The Emptiness Problem for Tree Automata with at Least One Disequality Constraint is NP-hard

Pierre-Cyrille Héam, Vincent Hugot, Olga Kouchnarenko

To cite this version:

Pierre-Cyrille Héam, Vincent Hugot, Olga Kouchnarenko. The Emptiness Problem for Tree Automata with at Least One Disequality Constraint is NP-hard. [Research Report] FEMTO-ST. 2014. hal-01089711

HAL Id: hal-01089711
https://inria.hal.science/hal-01089711
Submitted on 2 Dec 2014

HAL is a multi-disciplinary open access archive for the deposit and dissemination of scientific research documents, whether they are published or not. The documents may come from teaching and research institutions in France or abroad, or from public or private research centers.

L’archive ouverte pluridisciplinaire HAL, est destinée au dépôt et à la diffusion de documents scientifiques de niveau recherche, publiés ou non, émanant des établissements d’enseignement et de recherche français ou étrangers, des laboratoires publics ou privés.
The Emptiness Problem for Tree Automata with at Least One Disequality Constraint is NP-hard

P.-C. Héam* V. Hugot† O. Kouchnarenko‡
December 2, 2014

Abstract
The model of tree automata with equality and disequality constraints was introduced in 2007 by Filiot, Talbot and Tison. In this paper we show that if there is at least one disequality constraint, the emptiness problem is NP-hard.

1 Introduction
Tree automata are a pervasive tool of contemporary Computer Science, with applications running the gamut from XML processing [13] to program verification [3, 14, 12]. Since their original introduction, they have spawned an ever-growing family of variants, each with its own characteristics of expressiveness and decision complexity. Among them is the family of tree automata with equality and disequality constraints, providing several means for comparing subtrees. Examples of such automata are the original class introduced in [7], their restriction to constraints between brothers [2], and visibly tree automata with memory and constraints [5]. In this paper we focus on a recently introduced variant: tree automata with global equality and disequality constraints [8, 9, 10]. For this class of automata, the universality problem is undecidable [10], while membership is NP-complete [10], and emptiness is decidable [1]. Several complexity results for subclasses were pointed out in the literature: the membership problem is polynomial for rigid tree automata [14] as well as for tree automata with a fixed number of equality constraints [12] and no disequality constraints. The emptiness problem is EXPTIME-complete if there are only equality constraints [10], in NEXPTIME if there are only irreflexive disequality constraints [10], and in 3-EXPTIME if there are only reflexive disequality constraints [6]. In this paper we show that the emptiness problem is NP-hard for tree automata with global equality and disequality constraints if there is at least one disequality constraint.

*FEMTO-ST - INRIA - CNRS - Université de Franche-Comté
†LIFL - INRIA
‡FEMTO-ST - INRIA - CNRS - Université de Franche-Comté
2 Formal Background

A ranked alphabet is a finite set $\mathcal{F}$ of symbols equipped with an arity function arity from $\mathcal{F}$ into $\mathbb{N}$. The set of terms on $\mathcal{F}$, denoted $\mathcal{T}(\mathcal{F})$, is inductively defined as the smallest set satisfying: for every $t \in \mathcal{F}$ such that $\text{arity}(t) = 0$, $t \in \mathcal{T}(\mathcal{F})$; if $t_1, \ldots, t_n$ are in $\mathcal{T}(\mathcal{F})$ and if $f \in \mathcal{F}$ has arity $n$, then $f(t_1, \ldots, t_n) \in \mathcal{T}(\mathcal{F})$. The set of positions of a term $t$, denoted $\text{Pos}(t)$, is the subset of $\mathbb{N}^*$ (finite words over $\mathbb{N}$) inductively defined by: if $\text{arity}(t) = 0$, then $\text{Pos}(t) = \{\varepsilon\}$; if $t = f(t_1, \ldots, t_n)$, where $n$ is the arity of $f$, then $\text{Pos}(t) = \{\varepsilon\} \cup \{i \cdot \alpha_i \mid \alpha_i \in \text{Pos}(t_i)\}$. A term $t$ induces a function (also denoted $t$) from $\text{Pos}(t)$ into $\mathcal{F}$, where $t(\alpha)$ is the symbol of $\mathcal{F}$ occurring in $t$ at the position $\alpha$. The subterm of a term $t$ at position $\alpha \in \text{Pos}(t)$ is the term $t_{\alpha}$ such that $\text{Pos}(t_{\alpha}) = \{\beta \mid \alpha \cdot \beta \in \text{Pos}(t)\}$ and for all $\beta \in \text{Pos}(t_{\alpha})$, $t_{\alpha}(\beta) = t(\alpha \cdot \beta)$. For any pair of terms $t$ and $t'$, any $\alpha \in \text{Pos}(t)$, the term $t[t']_{\alpha}$ is the term obtained by substituting in $t$ the subterm rooted at position $\alpha$ by $t'$. Let $\mathcal{X}$ be an infinite countable set of variables such that $\mathcal{X} \cap \mathcal{F} = \emptyset$. A context $C$ is term in $\mathcal{T}(\mathcal{F} \cup \mathcal{X})$ (variables are constants) where each variable occurs at most once; it is denoted $C[X_1, \ldots, X_n]$ if the occurring variables are $X_1, \ldots, X_n$. If $t_1, \ldots, t_n$ are in $\mathcal{T}(\mathcal{F})$, $C[t_1, \ldots, t_n]$ is the term obtained from $C$ by substituting each $X_i$ by $t_i$.

