A Tree Automata Theory for Unification Modulo Equational Rewriting

Hitoshi Ohsaki and Toshinori Takai

National Institute of Advanced Industrial Science and Technology (AIST)
Nakoji 3–11–46, Amagasaki 661–0974, Japan

{ohsaki,takai}@ni.aist.go.jp

Abstract. An extension of tree automata framework, called equational tree automata, is presented. This theory is useful to deal with unification modulo equational rewriting. In the manuscript, we demonstrate how equational tree automata can be applied to unification problems.

1 Equational Tree Languages

Unification modulo equational theory is a central topic in automated reasoning. Tree automata are the powerful technique for handling unification modulo rewriting [2]. On the other hand, to model some network security problems like Diffie-Hellman key exchange algorithm, rewrite rules and equations (e.g. associativity and commutativity axioms) have to be separately dealt with in the underlying theory, but it causes the situation where the standard tree automata technique is useless. In our recent papers [5, 6], we have proposed an extension of tree automata, which is called equational tree automata. This framework subsumes Petri nets (Example 1). In a practical example, equational tree automata can be used to verify a security problem of Diffie-Hellman protocol (Example 2).

We start this section with basics of tree automata and the equational extension. A *tree automaton* (TA for short) $\mathcal{A}$ is defined by the 4-tuple $(\mathcal{F}, \mathcal{Q}, \mathcal{Q}_{fin}, \Delta)$: each of those components is a signature $\mathcal{F}$ (a finite set of function symbols with fixed arities), a finite set $\mathcal{Q}$ of states (special constants with $\mathcal{F} \cap \mathcal{Q} = \varnothing$), a subset $\mathcal{Q}_{fin}$ of $\mathcal{Q}$ consisting of so-called *final states* and a finite set Δ of transition rules in the following form:

- $f(p_1, \ldots, p_n) \to t$

for some $f \in \mathcal{F}$ with $\mathsf{arity}(f) = n$ and $p_1, \ldots, p_n \in \mathcal{Q}$ and $t \in \mathcal{T}(\{f\} \cup \mathcal{Q})$. Note that function symbols, except state symbols, in the right-hand side must be the same as one in the left-hand side. An *equational tree automaton* (ETA for short) $\mathcal{A}/\mathcal{E}$ is the combination of a TA $\mathcal{A}$ and an equational system (a set of equations; ES for short) $\mathcal{E}$ over the same signature $\mathcal{F}$. An ETA $\mathcal{A}/\mathcal{E}$ is called

- *regular* if the right-hand side t is a single state q,
- *monotone* if the right-hand side t is a single state q or a term $f(q_1, \ldots, q_n)$

for every transition rule $f(p_1, \ldots, p_n) \to t$ in Δ. Note that equational tree automata defined in [5, 6] are in the above monotone case. A term t is *accepted* by

$\mathcal{A}/\mathcal{E}$ if $t \in \mathcal{T}(\mathcal{F})$ and $t \to^*_{\mathcal{A}/\mathcal{E}} q$ for some $q \in \mathcal{Q}_{fin}$. Elements in the set $\mathcal{L}(\mathcal{A}/\mathcal{E})$ are ground terms accepted by $\mathcal{A}/\mathcal{E}$.

A *tree language* (TL for short) L over $\mathcal{F}$ is some subset of $\mathcal{T}(\mathcal{F})$. A TL L is $\mathcal{E}$-*recognizable* if there exists $\mathcal{A}/\mathcal{E}$ such that $L = \mathcal{L}(\mathcal{A}/\mathcal{E})$. Similarly, L is called $\mathcal{E}$-monotone ($\mathcal{E}$-regular) if $\mathcal{A}/\mathcal{E}$ is monotone (regular). If L is $\mathcal{E}$-recognizable with $\mathcal{E} = \varnothing$, we say L is recognizable. Likewise, we say L is monotone (regular) if L is $\varnothing$-monotone ($\varnothing$-regular). We say $\mathcal{A}/\mathcal{E}$ is a C-TA (A-TA, AC-TA) if $\mathcal{E} = \mathsf{C}$ ($\mathcal{E} = \mathsf{A}$, $\mathcal{E} = \mathsf{AC}$, respectively).

