Equational Tree Automata: Towards Automated Verification of Network Protocols[⋆]

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 several realistic unification examples, e.g. including a security problem of network protocols.

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, 7], 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 $(\mathcal{F}, \mathcal{Q}, \mathcal{Q}_{\mathit{fin}}, \Delta)$: 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 $\mathcal{Q}_{\mathit{fin}}$ 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 $p_1, \ldots, p_n \in \mathcal{Q}$. The right-hand side t is a term consisting of f and state symbols. A function symbol f in the right-hand side must be the same as one in the left-hand side.

Each of $\mathcal{F}_{\mathrm{A}}$ and $\mathcal{F}_{\mathrm{C}}$ consists of some binary function symbols of the signature $\mathcal{F}$. The intersection of $\mathcal{F}_{\mathrm{A}}$ and $\mathcal{F}_{\mathrm{C}}$ is denoted by $\mathcal{F}_{\mathrm{AC}}$. A set of associativity axioms $f(f(x, y), z) \approx f(x, f(y, z))$ for all $f \in \mathcal{F}_{\mathrm{A}}$ is denoted by $\mathsf{A}(\mathcal{F}_{\mathrm{A}})$. Likewise, a set of commutativity axioms $f(x, y) \approx f(y, x)$ for all $f \in \mathcal{F}_{\mathrm{C}}$ is $\mathsf{C}(\mathcal{F}_{\mathrm{C}})$. The union of of $\mathsf{A}(\mathcal{F}_{\mathrm{AC}})$ and $\mathsf{C}(\mathcal{F}_{\mathrm{AC}})$ is represented by $\mathsf{AC}(\mathcal{F}_{\mathrm{AC}})$. If unnecessary to be explicit,

⋆ This paper is a modified version of the authors' UNIF2002 paper [6].

we write A, C and AC, respectively. An *equational tree automaton* (ETA for short) $\mathcal{A}/\mathcal{E}$ is the combination of a TA $\mathcal{A}$ and a set $\mathcal{E}$ of equations 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 $f(q_1, \dots, q_n)$

for every transition rule $f(p_1, \dots, p_n) \to t$ in Δ. Equational tree automata defined in [4, 5, 7] are in the above monotone case.

A term t in $\mathcal{T}(\mathcal{F})$ is *accepted* by $\mathcal{A}/\mathcal{E}$ if $t \to^*_{\mathcal{A}/\mathcal{E}} q$ for some $q \in \mathcal{Q}_{\mathit{fin}}$. The set of terms accepted by $\mathcal{A}/\mathcal{E}$ is denoted by $\mathcal{L}(\mathcal{A}/\mathcal{E})$. 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. We suppose a tree language is recognizable with a C-TA $\mathcal{A}/\mathsf{C}$, where $\mathcal{A} = (\mathcal{F}, \mathcal{Q}, \mathcal{Q}_{\mathit{fin}}, \Delta)$. Define $\mathcal{B} = (\mathcal{F}, \mathcal{Q}, \mathcal{Q}_{\mathit{fin}}, \Delta')$ with $\Delta' = \{f(p_1, \dots, p_n) \to q \mid f(q_1, \dots, q_n) \to r \in \Delta$ such that $f(p_1, \dots, p_n) \sim_{\mathsf{C}} f(q_1, \dots, q_n)$ and $r \to^*_{\mathcal{A}/\mathsf{C}} q\}$. Then it can be proved that the regular TA $\mathcal{B}$ recognizes $\mathcal{L}(\mathcal{A}/\mathsf{C})$. $\square$

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

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

However, the classes of regular TL and $\mathcal{E}$*-recognizable TL are incomparable.*

Proof. The first inclusion relation is proved in [7]. For the second 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 TL is *maximal* for W. A TL 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 TL is maximal for W. It is known that recursively enumerable languages strictly include context-sensitive languages. The difference of the classes of regular TL and $\mathcal{E}$-recognizable TL are proved by taking the TL $L_1 = \{\mathsf{f}(\mathsf{f}(\mathsf{a}, \mathsf{a}), \mathsf{a})\}$ under the assumption of $\mathcal{F}_{\mathrm{A}} = \{\mathsf{f}\}$. The TL L_1 is regular (as it is finite), but it not recognizable with A-TA, because an A-TA which accepts $\mathsf{f}(\mathsf{f}(\mathsf{a}, \mathsf{a}), \mathsf{a})$ also accepts $\mathsf{f}(\mathsf{a}, \mathsf{f}(\mathsf{a}, \mathsf{a}))$. On the other hand, we take the TL $L_2 = \{t \mid |t|_{\mathsf{a}} = |t|_{\mathsf{b}}\}$ 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. $\square$

Remark 1. We know the same hierarchy holds also for $\mathcal{E} = \mathsf{AC}$, except

$\mathcal{E}$-monotone TL $\subsetneq$ $\mathcal{E}$-recognizable TL.