A tree automaton on a ranked alphabet $\mathcal{F}$ is a tuple $\mathcal{A} = (\mathcal{Q}, \Delta, F)$, where $\mathcal{Q}$ is a finite set of states, $F \subseteq \mathcal{Q}$ is the set of final sets and $\Delta$ is a finite set of rules of the form $f(q_1, \ldots, q_n) \rightarrow q$, where $f \in \mathcal{F}$ has arity $n$ and the $q_i$’s and $q$ are in $\mathcal{Q}$. A tree automaton $\mathcal{A} = (\mathcal{Q}, \Delta, F)$ induces a relation on $\mathcal{T}(\mathcal{F} \cup \mathcal{Q})$ (where elements of $\mathcal{Q}$ are constant, denoted $\rightarrow_{\mathcal{A}}$ or just $\rightarrow$, defined by $t \rightarrow_{\mathcal{A}} t'$ if there exists a transition $f(q_1, \ldots, q_n) \rightarrow q \in \Delta$ and $\alpha \in \text{Pos}(t)$ such that $t' = t[q]_{\alpha}$, $t(\alpha) = f$ and for every $1 \leq i \leq n$, $t(\alpha \cdot i) = q_i$. The reflexive transitive closure of $\rightarrow_{\mathcal{A}}$ is denoted $\rightarrow^*_{\mathcal{A}}$. A term $t \in \mathcal{T}(\mathcal{F})$ is accepted by $\mathcal{A}$ if there exists $q \in F$, such that $t \rightarrow^*_{\mathcal{A}} q$. An run $\rho$ for a term $t \in \mathcal{T}(\mathcal{F})$ in $\mathcal{A}$ is a function from $\text{Pos}(t)$ into $\mathcal{Q}$ such that if $\alpha \in \text{Pos}(t)$ and $t(\alpha)$ has arity $n$, then $t(\alpha)(\rho(\alpha \cdot 1), \ldots, \rho(\alpha \cdot n)) \rightarrow \rho(\alpha)$ is in $\Delta$. An accepting run is a run satisfying $\rho(\varepsilon) \in F$. It can be checked that a term $t$ is accepted by $\mathcal{A}$ iff there exists an accepting run $\rho$ for $t$ and, more generally, that $t \rightarrow^*_{\mathcal{A}} q$ if there exists a run $\rho$ for $t$ in $\mathcal{A}$ such that $\rho(\varepsilon) = q$. In this case we write $t \rightarrow^n_{\rho, \mathcal{A}} q$ or just $t \rightarrow^n q$ if $\mathcal{A}$ is clear from the context.