Lemma 1. *Every* C*-recognizable tree language is regular.*

Proof. Let $\mathcal{F}$ be the signature. Suppose L is a tree language recognizable with a C-TA $\mathcal{A}/\mathsf{C}$, where $\mathcal{A} = (\mathcal{F}, \mathcal{Q}, \mathcal{Q}_{fin}, \Delta)$. We define $\mathcal{B} = (\mathcal{F}, \mathcal{Q}, \mathcal{Q}_{fin}, \Delta')$, where

$$\Delta' = \left\{ f(p_1, \ldots, p_n) \to q \;\middle|\; \begin{array}{l} f(q_1, \ldots, q_n) \to r \in \Delta \text{ such that} \\ f(p_1, \ldots, p_n) \sim_{\mathsf{C}} f(q_1, \ldots, q_n) \text{ and } r \to^*_{\mathcal{A}/\mathsf{C}} q \end{array} \right\}.$$

Then it can be proved that the regular TA $\mathcal{B}$ recognizes L. □

Lemma 2. *The following language hierarchy holds if* $\mathcal{E} = \mathsf{A}$*:*

regular TL $\subsetneq$ $\mathcal{E}$*-regular TL* $\subsetneq$ $\mathcal{E}$*-monotone TL* $\subsetneq$ $\mathcal{E}$*-recognizable TL*

Proof. For the first inclusion, we take the tree language $L = \{t \mid |t|_{\mathsf{a}} = |t|_{\mathsf{b}}\}$ over the signature $\mathcal{F} = \{\mathsf{f}, \mathsf{a}, \mathsf{b}\}$, where $\mathsf{arity}(\mathsf{f}) = 2$ and a, b are constant symbols. If $\mathcal{F}_{\mathrm{A}} = \{\mathsf{f}\}$ then L is A-regular (Lemma 8, [5]), but is not regular. The second inclusion is proved in [6]. For the third inclusion we suppose $\mathcal{F} = \mathcal{F}_0 \cup \{\mathsf{f}\}$ with $\mathcal{F}_{\mathrm{A}} = \{\mathsf{f}\}$. Here $\mathcal{F}_0$ denotes a set of constant symbols. Then, a (word) language W over $\mathcal{F}_0$ is context-sensitive if and only if an A-monotone tree language is *maximal* for W. A tree language L is called maximal for a language W if for all terms t in $\mathcal{T}(\mathcal{F})$, $\mathsf{leaf}(t) \in W$ if and only if $t \in L$. Similarly, it holds that a language W is recursively enumerable if and only if an A-recognizable tree language is maximal for W. It is known that recursively enumerable languages strictly include context-sensitive languages. □

Question 1. Does the above hierarchy hold also for $\mathcal{E} = \mathsf{AC}$? Note that the first inclusion is trivial. Kudlek and Mitrana [3] showed in word case that multiset regular languages and multiset arbitrary languages are closed under homomorphisms and substitutions but multiset monotone languages are not. However, it does not imply the strict hierarchy of tree case, because of different definitions.

2 AC-Tree Automata for Unification Problems

In this section we discuss the applications of equational tree automata for unification problems. Our examples rely on the following decidability result.

