Synthesis of Deterministic Top-down Tree Transducers from Automatic Tree Relations

Christof Löding, Sarah Winter

Lehrstuhl Informatik 7, RWTH Aachen, 52056 Aachen, Germany

Abstract

We consider the synthesis of deterministic tree transducers from automaton definable specifications, given as binary relations, over finite trees. We consider the case of tree automatic specifications, meaning the specification is recognizable by a top-down tree automaton that reads the two given trees synchronously in parallel. In this setting we study tree transducers that are allowed to have either bounded delay or arbitrary delay. Delay is caused whenever the transducer reads a symbol from the input tree but does not produce output. For specifications that are deterministic top-down tree automatic, we provide decision procedures for both bounded and arbitrary delay that yield deterministic top-down tree transducers which realize the specification for valid input trees. For general nondeterministic tree-automatic specifications we solve two restricted cases, namely for unions of deterministic specifications and uniformizers without delay, as well as for all nondeterministic specifications where the uniformizers have bounded delay and do not swap subtrees. Similar to the case of relations over words, we use two-player games as the man technique to obtain our results.

1. Introduction

The synthesis problem asks, given a specification that relates possible inputs to allowed outputs, whether there is a program realizing the specification, and if so, construct one. This problem setting originates from Church's synthesis problem [1] which was already posed in 1957. Church considers the case where the input is an infinite bit sequence that has to be transformed, bit by bit, into an infinite bit sequence. The synthesis problem is then to decide whether there is a circuit which realizes the given input/output specification, and construct one if possible. A related notion is the one of uniformization of a (binary) relation, which is a function that selects for each element of the domain of the relation an element in its image. The synthesis problem asks for effective uniformization by functions that can be implemented in a specific way.

Specifications are usually written in some logical formalism, while the uniformization, in particular in the synthesis setting, is required to be implemented by some kind of device. Since many logics can be translated into automata, which can also serve as implementations of a uniformization, it is natural to study uniformization problems in automata theory. Relations (or specifications) can be defined using automata with two input tapes, and uniformizations can be realized by transducers, that is, automata with output.

A first uniformization result in such a setting has been obtained by Büchi and Landweber in [2], who showed that for specifications over infinite words in monadic second-order logic, it is decidable whether they have a uniformization by a synchronous transducer (that outputs one symbol for each input letter). The specifications considered in [2] can be translated into finite automata that read the two input words synchronously. Such relations are referred to as automatic relations over finite words, and as ω-automatic relations over infinite words.

The result of Büchi and Landweber has been extended to transducers with delay, that is, transducers that have the possibility to produce empty output in some transitions. For a bounded delay decidability was shown in [3], and for an unbounded delay in [4]. In the case of finite words, it was shown in [5] that it is decidable whether an automatic relation has a uniformization by a deterministic subsequential transducer, that is, a transducer that can output finite words on each transition.

Our aim is to study these uniformization questions for relations over trees. Tree automata are used in many fields, for example as tool for analyzing and manipulating rewrite systems or XML Schema languages (see [6]). Tree transformations that are realized by finite tree transducers thus become interesting in the setting of translations from one document scheme into another [7]. There are already some uniformization results for tree relations. For example, in [8] it is shown that each relation that can be defined by a nondeterministic top-down tree transducer, has a uniformization by a deterministic top-down tree transducer with regular lookahead. However, these results focus on the existence of a uniformization for each relation that can be specified by the considered model. In contrast to that, we are interested in the corresponding decision problem. More precisely, for a class $\mathcal{C}$ of tree relations and a class $\mathcal{F}$ of functions over trees, we are interested in a procedure that decides whether a given relation from $\mathcal{C}$ has a uniformization in $\mathcal{F}$.

In this paper, we start the investigation of such questions in the rich landscape of tree automaton and tree transducer models. We study uniformization of automatic tree relations over finite trees by deterministic top-down tree transducers. For automatic tree relations that can be defined by deterministic top-down automata (that is, an automaton deterministically reads the two input trees synchronously in top-down fashion), we show that it is decidable whether a given relation has a uniformization by a top-down tree transducer, and if possible construct one. For this result, we require that the transducer for the uniformization works correctly on all input trees that are in the domain of the specification. For trees outside the domain, the transducer can produce an arbitrary output. Since the transducer does not have to validate that the input is in the domain of the specification, we refer to this setting as uniformization without input validation. We also briefly comment on the problems in the case of uniformization with input validation, and solve the problem for the restricted class of uniformizers without delay (synchronously producing one output symbol for each input symbol). These decidability results are obtained by constructing strategies in two-player games of infinite duration.

For nondeterministic top-down tree automatic specifications we solve two restricted cases: For unions of deterministic specifications with disjoint domains we show that it is decidable whether there is a uniformizer without delay. For the proof, we extend the game-based methods that are used for deterministic specifications, and use automata on infinite trees (which are tightly related to infinite games [9]) as a tool. For general nondeterministic specifications, we restrict the class of uniformizers to transducers with bounded delay in which the position of a reading head an its associated position in the output tree are always on the same path. In this case we can use guidable automata (see [10]) to apply the game-based method from the deterministic case.

The paper is structured as follows. First, we fix some basic definitions and terminology. Then, in Section 3.1 and Section 3.2, we consider uniformization of deterministic top-down automatic tree relations by top-down tree transducers without input validation that have bounded delay and unbounded delay, respectively. In Section 3.3, we briefly consider the case of uniformization with input validation. Subsequently, in Section 4.1 and Section 4.2, we consider uniformization of non-deterministic automatic tree relations by top-down tree transducers.

A preliminary version of this work has appeared in [11].

2. Preliminaries

The set of natural numbers containing zero is denoted by $\mathbb{N}$. For a set S, the powerset of S is denoted by 2^S. An *alphabet* Σ is a finite non-empty set of letters. A finite *word* is a finite sequence of letters. The set of all finite words over Σ is denoted by Σ^*. The length of a word $w \in \Sigma^*$ is denoted by $|w|$, the empty word is denoted by ε. For $w = a_1 \dots a_n \in \Sigma^*$ for some $n \in \mathbb{N}$ and $a_1, \dots, a_n \in \Sigma$, let $w[i]$ denote the ith letter of w, i.e., $w[i] = a_i$. Furthermore, let $w[i,j]$ denote the infix from the ith to the jth letter of w, i.e., $w[i,j] = a_i \dots a_j$. We write $u \sqsubseteq w$ if $w = uv$ for $u, v \in \Sigma^*$. A subset $L \subseteq \Sigma^*$ is called *language* over Σ.

A *ranked alphabet* Σ is an alphabet where each letter $f \in \Sigma$ has a finite set of arities $rk(f) \subseteq \mathbb{N}$. The set of letters of arity i is denoted by Σ_i. A tree domain *dom* is a non-empty finite subset of $(\mathbb{N} \setminus \{0\})^*$ such that dom is prefix-closed and for each $u \in (\mathbb{N} \setminus \{0\})^*$ and $i \in \mathbb{N} \setminus \{0\}$ if $ui \in dom$, then $uj \in dom$ for all $1 \leq j < i$. We speak of ui as successor of u for each $u \in dom$ and $i \in \mathbb{N} \setminus \{0\}$.

A (finite Σ-labeled) *tree* is a pair $t = (dom_t, val_t)$ with a mapping $val_t : dom_t \to \Sigma$ such that for each node $u \in dom_t$ the number of successors of u corresponds to a rank of $val_t(u)$. The height h of a tree t is the length of its longest path, i.e., $h(t) = max\{|u| \mid u \in dom_t\}$. The set of all Σ-labeled trees is denoted by T_Σ. A subset $T \subseteq T_\Sigma$ is called *tree language* over Σ.

A *subtree* $t|_u$ of a tree t at node u is defined by $dom_{t|_u} = \{v \in \mathbb{N}^* \mid uv \in dom_t\}$ and $val_{t|_u}(v) = val_t(uv)$ for all $v \in dom_{t|_u}$. In order to formalize concatenation of trees, we introduce the notion of special trees. A *special tree* over Σ is a tree over $\Sigma \,\dot\cup \{\circ\}$ such that $\circ$ occurs exactly once at a leaf. Given $t \in T_\Sigma$ and $u \in dom_t$, we write $t[\circ/u]$ for the special tree that is obtained by deleting the subtree at u and replacing it by $\circ$. Let S_Σ be the set of special trees over Σ. For $t \in T_\Sigma$ or $t \in S_\Sigma$ and $s \in S_\Sigma$ let the *concatenation* $t \cdot s$ be the tree that is obtained from t by replacing $\circ$ with s.

Let X_n be a set of n variables $\{x_1, \ldots, x_n\}$ and Σ be a ranked alphabet. We denote by $T_\Sigma(X_n)$ the set of all trees over Σ which additionally can have variables from X_n at their leaves. Let $X = \bigcup_{n>0} X_n$. A tree from $T_\Sigma(X)$ is called *linar* if each variable occurs at most once. For $t \in T_\Sigma(X_n)$ let $t[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n]$ be the tree that is obtained by substituting each occurrence of $x_i \in X_n$ by $t_i \in T_\Sigma(X)$ for every $1 \leq i \leq n$.

A tree from $T_\Sigma(X_n)$ such that all variables from X_n occur exactly once and in the order $x_1, \ldots, x_n$ when reading the leaf nodes from left to right, is called *n-context* over Σ. A special tree can be seen as an 1-context. If C is an n-context and $t_1, \ldots, t_n \in T_\Sigma(X)$ we write $C[t_1, \ldots, t_n]$ instead of $C[x_1 \leftarrow t_1, \ldots, x_n \leftarrow t_n]$.

Tree automata. Tree automata can be viewed as a straightforward generalization of finite automata on finite words, when words are interpreted as trees over unary symbols. For a detailed introduction to tree automata see e.g. [12] or [6].

Let $\Sigma = \bigcup_{i=1}^m \Sigma_i$ be a ranked alphabet. A *non-deterministic top-down tree automaton* (an N↓TA) over Σ is of the form $\mathcal{A} = (Q, \Sigma, Q_0, \Delta)$ consisting of a finite set of states Q, a set $Q_0 \subseteq Q$ of initial states, and $\Delta \subseteq \bigcup_{i=0}^m (Q \times \Sigma_i \times Q^i)$ is the transition relation. For $i = 0$, we identify $Q \times \Sigma_i \times Q^i$ with $Q \times \Sigma_0$.

Let t be a tree and $\mathcal{A}$ be an N↓TA, a *run* of $\mathcal{A}$ on t is a mapping $\rho : dom_t \to Q$ compatible with Δ, i.e., $\rho(\varepsilon) \in Q_0$ and for each node $u \in dom_t$, if $val_t(u) \in \Sigma_i$ with $i \geq 0$, then $(\rho(u), val_t(u), \rho(u1), \ldots, \rho(ui)) \in \Delta$. A tree $t \in T_\Sigma$ is *accepted* if, and only if, there is a run of $\mathcal{A}$ on t. The tree language *recognized* by $\mathcal{A}$ is $T(\mathcal{A}) = \{t \in T_\Sigma \mid \mathcal{A} \text{ accepts } t\}$.

A tree language $T \subseteq T_\Sigma$ is called *regular* if T is recognizable by a non-deterministic top-down tree automaton. As the class of regular word languages, the class of regular tree languages is closed under Boolean operations.

A top-down tree automaton $\mathcal{A} = (Q, \Sigma, Q_0, \Delta)$ is *deterministic* (a D↓TA) if the set Q_0 is a singleton set and for each $f \in \Sigma_i$ and each $q \in Q$ there is at most one transition $(q, f, q_1, \ldots, q_i) \in \Delta$. However, non-deterministic and deterministic top-down automata are not equally expressive.

An extension to regular tree languages are (binary) *tree-automatic relations*. A way for a tree automaton to read a tuple of finite trees is to use a ranked vector alphabet. Thereby, all trees are read in parallel, processing one node from each tree in a computation step. Hence, the trees are required to have the same domain. Therefore we use a padding symbol to extend the trees if necessary. Formally, this is done in the following way.

