SAT Solvers for Queries over Tree Automata with Constraints

Pierre-Cyrille Héam, Vincent Hugot, Olga Kouchnarenko

{pcheam,okouchnarenko}@lifc.univ-fcomte.fr,

vhugot@edu.univ-fcomte.fr

Université de Franche-Comté
LIFC-INRIA/CASSIS

April 8, 2010

Plan of the talk

1. **Motivating XML example**

Plan of the talk

1. **Motivating XML example**
2. **Introduction of notions:**
    1. Tree Automata
    2. TAGEDs
    3. SAT problem

Plan of the talk

1. **Motivating XML example**
2. **Introduction of notions:**
   1. Tree Automata
   2. TAGEDs
   3. SAT problem
3. **Main contribution:**
   SAT encoding for TAGED Uniform Membership Problem

Plan of the talk

1. **Motivating XML example**
2. **Introduction of notions:**
   1. Tree Automata
   2. TAGEDs
   3. SAT problem
3. **Main contribution:**
   SAT encoding for TAGED Uniform Membership Problem
4. **Some experimental results:**
   1. Natural optimisations
   2. The prototype
   3. Conversion to CNF

Plan of the talk

1. **Motivating XML example**
2. **Introduction of notions:**
   1. Tree Automata
   2. TAGEDs
   3. SAT problem
3. **Main contribution:**
   SAT encoding for TAGED Uniform Membership Problem
4. **Some experimental results:**
   1. Natural optimisations
   2. The prototype
   3. Conversion to CNF
5. **Conclusion.**

A small example

Laboratory toy example

```
<university>
  <team>
    <member> Scotty </member>
    <member> Spock </member>
    <member> Uhura </member>
    <laboratory> Enterprise </laboratory>
  </team>
  <team>
    <member> McCoy </member>
    <member> Spock </member>
    <laboratory> Enterprise </laboratory>
  </team>
</university>
```

Objective: check that all teams belong to the same laboratory and no researcher is affected to two different teams.

A small example
Laboratory toy example

```
<university>
  <team>
    <member> Scotty </member>
    <member> Spock </member>
    <member> Uhura </member>
    <laboratory> Enterprise </laboratory>
  </team>
  <team>
    <member> McCoy </member>
    <member> Spock </member>
    <laboratory> Enterprise </laboratory>
  </team>
</university>
```

Objective: check that **all teams belong to the same laboratory** and no researcher is affected to two different teams.

A small example

Laboratory toy example

```
<university>
  <team>
    <member> Scotty </member>
    <member> Spock </member>
    <member> Uhura </member>
    <laboratory> Enterprise </laboratory>
  </team>
  <team>
    <member> McCoy </member>
    <member> Spock </member>
    <laboratory> Enterprise </laboratory>
  </team>
</university>
```

Objective: check that **all teams belong to the same laboratory** and **no researcher is affected to two different teams**.

Tree automata

Definition through an example

Tree automaton for True propositional formulæ

$$\mathcal{A} \stackrel{\text{def}}{=} \left(\Sigma = \{\, \wedge, \vee/_2, \neg/_1, 0, 1/_0 \,\},\ Q = \{\, q_0, q_1 \,\}, F = \{\, q_1 \,\}, \Delta\right)$$

$$\begin{aligned} \Delta = \{ & b \to q_b, \\ & \wedge(q_b, q_{b'}) \to q_{b\wedge b'}, \\ & \vee(q_b, q_{b'}) \to q_{b\vee b'}, \\ & \neg(q_b) \to q_{\neg b} \\ & \mid\ b, b' \in 0, 1\} \end{aligned}$$

Tree automata

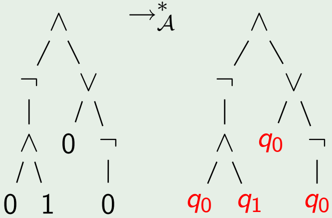
Definition through an example

```
       ∧
      / \
     ¬   ∨
     |   / \
     ∧  0   ¬
    / \     |
   0   1    0
```

Tree automata

Definition through an example

$0 \longrightarrow q_0, 1 \longrightarrow q_1 \in \Delta$



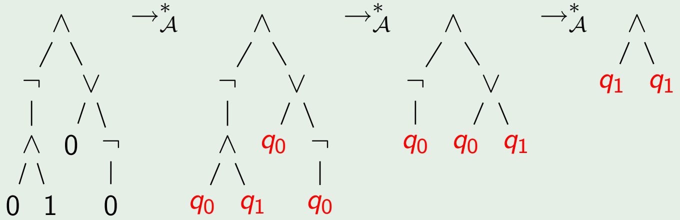
Tree automata
Definition through an example

$\wedge(q_0, q_1) \to q_0, \neg(q_0) \to q_1 \in \Delta$

$$\wedge(\neg(\wedge(0, 1)), \vee(0, \neg(0))) \to_{\mathcal{A}}^{*} \wedge(\neg(\wedge(q_0, q_1)), \vee(q_0, \neg(q_0))) \to_{\mathcal{A}}^{*} \wedge(\neg(q_0), \vee(q_0, q_1))$$

Tree automata
Definition through an example

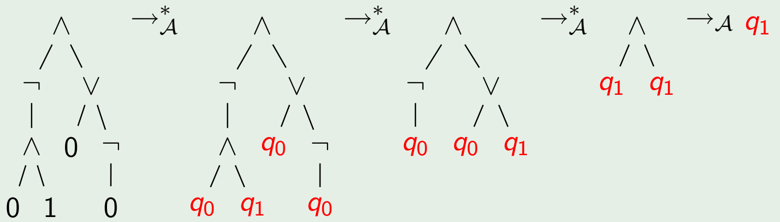
$\neg(q_0) \to q_1, \vee(q_0, q_1) \to q_1 \in \Delta$