Theorem 1 (Reachable property problem). *Given a ground AC-TRS* $\mathcal{R}/\mathsf{AC}$ *and tree languages* L_1, L_2 *over* $\mathcal{F}$ *with* $\mathcal{F}_{\mathrm{AC}}$*. If* L_1 *and* L_2 *are* AC*-recognizable tree languages, it is decidable whether there exist some* s *in* L_1 *and* t *in* L_2 *such that* $s \to^*_{\mathcal{R}/\mathsf{AC}} t$*, i.e.* $(\to^*_{\mathcal{R}/\mathsf{AC}})[L_1] \cap L_2 \neq \varnothing$ *is a computable question.*

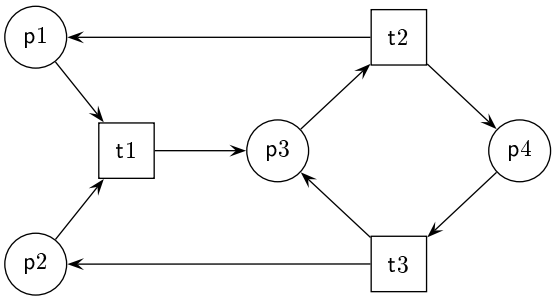

Fig. 1. A Petri net example: $\mathcal{P}$

Proof. For a singleton $\mathcal{F}_{\mathrm{AC}}$, the proof proceeds in the way of Lemma 4 in [5]. To extend $\mathcal{F}_{\mathrm{AC}}$ by allowing to have arbitrary many AC-symbols, we apply the similar argument of Section 3 in [5]. □

Example 1. Petri nets are known to be a special class of ground AC-TRSs. A Petri net is a triple (P, T, W), where P is a finite set of places, T is a finite set of transitions and W is a weight-function $(P \times T) \cup (T \times P) \to \mathbb{N}$. For instance, the Petri net $\mathcal{P}$ illustrated in Fig. 1 has W with $W(\mathsf{p1}, \mathsf{t1}) = 1$, $W(\mathsf{p2}, \mathsf{t1}) = 1$, $W(\mathsf{p3}, \mathsf{t2}) = 1$, $W(\mathsf{p4}, \mathsf{t3}) = 1$, $W(\mathsf{t1}, \mathsf{p3}) = 1$, $W(\mathsf{t2}, \mathsf{p1}) = 1$, $W(\mathsf{t2}, \mathsf{p4}) = 1$, $W(\mathsf{t3}, \mathsf{p2}) = 1$, $W(\mathsf{t3}, \mathsf{p3}) = 1$. In the figure, places are denoted by circles, and transitions are squares. The value of W determines the weight of directed arcs between places and transitions. Then, the associated ground AC-TRS $(\mathcal{F}, \mathcal{R}/\mathsf{AC})$ is defined by $\mathcal{F} = \{+\} \cup \{\epsilon, \mathsf{p1}, \ldots, \mathsf{p4}\}$, $\mathcal{F}_{\mathrm{AC}} = \{+\}$ and $\mathcal{R} = \{\mathsf{p1}+\mathsf{p2} \to \mathsf{p3},\ \mathsf{p3} \to \mathsf{p1}+\mathsf{p4},\ \mathsf{p4} \to \mathsf{p2}+\mathsf{p3}\} \cup \{\epsilon + \mathsf{p}i \to \mathsf{p}i,\ \mathsf{p}i \to \mathsf{p}i + \epsilon \mid 1 \leqslant i \leqslant 4\}$. In this setting, a state of a Petri net (the number of *tokens* on each place) is encoded by a multiset of place symbols. The empty multiset is represented by ϵ.

Given two sets L_1, L_2 of states of $\mathcal{P}$. According to Theorem 1, it is decidable whether there exist states $m_1 \in L_1$ and $m_2 \in L_2$ such that $m_1 \to_{\mathcal{P}}^* m_2$, provided L_1, L_2 are leaf-languages of AC-recognizable tree languages over $\mathcal{F}$. The binary relation $\to_{\mathcal{P}}^*$ is the reflexive-transitive closure of one-step transition relation. This decidability property generalizes the result of Mayr [4].