Let Σ, Γ be ranked alphabets and let $\Sigma_\perp = \Sigma \,\dot\cup \{\perp\}$, $\Gamma_\perp = \Gamma \,\dot\cup \{\perp\}$. The *convolution* of (t_1, t_2) with $t_1 \in T_\Sigma$, $t_2 \in T_\Gamma$ is the $\Sigma_\perp \times \Gamma_\perp$-labeled tree $t = t_1 \otimes t_2$ defined by $dom_t = dom_{t_1} \cup dom_{t_2}$, and $val_t(u) = (val^\perp_{t_1}(u), val^\perp_{t_2}(u))$ with $rk(val_t(u)) = max\left\{rk(val^\perp_{t_1}(u)), rk(val^\perp_{t_2}(u))\right\}$ for all $u \in dom_t$, where $val^\perp_{t_i}(u) = val_{t_i}(u)$ if $u \in dom_{t_i}$ and $val^\perp_{t_i}(u) = \perp$ otherwise for $i \in \{1, 2\}$. As a special case, given $t \in T_\Sigma$, we define $t \otimes \perp$ to be the tree with $dom_{t\otimes\perp} = dom_t$ and $val_{t\otimes\perp}(u) = (val_t(u), \perp)$ for all $u \in dom_t$. Analogously, we define $\perp \otimes t$. We define the *convolution of a tree relation* $R \subseteq T_\Sigma \times T_\Gamma$ to be the tree language $T_R := \{t_1 \otimes t_2 \mid (t_1, t_2) \in R\}$.

We call a (binary) relation R *tree-automatic* if there exists a regular tree language T such that $T = T_R$. For ease of presentation, we say a tree automaton $\mathcal{A}$ recognizes R if it recognizes the convolution T_R and denote by $R(\mathcal{A})$ the induced relation R.

A *uniformization* of a relation $R \subseteq X \times Y$ is a function $f_R : X \to Y$ such that for each domain element x the pair $(x, f_R(x))$ is in the relation, i.e., $(x, f_R(x)) \in R$ for all $x \in dom(R)$. In the following, we are interested for a given tree-automatic relation, whether there exists a uniformization which can be realized by a tree transducer.

Tree Transducers. Tree transducers are a generalization of word transducers. As top-down tree automata, a top-down tree transducer reads the tree from the root to the leafs, but can additionally in each computation step produce finite output trees which are attached to the already produced output. For an introduction to tree transducers the reader is referred to [6].

A *top-down tree transducer* (a TDT) is of the form $\mathcal{T} = (Q, \Sigma, \Gamma, q_0, \Delta)$ consisting of a finite set of states Q, a finite input alphabet Σ, a finite output alphabet Γ, an initial state $q_0 \in Q$, and Δ is a finite set of transition rules of the form

$$q(f(x_1, \dots, x_i)) \to w[q_1(x_{j_1}), \dots, q_n(x_{j_n})],$$

where $f \in \Sigma_i$, w is an n-context over Γ, $q, q_1, \dots, q_n \in Q$ and $j_1, \dots, j_n \in \{1, \dots, i\}$, or

$$q(x_1) \to w[q_1(x_1), \dots, q_n(x_1)] \quad (\varepsilon\text{-transition}),$$

where $f \in \Sigma_i$, w is an n-context over Γ, and $q, q_1, \dots, q_n \in Q$. A top-down tree transducer is *deterministic* (a DTDT) if it contains no ε-transitions and there are no two rules with the same left-hand side. A top-down tree transducer is *linear* if all the trees in the transitions are linar.

A *configuration* of a top-down tree transducer is a triple $c = (t, t', \varphi)$ of an input tree $t \in T_\Sigma$, an output tree $t' \in T_{\Gamma \cup Q}$ and a function $\varphi : D_{t'} \to dom_t$, where

- $val_{t'}(u) \in \Gamma_i$ for each $u \in dom_{t'}$ with $i > 0$ successors
- $val_{t'}(u) \in \Gamma_0$ or $val_{t'}(u) \in Q$ for each leaf $u \in dom_{t'}$
- $D_{t'} \subseteq dom_{t'}$ with $D_{t'} = \{u \in dom_{t'} \mid val_{t'}(u) \in Q\}$
  (φ maps every node from the output tree t' that has a state-label to a node of the input tree t)

Let $c_1 = (t, t_1, \varphi_1), c_2 = (t, t_2, \varphi_2)$ be configurations of a top-down tree transducer. We define a successor relation $\to_\mathcal{T}$ on configurations as usual by applying one rule, which looks formally (for the application of a non-ε-rule) as follows:

$$c_1 \to_\mathcal{T} c_2 :\Leftrightarrow \begin{cases} \exists u \in \mathrm{dom}_{t_1}, q \in Q, v \in \mathrm{dom}_t \text{ with } \mathrm{val}_{t_1}(u) = q \text{ and } \varphi_1(u) = v \\ \exists\, q(\mathrm{val}_t(v)(x_1, \dots, x_i)) \to w[q_1(x_{j_1}), \dots, q_n(x_{j_n})] \in \Delta \\ t_2 = s \cdot w[q_1, \dots, q_n] \text{ with } s = t_1[\circ/u] \\ \varphi_2 \text{ with } D_{t_2} = D_{t_1} \setminus \{u\} \cup \{u_i \mid u \sqsubseteq u_i, \mathrm{val}_{t_2}(u_i) = q_i, 1 \le i \le n\} \\ \forall\, u' \in D_{t_1} \setminus \{u\} : \varphi_2(u') = \varphi_1(u') \\ \forall\, u_i, u \sqsubseteq u_i, \mathrm{val}_{t_2}(u_i) = q_i : \varphi_2(u_i) = v.j_i \end{cases}$$

Furthermore, let $\to_\mathcal{T}^*$ be the reflexive and transitive closure of $\to_\mathcal{T}$ and $\to_\mathcal{T}^n$ the reachability relation for $\to_\mathcal{T}$ in n steps. The relation $R(\mathcal{T}) \subseteq T_\Sigma \times T_\Gamma$ induced by a top-down tree transducer $\mathcal{T}$ is

$$R(\mathcal{T}) = \{(t, t') \mid (t, q_0, \varphi_0) \to_\mathcal{T}^* (t, t', \varphi') \text{ with } \varphi_0(\varepsilon) = \varepsilon, \varphi' \text{ is empty and } t' \in T_\Gamma\}.$$

For a tree $t \in T_\Sigma$ let $\mathcal{T}(t)$ be the final transformed output of $\mathcal{T}$ for t. The class of relations definable by TDTs is called the class of *top-down tree transformations*.

Example 1 Let Σ be a ranked alphabet given by $\Sigma_2 = \{f\}$, $\Sigma_1 = \{g, h\}$, and $\Sigma_0 = \{a\}$. Consider the TDT $\mathcal{T}$ given by $(\{q\}, \Sigma, \Sigma, \{q\}, \Delta)$ with $\Delta = \{\ q(a) \to a,\ q(g(x_1)) \to q(x_1),\ q(h(x_1)) \to h(q(x_1)),\ q(f(x_1, x_2)) \to f(q(x_1), q(x_2))\ \}$. For each $t \in T_\Sigma$ the transducer deletes all occurrences of g in t.

Consider $t := f(g(h(a)), a)$. A possible sequence of configurations of $\mathcal{T}$ on t is $c_0 \to_\mathcal{T}^5 c_5$ such that $c_0 := (t, q, \varphi_0)$ with $\varphi_0(\varepsilon) = \varepsilon$, $c_1 := (t, f(q, q), \varphi_1)$ with $\varphi_1(1) = 1$, $\varphi_1(2) = 2$, $c_2 := (t, f(q, q), \varphi_2)$ with $\varphi_2(1) = 11$, $\varphi_2(2) = 2$, $c_3 := (t, f(q, a), \varphi_3)$ with $\varphi_3(1) = 11$, $c_4 := (t, f(h(q), a), \varphi_4)$ with $\varphi_4(11) = 111$, and $c_5 := (t, f(h(a), a), \varphi_5)$. A visualization of c_2 is shown in Figure 1. $\triangleleft$

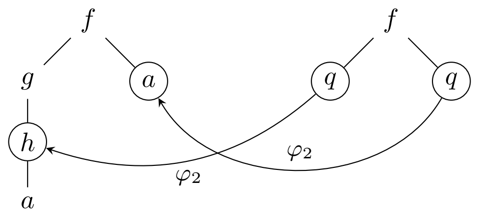

Figure 1: The configuration $c_2 = (t, f(q,q), \varphi_2)$ of $\mathcal{T}$ on t from Example 1.

Games. A safety game $\mathcal{G} = (V, V_0, V_1, E, S)$ is played by two players, Player 0 and Player 1, on a directed game graph $G = (V, E)$, where

- $V = V_0 \uplus V_1$ is a partition of the vertices into positions V_0 belonging to Player 0 and positions V_1 belonging to Player 1,
- $E \subseteq V \times V$ is the set of allowed moves, and
- $S \subseteq V$ is a set of safe vertices.

A play is a maximal finite or infinite sequence $v_0 v_1 v_2 \ldots$ of vertices compatible to the edges of the game graph starting from an initial vertex $v_0 \in V$. A play is maximal if it is either infinite or it ends in a vertex without outgoing edges. Player 0 wins a play if it stays inside the safe region, i.e., $v_i \in S$ for all i.

Let $i \in \{0, 1\}$, a strategy for Player i is a function $\sigma_i : V^* V_i \to V$ such that $\sigma_i(v_0 \ldots v_n) = v_{n+1}$ implies that $(v_n, v_{n+1}) \in E$. A strategy σ_i is a winning strategy from a vertex $v_0 \in V$ for Player i if the player wins every play starting in v_0, no matter how the opponent plays, if Player i plays according σ_i.

Safety games are positionally determined, cf. [13], i.e., for each vertex $v \in V$ one of the players has a winning strategy from v. Furthermore, the player always has a positional winning strategy σ, meaning that the strategy does not consider the previously seen vertices, but only the current vertex. More formally, a positional strategy σ_i for Player i is a mapping $\sigma_i : V_i \to V$ such that $(v, \sigma_i(v)) \in E$ for all $v \in V_i$.

3. Deterministic Top-down Specifications

Here we investigate uniformization of tree-automatic relations in the class of top-down tree transformations. We restrict ourselves in the scope of Section 3.1 and 3.2 to D↓TA-recognizable relations with total domain. Later on, in Section 3.3, we will briefly describe how we can deal with D↓TA-recognizable relations whose domain is not total but deterministic top-down tree automatic. In the course of this, we will consider two variants of uniformization. The classical uniformization setting, where invalid input trees have to be rejected and a more relaxed setting, where for each valid input tree the transducer selects one output tree, on each other input tree which is not part of the domain the transducer may behave arbitrarily. To distinguish between these issues, we will speak of uniformization with input validation and uniformization without input validation.

For Section 3.1 and Section 3.2, let $R \subseteq T_\Sigma \times T_\Gamma$ be a deterministic top-down tree automaton-definable relation with total domain and let $\mathcal{A} = (Q_\mathcal{A}, \Sigma_\perp \times \Gamma_\perp, q_0^\mathcal{A}, \Delta_\mathcal{A})$ be a D↓TA that recognizes R. For $q \in Q_\mathcal{A}$, let $\mathcal{A}_q$ be the automaton that results from $\mathcal{A}$ by using q as single initial state.

To begin with, we investigate the connection between input and output. A TDT $\mathcal{T}$ possibly reaches a point such that the position v of the input symbol under consideration and the position u of the correspondingly produced output are different (formally, the mapping φ from the configuration maps u to a node $v \neq u$). As a consequence, if u and v lie on divergent paths, then $\mathcal{T}$ selects the output at u (and below u) independent of the subtree at u of the input tree. In such a case, in order to satisfy a specification, the produced output tree at u has to match all possible input trees at u. Such cases are considered in the lemma below.

Lemma 2 *Let $q \in Q_\mathcal{A}$. It is decidable whether the following holds:*

1. $\forall t \in T_\Sigma : t \otimes \bot \in T(\mathcal{A}_q)$,
2. $\exists t' \in T_\Gamma : \bot \otimes t' \in T(\mathcal{A}_q)$,
3. $\exists t' \in T_\Gamma \, \forall t \in T_\Sigma : t \otimes t' \in T(\mathcal{A}_q)$.