A tree automaton with global equality and disequality constraints (TAGED for short) is a tuple $(\mathcal{A}, R_1, R_2)$, where $\mathcal{A} = (\mathcal{Q}, \Delta, F)$ is a tree automaton and $R_1, R_2$ are binary relations over $\mathcal{Q}$. The relation $R_1$ is called the set of equality constraints and the relation $R_2$ the set of disequality constraints. A term $t$ is accepted by $(\mathcal{A}, R_1, R_2)$ if there exists a successful run $\rho$ for $t$ in $\mathcal{A}$ such that: if $(\rho(\alpha), \rho(\beta)) \in R_1$, then $t_{\alpha} = t_{\beta}$, and if $(\rho(\alpha), \rho(\beta)) \in R_2$, then $t_{\alpha} \neq t_{\beta}$. For a ranked alphabet $\mathcal{F}$, let TAGED$(k', k)$ denote the class $(\mathcal{A}, R_1, R_2)$ of TAGED, where $\mathcal{A}$ is a tree automaton over $\mathcal{F}$, $|R_1| \leq k'$ and $|R_2| \leq k$. 

2
3 TAGED and the Hamiltonian Path Problem

The paper focuses on proving the following theorem.

**Theorem 1** The emptiness problem for TAGED(0,1) is NP-hard.

The proof of Theorem 1 is a reduction to the Hamiltonian Path Problem defined below.

**Hamiltonian Graph Problem**

**Input:** a directed finite graph $G = (V,E)$;

**Output:** 1 if there exists a path in $G$ visiting each element of $V$ exactly once, 0 otherwise.

The Hamiltonian Graph Problem is known to be NP-complete [11]. A path in a directed graph visiting each vertex exactly once is called a Hamiltonian path. Before proving Theorem 1, let us mention the following direct important consequence, which is the main result of the paper.

**Corollary 2** For every fixed $k \geq 1$, and every fixed $k' \geq 0$, the emptiness problem for TAGED($k',k$) is NP-hard.

We have divided the proof of Theorem 1 into a sequence of lemmas. Lemma 3, below, is immediately obtained by a cardinality argument.

**Lemma 3** In a directed graph $G$ with $n$ vertices, there exists a Hamiltonian path iff there is a path of length $n-1$ that does not visit the same vertex twice.

For any directed graph $G = (V,E)$, let $m_G$ denote the number of paths of length $|V| - 1$ in $G$.

**Lemma 4** Let $G = (V,E)$ be a directed graph. One can compute $m_G$ in polynomial time in the size of $G$.

**Proof.** Let us denote by $m_{G,k,u,v}$, for any $k \geq 1$, any $u \in V$ and any $v \in V$, the number of paths of length $k$ from $u$ to $v$ in $G$. One has $m_{G,k+1,u,v} = \sum_{(u',v') \in E} m_{G,k,u',v'}$. Therefore, every $m_{G,k,u,v}$, for $k \leq |V|$, can be computed recursively in polynomial time in $|V|$. Note that $m_G = \sum_{u,v \in V} m_{G,|V|,u,v}$, concluding the proof.

Let $\mathcal{F}_1 = \{f, g, A\}$, where $f$ has arity 2 and $g$ arity 3 and $A$ is a constant. The next construction aims to build in polynomial time a tree automaton accepting a unique term having exactly $m$ leaves.

**Construction 5** Let $m$ be a strictly positive integer and set $\alpha_1 \ldots \alpha_k$ the binary representation of $m$ ($\alpha_1 = 1$ and $\alpha_i \in \{0,1\}$). Let $A_m = (Q_1, \Delta_1, F_1)$ be the tree automaton over $\mathcal{F}_1$, where $Q_1 = \{q_i \mid 0 \leq i \leq k\}$, $F_1 = \{q_k\}$ and $\Delta_1 = \{A \rightarrow q_1\} \cup \{f(q_i, q_k) \rightarrow q_{i+1} \mid 1 \leq i \leq k-1 \text{ and } \alpha_i+1 = 0\} \cup \{g(q_i, q_i, q_1) \rightarrow q_{i+1} \mid 1 \leq i \leq k-1 \text{ and } \alpha_i+1 = 1\}$.
Lemma 6 The tree automaton $A_m$ can be computed in polynomial time in $k$. Moreover, $L(A_m)$ is reduced to a single term having exactly $m$ leaves, all labelled by $A$.

Proof. The proof is by induction on $k$. If $k = 1$, then $m = 1 = \alpha_1$ (since $m \neq 0$). In this case $Q_1 = F_1 = \{q_1\}$ and $\Delta_1 = \{A \to q_1\}$; therefore $L(A_1) = \{A\}$ and the lemma result holds.