Using the above property, for instance, we can solve *coverability* problem, which is a question of whether there exists m_3 such that $m_1 \to_{\mathcal{P}}^* m_3$ and $m_2 \subseteq m_3$. Actually, it is verified by solving the following question, which is decidable:

$$\exists \sigma ?.\ \ t_1 \to_{\mathcal{R}/\mathsf{AC}}^* t_2 + x\sigma.$$

Here t_1, t_2 are terms over $\mathcal{F}$ such that $\mathsf{leaf}(t_1) = m_1$ and $\mathsf{leaf}(t_2) = m_2$.

Example 2. We consider a simple network protocol. The protocol illustrated in Fig. 2 is called Diffie-Hellman key exchange algorithm (e.g., Section 22.1, [7]). In the protocol, a principal A chooses a prime number N and sends to B together with an integer $k(A) \circ N$ that is generated with a random number $k(A)$. Here we suppose that nobody else can guess $k(A)$ from $k(A) \circ N$. Then B returns $k(B) \circ N$

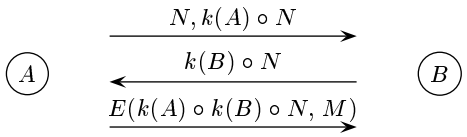

Fig. 2. Diffie-Hellman key exchange algorithm

to A. By assuming $\circ$ to be associative and commutative, $k(A) \circ k(B) \circ N$ can be used as a common secret key for A and B. It enables A to send only B a secret message M encrypted with this key. A security problem for this protocol is whether or not someone else can retrieve a secret message M by listening on the channel.

In term rewriting, the axiom of encryption and decryption and the property of keys are specified by the AC-rewrite system $\mathcal{R} = \{D(x, E(x, y)) \to y\}$ and $\mathsf{AC} = \{x \circ y \approx y \circ x,\ (x \circ y) \circ z \approx x \circ (y \circ z)\}$. On the other hand, a principal C wiretapping the channel can obtain N, $k(A) \circ N$, $k(B) \circ N$ and $E(k(A) \circ k(B) \circ N,\ M)$. Moreover, C is supposed to have personal data C, $k(C)$ and to be able to use function symbols $D, E, \circ$. So C's knowledge is the set L of terms constructible from these components. Then, the security problem is verified by solving the following unification problem:

$$\exists \sigma ?.\ \ x\sigma \to^*_{\mathcal{R}/\mathsf{AC}} M \ \text{ for some } \ x\sigma \in L.$$

In this setting, $(\to^*_{\mathcal{R}/\mathsf{AC}})[\,L\,]$ is an AC-monotone tree language. One should notice that in order to compute $(\to^*_{\mathcal{R}/\mathsf{AC}})[\,L\,]$ by using a modified algorithm of Kaji *et al.* [2], *intersection-emptiness* problem for AC-monotone tree languages must be decidable. Obviously a membership problem $M \in (\to^*_{\mathcal{R}/\mathsf{AC}})[\,L\,]$ is decidable.

Closure properties for equational tree languages are stated below.

Theorem 2. *$\mathcal{E}$-monotone tree languages are closed under union and intersection if $\mathcal{E} \in \{\,\mathsf{C}, \mathsf{A}, \mathsf{AC}\,\}$. The same closure property holds for $\mathcal{E}$-recognizable tree languages.*