Proof.

1. Let $\mathcal{A}_q^{\Sigma\times\bot}|_{T_\Sigma} = (Q_\mathcal{A}, \Sigma, q, \Delta'_\mathcal{A})$ be the automaton that results from $\mathcal{A}_q$ by using $\big(\bigcup_{i=1}^m (Q_\mathcal{A} \times (\Sigma \times \bot) \times Q_\mathcal{A}^i)\big) \cap \Delta_\mathcal{A}$ projected onto $\bigcup_{i=1}^m (Q_\mathcal{A} \times \Sigma \times Q_\mathcal{A}^i)$ as new transition set $\Delta'_\mathcal{A}$. It holds that $\forall t \in T_\Sigma : t \otimes \bot \in T(\mathcal{A}_q)$ if, and only if, $T_\Sigma = T(\mathcal{A}_q^{\Sigma\times\bot}|_{T_\Sigma})$.

2. Let $\mathcal{A}_q^{\bot\times\Gamma}|_{T_\Gamma} = (Q_\mathcal{A}, \Gamma, q, \Delta'_\mathcal{A})$ be the automaton that results from $\mathcal{A}_q$ by using $\big(\bigcup_{i=1}^m (Q_\mathcal{A} \times (\bot \times \Gamma) \times Q_\mathcal{A}^i)\big) \cap \Delta_\mathcal{A}$ projected onto $\bigcup_{i=1}^m (Q_\mathcal{A} \times \Gamma \times Q_\mathcal{A}^i)$ as new transition set $\Delta'_\mathcal{A}$. It holds that $\exists t' \in T_\Gamma : \bot \otimes t' \in T(\mathcal{A}_q)$ if, and only if, $T(\mathcal{A}_q^{\bot\times\Gamma}|_{T_\Sigma}) \neq \emptyset$.

3. First, we construct an N↓TA $\bar{\mathcal{A}}_q$ for $\overline{T(\mathcal{A}_q)}$. Secondly, we define the N↓TA $\mathcal{C} = (Q_{\bar{\mathcal{A}}_q} \uplus \{p, p_\bot\}), \Sigma_\bot \times \Gamma_\bot, q_0^{\bar{\mathcal{A}}_Q}, \Delta_\mathcal{C})$ as the product automaton of $\bar{\mathcal{A}}_q \times \mathcal{B}$ with $\Delta_\mathcal{C}$ constructed as follows:

   - For$f \in \Sigma_i$ and $(q, (f,g), q_1, \dots, q_n) \in \Delta_{\bar{\mathcal{A}}}$
   $$\big((q,p), (f,g), (q_1,p_1), \dots, (q_i,p_i), (q_{i+1}, p_\bot), \dots, (q_n, p_\bot)\big) \in \Delta_\mathcal{C},$$
   - for $p_\bot$ and $(q, (\bot, g), q_1, \dots, q_n) \in \Delta_\mathcal{A}$
   $$\big((q,p_\bot), (\bot,g), (q_1,p_\bot), \dots, (q_n, p_\bot)\big) \in \Delta_\mathcal{C}.$$

   We obtain $R(\mathcal{C}) = \{(t,t') \in T_\Sigma \times T_\Gamma \mid t \otimes t' \notin T(\mathcal{A}_q)\}$. Let $\mathcal{C}|_{T_\Gamma}$ be the automaton that results from $\mathcal{C}$ by projection of $\Delta_\mathcal{C}$ onto $\bigcup_{i=1}^m (Q_\mathcal{C} \times \Gamma \times Q_\mathcal{C}^i)$. $T(\mathcal{C}|_{T_\Gamma}) = \{t' \in T_\Gamma \mid \exists t \in T_\Sigma : t \otimes t' \notin T(\mathcal{A}_q)\}$. Thirdly, construct a tree automaton $\overline{\mathcal{C}|_{T_\Gamma}}$ that recognizes the complement $\overline{T(\mathcal{C}|_{T_\Gamma})} = \{t' \in T_\Gamma \mid \forall t \in T_\Sigma : t \otimes t' \in T(\mathcal{A}_q)\}$. It holds that $\exists t' \in T_\Gamma \, \forall t \in T_\Sigma : t \otimes t' \in T(\mathcal{A}_q)$ if, and only if, $T(\overline{\mathcal{C}|_{T_\Gamma}}) \neq \emptyset$.

□

3.1. Bounded delay

In this section, we consider the question whether there exists a uniformization of a D↓TA-recognizable relation with total domain such that the output delay is bounded. First, we formally define what is meant by output delay. Let $\mathcal{T}$ be a TDT and let $c = (t, t', \varphi)$ be a configuration of $\mathcal{T}$. Consider a node $u \in D_{t'}$. If $|\varphi(u)| \geq |u|$, we say the transducer has an *output delay* of $|\varphi(u)| - |u|$. If there is a $k \in \mathbb{N}$ such that for all reachable configurations $c = (t, t', \varphi)$ of $\mathcal{T}$ holds that for all $u \in D_{t'}$ the output delay $|\varphi(u)| - |u|$ is at most k, then the output delay is bounded by k.

We will solve the following problem.

Theorem 3 *Given $k > 0$, it is decidable whether a given D↓TA-recognizable relation with total domain has a uniformization by a deterministic top-down tree transducer with output delay bounded by k.*

Before we present a decision procedure, we introduce some notations that will simplify the presentation. Given $\Sigma = \bigcup_{i=0}^m \Sigma_i$, let $\mathrm{dir}_\Sigma = \{1, \dots, m\}$ be the set of directions compatible with Σ. For $\Sigma = \bigcup_{i=0}^m \Sigma_i$, the set Path_Σ of labeled paths over Σ is defined inductively by:

- ε is a labeled input path and each $f \in \Sigma$ is a labeled input path,
- given a labeled input path $\pi = x \cdot f$ with $f \in \Sigma_i \, (i > 0)$ over Σ, then $\pi \cdot jg$ with $j \in \{1, \dots, i\}$ and $g \in \Sigma$ is a labeled input path.

For $\pi \in \mathrm{Path}_\Sigma$, we define the path *path* and the word *lbls* induced by π inductively by:

- if $\pi = \varepsilon$ or $\pi = f$, then $path(\varepsilon) = path(f) = \varepsilon$, $lbls(\varepsilon) = \varepsilon$ and $lbls(f) = f$,
- if $\pi = x \cdot jf$ with $j \in \mathbb{N}$, $f \in \Sigma$, then $path(\pi) = path(x) \cdot j$, $lbls(\pi) = lbls(x) \cdot f$.

The length $|| \, ||$ of a labeled path over Σ is the length of the word induced by its path, i.e., $||\pi|| = |lbls(\pi)|$.

For $\pi \in \mathrm{Path}_\Sigma$ with $||\pi|| = k$ let

$$T_\Sigma^\pi := \{t \in T_\Sigma \mid val_t\big(path(\pi)[1 \dots (i-1)]\big) = lbls(\pi)[i] \text{ for } 1 \leq i \leq k\}$$

be the set of trees t over Σ such that π is a prefix of a labeled path through t. For $\Pi \subseteq \mathrm{Path}_\Sigma$ let

$$T_\Pi := \{t \in T_\Sigma \mid \exists \pi \in \Pi \text{ and } t \in T_\Sigma^\pi\}$$

be the set of trees such that each tree contains a labeled path starting with π for some $\pi \in \Pi$.

For $t \in T_\Sigma$ and $u \in \mathrm{dir}_\Sigma^*$, let $||t||^u := max\{|v| \mid v \in dom_t \text{ and } (u \sqsubseteq v \text{ or } v \sqsubseteq u)\}$ be the length of a maximal path through t along u.

Now, in order to solve the above decision problem, we consider a safety game between two players. The procedure is similar to a decision procedure presented in [5], where the question whether a uniformization of an automatic word relation by a word transducer exists, is reduced to the existence of winning strategies in a safety game. The game is played between In and Out, where In can follow any path from the root to a leaf in an input tree such that In plays one input symbol at a time. Out can either react with an output symbol, or delay the output and react with a direction in which In should continue with his input sequence.

The vertices in the game graph keep track of the state of $\mathcal{A}$ on the input combined with the output on the same path and additionally of the input that is ahead, which is bounded by k. We will see that it is not necessary to consider situations where input and output are on divergent paths. The intuition behind this is that D↓TAs cannot compare information on divergent paths through an input tree. Formally, the game graph $G_\mathcal{A}^k$ is constructed as follows.

- $V_{\mathsf{In}} = \{\big(q, \pi j\big) \in Q_\mathcal{A} \times \mathrm{Path}_\Sigma \cdot \mathrm{dir}_\Sigma \mid ||\pi|| \leq k, \pi \in \mathrm{Path}_\Sigma, j \in \mathrm{dir}_\Sigma\} \cup 2^{Q_\mathcal{A}}$ is the set of vertices of player In.
- $V_{\mathsf{Out}} = \{\big(q, \pi\big) \in Q_\mathcal{A} \times \mathrm{Path}_\Sigma \mid ||\pi|| \leq k\}$ is the set of vertices of player Out.
- From a vertex of In the following moves are possible:
  - i) $\big(q, \pi j\big) \rightarrow \big(q, \pi j f\big)$ for each $f \in \Sigma$ if $\|\pi\| < k$ (delay; In chooses the next input)
  - ii) $\{q_1, \dots, q_n\} \rightarrow \big(q_i, f\big)$ for each $i \in \{1, \dots, n\}$

    (no delay; In chooses the next direction and input)
- From a vertex of Out the following moves are possible:
  - iii) $\big(q, f\big) \xrightarrow{r} \{q_1, \dots, q_i\}$ if there is $r = (q, (f, g), q_1, \dots, q_n) \in \Delta_\mathcal{A}$, $f \in \Sigma$ is i-ary, $g \in \Sigma_\perp$ is j-ary, and if $j > i$, there exist trees $t_{i+1}, \dots, t_j \in T_\Gamma$ such that $\perp \otimes t_l \in T(\mathcal{A}_{q_l})$ for all $i < l \leq j$.

    (no delay; Out applies a transition; Out can pick output trees for all directions where the input has ended; In can continue from the other directions)
  - iv) $\big(q, \pi j' f'\big) \xrightarrow{r} \big(q', \pi' j' f'\big)$ for each $g \in \Gamma_\perp$ such that $\pi = f j \pi'$, there is $r = (q, (f, g), q_1, \dots, q_n) \in \Delta_\mathcal{A}$ with $q' = q_j$, and for each $l \neq j$ with $l \in \{1, \dots, n\}$ holds
    - if $l \leq rk(f), rk(g)$, then $\exists t' \in T_\Gamma \forall t \in T_\Sigma : t \otimes t' \in T(\mathcal{A}_{q_l})$
    - if $rk(g) < l \leq rk(f)$, then $\forall t \in T_\Sigma : t \otimes \perp \in T(\mathcal{A}_{q_l})$
    - if $rk(f) < l \leq rk(g)$, then $\exists t' \in T_\Gamma : \perp \otimes t' \in T(\mathcal{A}_{q_l})$

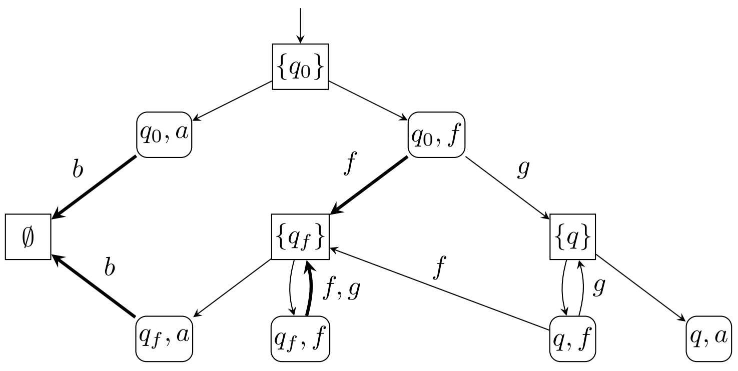

Figure 2: The game graph $G^1_\mathcal{A}$ constructed from the D↓TA $\mathcal{A}$ from Example 4. A possible winning strategy for Out in $\mathcal{G}^1_\mathcal{A}$ is emphasized in the graph.