Tree automata

Definition through an example

$\wedge(q_1, q_1) \to q_1 \in \Delta$



Tree automata

Definition through an example

$$\wedge(\neg(\wedge(0,1)), \vee(0, \neg(0))) \rightarrow^*_{\mathcal{A}} \wedge(\neg(\wedge({\color{red}q_0},{\color{red}q_1})), \vee({\color{red}q_0}, \neg({\color{red}q_0}))) \rightarrow^*_{\mathcal{A}} \wedge(\neg({\color{red}q_0}), \vee({\color{red}q_0}, {\color{red}q_1})) \rightarrow^*_{\mathcal{A}} \wedge({\color{red}q_1}, {\color{red}q_1}) \rightarrow_{\mathcal{A}} {\color{red}q_1}$$

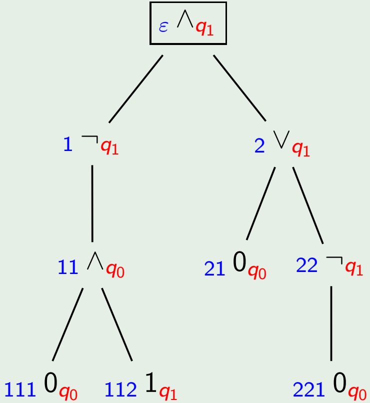
Definition: run of $\mathcal{A}$ on a term $t \in \mathcal{T}(\Sigma)$

A run ρ is a mapping from $\mathcal{P}os(t)$ to Q compatible with the transition rules.

Tree automata

Definition through an example

$\rho =$



TAGEDs

Tree Automata With Global Equality and Disequality Constraints

Introduced in Emmanuel Filiot's PhD thesis on XML query languages. See [Filiot et al., 2008].

A TAGED is a tuple $\mathcal{A} = (\Sigma, Q, F, \Delta, =_\mathcal{A}, \neq_\mathcal{A})$, where

- (Σ, Q, F, Δ) is a tree automaton
- $=_\mathcal{A}$ is a reflexive symmetric binary relation on a subset of Q
- $\neq_\mathcal{A}$ is an irreflexive and symmetric binary relation on Q. Note that in our work, we have dealt with a slightly more general case, where $\neq_\mathcal{A}$ is not necessarily irreflexive.

A TAGED $\mathcal{A}$ is said to be *positive* if $\neq_\mathcal{A}$ is empty and *negative* if $=_\mathcal{A}$ is empty.

Runs must be compatible with equality and disequality constraints.

TAGEDs

Compatibility with global constraints

Le ρ be a run of the TAGED $\mathcal{A}$ on a tree t:

Compatibility with the equality constraint $=_\mathcal{A}$

$$\forall \alpha, \beta \in \mathcal{P}os(t) : \rho(\alpha) =_\mathcal{A} \rho(\beta) \implies t|_\alpha = t|_\beta \,.$$

Compatibility with the disequality constraint $\neq_\mathcal{A}$ (irreflexive)

$$\forall \alpha, \beta \in \mathcal{P}os(t) : \rho(\alpha) \neq_\mathcal{A} \rho(\beta) \implies t|_\alpha \neq t|_\beta \,.$$

Compatibility with the disequality constraint $\neq_\mathcal{A}$ (non irreflexive)

$$\forall \alpha, \beta \in \mathcal{P}os(t) : \alpha \neq \beta \wedge \rho(\alpha) \neq_\mathcal{A} \rho(\beta) \implies t|_\alpha \neq t|_\beta \,.$$

TAGEDs

A non-regular language accepted by TAGEDs

TAGED for $\{\, f(t,t) \mid\ f \in \Sigma, t \in \mathcal{T}(\Sigma) \,\}$

$$\mathcal{A} \stackrel{\text{def}}{=} (\Sigma = \{\, a, f \,\},\ Q = \{\, q, \hat{q}, q_f \,\},\ F = \{\, q_f \,\},$$
$$\Delta,\ \hat{q} =_{\mathcal{A}} \hat{q}),$$
$$\text{where } \Delta \stackrel{\text{def}}{=} \{f(\hat{q},\hat{q}) \to q_f,\ f(q,q) \to q,\ f(q,q) \to \hat{q},$$
$$a \to q,\ a \to \hat{q}, \}$$

$$f(f(a,a),f(a,a))$$

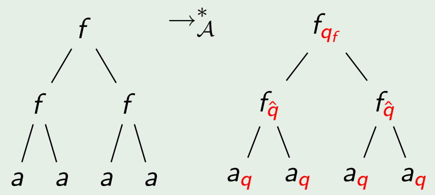
TAGEDs

A non-regular language accepted by TAGEDs

TAGED for $\{\, f(t,t) \mid f \in \Sigma, t \in \mathcal{T}(\Sigma) \,\}$

$$\mathcal{A} \overset{\text{def}}{=} (\Sigma = \{\, a, f \,\},\ Q = \{\, q, \hat{q}, q_f \,\},\ F = \{\, q_f \,\},\ \Delta,\ \hat{q} =_{\mathcal{A}} \hat{q}),$$

$$\text{where } \Delta \overset{\text{def}}{=} \{ f(\hat{q},\hat{q}) \to q_f,\ f(q,q) \to q,\ f(q,q) \to \hat{q},\ a \to q,\ a \to \hat{q}, \}$$