The above relation remains as an open question.

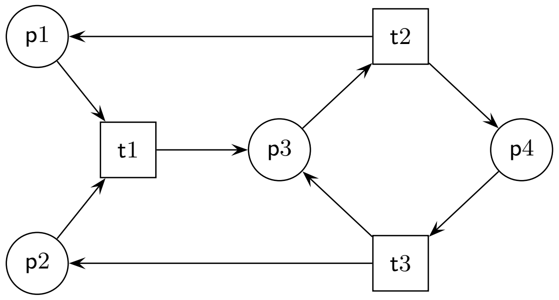

Fig. 1. A Petri net example: $\mathcal{P}$

2 AC-Tree Automata for Unification Problems

In this section we discuss the applications of equational tree automata, in particular AC-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.*

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 [3].

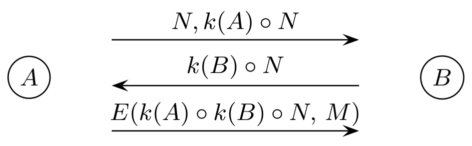

Fig. 2. Diffie-Hellman key exchange algorithm

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, [8]). 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 $k(B) \circ N$ 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.

Decidability results and closure properties for equational tree languages are summarized in Fig.3. In the figure, the check mark ✓ means “positive” and the cross $\times$ is “negative”. The question mark ? means “open”. If the same result holds in both regular and non-regular cases, it is represented by a single mark in a large column.

		C	A	AC
$\mathcal{L}(\mathring{A}/\mathcal{E}) = \varnothing$?	regular	✓	✓	✓
	non-regular		×	
$\mathcal{L}(\mathring{A}/\mathcal{E}) \subseteq \mathcal{L}(\mathcal{B}/\mathcal{E})$?	regular	✓	×	✓
	non-regular			?
$\mathcal{L}(\mathring{A}/\mathcal{E}) = \mathcal{T}(\mathcal{F})$?	regular	✓	×	✓
	non-regular			?
$\mathcal{L}(\mathring{A}/\mathcal{E}) \cap \mathcal{L}(\mathcal{B}/\mathcal{E}) = \varnothing$?	regular	✓	×	✓
	non-regular			

		C	A	AC
closed under $\cup$	regular	✓	✓	✓
	non-regular			
closed under $\cap$	regular	✓	×	✓
	non-regular		✓	
closed under $\overline{()}$	regular	✓	×	✓
	non-regular		✓	?

Fig. 3. Decidability results and closure properties

References

1. H. Comon, M. Dauchet, R. Gilleron, F. Jacquemard, D. Lugiez, S. Tison, and M. Tommasi: *Tree Automata Techniques and Applications*, 2002. Draft 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. E.W. Mayr: *An Algorithm for the General Petri Net Reachability Problem*, SIAM J. Comput. 13(3), pp. 441–460, 1984.
4. H. Ohsaki, H. Seki, and T. Takai: *Recognizing Boolean Closed A-Tree Languages with Membership Conditional Rewriting Mechanism*, Proc. of 14th RTA, 2003. To appear in LNCS.
5. H. Ohsaki and T. Takai: *Reachability and Closure Properties of Equational Tree Languages*, Proc. of 13th RTA, LNCS 2378, pp. 114-128, 2002.
6. H. Ohsaki and T. Takai: *A Tree Automata Theory for Unification Modulo Equational Rewriting*, Proc. of 16th UNIF, 2002.
7. H. Ohsaki: *Beyond Regularity: Equational Tree Automata for Associative and Commutative Theories*, Proc. of 15th CSL, LNCS 2142, pp. 539–553, 2001.
8. B. Schneier: *Applied Cryptography: Protocols, Algorithms, and Source Code in C*, Second Edition, John Wiley & Sons, 1996.