(delay; Out applies a transition, removes the leftmost input and advances in direction of the labeled path ahead; Out can pick output trees for all divergent directions)

v) $\big(q, \pi j f\big) \to \big(q, \pi j f j'\big)$ for each $j' \in \{1, \dots, i\}$ for $f \in \Sigma_i$ if $\|\pi j f\| < k$

(Out delays and chooses a direction from where In should continue)

- The initial vertex is $\{q_0^\mathcal{A}\}$.

Note that the game graph can effectively be constructed, because Lemma 2 implies that it is decidable whether the edge constraints are satisfied.

The winning condition should express that player Out loses the game if the input can be extended, but no valid output can be produced. This is represented in the game graph by a set of bad vertices B that contains all vertices of Out with no outgoing edges. If one of these vertices is reached during a play, Out loses the game. Thus, we define $\mathcal{G}^k_\mathcal{A} = (G^k_\mathcal{A}, V \setminus B)$ as safety game for Out.

Example 4 Let Σ be an input alphabet given by $\Sigma_2 = \{f\}$ and $\Sigma_0 = \{a\}$ and let Γ be an output alphabet given by $\Gamma_2 = \{f, g\}$ and $\Gamma_0 = \{b\}$. Consider the relation R that contains exactly the pairs of trees $(t, t') \in T_\Sigma \times T_\Gamma$ such that t and t' have the same domain and on every path through t' occurs an f if $t \neq a$.

It is easy to see that D↓TA $\mathcal{A} = (\{q_0, q, q_f\}, \Sigma \times \Gamma, q_0, \Delta_\mathcal{A})$ with $\Delta_\mathcal{A} = \{(q_0, (a, b)),\ (q_0, (f, f), q_f, q_f),\ (q_0, (f, g), q, q),\ (q, (f, f), q_f, q_f),\ (q, (f, g), q, q),\ (q_f, (a, b)),\ (q_f, (f, f), q_f, q_f),\ (q_f, (f, g), q_f, q_f)\}$ recognizes R. For $k = 1$, the corresponding game graph $G^1_\mathcal{A}$ is depicted in Figure 2. $\triangleleft$

The following two lemmata show that from the existence of a winning strategy a top-down tree transducer that uniformizes the relation can be obtained and vice versa.

Lemma 5 *If Out has a winning strategy in $\mathcal{G}^k_\mathcal{A}$, then R has a uniformization by a DTDT in which the output delay is bounded by k.*

Proof. Assume that Out has a winning strategy in the safety game $\mathcal{G}^k_\mathcal{A}$, then there is also a positional one. We can represent a positional winning strategy by a function $\sigma : V_\mathsf{Out} \to \Delta_\mathcal{A} \cup \mathrm{dir}_\Sigma$, because Out either plays one output symbol (corresponding to a unique transition in $\Delta_\mathcal{A}$), or a new direction for an additional input symbol.

We construct a deterministic TDT $\mathcal{T} = (Q_\mathcal{A} \cup \{\big(q, \pi j\big) \mid \big(q, \pi j f\big) \in V_\mathsf{Out}\}, \Sigma, \Gamma, q_0^\mathcal{A}, \Delta)$ from such a positional winning strategy σ as follows:

a) For each $\sigma : \big(q, f\big) \overset{r}{\mapsto} \{q_1, \dots, q_i\}$ with $r = (q, (f, g), q_1, \dots, q_n) \in \Delta_\mathcal{A}$:

- add $q(f(x_1, \dots, x_i)) \to g(q_1(x_1), \dots, q_j(x_j))$ to Δ if $j \leq i$, or

- add $q(f(x_1,\dots,x_i)) \to g(q_1(x_1),\dots,q_i(x_i),t_{i+1},\dots,t_j)$ to Δ if $j > i$

where $f \in \Sigma_i$, $g \in \Gamma_j$ and $t_{i+1},\dots,t_j \in T_\Gamma$ chosen according to the r-edge constraints in $\big(q,f\big)$.

b) For each $\sigma : \big(q,\pi jf\big) \mapsto \big(q,\pi jfj'\big)$ add $\big(q,\pi j\big)(f(x_1,\dots,x_i)) \to \big(q,\pi jfj'\big)(x_{j'})$ to Δ.

If the strategy σ defines a sequence of moves of Out inside vertices of V_{Out}, that is a sequence of moves of type iv), then this corresponds to an output sequence that is produced without reading further input. Each output of these moves can be represented by a special tree s as follows. A move of type iv) has the form $(q,fj\pi) \xrightarrow{r} (q',\pi)$ with $r = (q,(f,g),q_1,\dots,q_n)$ and $q' = q_j$. Then, let $s = g(t_1,\dots,t_{j-1},\circ,t_{j+1},\dots,t_n) \in S_\Sigma$ be the special tree, where each $t_l \in T_\Gamma$ is chosen according to the r-edge constraints in $(q,fj\pi)$ for $l \neq j, 1 \leq l \leq n$. Eventually, the strategy defines a move of Out of type iii) or v) to a node of V_{In}, otherwise σ is not a winning strategy. These parts of the strategy are transformed as follows:

c) For each $\big(q,\pi jf\big) \xrightarrow{r_1} \dots \xrightarrow{r_{l-1}} \big(q',\pi'jf\big) \to \big(q',\pi'jfj'\big)$ add $\big(q,\pi j\big)(f(x_1,\dots,x_i)) \to s_1 \cdot \dots \cdot s_{l-1} \cdot \big(q',\pi'jfj'\big)(x_{j'})$ to Δ, where each $s_i \in S_\Gamma$ is a special tree corresponding to the r_i-edge in the ith move.

d) For each $\big(q,\pi jf\big) \xrightarrow{r_1} \dots \xrightarrow{r_{l-1}} \big(q',f\big) \xrightarrow{r_l} \{q_1,\dots,q_i\}$ add $\big(q,\pi j\big)(f(x_1,\dots,x_i)) \to s_1 \cdot \dots \cdot s_{l-1} \cdot s$ to Δ, where each $s_i \in S_\Gamma$ is a special tree corresponding to the r_i-edge in the ith move and s is an output corresponding to r_l constructed as described in step a).

We now verify that $\mathcal{T}$ defines a uniformization of R. Let $t \in T_\Sigma$. We can show by induction on the number of steps needed to reach a configuration from the initial configuration $(t,q_0^{\mathcal{A}},\varphi_0)$ that for each configuration $c = (t,t',\varphi)$ such that $D_{t'} \neq \emptyset$, in other words $t' \notin T_\Gamma$, there exists a successor configuration c'. Thus, $(t,\mathcal{T}(t)) \in R$.

In $\mathcal{T}$ the output delay is bounded by k, because the existence of a winning strategy σ guarantees that from a vertex (q,π) with $|\pi| = k$, that is reachable by playing according to σ, a move of Out follows. It follows from the construction that $\mathcal{T}$ produces output accordingly. $\square$

The size of $G^k_{\mathcal{A}}$ is at most $Q_{\mathcal{A}} \cdot (|\Sigma| \cdot |\mathrm{dir}_\Sigma|)^{k-1} \cdot |\Sigma| + 2^{Q_{\mathcal{A}}}$. For the winning player, a positional winning strategy can be determined in linear time in the size of $G^k_{\mathcal{A}}$ (see Theorem 3.1.2 in [14], which can easily be adapted to safety games). For a positional winning strategy of Out, the above construction yields a DTDT with delay bounded by k that uses at most $Q_{\mathcal{A}} \cdot (|\Sigma| \cdot \mathrm{dir}_\Sigma)^{k-1}$ states.

We now show the other direction.

Lemma 6 *If R has a uniformization by a DTDT in which the output delay is bounded by k, then* Out *has a winning strategy in $\mathcal{G}^k_{\mathcal{A}}$.*

Proof. Assume that R has a uniformization by some DTDT $\mathcal{T} = (Q,\Sigma,\Gamma,q_0,\Delta)$ in which the output delay in bounded by k. A winning strategy for Out basically takes the moves corresponding to the output sequence that $\mathcal{T}$ produces for a read input sequence induced by the moves of In. We construct the strategy inductively. In a play, a vertex $\big(q,y\big)$ is reached by a sequence of moves that describe a labeled path $xiy \in \mathrm{Path}_\Sigma$ with $x,y \in \mathrm{Path}_\Sigma$, and $i \in \mathrm{dir}_\Sigma$. Let $\mathit{path}(xi) = u$ and $\mathit{path}(xiy) = v$. The strategy in $G^k_{\mathcal{A}}$ can be chosen such that in every play according to the strategy for each reached vertex $\big(q,y\big)$ the following property is satisfied. There is a $t \in T^{xiy}_\Sigma$ such that the deterministic run $\rho_{\mathcal{A}}$ of $\mathcal{A}$ on $t \otimes \mathcal{T}(t)$ yields $\rho_{\mathcal{A}}(u) = q$. Further, if $||y|| > 1$, then there is a configuration (t,t',φ) of $\mathcal{T}$ with $(t,q_0,\varphi_0) \to^*_{\mathcal{T}} (t,t',\varphi)$ such that $u \in D_{t'}$ with $\varphi(u) = v$, or there is $u' \in D_{t'}$ with $u \sqsubset u'$ and $\varphi(u') = vd$ for some $d \in \mathrm{dir}_\Sigma$.

We define the strategy as follows. First, we consider the case $y = f \in \Sigma$, i.e., $||y|| = 1$ and $u = v$. For a $t \in T^{xif}_\Sigma$ let $s = t[\circ/u] \cdot f$ be the tree that is obtained by deleting all nodes below u. Then, Out can make her next move according to the result of the computation of $\mathcal{T}$ on s. If output was produced that is mapped to u, then there has to exist a rule r of the form $(q,\mathit{val}_{t\otimes\mathcal{T}(t)}(u),q_1,\dots,q_n) \in \Delta_{\mathcal{A}}$ since $\mathcal{T}$ uniformizes R. Also, there has to exist an outgoing r-edge from (q,f) that Out can take. Otherwise, if no output was produced that is mapped to u, then the output is dependent on a node below u. Thus, it must hold that there is a reachable configuration (t, t', φ) with $\varphi(u) = v$ and there exists a successor configuration (t, t'', φ') with $\varphi'(u) = vj$ for some $j \in \text{dir}_\Sigma$. Out delays the output and chooses direction j as next move.