TAGEDs

A non-regular language accepted by TAGEDs

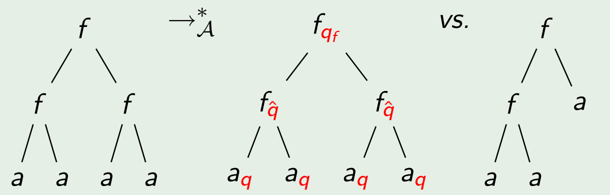
TAGED for $\{\, f(t,t) \mid\ f \in \Sigma, t \in \mathcal{T}(\Sigma) \,\}$

$$\mathcal{A} \stackrel{\text{def}}{=} (\Sigma = \{\, a, f \,\},\ Q = \{\, q, \hat{q}, q_f \,\},\ F = \{\, q_f \,\},$$

$$\Delta,\ \hat{q} =_{\mathcal{A}} \hat{q}),$$

$$\text{where } \Delta \stackrel{\text{def}}{=} \{f(\hat{q},\hat{q}) \to q_f,\ f(q,q) \to q,\ f(q,q) \to \hat{q},$$

$$a \to q,\ a \to \hat{q}, \}$$



TAGEDs

A non-regular language accepted by TAGEDs

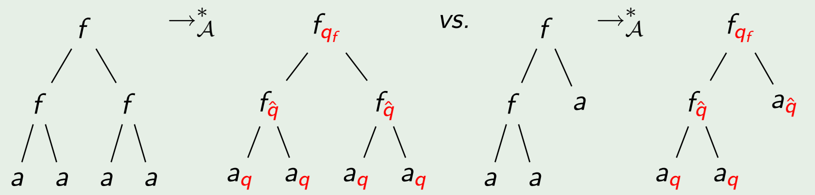
TAGED for $\{\, f(t,t) \mid f \in \Sigma, t \in \mathcal{T}(\Sigma) \,\}$

$$\mathcal{A} \stackrel{\text{def}}{=} (\Sigma = \{\, a, f \,\},\ Q = \{\, q, \hat{q}, q_f \,\},\ F = \{\, q_f \,\},$$

$$\Delta,\ \hat{q} =_{\mathcal{A}} \hat{q}),$$

$$\text{where } \Delta \stackrel{\text{def}}{=} \{f(\hat{q},\hat{q}) \to q_f,\ f(q,q) \to q,\ f(q,q) \to \hat{q},$$

$$a \to q,\ a \to \hat{q},\}$$



TAGED membership

... through SAT solvers

Uniform Membership Problem

INPUT: $\mathcal{A}$ a TAGED and $t \in \mathcal{T}(\Sigma)$ a term.
OUTPUT: Is t accepted by $\mathcal{A}$?

TAGED membership
. . . through SAT solvers

Uniform Membership Problem

INPUT: $\mathcal{A}$ a TAGED and $t \in \mathcal{T}(\Sigma)$ a term.
OUTPUT: Is t accepted by $\mathcal{A}$?

Theorem [Filiot2008]

The Uniform Membership Problem for TAGEDs is *NP*-complete.

TAGED membership

. . . through SAT solvers

Uniform Membership Problem

INPUT: $\mathcal{A}$ a TAGED and $t \in \mathcal{T}(\Sigma)$ a term.
OUTPUT: Is t accepted by $\mathcal{A}$?

Theorem [Filiot2008]

The Uniform Membership Problem for TAGEDs is *NP*-complete.

Possible practical approach

XML/. . . $\Rightarrow$ TAGED membership

TAGED membership
...through SAT solvers

Uniform Membership Problem

INPUT: $\mathcal{A}$ a TAGED and $t \in \mathcal{T}(\Sigma)$ a term.
OUTPUT: Is t accepted by $\mathcal{A}$?

Theorem [Filiot2008]

The Uniform Membership Problem for TAGEDs is *NP*-complete.

Possible practical approach

XML/... $\Rightarrow$ TAGED membership $\Rightarrow$ SAT problem

TAGED membership

. . . through SAT solvers

Uniform Membership Problem

INPUT: $\mathcal{A}$ a TAGED and $t \in \mathcal{T}(\Sigma)$ a term.
OUTPUT: Is t accepted by $\mathcal{A}$?

Theorem [Filiot2008]

The Uniform Membership Problem for TAGEDs is *NP*-complete.

Possible practical approach

XML/. . . $\Rightarrow$ TAGED membership $\Rightarrow$ SAT problem

TAGED membership
. . . through SAT solvers

Uniform Membership Problem

INPUT: $\mathcal{A}$ a TAGED and $t \in \mathcal{T}(\Sigma)$ a term.
OUTPUT: Is t accepted by $\mathcal{A}$?

Theorem [Filiot2008]

The Uniform Membership Problem for TAGEDs is *NP*-complete.

Possible practical approach

XML/. . . $\Rightarrow$ TAGED membership $\Rightarrow$ SAT problem $\Rightarrow$ *answer*

The SAT Problem
. . . and applications

Definition: The SAT problem

Given a propositional formula, for instance

$$\varphi = X \vee (\neg X \wedge \neg Y) \text{ or } \psi = X \wedge (\neg X \wedge \neg Y),$$

is there a valuation such that the formula evaluates to true?