Now assume that the lemma is true for a fixed $k \geq 1$. Let $2^{k+1} \leq m < 2^{k+2}$ and set $m = \alpha_1 \ldots \alpha_k \alpha_{k+1}$, the binary representation of $m$. Two cases may arise:

- $\alpha_{k+1} = 0$: In this case, by construction, the terms accepted by $A_m$ are exactly the terms of the form $f(t_1, t_2)$, with $t_1 \to_A^* q_{k-1}$ and $t_2 \to_A^* q_k$. They correspond to the terms $f(t_1, t_2)$, with $t_1, t_2 \in L(A_{\frac{m}{2}})$. By induction hypothesis, $L(A_{\frac{m}{2}})$ is a singleton containing a unique term with $\frac{m}{2}$ leaves, all labelled by $A$. It follows that $L(A_m)$ accepts a unique term with $2 \cdot \frac{m}{2} = m$ leaves, all labelled by $A$.

- $\alpha_{k+1} = 1$: Similarly, the terms accepted by $A_m$ are exactly the terms of the form $g(t_1, t_2, A)$, with $t_1, t_2 \in L(A_{\frac{m-1}{2}})$. By induction, it follows that $L(A_m)$ accepts a unique term with $1 + 2 \cdot \frac{m-1}{2} = m$ leaves, all labelled by $A$.

Therefore, the lemma result holds also for $k + 1$, which concludes the proof. $\square$

Note that since $m_G \leq |V|^{|V| - 1}$ the binary encoding of $m_G$ is of the size polynomial in $|V|$. By Lemma 4, $m_G$ can be computed in polynomial time and $k$ is polynomial in $|V|$. Therefore, the construction of $A_{m_G}$ can be done in polynomial time in $|V|$, proving the following lemma.

Lemma 7 Let $G$ be a directed graph satisfying $m_G \neq 0$. The tree automaton $A_{m_G}$ can be computed in polynomial time.

The next construction is dedicated to a tree automaton $P_G$ accepting terms encoding paths of length $|V| - 1$.

Construction 8 Let $G = (V, E)$ be a non-empty directed graph and let $n = |V| - 1$. Let $F_1 = \{h\} \cup \{A_v \mid v \in V\}$, where $h$ is of arity 2 and the $A_v$'s are constants. Let $P_G = (Q_2, \Delta_2, F_2)$ be the tree automaton over $F_2$, where $Q_2 = \{q_w \mid 0 \leq i \leq n - 1, \ w \in V\}$, $F_2 = \{q_w^{n-1} \mid w \in V\}$, and

\[ \Delta_2 = \{A_v \to q_w^i \mid w \in V\} \cup \{h(q_v^i, q_w^i) \to q_v^{i+1} \mid 1 \leq i \leq n - 2, \ (w, v) \in E\} \]

Note that the construction of $P_G$ can be done in polynomial time. For a given graph $G = (V, E)$ and a given finite set $Q$, an $h$-term on $Q$ is a term either of the form $\beta_0$ or $h(\beta_k, h(\beta_{k-1}, h(\ldots, h(\beta_1, \beta_0) \ldots)))$, where $\beta_i \in \{A_v \mid v \in V\} \cup Q$. Such an $h$-term is denoted $[\beta_k \beta_{k-1} \ldots \beta_0]_Q$. If $Q$ is clear from the context, the index $Q$ is omitted.
Lemma 9 Let $G = (V, E)$ be a non empty directed graph. A term $t$ is accepted by $\mathcal{P}_G$ iff there exists a path $(w_0, w_1)(w_1, w_2) \ldots (w_{n-1}, w_n)$ in $G$ such that $t = [A_{w_{n-1}}A_{w_{n-2}} \ldots A_{w_1}A_{w_0}]_{Q_3}$.