Secondly, we consider the case $||y|| > 1$. There are two possibilities. Either $\mathcal{T}$ reaches a configuration (t, t', φ) with $\varphi(u) = v$ and produces no output in the next configuration step, or $\mathcal{T}$ reaches (t, t', φ) with $\varphi(u') = vd$ for some u' with $u \sqsubset u'$ and some $d \in \text{dir}_\Sigma$, meaning that $\mathcal{T}$ has produced output at u after reading the input symbol at node v. If no output was produced, Out also delays. In this case, there is a successor configuration (t, t'', φ') with $\varphi'(u) = vj$ for some $j \in \text{dir}_\Sigma$ and Out chooses direction j. This can happen at most k times in a row, since the output delay in $\mathcal{T}$ is bounded by k. Otherwise, output was produced. Since $\mathcal{T}$ uniformizes R, there has to exist a rule r of the form $(q, val_{t\otimes\mathcal{T}(t)}(u), q_1, \dots, q_n) \in \Delta_\mathcal{A}$. We show that there also exists an r-labeled edge outgoing from (q, y) that Out can choose. Let $y = fjy'$ with $j \in \text{dir}_\Sigma$. Furthermore, let $val_{t\otimes\mathcal{T}(t)}(u) = (f, g) \in \Sigma \times \Gamma_\perp$. We have to prove that the r-edge constraints of type iv) are satisfied. That means, we have to show for each direction $l \neq j$ with $l \in \{1, \dots, n\}$ that there exists an output tree that matches all possible input trees. More formally, it holds that $\rho_\mathcal{A}(u) = q$ and $\rho_\mathcal{A}(ul) = q_l$ for each $1 \leq l \leq n$. We have to show that for each $l \neq j$ that there exists an output tree s_l such that $T_\Sigma \times \{s_l\} \subseteq T(\mathcal{A}_{q_l})$ (or $\{\perp\} \times \{s_l\} \subseteq T(\mathcal{A}_{q_l})$ if $l > rk(f)$). Clearly, this holds if the right-hand side of the applied rule is of the form $g[s_1, \dots, s_{j-1}, w[\dots], s_{j+1}, \dots, s_{rk(g)}]$, where $s_1, \dots, s_{j-1}, s_{j+1}, \dots, s_{rk(g)} \in T_\Gamma$ and w is a Γ-context. However, in general, the applied rule is of the form $g[w_1[\dots], \dots, w_{rk(g)}[\dots]]$, where $w_1, \dots, w_{rk(g)}$ are Γ-contexts. Consequently, $\mathcal{T}(t)|_{ul}$ is produced by $\mathcal{T}$ reading input below v in t, i.e., it is depended on $t|_v$. We will show that for a direction $l \neq j$ we can pick the desired output tree s_l as $\mathcal{T}(t)|_{ul}$. Intuitively this is possible, because ul is not on the same path as v for $l \neq j$, which means that in the successful run of $\mathcal{A}$ on $t \otimes \mathcal{T}(t)$ the reached state at ul is independent of the reached state at v. Now, for the formal proof we distinguish three possibilities for l. First, if $l \leq rk(f), rk(g)$, then $t_l \otimes \mathcal{T}(t)|_{ul} \in T(\mathcal{A}_{ql})$ for all $t_l \in T_\Sigma$. Assume this is not true, then there exists $t_l' \in T_\Sigma$ such that $t_l' \otimes \mathcal{T}(t)|_{ul} \notin T(\mathcal{A}_{ql})$. Since $dom_{t|ul} \cap dom_{t|v} = \emptyset$, we can pick $t' = t[\circ/ul] \cdot t_l'$ and obtain $\mathcal{T}(t) = \mathcal{T}(t')$. Thus, $t_l' \otimes \mathcal{T}(t)|_{ul} \in T(\mathcal{A}_{ql})$ which is a contradiction. Secondly, if $rk(g) < l \leq rk(f)$, then $t_l \otimes \perp \in T(\mathcal{A}_{ql})$ for all $t_l \in T_\Sigma$ and thirdly, if $rk(g) < l \leq rk(f)$, then $\perp \otimes \mathcal{T}(t)|_{ul} \in T(\mathcal{A}_{ql})$. For the correctness, the same argumentation as in the case $l \leq rk(f), rk(g)$ can be applied.

As we have seen, Out never reaches a vertex without outgoing edges and therefore wins. ☐

As a consequence of Lemma 5 and Lemma 6 together with the fact that a winning strategy for Out can effectively be computed in $\mathcal{G}^k_\mathcal{A}$ we immediately obtain Theorem 3.

3.2. Unbounded delay

Previously, we considered the question whether there exists a uniformization without input validation of a D↓TA-recognizable relation with D↓TA-recognizable domain such that the output delay is bounded. In this section, we will show, that this question is also decidable if the output delay is unbounded. Similar to [5] for automatic word relations, we will see that if the output delay exceeds a certain bound, then we can decide whether the uniformization is possible or not.

The intuition is that if it is necessary to have such a long delay between input and output, then only one path in the tree is relevant to determine an output tree. We can define this property by introducing the term path-recognizable function. If a relation is uniformizable by a path recognizable function, then the relation has a uniformization by a DTDT that first deterministically reads one path of the input tree and then outputs a matching output tree.

Formally, we say a relation R is *uniformizable by a path-recognizable function*, if there exists a DTDT $\mathcal{T}$ that uniformizes R such that $\Delta_\mathcal{T}$ only contains transitions of the following form:

$$q(f(x_1, \dots, x_i)) \to q'(x_{j_1}) \quad \text{or} \quad q(a) \to t,$$

where $f \in \Sigma_i$, $i > 0$, $a \in \Sigma_0$, $q, q' \in Q$ and $j_1 \in \{1, \dots, i\}$ and $t \in T_\Gamma$.

In the following, we will show that there exists a bound on the output delay that we have to consider in order to decide whether a uniformization by a path-recognizable function is possible.

Beforehand, we need to fix some notations. For R, $\pi \in \text{Path}_\Sigma$ and $q \in Q_\mathcal{A}$ let

$$R^\pi := \{(t, t') \in R \mid t \in T^\pi_\Sigma\} \text{ and } R^\pi_q := \{(t, t') \in R(\mathcal{A}_q) \mid t \in T^\pi_\Sigma\}.$$

If $q = q_0^{\mathcal{A}}$, then R_q^π corresponds to R^π, if additionally $\pi = \varepsilon$, then R_q^π corresponds to R. Note, a D↓TA that recognizes R_q^π can be easily constructed from $\mathcal{A}$.

Since we will consider labeled paths through trees, it is convenient to define the notion of convolution also for labeled paths. For a labeled path $x \in \mathrm{Path}_\Sigma$ with $||x|| > 0$, let $dom_x := \{u \in \mathrm{dir}_\Sigma^* \mid u \sqsubseteq path(x)\}$ and $val_x : dom_x \to \Sigma$, where $val_x(u) = lbls(x)[i]$ if $u \in dom_x$ with $|u| = i + 1$. Let $x \in \mathrm{Path}_\Sigma$, $y \in \mathrm{Path}_\Gamma$ with $path(y) \sqsubseteq path(x)$ or $path(x) \sqsubseteq path(y)$, then the *convolution* of x and y is $x \otimes y$ defined by $dom_{x\otimes y} = dom_x \cup dom_y$, and $val_{x\otimes y}(u) = (val_x^\perp(u), val_y^\perp(u))$ for all $u \in dom_{x\otimes y}$, where $val_x^\perp(u) = val_x(u)$ if $u \in dom_x$ and $val_x^\perp(u) = \perp$ otherwise, analogously defined for $val_y^\perp(u)$.

Furthermore, it is useful to relax the notion of runs to labeled paths. Let $i \in \mathrm{dir}_\Sigma$, $x \in \mathrm{Path}_\Sigma$, $y \in \mathrm{Path}_\Gamma$ such that $x \otimes y$ is defined, i.e., $path(y) \sqsubseteq path(x)$ or $path(x) \sqsubseteq path(y)$. We define $\rho_{\mathcal{A}} : \mathrm{dir}_\Sigma^* \to Q_{\mathcal{A}}$ to be the partial function with $\rho_{\mathcal{A}}(\varepsilon) = q_0^{\mathcal{A}}$, and for each $u \in dom_{x\otimes y}$: if $q := \rho_{\mathcal{A}}(u)$ is defined and there is a transition $(q, val_{x\otimes y}(u), q_1, \dots, q_i) \in \Delta_{\mathcal{A}}$, then $\rho_{\mathcal{A}}(u.j) = q_j$ for all $j \in \{1, \dots, i\}$. Let $path(x \otimes y) = v$. Shorthand, we write

$$\mathcal{A} : q_0^{\mathcal{A}} \xrightarrow{x\otimes y}_i q,$$

if $q := \rho_{\mathcal{A}}(vi)$ is defined. We write $\mathcal{A} : q_0^{\mathcal{A}} \xrightarrow{x\otimes y} F_{\mathcal{A}}$ if $(\rho_{\mathcal{A}}(v), val_{x\otimes y}(v)) \in \Delta_{\mathcal{A}}$ to indicate that the (partial) run $\rho_{\mathcal{A}}$ of $\mathcal{A}$ on $x \otimes y$ is accepting.

Sometimes it is sufficient to consider only the output that is mapped to a certain path. For an input tree $t \in T_\Sigma$ or $t \in S_\Sigma$ and a path $u \in \mathrm{dir}_\Sigma^*$, we define

$$out_{\mathcal{T}}(t, u) := \{\pi \in \mathrm{Path}_\Gamma \mid \mathcal{T}(t) \in T_\Gamma^\pi \text{ and } (path(\pi) \sqsubseteq u \text{ or } u \sqsubseteq path(\pi))\}$$

to be the set of labeled paths through the output tree $\mathcal{T}(t)$ along u. Note, that if $\mathcal{T}$ is deterministic and $||\mathcal{T}(t)||^u < |u|$, then $out_{\mathcal{T}}(t, u)$ is a singleton set.

We introduce a partial function that yields the state transformations on a path π induced by the input sequence of π together with some output sequence on the same path of same or smaller length. Formally, for $x \in \mathrm{Path}_\Sigma$, $y \in \mathrm{Path}_\Gamma$ and a direction $i \in \mathrm{dir}_\Sigma$ such that $path(y) \sqsubseteq path(x)$, we define the partial function $\tau_{xi,y} : Q_{\mathcal{A}} \to Q_{\mathcal{A}}$ with $\tau_{xi,y}(q) := q'$ if $\mathcal{A} : q \xrightarrow{x\otimes y}_i q'$ and for each uj with $u \in dom_x : uj \not\sqsubseteq path(xi)$ and $j \in \{1, \dots, rk\big((val_x^\perp(u), val_y^\perp(u))\big)\}$ holds

- if $r := \rho_{\mathcal{A}}(uj)$ and $j \le rk(val_x^\perp(u))$, then there exists $t' \in T_\Gamma$ such that for all $t \in T_\Sigma$ holds $t \otimes t' \in T(\mathcal{A}_r)$, and
- if $r := \rho_{\mathcal{A}}(uj)$ and $j > rk(val_x^\perp(u))$, then there exists $t' \in T_\Gamma$ such that $\perp \otimes t' \in T(\mathcal{A}_r)$,

where $\rho_{\mathcal{A}}$ is the run of $\mathcal{A}_q$ on $x \otimes y$. Lemma 2 implies that it is decidable whether $\tau_{xi,y}(q)$ is defined. Basically, if $q' := \tau_{xi,y}(q)$ is defined, then there exists a fixed (partial) output tree $s' \in S_\Gamma^{yi\circ}$ such that for each input tree $t \in T_\Sigma^x$ there exists some $t' \in T_\Gamma$ such that $t \otimes (s' \cdot t') \in T(\mathcal{A}_q)$.

We define the profile of a labeled path segment xi to be the set that contains all possible state transformations induced by x together with some y of same or smaller length. Formally, let $x \in \mathrm{Path}_\Sigma$ and $i \in \mathrm{dir}_\Sigma$, we define the profile of xi to be $P_{xi} = (P_{xi,=}, P_{xi,<}, P_{xi,\varepsilon})$ with

- $P_{xi,=} := \{\tau_{xi,y} \mid |y| = |x|\}$
- $P_{xi,<} := \{\tau_{xi,y} \mid y \ne \varepsilon \text{ and } |y| < |x|\}$
- $P_{xi,\varepsilon} := \{\tau_{xi,y} \mid y = \varepsilon\}$.

A segment $xi \in (\Sigma\mathrm{dir}_\Sigma)^*\mathrm{dir}_\Sigma$ of a labeled path is called idempotent if $P_{xi} = P_{xixi}$.

As a consequence of Ramsey's Theorem [15], we obtain the next remark.

Remark 7 *There exists a bound $K \in \mathbb{N}$ such that each labeled path $\pi \in \mathrm{Path}_\Sigma$ with $||\pi|| \ge K$ contains an idempotent factor.*

Proof. Ramsey's Theorem yields that for any number of colors c, and any number r, there exists a number $K \in \mathbb{N}$ such that if the edges of a complete graph with at least K vertices are colored with c colors, then the graph must contain a complete subgraph with r vertices such that all edges have the same color, c.f. [16].

Let $\pi \in \mathrm{Path}_\Sigma$ with the factorization $\pi = f_1 j_1 \dots j_{n-1} f_n$, $f_1, \dots, f_n \in \Sigma$ and $j_1, \dots, j_{n-1} \in \mathrm{dir}$. Consider the complete graph $G = (V, E, col)$ with edge-coloring $col : E \to Cols$, where $V := \{f_i j_i \mid 1 \leq i < n\}$, $E := V \times V$, $Cols$ is the finite set of profiles and $col(e) := P_{f_i j_i \dots f_k j_k}$ if $e = (f_i j_i, f_k j_k)$ for all $e \in E$. If there exist $i, j, k \in \mathbb{N}$ with $i < j < k \leq n$ such that the edges $(f_i j_i, f_k j_k)$, $(f_i j_i, f_j j_j)$ and $(f_j j_j, f_k j_k)$ have the same color, i.e.,, the respective profiles are the same, then π has a factorization that contains an idempotent factor.