NP-complete

The SAT problem is the first known *NP*-complete decision problem (Cook, 1971).

In practice

There are very efficient heuristics implanted in modern SAT solvers.

Plan

1. The SAT encoding

2. Implementation and Experiments

The SAT encoding

Choice of variables and $\Theta_{\twoheadrightarrow}$

Variables: X_q^α

A run is a mapping from $\mathcal{P}os(t)$ to Q. So we take variables of the form X_q^α, meaning:

$$\forall \alpha \in \mathcal{P}os(t), q \in Q : X_q^\alpha \iff \rho(\alpha) = q$$

The SAT encoding

Choice of variables and $\Theta_{\nrightarrow}$

Variables: X_q^α

A run is a mapping from $\mathcal{P}os(t)$ to Q. So we take variables of the form X_q^α, meaning:

$$\forall \alpha \in \mathcal{P}os(t), q \in Q : X_q^\alpha \iff \rho(\alpha) = q$$

Partial function constraint $\Theta_{\nrightarrow}$: "ρ is a function"

$$\Theta_{\nrightarrow} \stackrel{\text{def}}{=} \bigwedge_{\substack{\alpha \in \mathcal{P}os(t) \\ q \in Q}} \left[X_q^\alpha \implies \bigwedge_{\substack{p \in Q \\ p \neq q}} \neg X_p^\alpha \right]$$

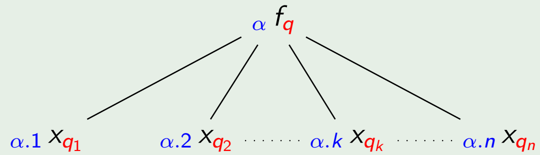
The SAT encoding

Rule application and compatibility: $\Psi^{\alpha}(r)$ and $\Phi^{\varepsilon}(t)$

Rule application constraint $\Psi^{\alpha}(r)$

We define, for any $\alpha \in \mathcal{P}os(t)$, and any transition rule $f(q_1, \dots, q_n) \to q \in \Delta$,

$$\Psi^{\alpha}(f(q_1, \dots, q_n) \to q) \stackrel{\text{def}}{=} X_q^{\alpha} \wedge \bigwedge_{k=1}^{n} X_{q_k}^{\alpha.k}.$$



The SAT encoding

Rule application and compatibility: $\Psi^\alpha(r)$ and $\Phi^\varepsilon(t)$

Rule application constraint $\Psi^\alpha(r)$

We define, for any $\alpha \in \mathcal{P}os(t)$, and any transition rule $f(q_1, \dots, q_n) \to q \in \Delta$,

$$\Psi^\alpha(f(q_1, \dots, q_n) \to q) \stackrel{\text{def}}{=} X_q^\alpha \wedge \bigwedge_{k=1}^{n} X_{q_k}^{\alpha.k}.$$

Rules compatibility constraint $\Phi^\varepsilon(t)$

$$\Phi^\varepsilon(t) \stackrel{\text{def}}{=} \bigwedge_{\alpha \in \mathcal{P}os(t)} \left[\bigvee_{r \in \Delta_{t(\alpha)}} \Psi^\alpha(r) \right]$$

where $\Delta_f = \{ f(\dots) \to \dots \in \Delta \}$.

The SAT encoding

Connecting to TAGEDs: $\Theta_{\leftrightarrows}$

Accepting run for tree automata

We have coded runs for tree automata: one more constraint: $\bigvee_{q \in F} X_q^{\varepsilon}$ makes sure we end up in a final state.

The SAT encoding
Connecting to TAGEDs: $\Theta_{\leftrightarrows}$

Accepting run for tree automata

We have coded runs for tree automata: one more constraint: $\bigvee_{q \in F} X_q^{\varepsilon}$ makes sure we end up in a final state.

New variables: T_u^q

We need new variables to link subterms and states: T_u^q denotes "the subterm u evaluates to q", for any $u \trianglelefteq t$ and $q \in Q$.

The SAT encoding
Connecting to TAGEDs: $\Theta_{\leftrightarrows}$

Accepting run for tree automata

We have coded runs for tree automata: one more constraint: $\bigvee_{q\in F} X_q^\varepsilon$ makes sure we end up in a final state.

New variables: T_u^q

We need new variables to link subterms and states: T_u^q denotes "the subterm u evaluates to q", for any $u \trianglelefteq t$ and $q \in Q$.

Structural glue: $\Theta_{\leftrightarrows}$

$$\Theta_{\leftrightarrows} \stackrel{\text{def}}{=} \bigwedge_{\substack{\alpha\in\mathcal{P}os(t)\\ q\in Q}} \left[X_q^\alpha \implies T_{t|_\alpha}^q\right].$$

The SAT encoding
Compatibility with $=_{\mathcal{A}}$ and $\neq_{\mathcal{A}}$

Compatibility with $=_{\mathcal{A}}$: $\Theta_{=_{\mathcal{A}}}$

$$\Theta_{=_{\mathcal{A}}} \stackrel{\text{def}}{=} \bigwedge_{\substack{\alpha \in \mathcal{P}os(t) \\ q \in Q}} \left[X_q^\alpha \implies \bigwedge_{\substack{p \in Q \\ p =_{\mathcal{A}} q}} \bigwedge_{\substack{u \trianglelefteq t \\ u \neq t|_\alpha}} \neg T_u^p \right]$$