Proof. If $t$ is accepted by $\mathcal{P}_G$, then there exists $w_{n-1} \in V$ such that $t \rightarrow^* q_{w_{n-1}}^{n-1}$. Looking right-hand sides of the transitions, it follows that there exists $w_{n-2} \in V$ such that $t \rightarrow^* h(q_0^{w_{n-2}}, q_{w_{n-2}}^{n-2}) \rightarrow q_{w_{n-1}}^{n-1}$. The unique rule with right-hand side $q_0^{w_{n-2}}$ is $A_{w_{n-1}} \rightarrow q_{w_{n-1}}^{n-1}$. Therefore $t$ is of the form $t = h(A_{w_{n-1}}, t')$ with $t' \rightarrow^* q_{w_1}^{n-2}$ and $(w_{n-2}, w_{n-1}) \in E$. By a direct induction on $n$, one has $t = [A_{w_{n-1}}A_{w_{n-2}} \ldots A_{w_1}A_{w_0}]$, where $(w_0, w_1)(w_1, w_2) \ldots (w_{n-2}, w_{n-1})$ is a path in $G$.

Conversely, assume that $t = [A_{w_{n-1}}A_{w_{n-2}} \ldots A_{w_1}A_{w_0}]$ and that the sequence $(w_0, w_1)(w_1, w_2) \ldots (w_{n-2}, w_{n-1})$ is a path in $G$. For each $1 \leq i \leq n - 1$, let $t_i = [A_{w_{i+1}}A_{w_{i+2}} \ldots A_{w_i}A_{w_0}]$. One has $t_1 = h(A_{w_1}, A_{w_0})$, with $(w_0, w_1) \in E$. Therefore $t_1 \rightarrow^* q_{w_1}^{0}$. By a direct induction, one has $t_i \rightarrow^* q_{w_i}^{i-1}$. Consequently $t_{n-1} \rightarrow^* q_{w_{n-1}}^{n-1}$. It follows that $t_{n-1}$ is accepted by $\mathcal{P}_G$. It suffices to note that $t_{n-1} = t$ to conclude the proof. □

The next construction designs a tree automaton $\mathcal{C}_G$ accepting terms of the form $[A_{w_k}A_{w_{k-1}} \ldots A_{w_1}A_{w_0}]$, where $k \geq 1$ and there exist $j \neq i$ such that $w_i = w_j$.

Construction 10 Let $G = (V, E)$ be a non empty directed graph. Let $\mathcal{F}_2 = \{h\} \cup \{A_v | v \in V\}$, where $h$ has arity 2 and the $A_v$’s are constants. Without loss of generality we assume that $0, 1, f \notin V$. Let $\mathcal{C}_G = (Q_3, \Delta_3, \mathcal{F}_2)$ be the tree automaton over $\mathcal{F}_2$, where $Q_3 = \{p_w, p_w' | w \in V\} \cup \{p_0, p_1, p_f\}$, $F_3 = \{p_f\}$, and

$\Delta_3 = \{A_w \rightarrow p_0, A_w \rightarrow p_w, A_w \rightarrow p_w' | w \in V\}$
$\cup \{h(p_0, p_0) \rightarrow p_1, h(p_0, p_1) \rightarrow p_1, h(p_0, p_0) \rightarrow p_w'\}$
$\cup \{h(p_w, p_w') \rightarrow p_f, h(p_0, p_0) \rightarrow p_w', h(p_0, p_f) \rightarrow p_f, h(p_w, p_1) \rightarrow p_w'\}$.

Lemma 11 Let $G = (V, E)$ be a non empty directed graph. For any term $t$, one has $t \rightarrow^* p_1$ iff $t = [A_{w_k}A_{w_{k-1}} \ldots A_{w_1}A_{w_0}]_{Q_3}$ with $k \geq 1$.

Proof. If $t = h(A_{w_k}, h(A_{w_{k-1}}, h(\ldots, h(A_{w_1}, A_{w_0}) \ldots)))$, then by a direct induction on $k$, and using the transitions $A_w \rightarrow p_0$ and $h(p_0, p_1) \rightarrow p_1$, one has $t \rightarrow^* p_1$.

