as: $\mu\xi.(a[\xi] \vee 0)$

$$\begin{array}{lcl} a(\ a(\ 0\ )) & \models & \mu\xi.(a[\xi] \vee 0) \\ a(\ a(\ 0\ )) & \models & a[\xi] \vee 0 \\ a(\ a(\ 0\ )) & \models & a[\xi] \end{array}$$

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

- unary trees labeled only by as: $\mu\xi.(a[\xi] \vee 0)$

$$\begin{array}{rcl}
a(\ a(\ 0\ )) & \models & \mu\xi.(a[\xi] \vee 0) \\
a(\ a(\ 0\ )) & \models & a[\xi] \vee 0 \\
a(\ a(\ 0\ )) & \models & a[\xi] \\
a(\ 0\ ) & \models & \xi
\end{array}$$

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

- unary trees labeled only by as: $\mu\xi.(a[\xi] \vee 0)$

$$\begin{array}{rcl}
a(\ a(\ 0\ )) & \models & \mu\xi.(a[\xi] \vee 0) \\
a(\ a(\ 0\ )) & \models & a[\xi] \vee 0 \\
a(\ a(\ 0\ )) & \models & a[\xi] \\
a(\ 0\ ) & \models & \xi \\
a(\ 0\ ) & \models & a[\xi] \vee 0
\end{array}$$

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

- unary trees labeled only by as: $\mu\xi.(a[\xi] \vee 0)$

$$\begin{array}{rcl}
a(\ a(\ 0\ )) & \models & \mu\xi.(a[\xi] \vee 0) \\
a(\ a(\ 0\ )) & \models & a[\xi] \vee 0 \\
a(\ a(\ 0\ )) & \models & a[\xi] \\
a(\ 0\ ) & \models & \xi \\
a(\ 0\ ) & \models & a[\xi] \vee 0 \\
a(\ 0\ ) & \models & a[\xi]
\end{array}$$

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

- unary trees labeled only by as: $\mu\xi.(a[\xi] \vee 0)$

$$\begin{array}{rcl}
a(\ a(\ 0\ )) & \models & \mu\xi.(a[\xi] \vee 0) \\
a(\ a(\ 0\ )) & \models & a[\xi] \vee 0 \\
a(\ a(\ 0\ )) & \models & a[\xi] \\
a(\ 0\ ) & \models & \xi \\
a(\ 0\ ) & \models & a[\xi] \vee 0 \\
a(\ 0\ ) & \models & a[\xi] \\
0 & \models & \xi
\end{array}$$

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

- unary trees labeled only by as: $\mu\xi.(a[\xi] \vee 0)$

$$\begin{array}{rcl}
a(\ a(\ 0\ )) & \models & \mu\xi.(a[\xi] \vee 0) \\
a(\ a(\ 0\ )) & \models & a[\xi] \vee 0 \\
a(\ a(\ 0\ )) & \models & a[\xi] \\
a(\ 0\ ) & \models & \xi \\
a(\ 0\ ) & \models & a[\xi] \vee 0 \\
a(\ 0\ ) & \models & a[\xi] \\
0 & \models & \xi \\
0 & \models & a[\xi] \vee 0
\end{array}$$

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

- unary trees labeled only by as: $\mu\xi.(a[\xi] \ \vee \ 0)$

$$\begin{array}{rcl}
a(\ a(\ 0\ )) & \models & \mu\xi.(a[\xi] \ \vee \ 0) \\
a(\ a(\ 0\ )) & \models & a[\xi] \ \vee \ 0 \\
a(\ a(\ 0\ )) & \models & a[\xi] \\
a(\ 0\ ) & \models & \xi \\
a(\ 0\ ) & \models & a[\xi] \ \vee \ 0 \\
a(\ 0\ ) & \models & a[\xi] \\
0 & \models & \xi \\
0 & \models & a[\xi] \ \vee \ 0 \\
0 & \models & 0
\end{array}$$