Compatibility with $\neq_{\mathcal{A}}$ ($p \neq q$): $\Theta_{\neq_{\mathcal{A}}}$

$$\Theta_{\neq_{\mathcal{A}}} \stackrel{\text{def}}{=} \bigwedge_{\substack{\alpha \in \mathcal{P}os(t) \\ q \in Q}} \left[X_q^\alpha \implies \bigwedge_{\substack{p \in Q \\ p \neq_{\mathcal{A}} q \\ p \neq q}} \neg T_{t|_\alpha}^p \right]$$

The SAT encoding

Compatibility with $=_\mathcal{A}$ and $\neq_\mathcal{A}$

Compatibility with $\neq_\mathcal{A}$ with structural glue $\Theta_\leftrightarrows$

$$\Theta_{\neq_\mathcal{A}} \wedge \Theta_\leftrightarrows \stackrel{\text{def}}{=} \bigwedge_{\substack{\alpha\in\mathcal{P}os(t)\\ q\in Q}} \left[X_q^\alpha \implies T^q_{t|_\alpha} \wedge \bigwedge_{\substack{p\in Q\\ p\neq_\mathcal{A} q\\ p\neq q}} \neg T^p_{t|_\alpha} \right]$$

The idea: a counterexample

Let $t \in \mathcal{T}(\Sigma)$, $\alpha, \beta \in \mathcal{P}os(t)$ and $u = t|_\alpha = t|_\beta$. Suppose that ρ is a run such that $\rho(\alpha) = p$ and $\rho(\beta) = q$ with $p \neq_\mathcal{A} q$. Then we have X_p^α, X_q^β,

The SAT encoding
Compatibility with $=_\mathcal{A}$ and $\neq_\mathcal{A}$

Compatibility with $\neq_\mathcal{A}$ with structural glue $\Theta_\leftrightarrows$

$$\Theta_{\neq_\mathcal{A}} \wedge \Theta_\leftrightarrows \stackrel{\text{def}}{=} \bigwedge_{\substack{\alpha\in\mathcal{P}os(t)\\ q\in Q}} \left[X_q^\alpha \implies T_{t|_\alpha}^q \wedge \bigwedge_{\substack{p\in Q\\ p\neq_\mathcal{A} q\\ p\neq q}} \neg T_{t|_\alpha}^p \right]$$

The idea: a counterexample

Let $t \in \mathcal{T}(\Sigma)$, $\alpha, \beta \in \mathcal{P}os(t)$ and $u = t|_\alpha = t|_\beta$. Suppose that ρ is a run such that $\rho(\alpha) = p$ and $\rho(\beta) = q$ with $p \neq_\mathcal{A} q$. Then we have $X_p^\alpha, X_q^\beta, T_{t|_\alpha}^p, \neg T_{t|_\alpha}^q,$

The SAT encoding

Compatibility with $=_\mathcal{A}$ and $\neq_\mathcal{A}$

Compatibility with $\neq_\mathcal{A}$ with structural glue $\Theta_\leftrightarrows$

$$\Theta_{\neq_\mathcal{A}} \wedge \Theta_\leftrightarrows \stackrel{\text{def}}{=} \bigwedge_{\substack{\alpha \in \mathcal{P}os(t) \\ q \in Q}} \left[X_q^\alpha \implies T_{t|_\alpha}^q \wedge \bigwedge_{\substack{p \in Q \\ p \neq_\mathcal{A} q \\ p \neq q}} \neg T_{t|_\alpha}^p \right]$$

The idea: a counterexample

Let $t \in \mathcal{T}(\Sigma)$, $\alpha, \beta \in \mathcal{P}os(t)$ and $u = t|_\alpha = t|_\beta$. Suppose that ρ is a run such that $\rho(\alpha) = p$ and $\rho(\beta) = q$ with $p \neq_\mathcal{A} q$. Then we have $X_p^\alpha, X_q^\beta, T_{t|_\alpha}^p, \neg T_{t|_\alpha}^q, T_{t|_\beta}^q, \neg T_{t|_\beta}^p$.

The SAT encoding

Compatibility with $=_\mathcal{A}$ and $\neq_\mathcal{A}$

Compatibility with $\neq_\mathcal{A}$ with structural glue $\Theta_\leftrightarrows$

$$\Theta_{\neq_\mathcal{A}} \wedge \Theta_\leftrightarrows \stackrel{\text{def}}{=} \bigwedge_{\substack{\alpha \in \mathcal{P}os(t) \\ q \in Q}} \left[X_q^\alpha \implies T^q_{t|_\alpha} \wedge \bigwedge_{\substack{p \in Q \\ p \neq_\mathcal{A} q \\ p \neq q}} \neg T^p_{t|_\alpha} \right]$$

The idea: a counterexample

Let $t \in \mathcal{T}(\Sigma)$, $\alpha, \beta \in \mathcal{P}os(t)$ and $u = t|_\alpha = t|_\beta$. Suppose that ρ is a run such that $\rho(\alpha) = p$ and $\rho(\beta) = q$ with $p \neq_\mathcal{A} q$. Then we have $X_p^\alpha, X_q^\beta, T^p_{t|_\alpha}, \neg T^q_{t|_\alpha}, T^q_{t|_\beta}, \neg T^p_{t|_\beta}$. Since $u = t|_\alpha = t|_\beta$ we have $T_u^q, \neg T_u^q, T_u^p, \neg T_u^p$, hence the formula is not satisfiable.