Now, if $t \rightarrow^* p_1$, then the last transition used to reduce $t$ is $h(p_0, p_1) \rightarrow p_1$. Therefore there exists $w \in V$ such that $t = h(A_w, t')$ with $t' \rightarrow^* p_1$. By a direct induction on the depth of $t$, one can conclude the proof. □

Lemma 12 Let $G = (V, E)$ be a non empty directed graph. For any term $t$, one has $t \rightarrow^* p_w'$ iff $t$ is of the form $t = [A_{w_k}A_{w_{k-1}} \ldots A_{w_1}A_{w_0}]_{Q_3}$, where $k \geq 1$ and at least one of the $w_i$ is equal to $w$. 5
Assume first that there exists $j$ such that $w_i = w$. Assume now that the depth of $t$ is $n$. The four transitions having $p_w'$ as right-hand side are $A_w \rightarrow p_w'$, $h(p_w, p_0) \rightarrow p_w'$, $h(p_0, p_w') \rightarrow p_w'$ and $h(p_w, p_1) \rightarrow p_w'$. If the last transition used to reduce $t$ is $A_w \rightarrow p_w'$, then $t = A_w$; $t$ is of the expected form. If the last transition used to reduce $t$ is $h(p_w, p_1) \rightarrow p_w'$, then $t = h(A_w, t')$. Using Lemma 11, $t$ is of the expected form. If the last transition used to reduce $t$ is $h(p_w, p_0) \rightarrow p_w'$, then there exists $A_w$ such that $t = h(A_w, A_w')$; $t$ is of the expected form. If the last transition used to reduce $t$ is $h(p_0, p_w') \rightarrow p_w'$, then there exists $w'$ and $t'$ such that $t = h(A_w, t')$ and $t' \rightarrow^* p_w'$. By induction hypothesis on $t'$, $t$ is of the expected form, concluding the induction and proving the lemma.

**Lemma 13** Let $G = (V, E)$ be a non empty directed graph. A term $t$ is accepted by $C_G$ iff it is of the form $t = [A_{w_1} A_{w_2} \ldots A_{w_k} A_{w_0}] Q_3$, where $k \geq 1$ and there exist $j \neq i$ such that $w_i = w_j$.

**Proof.** Assume first that $t = [A_{w_1} A_{w_2} \ldots A_{w_k} A_{w_0}] Q_3$, with $k \geq 2$ and there exist $j \neq i$ such that $w_i = w_j$. If $j \geq 2$, one has

$$
A_{w_1} A_{w_2} \ldots A_{w_j} A_{w_0} \rightarrow^* [A_{w_1} A_{w_2} \ldots A_{w_j} A_{w_0} (A_{w_1} A_{w_2} \ldots A_{w_j} A_{w_0}')] Q_3 + [A_{w_1} A_{w_2} \ldots A_{w_j} A_{w_0} A_{w_j+1} A_{w_0}] Q_3.$$

If $j = 1$, then $t \rightarrow [A_{w_1} A_{w_2} \ldots A_{w_1} A_{w_0}] = [A_{w_1} A_{w_2} \ldots A_{w_j+1} A_{w_0}]$. If $j = 0$, then $t \rightarrow [A_{w_1} A_{w_2} \ldots A_{w_0} A_{w_0}] = [A_{w_1} A_{w_2} \ldots A_{w_j+1} A_{w_0}]$. In every case one has $t \rightarrow^* [A_{w_1} A_{w_2} \ldots A_{w_j+1}] Q_3$. Moreover $[A_{w_1} A_{w_2} \ldots A_{w_j+1}] Q_3 \rightarrow [A_{w_1} A_{w_2} \ldots A_{w_j+1} A_{w_j}]$. Since $w_i = w_j$, $[A_{w_1} A_{w_2} \ldots A_{w_i} A_{w_j}] \rightarrow [A_{w_1} A_{w_2} \ldots A_{w_i} A_{w_j}]$. It follows that $t$ is accepted by $C_G$.

Conversely, assume now that $t \in L(C_G)$. We prove by induction on the depth of $t$ that it is of the form $t = [A_{w_1} A_{w_2} \ldots A_{w_i} A_{w_0}]$, with $k \geq 2$ and such that there exists $j \neq i$ satisfying $w_i = w_j$.

No constant is accepted by $C_G$. If $t \in L(C_G)$ has depth 2, then $t \rightarrow^* p_f$. The last transition used to reduce $t$ cannot be $h(p_0, p_f) \rightarrow p_f$; otherwise $t$ would have a depth strictly greater than 2. It follows that there exists $w$ such that $t \rightarrow h(p_w, p_w')$. Consequently, $t \rightarrow h(A_w, p_w')$ since the unique transition having $p_w$ as right hand side is $A_w \rightarrow p_w$. Now, since $t$ has depth 2, the unique possibility is that $t = f(A_w, A_w)$. The property is therefore true for term of depth 2. Now let $t$ be a term of depth $k - 1$ belonging to $L(C_G)$. There exists
a successful run $\rho$ such that $t \rightarrow^*_\rho p_f$. Therefore, either $t \rightarrow^*_\rho h(p_0, p_f)$ or there exists $w_k$ such that $t \rightarrow^*_\rho h(p_{w_k}, p'_{w_k})$.