Examples without Tree Variables

- set of trees:

$$\Sigma[\_]$$

- unary trees labeled only by as: $\mu\xi.(a[\xi] \ \vee \ 0)$

$$0 \quad \models \quad 0$$

- all books have been published in 2006:

$$\text{bib}[\ (\ \text{book}[\_ \,;\, \text{year}[2006]]\ )^*\ ]$$

Examples with Tree Variables

- select all books published in 2006:

$$\mathrm{bib}[\ \_\ ⨟\ (X \wedge \mathrm{book}[\ \_ ⨟ \mathrm{year}[2006]])\ ⨟\ \_\ ]$$

- there is a year during which two books have been published:

$$\mathrm{bib}[\ \_\ ⨟\ \mathrm{book}[\_ ⨟ \mathrm{year}[X]] \wedge Y\ ⨟\ \_\ ⨟\ \mathrm{book}[\_ ⨟ \mathrm{year}[X]] \wedge \neg Y\ ⨟\ \_\ ]$$

More Examples

- trees of the form $a(t, t, t, t, \ldots, t)$, for all trees t:

$$a[X^*]$$

- context-free language $a^n b^n$:

$$\mu\xi.(\ a[0] \,;\, \xi \,;\, b[0] \ \vee\ 0)$$

Outline

1. Hedges, Automata and TQL
2. **Towards a Decidable Fragment of TQL**
3. Tree Automata with Global Equalities and Disequalities (TAGED)
4. MSO with Tree Isomorphisms Tests

Undecidability of TQL

Satisfiability Problem

Given a recursion-closed formula ϕ, are there an assignment ρ of tree variables and a hedge h such that $h \in [\![\phi]\!]_\rho$?

Theorem

The satisfiability problem is undecidable for TQL formulas.

By reduction from emptiness test of intersection of two context-free grammars.

Guarded Fragment without Tree Variables

Definition

- no tree variables
- all recursion variables are **guarded**, i.e. must occur under a location: $\mu\xi.(a[0] \,⨾\, \xi \,⨾\, b[0] \;\vee\; 0)$ is **not** guarded, while $\mu\xi.(a[\xi] \vee 0)$ **is**.

Theorem (Satisfiability and Expressivity)

- *satisfiability of guarded formulas without tree variables is decidable;*
- *guarded formulas without tree variables can define all regular hedge languages.*

Adding Tree Variables: Bounded Fragment

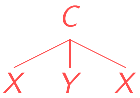
- recursions are guarded
- the number of positions where a tree is captured by a variable is bounded
- we provide a syntactic definition in the paper

Examples

$a[X^*]$	$a(t, t, \ldots, t)$	not bounded
$a[\mu\xi.(\ X \,⨾\, \xi\ \vee\ 0)]$	$a(t, t, \ldots, t)$	not bounded
$a[X \,⨾\, X]$	$a(t, t)$	bounded
$\mu\xi.(\ a[\xi]\ \vee\ X)$	$a(a(a(\ldots a(t))))$	bounded

What remains?

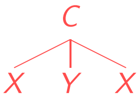
- use recursion $\mu\xi.\phi$ to navigate by depth
- use iteration ϕ^* to navigate by width
- cannot test an unbounded number of tree equalities
- but can express at least: **non-linear tree patterns** with membership constraints of this form



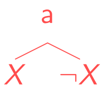
$C \in L_1 \quad X \in L_2 \quad Y \in L_3$

What remains?

- use recursion $\mu\xi.\phi$ to navigate by depth
- use iteration ϕ^* to navigate by width
- cannot test an unbounded number of tree equalities
- but can express at least: **non-linear tree patterns** with membership constraints of this form



$C \in L_1 \quad X \in L_2 \quad Y \in L_3$



(*Anti-patterns*, Kirchner, Kopetz, Moreau, ESOP'07)

Main Theorem

Theorem

Satisfiability of bounded TQL formulas is decidable.

By reduction to emptiness test of bounded TAGED.

bounded TQL formula		bounded TAGED
ϕ	$\rightarrow$	$[\![\exists X_1 \dots \exists X_n\ \phi]\!]$

where $X_1, \dots, X_n$ are the tree variables occuring in ϕ.