As a consequence of Ramsey's Theorem, for $r = 3$ and $c = |Cols|$, if $n \geq K$, then π must contain an idempotent factor. ☐

For the rest of this paper we fix how we repeat the part of a tree that contains an idempotent factor in a labeled path segment. Let $x, y \in \mathrm{Path}_\Sigma$, $i, j \in \mathbb{N}$ with $y \neq \varepsilon$ and yj idempotent. For any $t \in T_\Sigma^{xiy}$ we fix t^n to be the tree that results from repeating the idempotent factor n times. More formally, let $path(x) = u$ and $path(y) = v$, we define

$$t^n := \underbrace{t[\circ/ui]}_{s_x} \cdot \underbrace{(t|_{ui}[\circ/uivj])}_{s_y}{}^n \cdot \underbrace{t|_{uivj}}_{\hat{t}} \text{ for } n \in \mathbb{N}.$$

The following Lemma shows that is it decidable whether a relation has a uniformization by a path-recognizable function.

Lemma 8 *For $q \in Q_\mathcal{A}$, $x, y \in \mathrm{Path}_\Sigma$, $i, j \in \mathbb{N}$ with $path(x) = u$, $path(y) = v$, $y \neq \varepsilon$ and yj idempotent, it is decidable whether R_q^{xiy} can be uniformized by a path-recognizable function.*

Proof. First, we show if R_q^{xiy} is uniformized by a path-recognizable function realized by a DTDT $\mathcal{T}$ such that uiv is a prefix of each path that $\mathcal{T}$ reads, then R_q^{xiy} can also be uniformized by a path-recognizable function realized by a DTDT $\mathcal{T}'$ such that $||\mathcal{T}'(t)||^{uiv} \leq |uiv|$ for all $t \in dom(R_q^{xiy})$.

Let $f_\Pi = \bigcup_{k=1}^n (T_{\Pi_k} \cap dom(R_q^{xiy})) \times \{t_k\}$ denote the uniformization of R_q^{xiy} that is realized by $\mathcal{T}$. For any $k \in \{1, \dots, n\}$ such that $||t_k||^{uiv} > |uiv|$ consider for an arbitrary $xiyjz \in \Pi_k$ with $path(z) = w$ an arbitrary tree $t \in T_\Sigma^{xiyjz} \subseteq T_{\Pi_k}$. We choose n such that $||\mathcal{T}(t^n)||^{ui(vj)^n w} < |ui(vj)^n|$. This is always possible, because for each $t \in dom(R_q^{xiy})$ holds $||\mathcal{T}(t)|| \leq max\{||t_1||, \dots, ||t_n||\}$. Let $o = out_\mathcal{T}(t^n, ui(vj)^n w)$, and let $o = o'o''$ such that $||o'|| = ||x||$. Using this factorization, we now construct some $t'_k \in T_\Gamma$ with $||t'_k||^{uivj} \leq |uivj|$. Since yj is idempotent, $P_{yi} = P_{(yi)^n}$, i.e.,, there exists some $m \in \mathrm{Path}_\Gamma$ with $||m|| \leq ||y||$ such that $\tau_{y,m} = \tau_{y^n, o''}$. For $o'm$, let t'_k be the corresponding tree in $T_\Gamma^{o'm}$. Thus, the run ρ_q of $\mathcal{A}_q$ on $t^n \otimes \mathcal{T}(t^n)$ results in the same state at $ui(vj)^n$ as the run ρ_q of $\mathcal{A}$ on $t \otimes t'_k$ at $uivj$. Consequently, $(t, t'_k) \in R_q^{xiy}$.

We now show that $(t', t'_k) \in R_q^{xiy}$ for any other $t' \in T_{\Pi_k}$ and thus it suffices to replace t_k with some suitable t'_k to obtain a uniformization of R_q^{xiy} such that $||t'_k||^{uivj} \leq |uivj|$ for each $k \in \{1, \dots, n\}$. If Π is of the form $L(xi(vj)^+) \cdot \Pi'$, then $\mathcal{T}(t_d) = \mathcal{T}(t_d^n)$ for all $t_d \in dom(R_q^{xiy})$. In this case, since $\mathcal{T}(t) = \mathcal{T}(t')$ we obtain $\mathcal{T}(t^n) = \mathcal{T}(t'^n)$ and it follows $(t', t'_k) \in R_q^{xiy}$. Otherwise, if Π is not of this form, there exists $m \in \mathbb{N}$ such that for each $n > m$ and each tree $t_d \in T_\Sigma^{xiy}$ holds that a configuration of $\mathcal{T}$ for t_d^n is reachable with $\varphi(v) = \varepsilon$ and $v \not\sqsubseteq xi(yj)^n$ with $v \in dom_{t_d^n}$. This means $\mathcal{T}(t_d^n)$ is independent of $\hat{t}_d$ for each $n > m$. Wlog, we can assume that we have chosen $n > m$ in order to find t'_k as alternative output for $t = s_x \cdot s_y \cdot \hat{t}$. Consider z' with $xiyjz' \in \Pi_k$ and an arbitrary tree $t' = s'_x \cdot s'_y \cdot \hat{t}' \in T_\Sigma^{xiyjz'} \subseteq T_{\Pi_k}$. Since $\mathcal{T}\left(s_x \cdot s_y^n \cdot \hat{t}\right) = \mathcal{T}\left(s_x \cdot s_y^n \cdot \hat{t}'\right)$, it follows directly that $(s_x \cdot s_y \cdot \hat{t}', t'_k) \in R_q^{xiy}$, but then also $(s'_x \cdot s'_y \cdot \hat{t}', t'_k) = (t', t'_k) \in R_q^{xiy}$.

Secondly, we describe a process how to check whether there exists such a uniformization. Our requirement is that for an arbitrary tree $t \in dom(R_q^{xiy}) \subseteq T_\Sigma^{xiy}$ only labeled paths beginning with $xiyj$ are relevant. Furthermore, we know that if such a uniformization exists, then it is sufficient to consider output trees in which the length on the path uiv is restricted to $|uiv|$. Therefore, let

$$O := \{o \in \mathrm{Path}_\Gamma \mid path(o) \sqsubseteq path(xiy) \wedge \tau_{xiyj,o}(q) = r \text{ for some } r \in Q_\mathcal{A}\}$$

be the set of possible outputs for xiy of same or shorter length that fulfill this requirement. For each $o \in O$, let t_o denote a corresponding output tree. The set O identifies a finite set of possible output trees. We first check if for each $t \in dom(R_q^{xiy})$ there exists an $o \in O$ such that $(t, t_o) \in R_q^{xiy}$. Hence, let

$$T := \bigcup_{o \in O} \{t \in T_\Sigma \mid (t, t_o) \in R_q^{xiy}\} = \bigcup_{o \in O} \left(R_q^{xiy} \cap (T_\Sigma \times \{t_o\})\right)$$

be the set of trees that are possible input trees for some output tree, and check whether $T = dom(R_q^{xiy})$ holds. If this is the case, we can continue, otherwise there exists no uniformization by a path-recognizable function. This is decidable, because a tree automaton for T can be constructed from an automaton for R_q^{xiy} (using that regular tree languages are closed under intersection and union).

Since the length of the output is restricted, we know that for each t with $(t, t_o) \in R_q^{xiy}$ for some $o \in O$ holds that $(t \times t_o)|_{uivj}$ is of the form $t|_{uivj} \times \bot$ and since $\tau_{xiyj,o}(q) = r \in Q_\mathcal{A}$ we have $t|_{uivj} \times \bot \in T(\mathcal{A}_r)$. Let $\mathcal{A}_r^\bot$ denote the automaton that results from $\mathcal{A}_r$ by removing every transition in which the second component of the letter is not $\bot$. Let

$$P := \bigcup_{o \in O} \{r \in Q_\mathcal{A} \mid \tau_{xiyj,o}(q) = r\}$$

be the set of states that $\mathcal{A}$ can reach at $uivj$ on the input xiy together with some output $o \in O$.

It holds that if each transition in $\mathcal{A}_r^\bot$ has at most one non-trivial successor state, then the output was dependent on an unambiguous path through the input. In this case, let $\Pi_r(\mathcal{A}_r^\bot)$ denote the induced language of unambiguous labeled paths. If for each $r \in P$ this is the case and also the union of all induced labeled paths languages still is a language of unambiguous labeled path, then R_q^{xiy} is uniformizable by a path-recognizable function. $\mathcal{A}_r^\bot$ has at most one non-trivial successor state in each transition if, and only if, $T(\mathcal{A}_r^\bot)$ and $\overline{T(\mathcal{A}_r^\bot)}$ are D↓TA-recognizable (see above). We obtain that R_q^{xiy} can be uniformized by a path-recognizable function such that $xiyj$ is a prefix of each labeled path if

- $T(\mathcal{A}_r^\bot)$ and $\overline{T(\mathcal{A}_r^\bot)}$ are D↓TA-recognizable for all $r \in P$,
- For a DFA $\mathcal{C} = (Q_\mathcal{C}, \Sigma \cup \mathrm{dir}, \Delta_\mathcal{C}, F_\mathcal{C})$ that recognizes $\bigcup_{r \in P} \Pi_r(\mathcal{A}_r^\bot)$ holds for all $q \in Q_\mathcal{C}$ and for all $i, j \in \mathrm{dir}$:

$$\text{If } (q, i, q') \in \Delta_\mathcal{C} \wedge (q, j, q'') \in \Delta_\mathcal{C}, \text{ then } i = j \wedge q = q'.$$

This is decidable, because D↓TA-recognizability for regular tree-languages is decidable, cf. [12]. A DTDT that realizes a path-recognizable function that uniformizes R_q^{xiy} can then easily be obtained from $\mathcal{C}$. □

The following Lemma establishes the connection between long output delay and path-recognizable functions. Basically, the lemma states that if there exists a uniformization by a DTDT such that an idempotent path segment can be repeated any number of times and the length of the output on the repetition is bounded, i.e., the output delay is unbounded, then there also exists a uniformization by a path-recognizable function.

Lemma 9 *Let $q \in Q_\mathcal{A}$, $x, y \in \mathrm{Path}_\Sigma$, $i, j \in \mathbb{N}$ with $path(x) = u$, $path(y) = v$, $y \neq \varepsilon$ and yj idempotent. If R_q^{xiy} is uniformized by a DTDT $\mathcal{T}$ such that $||\mathcal{T}(s_x \cdot s_y^n)||^{ui(vj)^n} \leq |ui|$ for each $t \in T_\Sigma^{xiy}$ and for each $n \in \mathbb{N}$, then R_q^{xiy} can be uniformized by a path-recognizable function.*

Proof. Our goal is to construct a DTDT $\mathcal{T}'$ that realizes a path-recognizable function which uniformizes R_q^{xiy} from $\mathcal{T}$. For our construction we use the following observation. For any $t = s_x \cdot s_y \cdot \hat{t} \in T_\Sigma^{xiy}$ holds, that if the idempotent factor is repeated often enough, say n times, then $||\mathcal{T}(s_x \cdot s_y^n \cdot \hat{t})||^{ui(vj)^n} = ||\mathcal{T}(t)^n||^{ui(vj)^n} \leq |ui(vj)^n|$.

We assume that $\mathcal{T}$ reaches a configuration $c = (t^n, t', \varphi)$ such that there exists $u' \in D_{t'}$ with $\varphi(u') = ui(vj)^n$ for each $t \in T_\Sigma^{xiy}$ and each $n \in \mathbb{N}$. Otherwise, there exists $m \in \mathbb{N}$, such that for each $t \in T_\Sigma^{xiy}$ holds $\mathcal{T}(t^n)$ is independent of $\hat{t}$ for all $n \geq m$. In this case we can choose the same output tree for all input trees as seen in the proof of Lemma 8.

The idea is that we construct $\mathcal{T}'$ such that for the input tree t the transducer simulates $\mathcal{T}$ on t^n. Since the output which is mapped to the repetition of the idempotent factor, that is to the path $ui(vj)^n$, is shorter than $ui(vj)^n$, we can keep track which path w in $\hat{t}$ is read by $\mathcal{T}$ to produce output on $ui(vj)^n$. We will see that $uivjw$ is the path that $\mathcal{T}'$ has to read in t in order to determine a matching output tree t' for t.