- If $t \rightarrow^*_\rho h(p_0, p_f)$, then there exists $w_k$ such that $t = h(A_{w_k}, t')$, with $t' \in L(C_G)$, and $t'$ has depth $k - 1$. By induction on the depth, $t$ has the wanted form.

- If $t \rightarrow^*_\rho h(p_{w_k}, p'_{w_k})$, then $t = h(A_{w_k}, t')$ and $t' \rightarrow^*_w w'_k$. Using Lemma 12, $t' = [A_{w_{k-1}} A_{w_{k-2}} \ldots A_{w_0} A_{w_0}]$, where at least one of the $w_i (i \leq k - 1)$ is equal to $w_k$, proving the induction and concluding the proof.

\[ \square \]

**Lemma 14** Given a directed non-empty graph $G = (V, E)$, one can compute in time polynomial in the size of $G$ a tree automaton $B_G$ with a unique final state, accepting exactly the set of terms of the form $t = [A_{w_{|V|-2}} A_{w_{|V|-3}} \ldots A_{w_1} A_{w_0}] \emptyset$, such that $(w_0, w_1) \ldots (w_{|V|-2}, w_{|V|-1})$ is a non Hamiltonian path of $G$.

**Proof.** The automata $C_G$ – checking that a vertex is visited twice – and $P_G$ – checking the length of the path – can both be computed in polynomial time. Therefore, using the classical product construction, one can compute a tree automaton accepting $L(C_G) \cap L(P_G)$ in polynomial time. Transforming this automaton into an automaton with a unique final state can also be done in polynomial time using classical $\varepsilon$-transition removal, proving the lemma. The obtained automaton is $B_G$.

\[ \square \]

We can now give the last construction to prove the main result.

**Construction 15** Set $B_G = (Q, \Delta, \{q_f\})$. Without loss of generality, one can assume that $q_f = q_1$ and that $Q \cap Q_1 = \{q_1\}$. We consider the automaton $D_G = (Q_4, \Delta_4, F_4)$ over $F_1 \cup F_2$ defined by: $Q_4 = Q \cup Q_1$, $F_4 = \{q_k\}$ and $\Delta_4 = (\Delta \cup \Delta_1) \setminus \{A \rightarrow q_1\}$.

**Lemma 16** The TAGED $(D_G, \emptyset, \{q_1, q_1\})$ can be constructed in polynomial time in the size of $G$. Moreover, it accepts the empty language iff there exists a Hamiltonian path in $G$.

**Proof.** Using Lemma 6, the term accepted by $D_G$ are those of the form $C[t_1, \ldots, t_{m_G}]$, where $C[A, \ldots, A]$ is the unique term accepted by $A_{m_G}$ and each $t_i$ is accepted by $B_G$. With the inequality constraint, $(D_G, \emptyset, \{q_1, q_1\})$ accepts an empty language iff $|L(B_G)| < m_G$. By Lemma 14, $|L(B_G)|$ is the number of non-Hamiltonian paths in $G$. Since $m_G$ is the number of paths of length $|V| - 1$ in $G$, using Lemma 3, $L((D_G, \emptyset, (q_1, q_1))) = \emptyset$ iff there exists a Hamiltonian path of length $|V| - 1$ in $G$.

\[ \square \]

Theorem 1 is a direct consequence of Lemma 16 and of the polynomial time construction of $D_G$. 

7
4 Conclusion

In this paper we have proved that the emptiness problem for TAGED is NP-hard if there is at least one negative constraint. It is known that the emptiness problem for TAGED with only irreflexive disequality constraints is in NEXP-TIME [10], and that it is NP-hard – by reduction of emptiness for DAG automata [4]. If there are only reflexive disequality constraints, emptiness is known to be solvable in 3-EXPTIME [6]. The gap between these bounds is large and deserves to be refined.

References