Proof. If $\mathcal{E} = \mathsf{C}$, it is trivial from Lemma 1. Moreover, $\mathcal{E}$-monotone case is proved in [6]. For $\mathcal{E}$-recognizable tree languages, we take $\mathcal{A}$ and $\mathcal{B}$ where $\mathcal{A} = (\mathcal{F}, \mathcal{Q}_1, \mathcal{Q}_{1fin}, \Delta_1)$ and $\mathcal{B} = (\mathcal{F}, \mathcal{Q}_2, \mathcal{Q}_{2fin}, \Delta_2)$. Here we suppose $\mathcal{Q}_1 \cap \mathcal{Q}_2 = \varnothing$. We define $\mathcal{C} = (\mathcal{F}, \mathcal{Q}, \mathcal{Q}_{fin}, \Delta_\cup)$ where $\mathcal{Q} = \mathcal{Q}_1 \cup \mathcal{Q}_2$, $\mathcal{Q}_{fin} = \mathcal{Q}_{1fin} \cup \mathcal{Q}_{2fin}$ and $\Delta_\cup = \Delta_1 \cup \Delta_2$. By construction, $\mathcal{L}(\mathcal{A}/\mathcal{E}) \cup \mathcal{L}(\mathcal{B}/\mathcal{E}) \subseteq \mathcal{L}(\mathcal{C}/\mathcal{E})$ is obvious. And the reverse can be proved easily, because if $t \to^*_{\mathcal{C}/\mathcal{E}} q \in \mathcal{Q}_i$ then $t \in \mathcal{T}(\mathcal{F} \cup \mathcal{Q}_i)$. Note that for all transition rules $l \to r \in \Delta_\cup$, if $|r|_\mathcal{Q} = 0$ then $l = r$ (l, r are the same constant symbols), because $r \in \mathcal{T}(\{\mathsf{root}(l)\} \cup \mathcal{Q})$. Moreover, $s \to_{\mathcal{C}/\mathcal{E}} s'$ with $|s|_{\mathcal{Q}_i} \geqslant 1$ implies $|s'|_{\mathcal{Q}_i} \geqslant 1$ for each $1 \leqslant i \leqslant 2$.

For the intersection of $\mathcal{L}(\mathcal{A}/\mathcal{E})$ and $\mathcal{L}(\mathcal{B}/\mathcal{E})$, we define $\mathcal{D} = (\mathcal{F}, \mathcal{P}, \mathcal{P}_{fin}, \Delta_\cap)$:

- $\mathcal{P} = \{\langle p\rangle,\ \langle q\rangle,\ \langle p, q\rangle \mid p \in \mathcal{Q}_1 \text{ and } q \in \mathcal{Q}_2\}$,
- $\mathcal{P}_{fin} = \{\langle p, q\rangle \mid p \in \mathcal{Q}_{1fin} \text{ and } q \in \mathcal{Q}_{2fin}\}$,

– $\Delta_\cap = \Delta_\times \cup \Delta_{\overline{1}} \cup \Delta_{\overline{2}} \cup \Delta_\leftrightarrow$, where

$\Delta_\times$: $f(\langle p_1, q_1\rangle, \ldots, \langle p_n, q_n\rangle) \to \langle p, q\rangle$
if $f(p_1, \ldots, p_n) \to^*_{\mathcal{A}/\mathcal{E}} p$ and $f(q_1, \ldots, q_n) \to^*_{\mathcal{B}/\mathcal{E}} q$ with $f \notin \mathsf{fun}(\mathcal{E})$,

$\Delta_{\overline{1}}$: $f(\langle p_1, q_1\rangle, \langle p_2, q_2\rangle) \to C_f[\![\langle p'_1, q_1\rangle, \langle p'_2, q_2\rangle, \langle p'_3\rangle \ldots, \langle p'_m\rangle]\!]$
$f(\langle p_1, q_1\rangle, \langle p_2\rangle) \to C_f[\![\langle p'_1, q_1\rangle, \langle p'_2\rangle, \ldots, \langle p'_m\rangle]\!]$
if $f(p_1, p_2) \to C_f[\![p'_1, \ldots, p'_m]\!] \in \Delta_1$ with $f \in \mathsf{fun}(\mathcal{E})$;
$f(\langle p_1, q_1\rangle, \langle p_2, q_2\rangle) \to f(\langle p'_1, q_1\rangle, \langle q_2\rangle)$
$f(\langle p_1, q_1\rangle, \langle p_2\rangle) \to \langle p'_1, q_1\rangle$
if $f(p_1, p_2) \to p'_1 \in \Delta_1$ with $f \in \mathsf{fun}(\mathcal{E})$,