The SAT encoding

Compatibility with $\neq_{\mathcal{A}}$, non-irreflexive case: $\Omega_{\neq_{\mathcal{A}}}$

Compatibility with $\neq_{\mathcal{A}}$ $(p \neq q)$: $\Theta_{\neq_{\mathcal{A}}}$

$$\Theta_{\neq_{\mathcal{A}}} \stackrel{\text{def}}{=} \bigwedge_{\substack{\alpha \in \mathcal{P}os(t) \\ q \in Q}} \left[X_q^\alpha \implies \bigwedge_{\substack{p \in Q \\ p \neq_{\mathcal{A}} q \\ p \neq q}} \neg T_{t|_\alpha}^p \right]$$

New variables: S_u^α

We need new variables to link subterms and positions: S_u^α encodes the statement "the subterm u is rooted in α".

Compatibility with $\neq_{\mathcal{A}}$ (non-irreflexive; $q \neq_{\mathcal{A}} q$): $\Omega_{\neq_{\mathcal{A}}}$

$$\Omega_{\neq_{\mathcal{A}}} \stackrel{\text{def}}{=} \bigwedge_{\alpha \in \mathcal{P}os(t)} S_{t|_\alpha}^\alpha \wedge \bigwedge_{\substack{\alpha \neq \beta \in \mathcal{P}os(t) \\ q \neq_{\mathcal{A}} q}} \left[X_q^\alpha \wedge X_q^\beta \implies \neg S_{t|_\beta}^\alpha \right]$$

The SAT encoding
Completeness and soundness

Definition (SAT encoding of TAGED membership problem $\Delta_{\mathcal{A}}(t)$)

Let $\mathcal{A} = (\Sigma, \Delta, Q, F, =_{\mathcal{A}}, \neq_{\mathcal{A}})$ be a TAGED and $t \in \mathcal{T}(\Sigma)$; then we define

$$\Delta_{\mathcal{A}}(t) \stackrel{\text{def}}{=} \Theta_{\nrightarrow} \wedge \Phi^{\varepsilon}(t) \wedge \bigvee_{q \in F} X_q^{\varepsilon} \wedge \Theta_{=_{\mathcal{A}}} \wedge \Theta_{\neq_{\mathcal{A}}} \wedge \Omega_{\neq_{\mathcal{A}}}.$$

Theorem (TAGED membership, correctness and soundness)

There exists a successful run ρ of the TAGED $\mathcal{A}$ on a term t iff $\Delta_{\mathcal{A}}(t)$ is satisfiable. Moreover, if $I \models \Delta_{\mathcal{A}}(t)$, then for any $\alpha \in \mathcal{P}os(t)$ we have $\rho(\alpha) = q \iff I \models X_q^{\alpha}$.

Plan

Possible optimisations

... from simple observations

The formulæ can be trimmed down: consider

Structural glue: $\Theta_{\leftrightarrows}$

$$\Theta_{\leftrightarrows} \stackrel{\text{def}}{=} \bigwedge_{\substack{\alpha \in \mathcal{P}os(t) \\ q \in Q}} \left[X_q^\alpha \implies T_{t|_\alpha}^q \right].$$

Not all couples (α, q) are necessary because not all states are obtainable at any given position.

Definition (Possibly obtainable states at position α)

$$\sigma(\alpha) \stackrel{\text{def}}{=} \{ q \in Q / \exists t(\alpha)(\dots) \to q \in \Delta \}$$

Given a position α, we only need to deal with $q \in \sigma(\alpha)$.

Possible optimisations

. . . from simple observations

The formulæ can be trimmed down: consider

Structural glue: $\Theta_{\leftrightarrows}$

$$\Theta_{\leftrightarrows} \stackrel{\text{def}}{=} \bigwedge_{\substack{\alpha \in \mathcal{P}os(t) \\ q \in \sigma(\alpha)}} \left[X_q^\alpha \implies T^q_{t|_\alpha} \right] .$$

Not all couples (α, q) are necessary because not all states are obtainable at any given position.

Definition (Possibly obtainable states at position α)

$$\sigma(\alpha) \stackrel{\text{def}}{=} \{ q \in Q / \ \exists t(\alpha)(\dots) \to q \in \Delta \}$$

Given a position α, we only need to deal with $q \in \sigma(\alpha)$.

The prototype

Intro/ Input format

Prototype (`OCaml`). Takes a `TAGED` and a term as input (syntax close to the Tree Automata library *Timbuk*).

```
(** TAGED Automaton for {f(x,x)} *)
  Taged fxxA
  Alphabet f a b
  States q qq qf
  Final qf
  Rules
    f qq qq : qf
    f q q   : q
    f q q   : qq
    a:q a:qq
    b:q b:qq
  Equal
    qq qq
  Different
    qq qf
```

```
f(f(a,a), f(a,a))
// in a_fxx
```

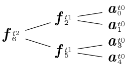
The prototype
Generated formula

Automaton: **fxxA**
Alphabet: $\{\boldsymbol{f},\ \boldsymbol{a},\ \boldsymbol{b}\}$
States: $\{\mathsf{q}, \mathsf{qq}, \mathsf{qf}\}$
Final States: $\{\mathsf{qf}\}$
Transition Rules: $\{$

- $\boldsymbol{f}(\mathsf{qq}, \mathsf{qq}) \rightarrow \mathsf{qf}$,
- $\boldsymbol{f}(\mathsf{q}, \mathsf{q}) \rightarrow \mathsf{q}$,
- $\boldsymbol{f}(\mathsf{q}, \mathsf{q}) \rightarrow \mathsf{qq}$,
- $\boldsymbol{a} \rightarrow \mathsf{q}$,
- $\boldsymbol{a} \rightarrow \mathsf{qq}$,
- $\boldsymbol{b} \rightarrow \mathsf{q}$,
- $\boldsymbol{b} \rightarrow \mathsf{qq}$

$\}$
Global State Equality: $\{\mathsf{qq} = \mathsf{qq}\}$
Global State Disequality: $\{\mathsf{qq} \neq \mathsf{qf}\}$
End Automaton.