For this purpose, we encode in the state space of $\mathcal{T}'$ the current state of $\mathcal{T}$ at the read input symbol in $\hat{t}$, the current state of $\mathcal{A}$ as well as the current position in x resp. in the repetition of y to which the next output of $\mathcal{T}$ is mapped. Since the current position in x resp. (in a repetition of) y is known, the direction in which the input has to be pursued can be determined from the applied transition of $\mathcal{T}$.

Let x be of the form $f_1 i_1 f_2 i_2 \dots i_{m-1} f_m$ and y be of the form $g_1 j_1 g_2 j_2 \dots j_{n-1} g_n$. First, for any input tree $t \in T_\Sigma^{xiy}$, $\mathcal{T}'$ reads xiy and then continues in direction j and switches to a state $(s_s, q_s, \pi_s d_s)$ in the new state space $Q_\mathcal{T} \times Q_\mathcal{A} \times \{\pi \in \mathrm{Path}_\Sigma \mid ||\pi|| \leq ||xiy||\} \cdot \mathrm{dir}_\Sigma$ of $\mathcal{T}'$. The components $s_s \in Q_\mathcal{T}$, $q_s \in Q_\mathcal{A}$ and $\pi_s d_s \in \mathrm{Path}_\Sigma \cdot \mathrm{dir}_\Sigma$ are chosen as follows:

a) Let s_n denote the state that $\mathcal{T}$ reaches after reading $ui(vj)^n$, $n \in \mathbb{N}$. For $n > |Q_\mathcal{T}|$, there has to be a state that occurs again in the sequence $s_1 s_2 \dots$. Let $s_s \in Q_\mathcal{T}$ be such a state.

b) Since $|out_\mathcal{T}(s_x \cdot s_y^n, ui(vj)^n)| \leq ui$ for each $n \in \mathbb{N}$, there is $m \in \mathbb{N}$ such that $out_\mathcal{T}(s_x \cdot s_y^n, ui(vj)^n)$ $= out_\mathcal{T}(s_x \cdot s_y^{n+1}, ui(vj)^{n+1})$ for all $n \geq m$. Let $||out_\mathcal{T}(s_x \cdot s_y^m, ui(vj)^m)|| = l$. Let $q_s \in Q_\mathcal{A}$ with $\mathcal{A} : q \xrightarrow{f_1 i_1 f_2 i_2 \dots i_{l-1} f_l \otimes out_\mathcal{T}(s_x \cdot s_y^m, ui(vj)^m)}_{i_l} q_s$.

c) Let $\pi_s d_s := f_{l+1} i_{l+1} \dots i_{m-1} f_m iyj$.

After the switch to $(s_s, q_s, \pi_s d_s)$, the transitions of $\mathcal{T}'$ look as follows (based on $\mathcal{T}$):

d) $\mathcal{T}$ produces no output and reaches s' in direction j_1, $\mathcal{T}'$ continues to read:

$$(s, q, \pi d)\big(f(x_1, \dots, x_i)\big) \to (s', q, \pi d)(x_{j_1}),$$

if $s\big(f(x_1, \dots, x_i)\big) \to s'(x_{j_1}) \in \Delta_\mathcal{T}$.

e) $\mathcal{T}$ produces output that is mapped to $i_1 \dots i_{k-1}$ and reaches s' in direction i_k, $\mathcal{T}'$ continues to read:

$$(s, q, f_k i_k \dots i_{m-1} f_m iy)\big(f(x_1, \dots, x_i)\big) \to (s', q', f_{k+1} i_{k+1} \dots i_{m-1} f_m iy)(x_{i_k}),$$

if $s\big(f(x_1, \dots, x_i)\big) \to g(\dots, s'(x_{i_k}), \dots) \in \Delta_\mathcal{T}$ and $\mathcal{A} : q \xrightarrow{f_k \otimes g}_{i_k} q'$ for $l \leq k \leq m$.

f) $\mathcal{T}$ produces output that is mapped to $ui(vj)^n j_1 \dots j_{k-1}$ for some n and reaches s' in direction j_k, $\mathcal{T}'$ continues to read:

$(s, q, \underbrace{g_k j_k \dots j_{n-1} g_n}_{y'} j \underbrace{g_1 j_1 \dots g_{k-1} j_{k-1}}_{y'' \text{ with } yj = y''y'j})\big(f(x_1, \dots, x_i)\big) \to (s', q', g_{k+1} j_{k+1} \dots j_{n-1} g_n j g_1 j_1 \dots g_k j_k)(x_{j_k})$, if $s\big(f(x_1, \dots, x_i)\big) \to g(\dots, s'(x_{j_k}), \dots) \in \Delta_\mathcal{T}$ and $\mathcal{A} : q \xrightarrow{g_k \otimes g}_{j_k} q'$ for $1 \leq k \leq m$.

g) Finally, when a leaf is reached $\mathcal{T}'$ produces an output tree:

$$r := (s, q, \pi d)(a) \to t'_{ra},$$

where $t'_{ra} \in T_\Gamma$ is a tree chosen as explained below.

Let Π_{ra} denote the set of labeled paths that $\mathcal{T}'$ can read to reach a leaf a at state r. We enumerate all reachable combinations from 1 to n. We show that there exist trees $t'_1, \dots, t'_n \in T_\Gamma$ such that $\mathcal{T}'$ realizes a uniformization of R_q^{xiy}. Consider an arbitrary $\pi_k \in \Pi_k$ and an arbitrary tree $t_k \in T_\Sigma^{\pi_k}$. Choose n_1 such that $||\mathcal{T}(t_k^{n_1})|| \leq |ui(vj)^{n_1}|$ and a configuration $c = (t_k^{n_1}, t', \varphi)$ of $\mathcal{T}$ such that $\varphi(path(\pi_s) d_s) = ui(vj)^{n_1}$ and $val_{t'}(path(\pi_s) d_s) = s_s$ is reached. Let $x = x' j_l x''$ with $x' = f_1 i_1 \dots i_{l-1} f_{l-1}$, $x'' = f_l i_1 \dots i_{m-1} f_m$ with $f_1, \dots, f_m \in \Sigma$ and $i_1, \dots, i_{m-1} \in \mathrm{dir}_\Sigma$. For $r_k = (s, q_2, g_{i+1} j_{i+1} \dots g_i j_i, a)$ and $path(\pi_k) = w$ consider the factorization $out_\mathcal{T}(t_k^{n_1}, ui(vj)^{n_1} w) = o_1 o_2 o_3 o_4$ such that $|o_1| = |x'|$, $|o_2| = |x''|$, and $|o_3| = |(yj)^{n_1 - 1} g_1 \dots g_i|$. The run of $\mathcal{A}$ on $t_k^{n_1} \otimes \mathcal{T}(t_k^{n_1})$ has the following property:

$$\mathcal{A} : q \xrightarrow{x'\otimes o_1}_{i_l} q_s \xrightarrow{x''\otimes o_2}_{i} q_1 \xrightarrow{(yj)^{n_1-1}g_1\cdots g_i\otimes o_3}_{j_i} q_2 \xrightarrow{g_{i+1}\cdots g_n\otimes o_4}_{j} q_3 \xrightarrow{\pi_k\otimes\varepsilon} F_{\mathcal{A}}.$$

Since yj is idempotent, we can choose some $\tau_{yj,o}$ such that $\tau_{(yj)^{n_1},o_3o_4} = \tau_{yj,o}$. Hence, we choose t'_k to be a tree compatible to $\tau_{yj,o}$. Therefore, the run of $\mathcal{A}$ on $t_k \otimes t'_k$ looks as follows:

$$\mathcal{A} : q \xrightarrow{x'\otimes o_1}_{i_l} q_s \xrightarrow{x''\otimes o_2}_{i} q_1 \xrightarrow{y\otimes o}_{j} q_3 \xrightarrow{\pi_k\otimes\varepsilon} F_{\mathcal{A}}.$$

Now, for an arbitrary $t \in T_{\Pi_k}$ with $t \in T_\Sigma^{xiyj\pi}$ we choose a suitable n_2 such that $||\mathcal{T}(t^{n_2})|| \leq |ui(vj)^{n_2}|$ and a configuration $c = (t^{n_2}, t', \varphi)$ of $\mathcal{T}$ such that $\varphi(path(\pi_s)d_s) = ui(vj)^{n_2}$ and $val_{t'}(path(\pi_s)d_s) = s_s$ is reached. Let $path(\pi) = w'$, and consider the output of $\mathcal{T}$ on the path $ui(vj)^{n_2}w'$. The beginning and the end of the output on this path are the same as for $t_k^{n_1}$, because in $\mathcal{C}$ reading π leads to the same state as π_k, which means $\mathcal{T}$ produces the same final output at the leaf. Thus, we have $out_{\mathcal{T}}(t^{n_2}, ui(vj)^{n_2}w') = o_1o'_2o'_3o_4$ with $|o'_2| = |o_2|$. Meaning the run of $\mathcal{A}$ on $t^{n_2} \otimes \mathcal{T}(t^{n_2})$ looks as follows:

$$\mathcal{A} : q \xrightarrow{x'\otimes o_1}_{i_l} q_s \xrightarrow{x''\otimes o'_2}_{i} q'_1 \xrightarrow{(yj)^{n_2-1}g_1\cdots g_i\otimes o'_3}_{j_i} q_2 \xrightarrow{g_{i+1}\cdots g_n\otimes o_4}_{j} q_3 \xrightarrow{\pi\otimes\varepsilon} F_{\mathcal{A}}.$$

We see, that $(x''i(yj)^{n_2-1}y, o'_2o'_3o_4)$ induces the same state transformation on $\mathcal{A}$ from q_s as $(x''iy, o_2o)$. Hence, the run of $\mathcal{A}$ on $t \otimes t'_k$ results in

$$\mathcal{A} : q \xrightarrow{x'\otimes o_1}_{i_l} q_s \xrightarrow{x''\otimes o_2}_{i} q_1 \xrightarrow{y\otimes o}_{j} q_3 \xrightarrow{\pi\otimes\varepsilon} F_{\mathcal{A}},$$

i.e., $(t, t'_k) \in R_q^{xiy}$. □

As we have seen, if a transducer that uniformizes a relation introduces long output delay, then the relation can also be uniformized by a path-recognizable function.

Now that we have completed all preparations, we present a decision procedure for the case of unbounded delay. Therefore, we consider a similar safety game as in the previous section on uniformization with bounded output delay. We only have to adapt the game graph if the input sequence is ahead K steps. Let $\mathcal{G}_{\mathcal{A}}^K$ denote the modified game. From each vertex $(q, \pi) \in V_{\mathsf{Out}}$ with $||\pi|| = K$ we add a move that allows Out to stay in this vertex if there exists a factorization of $\pi = xiyjz$ with $x, y, z \in \mathrm{Path}_\Sigma$, $i, j \in \mathrm{dir}$ and yj is idempotent such that R_q^{xiy} can be uniformized by a path-recognizable function without input validation. These changes to the game graph can be made, because if the input is K steps ahead, then there exists a factorization of the input sequence that contains an idempotent factor and Lemma 8 implies that it is decidable whether there exists a corresponding uniformization by a path-recognizable function.

Lemma 10 *R has a uniformization if, and only if, Out has a winning strategy in the safety game $\mathcal{G}_{\mathcal{A}}^K$.*

Proof. Assume that Out has a winning strategy in $\mathcal{G}_{\mathcal{A}}^K$, then there also exists a positional winning strategy for Out. To construct a DTDT $\mathcal{T}$ that uniformizes R, we proceed as presented in the proof of Lemma 5 with one addition. We construct for each $(q, \pi) \in V_{\mathsf{Out}}$ such that $||\pi|| = K$ and there is $\pi = xiyjz$ such that R_q^{xiy} can be uniformized by a path-recognizable function, a DTDT $\mathcal{T}_q^{xiy}$ that uniformizes R_q^{xiy}. In $\mathcal{T}$ we switch to $\mathcal{T}_q^{xiy}$ at the respective states.