Outline

1. Hedges, Automata and TQL
2. Towards a Decidable Fragment of TQL
3. **Tree Automata with Global Equalities and Disequalities (TAGED)**
4. MSO with Tree Isomorphisms Tests

Tree Automata with Global Equalities and Disequalities

A **tree automata** A **with global equalities and disequalities** (TAGED) is given by:

$$\left.\begin{array}{ll} Q & \text{set of states} \\ F & \text{set of final states} \\ \Delta & \text{set of rules} \end{array}\right\} \text{ hedge automaton}$$

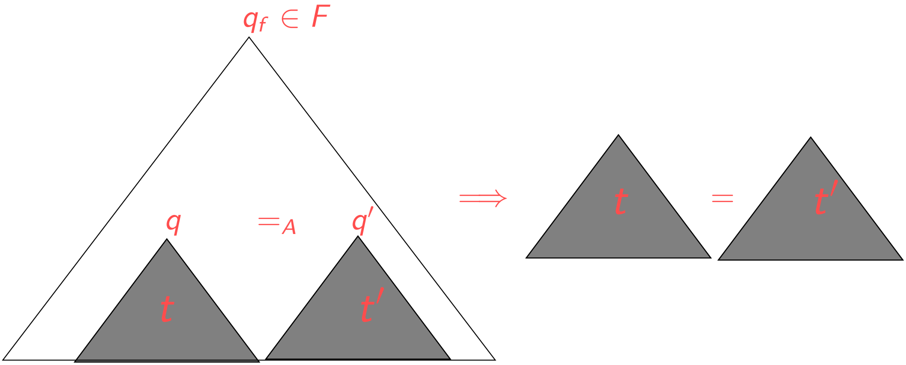
Tree Automata with Global Equalities and Disequalities

A **tree automata** A **with global equalities and disequalities** (TAGED) is given by:

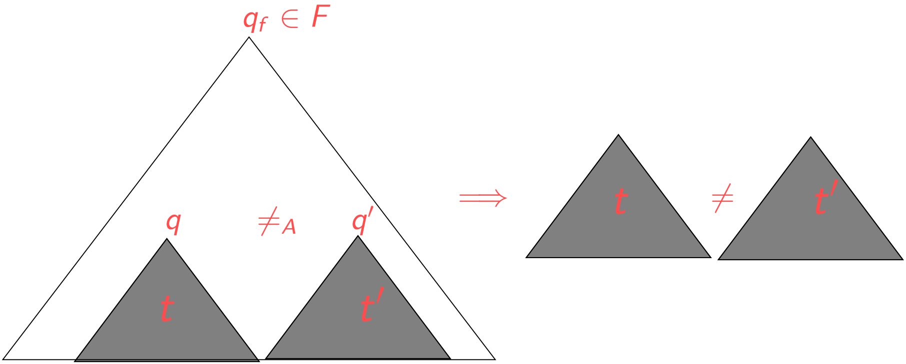
$$\left.\begin{array}{ll} Q & \text{set of states} \\ F & \text{set of final states} \\ \Delta & \text{set of rules} \end{array}\right\} \quad \text{hedge automaton}$$

$=_A \subseteq Q^2$	equivalence relation on a **subset** of Q
$\neq_A \subseteq Q^2$	non-reflexive symmetric relation