Term as expression: $\boldsymbol{f}\{\boldsymbol{f}[\boldsymbol{a},\ \boldsymbol{a}],\ \boldsymbol{f}[\boldsymbol{a},\ \boldsymbol{a}]\}$
Term as tree:



Membership formula $= [[(X^3_{\mathsf{q}} \vee X^3_{\mathsf{qq}}) \wedge ([X^5_{\mathsf{qf}} \wedge X^3_{\mathsf{qq}} \wedge X^4_{\mathsf{qq}}] \vee [X^5_{\mathsf{q}} \wedge X^3_{\mathsf{q}} \wedge X^4_{\mathsf{q}}] \vee [X^5_{\mathsf{qq}} \wedge X^3_{\mathsf{q}} \wedge X^4_{\mathsf{q}}]) \wedge ([X^2_{\mathsf{qf}} \wedge X^0_{\mathsf{qq}} \wedge X^1_{\mathsf{qq}}] \vee [X^2_{\mathsf{q}} \wedge X^0_{\mathsf{q}} \wedge X^1_{\mathsf{q}}] \vee [X^2_{\mathsf{qq}} \wedge X^0_{\mathsf{q}} \wedge X^1_{\mathsf{q}}]) \wedge (X^1_{\mathsf{q}} \vee X^1_{\mathsf{qq}}) \wedge ([X^6_{\mathsf{qf}} \wedge X^2_{\mathsf{qq}} \wedge X^5_{\mathsf{qq}}] \vee [X^6_{\mathsf{q}} \wedge X^2_{\mathsf{q}} \wedge X^5_{\mathsf{q}}] \vee [X^6_{\mathsf{qq}} \wedge X^2_{\mathsf{q}} \wedge X^5_{\mathsf{q}}]) \wedge (X^0_{\mathsf{q}} \vee X^0_{\mathsf{qq}}) \wedge (X^4_{\mathsf{q}} \vee X^4_{\mathsf{qq}})] \wedge [\neg X^6_{\mathsf{q}} \wedge \neg X^6_{\mathsf{qq}}] \wedge [\{X^3_{\mathsf{qq}} \Rightarrow T^{\mathsf{qq}}_0\} \wedge \{X^5_{\mathsf{qq}} \Rightarrow T^{\mathsf{qq}}_1\} \wedge \{X^2_{\mathsf{qq}} \Rightarrow T^{\mathsf{qq}}_1\} \wedge \{X^1_{\mathsf{qq}} \Rightarrow T^{\mathsf{qq}}_0\} \wedge \{X^6_{\mathsf{qq}} \Rightarrow T^{\mathsf{qq}}_2\} \wedge \{X^0_{\mathsf{qq}} \Rightarrow T^{\mathsf{qq}}_0\} \wedge \{X^4_{\mathsf{qq}} \Rightarrow T^{\mathsf{qq}}_0\}] \wedge [\{X^5_{\mathsf{qq}} \Rightarrow \neg T^{\mathsf{qf}}_1\} \wedge \{X^2_{\mathsf{qq}} \Rightarrow \neg T^{\mathsf{qf}}_1\} \wedge \{X^6_{\mathsf{qq}} \Rightarrow \neg T^{\mathsf{qf}}_2\}] \wedge [\{X^3_{\mathsf{qq}} \Rightarrow [\neg T^{\mathsf{qq}}_2 \wedge \neg T^{\mathsf{qq}}_1]\} \wedge \{X^5_{\mathsf{qq}} \Rightarrow [\neg T^{\mathsf{qq}}_2 \wedge \neg T^{\mathsf{qq}}_0]\} \wedge \{X^2_{\mathsf{qq}} \Rightarrow [\neg T^{\mathsf{qq}}_2 \wedge \neg T^{\mathsf{qq}}_0]\} \wedge \{X^1_{\mathsf{qq}} \Rightarrow [\neg T^{\mathsf{qq}}_2 \wedge \neg T^{\mathsf{qq}}_1]\} \wedge \{X^6_{\mathsf{qq}} \Rightarrow [\neg T^{\mathsf{qq}}_1 \wedge \neg T^{\mathsf{qq}}_0]\} \wedge \{X^0_{\mathsf{qq}} \Rightarrow [\neg T^{\mathsf{qq}}_2 \wedge \neg T^{\mathsf{qq}}_1]\} \wedge \{X^4_{\mathsf{qq}} \Rightarrow [\neg T^{\mathsf{qq}}_2 \wedge \neg T^{\mathsf{qq}}_1]\}]]$

The prototype
CNF Conversion: the BAT

Definition: Conjunctive Normal Form

A formula is in Conjunctive Normal Form (CNF) if it is a conjunction of disjunctions of literals and contains only $\neg$, $\wedge$ or $\vee$.