$\Delta_{\overline{2}}$: $f(\langle p_1, q_1\rangle, \langle q_2\rangle) \to C_f[\![\langle p_1, q'_1\rangle, \langle q'_2\rangle, \ldots, \langle q'_m\rangle]\!]$
$f(\langle q_1\rangle, \langle q_2\rangle) \to C_f[\![\langle q'_1\rangle, \ldots, \langle q'_m\rangle]\!]$
if $f(q_1, q_2) \to C_f[\![q'_1, \ldots, q'_m]\!] \in \Delta_2$ with $f \in \mathsf{fun}(\mathcal{E})$;
$f(\langle p_1, q_1\rangle, \langle q_2\rangle) \to \langle p_1, q'_1\rangle$
$f(\langle q_1\rangle, \langle q_2\rangle) \to \langle q'_1\rangle$
if $f(q_1, q_2) \to q'_1 \in \Delta_2$ with $f \in \mathsf{fun}(\mathcal{E})$,

$\Delta_\leftrightarrow$: $f(\langle p_1, q_1\rangle, \langle p_2\rangle) \to f(\langle p_1\rangle, \langle p_2, q_1\rangle)$
$f(\langle p_1\rangle, \langle p_2, q_2\rangle) \to f(\langle p_1, q_2\rangle, \langle p_2\rangle)$
$f(\langle p_1, q_1\rangle, \langle q_2\rangle) \to f(\langle q_1\rangle, \langle p_1, q_2\rangle)$
$f(\langle q_1\rangle, \langle p_2, q_2\rangle) \to f(\langle p_2, q_1\rangle, \langle q_2\rangle)$
if $f \in \mathsf{fun}(\mathcal{E})$

for all $f \in \mathcal{F}$, $p_1, \ldots, p_n, p'_1, \ldots, p'_m, p \in \mathcal{Q}_1$ and $q_1, \ldots, q_n, q'_1, \ldots, q'_m, q \in \mathcal{Q}_2$. In the above definition, C_f denotes a non-empty context consisting of f. (If $\mathsf{arity}(f) = n$, C_f has at least n holes.) The ETA $\mathcal{D}/\mathcal{E}$ satisfies $\mathcal{L}(\mathcal{D}/\mathcal{E}) = \mathcal{L}(\mathcal{A}/\mathcal{E}) \cap \mathcal{L}(\mathcal{B}/\mathcal{E})$ if $\mathcal{E} = \mathsf{A}$ ($\mathcal{E} = \mathsf{AC}$). □

Remark 1. $\mathcal{E}$-regular tree languages are closed under union. Moreover, C-regular TL are closed under intersection, but A-regular TL are not closed intersection. On the other hand, closure under intersection of AC-regular TL is open [5].

References

1. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison and M. Tommasi: *Tree Automata Techniques and Applications*, draft, 1999. Available on `http://www.grappa.univ-lille3.fr/tata/`
2. Y. Kaji, T. Fujiwara and T. Kasami: *Solving a Unification Problem under Constrained Substitutions Using Tree Automata*, JSC 23, pp. 79–117, 1997.
3. M. Kudlek and V. Mitrana: *Normal Forms of Grammars, Finite Automata, Abstract Families, and Closure Properties of Macrosets*, Multiset Processing, LNCS 2235, pp. 135–146, 2001.
4. E.W. Mayr: *An Algorithm for the General Petri Net Reachability Problem*, SIAM J. Comput. 13(3), pp. 441–460, 1984.
5. H. Ohsaki and T. Takai: *Reachability and Closure Properties of Equational Tree Languages*, Proc. of 13th RTA, 2002. To appear in LNCS.
6. H. Ohsaki: *Beyond Regularity: Equational Tree Automata for Associative and Commutative Theories*, Proc. of 15th CSL, LNCS 2142, pp. 539–553, 2001.
7. B. Schneier: *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, Second Edition, John Wiley & Sons, 1996.