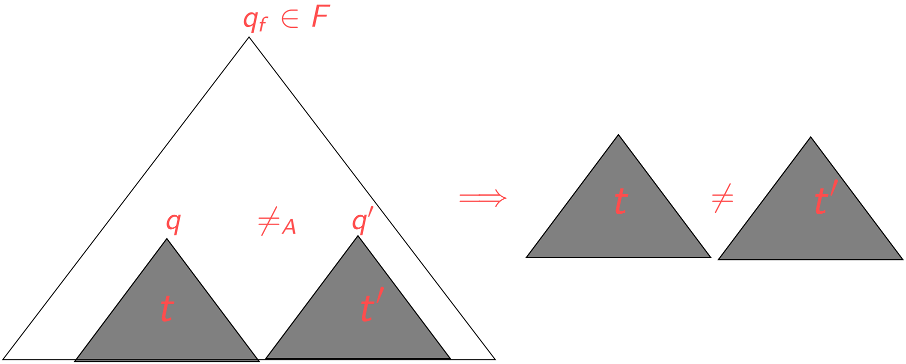
Accepting Runs



Accepting Runs



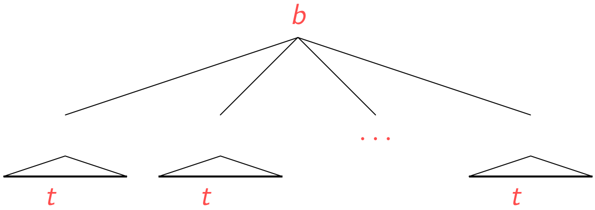
Accepting Runs



- equalities and disequalities can be tested arbitrarily faraway
- different from usual **Automata with Constraints** where tests are **local** (Bogaert, Tison, STACS'92) (Dauchet, Caron, Coquidé, JCS'95) (Karianto, Löding, ICALP'07)

Example

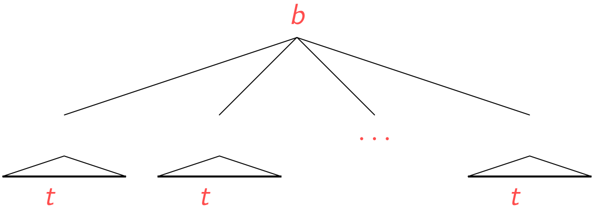
Set of trees of the form:



with t labeled only by as

Example

Set of trees of the form:



with t labeled only by as

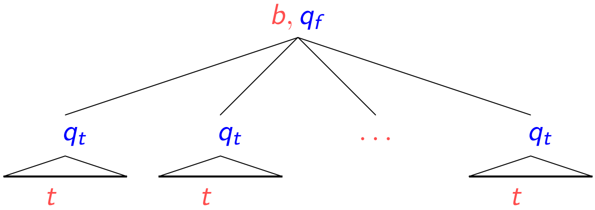
- states q, q_t, q_f
- final state: q_f
- transitions:

$$a(q^*) \to q \qquad a(q^*) \to q_t$$
$$b({q_t}^*) \to q_f$$

- equalities: $q_t =_A q_t$

Example

Set of trees of the form:



with t labeled only by as

- states q, q_t, q_f
- final state: q_f
- transitions:

$$a(q^*) \to q \qquad a(q^*) \to q_t$$
$$b({q_t}^*) \to q_f$$

- equalities: $q_t =_A q_t$

Bounded TAGED

Definition

A bounded TAGED is a pair (A, k) where A is a TAGED and $k \in \mathbb{N}$ is a natural.

Definition (Accepting Runs)

A run is accepting if every state in the domain of $=_A$ and $\neq_A$ occurs at most k times.

Examples

$\{\ b(t, t, \ldots, t) \mid t \in \mathrm{Trees}\}$ **not definable** by a bounded TAGED.

$\{\ b(t, t) \mid t \in \mathrm{Trees}\}$ **definable** by a bounded TAGED.

Emptiness Problem

- **Input**: a (bounded) TAGED A
- **Output**: is there a tree accepted by A?

Theorem

Emptiness problem for bounded TAGED is decidable.

Idea:

- decomposition into *configurations*
- emptiness test of every subpart of configurations
- context disunification procedure to manage inequalities

Relation to TQL

Theorem

- *Guarded TQL* $\Rightarrow$ *TAGED*
- *Bounded TQL* $\Rightarrow$ *bounded TAGED*
- *i.e. for all formula* ϕ *of guarded TQL (resp. bounded TQL) over* $X_1, \ldots, X_n$*, the language* $\exists X_1 \ldots \exists X_n \phi$ *is definable by a computable TAGED (resp. bounded TAGED).*

Idea Non-trivial generalization of the proof for the variable-free fragment.