DIMACS CNF

$$\varphi = (X \vee \neg Z) \wedge (Y \vee Z \vee \neg X).$$

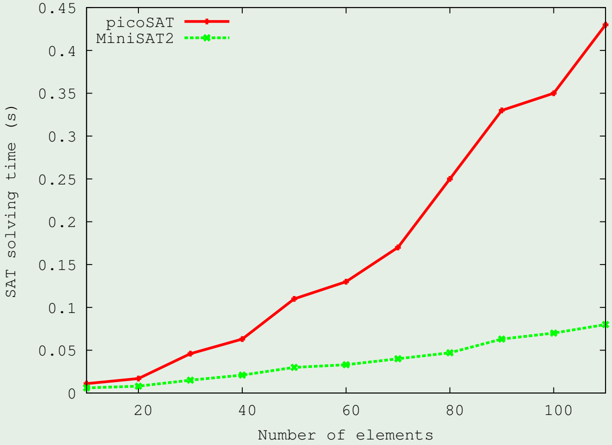
```
c DIMACS CNF for φ
p cnf 3 2
1 -3 0
2 3 -1 0
```

Problem!

Our formula is not in CNF! We must convert it. The BAT solves it.

The prototype
Results: Laboratory example

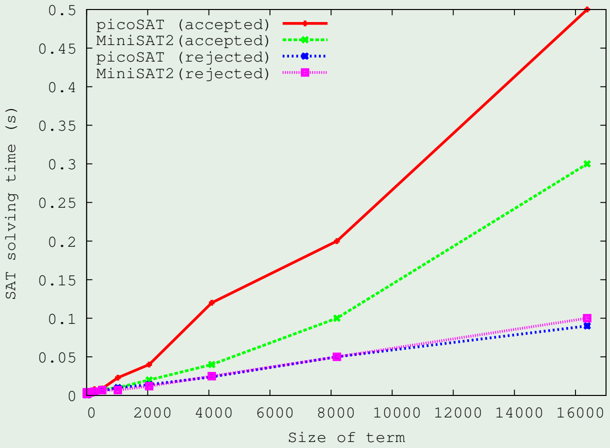
CNF solving time, Laboratory example



The prototype

Results: $\{\, f(t,t) \mid\; f \in \Sigma, t \in \mathcal{T}(\Sigma) \,\}$ example

CNF solving time, $\{\, f(t,t) \mid\; f \in \Sigma, t \in \mathcal{T}(\Sigma) \,\}$



Conclusion, step by step

$$\text{XML} \Rightarrow \texttt{TAGED} \Rightarrow \Delta_{\mathcal{A}}(t) \Rightarrow \text{CNF} \Rightarrow \text{SAT solver} \Rightarrow \textit{answer}$$

Conclusion, step by step

XML $\Rightarrow$ `TAGED` $\Rightarrow$ $\Delta_{\mathcal{A}}(t)$ $\Rightarrow$ CNF $\Rightarrow$ SAT solver $\Rightarrow$ *answer*

Theoretical contribution

SAT encoding for the `TAGED` uniform membership problem. Natural optimisations.

Complexity

The formula is quadratic in the size of the input, and so is the generation time (worst case).

Implementation

Unoptimised `OCaml` prototype generating optimised $\Delta_{\mathcal{A}}(t)$ from input `TAGED` $\mathcal{A}$ and term t.

Conclusion, step by step

XML $\Rightarrow$ TAGED $\Rightarrow$ $\Delta_{\mathcal{A}}(t)$ $\Rightarrow$ CNF $\Rightarrow$ SAT solver $\Rightarrow$ *answer*

Conversion to CNF

We use an existing tool, BAT [Manolios and Vroon, 2009], to convert our formula to DIMACS CNF.

Caveats

For now, CNF conversion is the bottleneck of our experiments:

- BAT is 4.5 times slower than formula generation on big formulæ.
- BAT crashed (stack overflow) on big formulæ

So we could not produce tests big enough to really push the SAT solvers to their limits.

Conclusion, step by step

$\text{XML} \Rightarrow \texttt{TAGED} \Rightarrow \Delta_{\mathcal{A}}(t) \Rightarrow \text{CNF} \Rightarrow \text{SAT solver} \Rightarrow \textit{answer}$

Tested SAT solvers

Solvers `picoSAT` and `MiniSAT2` both display good performances, `MiniSAT2` seeming faster in general.

Results

Efficient (sub-second) SAT solving even in largest tested cases[a]

[a] Order of magnitude: Term of 20 000 nodes, formula of 70 000 variables, 120 000 clauses and 250 000 literals (in CNF)

Conclusion, step by step

XML $\Rightarrow$ `TAGED` $\Rightarrow$ $\Delta_{\mathcal{A}}(t)$ $\Rightarrow$ CNF $\Rightarrow$ SAT solver $\Rightarrow$ *answer*

Conclusion

Overall, the current experimental limitations lie in the computationally easier part of the problem, while its inherent difficulty (*NP*-completeness) seems well overcome by the heuristics of tested SAT solvers.

Some references

[Comon et al., 2007, Filiot et al., 2008, Manolios and Vroon, 2009]

Comon, H., Dauchet, M., Gilleron, R., Löding, C., Jacquemard, F., Lugiez, D., Tison, S., and Tommasi, M. (2007).
Tree automata techniques and applications.
release October, 12th 2007.

Filiot, E., Talbot, J.-M., and Tison, S. (2008).
Tree Automata with Global Constraints.
In *12th International Conference on Developments in Language Theory (DLT)*, pages 314–326, Kyoto Japon.

Manolios, B. and Vroon, D. (2009).
Faster SAT Solving with Better CNF Generation.
Design, Automation and Test in Europe, DATE.