Relation to TQL

Theorem

- *Guarded TQL $\Rightarrow$ TAGED*
- *Bounded TQL $\Rightarrow$ bounded TAGED*
- *i.e. for all formula ϕ of guarded TQL (resp. bounded TQL) over $X_1, \ldots, X_n$, the language $\exists X_1 \ldots \exists X_n \phi$ is definable by a computable TAGED (resp. bounded TAGED).*

Idea Non-trivial generalization of the proof for the variable-free fragment.

Corollary

Satisfiability of bounded TQL formulas is decidable.

Still open for the full guarded TQL fragment.

Outline

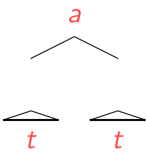
1. Hedge Algebra, Hedge Automata and TQL
2. Towards a Decidable Fragment of TQL
3. Tree Automata with Global Equalities and Disequalities (TAGED)
4. **MSO with Tree Isomorphisms Tests**

MSO with Tree Isomorphism Tests: MSO($\sim$)

- hedges h viewed as structures over the signature next-sibling, first-child, label$_a$, $a \in \Sigma$
- first-order variables denote **nodes**
- second-order variables denote **set of nodes**
- an new predicate $x \sim y$ to test tree isomorphisms between subtrees rooted at x and y respectively

Example

The language



in binary trees can be defined by:

$$\exists x \exists x_1 \exists x_2, \quad \text{root}_a(x) \wedge \text{first-child}(x, x_1) \wedge \text{next-sibling}(x_1, x_2) \wedge x_1 \sim x_2 \wedge \phi_{bin}$$

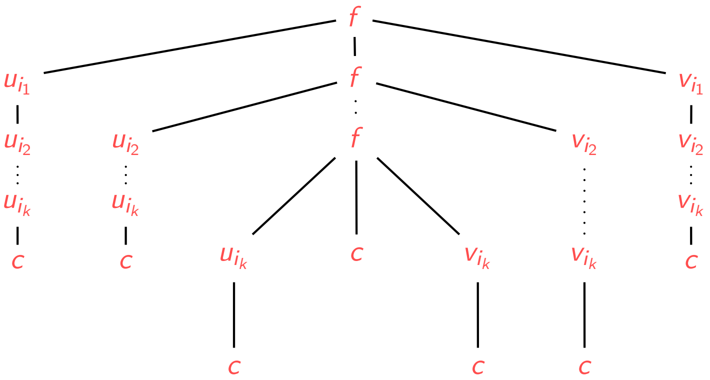
Satisfiability of MSO($\sim$)

Theorem

Satisfiability of MSO($\sim$) is undecidable.

Idea (adapted from Mongy, 81)

- Start from a PCP instance $(u_1, v_1), \ldots, (u_n, v_n)$
- Encode solutions $u_{i_1} \ldots u_{i_k} = v_{i_1} \ldots v_{i_k}$ by trees of the form:



Existential Fragment: MSO$^{\exists}(\sim)$

- Formulas of the form:

$$\exists x_1 \ldots \exists x_n \psi(x_1, \ldots, x_n)$$

- tests $x_i \sim x_j$ **only** on $x_1, \ldots, x_n$ in ψ

Theorem

- **expressivity**: *MSO$^{\exists}(\sim)$ sentences and bounded TAGED can effectively define the same hedge languages;*
- **satisfiability**: *decidable for MSO$^{\exists}(\sim)$.*

Work in progress: another application

Unification with membership constraints

- atoms of the form $s = s'$ or $x \in L$
- s, s' are terms with variables
- FO over these atoms is decidable (Comon, Delor, ICALP'90)

Work in progress: another application

Unification with membership constraints

- atoms of the form $s = s'$ or $x \in L$
- s, s' are terms with variables
- FO over these atoms is decidable (Comon, Delor, ICALP'90)
- add context variables C and atoms $C \in L$
- **restriction:** cannot use the same context variable in two different terms
- full FO is undecidable (even with the restriction)
- decidable for Existential FO (by using bounded TAGED)

Future Work: Emptiness of TAGED

$=$ \ $\neq$	no test	bounded	unbounded
no test	linear	decidable	??
bounded	EXPTIME	decidable (paper)	??
unbounded	EXPTIME-complete	decidable	??