For the other direction, assume that R is uniformized by some DTDT $\mathcal{T}$. Again, the proof is similar to the proof of Lemma 6. Thus, we only describe how the strategy is chosen if the output delay in $\mathcal{T}$ exceeds K. If the play reaches a vertex $(q, \pi) \in V_{\mathsf{Out}}$ with $||\pi|| = K$, there is a factorization of $\pi = xiyjz$ with $x, y, z \in \mathrm{Path}_\Sigma$, $i, j \in \mathrm{dir}$ such that yj is idempotent. Let $path(x) = u, path(y) = v$ and $path(z) = w$. Let $\mathcal{T}_s$ uniformize R_q and pick any $t \in T_\Sigma^\pi$, if $||out_{\mathcal{T}_s}(t^n = s_x \cdot s_y^n \cdot \hat{t}, ui(vj)^n)|| < ui(vj)^n$ for all $n \in \mathbb{N}$, then Lemma 9 implies that R^{xiy} can be uniformized by path-recognizable function. In this case, Out stays in this vertex from then on and wins.

Otherwise, there exists $m \in \mathbb{N}$ such that $||out_{\mathcal{T}_s}(s_x \cdot s_y^m \cdot \hat{t}, ui(vj)^m)|| \geq ui(vj)^m$. Consider the factorization of $out_{\mathcal{T}_s}(s_x \cdot s_y^m \cdot \hat{t}, ui(vj)^m) = o_1io_2jo_3$ such that $|o_1i| = |xi|$ and $|o_2j| = (yj)^m$. Since yj is idempotent we can choose some o of length K such that $\mathcal{A} : q \xrightarrow{xiy\otimes o_1o}_{j} q'$ and $\mathcal{A} : q \xrightarrow{xi(yj)^{m-1}y\otimes o_1o_2}_{j} q''$ with $q' = q''$. Then Out makes K moves according to o leading to some $(q', z) \in V_{\mathsf{Out}}$. From there, Out takes the transitions according to o_3. □

As a consequence of Lemma 10 and the fact that a winning strategy for Out in $\mathcal{G}_{\mathcal{A}}^K$ can effectively be computed we immediately obtain our main result.

Theorem 11 *It is decidable whether a D↓TA-recognizable relation with total domain has a uniformization by a deterministic top-down tree transducer.*

As mentioned in the beginning, the presented results are also valid for D↓TA-recognizable relations with D↓TA-recognizable domain in the sense that a TDT that realizes a uniformization of a relation may behave arbitrarily on trees that are not part of the domain. The presented constructions have to be adapted such that In, given a D↓TA for the domain, also keeps track of the state in the input tree in order to play only correct input symbols.

3.3. Input validation

In the former section, we assumed that a top-down tree transducer that implements a uniformization of a relation is only given valid input trees. In this section we consider the case that a top-down transducer also has to validate the correctness of a given input tree.

We will see that in this case it can be necessary that a transducer takes divergent paths for input and output. The following example shows that there exists a D↓TA-recognizable relation with D↓TA-recognizable domain that can be uniformized by a DTDT, but every such DTDT has a reachable configuration (t, t', φ) such that $\varphi(u) \not\sqsubseteq u$ and $u \not\sqsubseteq \varphi(u)$ for some node u.

Example 12 Let Σ be given by $\Sigma_2 = \{f\}$ and $\Sigma_0 = \{a, b\}$. We consider the relation $R_1 \subseteq T_\Sigma \times T_\Sigma$ defined by $\{(f(b,t), f(t',b)) \mid \neg\exists u \in dom_t : val_t(u) = b\}$. Clearly, both R_1 and $dom(R_1)$ are D↓TA-recognizable. Intuitively, a DTDT $\mathcal{T}$ that uniformizes R_1 must read the whole right subtree $t|_2$ of an input tree t to verify that there is no occurrence of b. If an f in $t|_2$ is read and no output is produced, a DTDT can either continue to read left or right, but cannot verify both subtrees. Therefore, in order to verify $t|_2$, a DTDT has to produce an output symbol at each read inner node which results in an output tree of the same size. Clearly, the relation R_1 is uniformized by the following DTDT $\mathcal{T} = (\{q_0, q_1, q_2\}, \Sigma, \Sigma, q_0, \Delta)$ with $\Delta =$

$$\begin{aligned}\{ \quad & q_0(f(x_1,x_2)) \to f(q_1(x_2), q_2(x_1)), \quad && q_1(a) \to b, \\ & q_1(f(x_1,x_2)) \to f(q_1(x_1), q_1(x_2)), \quad && q_2(b) \to b \quad \}.\end{aligned}$$

However, there exists no DTDT $\mathcal{T}'$ that uniformizes R_1 such that the read input sequence and the produced output are on the same path. Assume such a DTDT $\mathcal{T}'$ exists, then for an initial state q_0 there is a transition of the form $q_0(f(x_1,x_2)) \to f(q_1(x_1), q_2(x_2))$. It follows that $\mathcal{T}'_{q_2}$ must induce the relation $\{(t,b) \mid t \in T_\Sigma \wedge \neg\exists u \in dom_t : val_t(u) = b\}$. The only output that $\mathcal{T}'_{q_2}$ can produce is exactly one b. Thus, there is a transition with left-hand side $q_2(f(x_1,x_2))$ that has one of the following right-hand sides: b, $q_3(x_1)$, or $q_3(x_2)$. No matter which right-hand side is chosen, $dom(R(\mathcal{T}'_{q_2}))$ must also contain trees with occurrences of b. ◁

It follows directly from the above example that the presented decision procedure is invalid if the domain of a considered relation is not total. However, if we restrict ourselves to uniformizations such that a DTDT only contains rules of the form $q(f(x_1,\dots,x_i)) \to w[q_1(x_{j_1}),\dots,q_n(x_{j_n})]$, where $w \in \Gamma(X_n)$ i.e., read input symbol and correspondingly produced output begin always on the same tree level, it is possible to adapt the presented decision procedure from Section 3.1. We refer to this kind of DTDTs as DTDTs without delay.

Theorem 13 *It is decidable whether a D↓TA-recognizable relation with D↓TA-recognizable domain has a uniformization by a deterministic top-down tree transducer without delay.*

For this purpose, we can change the game graph in the following way. Let $\mathcal{A}$ be a D↓TA for a relation and $\mathcal{B}$ be a D↓TA for its domain. The main differences to the previous section is that the vertices in the game graph keep track of the current state of $\mathcal{B}$ on the input sequence played by In and keep track of the state of $\mathcal{A}$ on the combined part of all possible input sequences and the current output sequence of Out which is not necessarily the same as the input sequence played by In. The move constraints for Out will be chosen such that it is guaranteed that the input sequence is valid, and the combined part of all possible input sequences together with her output sequence is valid. Details for this construction can be found in [17].

4. Non-deterministic Top-down Specifications

In Section 3, we considered relations that are deterministic top-down tree automaton-recognizable. In the remainder we consider non-deterministic tree automatic relations.

We face two difficulties. Firstly, not every non-deterministic tree automatic specification has a representation by a deterministic top-down tree automaton. Thus, synthesis methods based on a deterministic representation of a given specification can not be applied. Secondly, in Section 3.3 we have seen that already for deterministic top-down specifications it is necessary that a uniformizer takes divergent paths for read input and produced output.

In order to solve the uniformization problem for non-deterministic top-down specifications to some extent we present two approaches. In Section 4.1, we consider a restricted class of non-deterministic top-down specifications and ask whether there is a uniformizer without delay, whereas in Section 4.2 we consider general non-deterministic top-down specifications and restrict the uniformizer.

4.1. Union of top-down deterministic specifications

In this section, we assume that $R \subseteq T_\Sigma \times T_\Gamma$ is given as the union $\bigcup_{i=1}^{n} R_i$ of n relations with pairwise disjoint domains, where each R_i is recognized by a D↓TA $\mathcal{A}_i$ and its domain is recognized by a D↓TA $\mathcal{B}_i$. Furthermore, we assume that the domain of the relation is D↓TA-recognizable, otherwise there exists no uniformization by a deterministic top-down tree transducer. It is decidable whether a tree automatic language is D↓TA-recognizable, see [12], and if possible, a D↓TA can be obtained from a given N↓TA for a tree automatic language. Let $dom(R)$ be recognized by a D↓TA $\mathcal{D}$.

We start by giving an example for a non-deterministic top-down specification that can be defined as the union of deterministic top-down specifications. We will see, that a uniformizer without delay for this example specification has to be non-linear, i.e., has to copy a subtree.

Example 14 Let Σ be an input alphabet given by $\Sigma_1 = \{h\}$ and $\Sigma_0 = \{c, d\}$ and let Γ be an output alphabet given by $\Gamma_2 = \{f\}$, $\Gamma_1 = \{h\}$ and $\Gamma_0 = \{c, d\}$. We consider the relation $R \subseteq T_\Sigma \times T_\Gamma$ defined by $\{(h(t), f(t, t')) \mid t, t' \in T_\Sigma$ such that t and t' have the same leaf symbol$\}$.

This specification can be obtained by the union of two deterministic top-down specifications, one specification for each leaf symbol. Clearly, a deterministic top-down transducer can realize the specification by producing $f(t, t)$ for a unary input tree $h(t)$, e.g., by starting with $q_0(h(x_1)) \to f(q(x_1), q(x_1))$.

However, there is no linear uniformizer without delay for R, because in the first step a linear DTDT would have to pick for either the right or the left subtree an output tree with a fixed leaf symbol. As the actual leaf symbol of the input tree is yet unknown it is not possible to fix a correct output tree. $\lhd$

We will provide a decision procedure for the following problem.

Theorem 15 *It is decidable whether the union of D↓TA-recognizable relations with pairwise disjoint D↓TA-recognizable domains has a uniformization by a deterministic top-down tree transducer without delay.*

Previously, in Section 3, we used two-player games to obtain our results on uniformization of D↓TA-recognizable specifications. Recall the game $\mathcal{G}_{\mathcal{A}}^k$ presented in Section 3.1. Deciding whether there exists a uniformizer for a specification given by a D↓TA $\mathcal{A}$ reduces to deciding the winner in $\mathcal{G}_{\mathcal{A}}^k$. For $k = 0$, this corresponds to deciding whether there is a uniformizer without delay for the specification. Here, we are asking whether there is a uniformizer without delay, but for an N↓TA-recognizable specification. In $\mathcal{G}_{\mathcal{A}}^k$, in a play between In and Out, the moves of In correspond to an input sequence – a labeled path through an input tree – and the moves of Out to the correspondingly produced output sequence. Accordingly, a winning strategy in $\mathcal{G}^k_{\mathcal{A}}$ for Out means that for every path through an input tree valid output can be produced. Given a top-down deterministic specification, a winning strategy then indeed corresponds to a uniformizer. However, here we consider non-deterministic specifications. Hence, we can not reason over labeled input paths individually, but have to reason over input trees. Therefore, instead of a game but similarly to $\mathcal{G}^k_{\mathcal{A}}$, we want to define a regular infinite tree [9], given as the unfolding of a finite graph, that allows us to view output strategies in relation to input trees.

Unlike in $\mathcal{G}^k_{\mathcal{A}}$, we have to model that a uniformizer can switch and copy subtrees. As a result, output at different positions in the output tree can be depended on the same position in the input tree. Consequently, the number of output choices that depend on an input position is unbounded as the size of the input trees is generally unbounded. An infinite tree that includes for every input sequence every possibly dependent output choice would have to use infinitely many different labels.

To overcome this problem, we will give a construction for a regular infinite tree where the number of output choices that depend on a single input position can be bounded. To achieve this we make use of the specification automata. Naturally, the number of state transformations that different output sequences together with an input sequence can induce in an automaton is bounded. Our construction will take this into account. First, we specify a finite graph based on the specification automata and then obtain a regular infinite tree by using its unfolding.

Before we formally define the graph, we describe its components. We want to keep track of the state of $\mathcal{D}$ on the input, and also keep track of the states of $\mathcal{A}_1, \dots, \mathcal{A}_n$ on the combined part of input and output. For the latter we will use vectors with n elements. We define a function τ_ℓ that returns the ℓth element of a vector, for each $1 \leq \ell \leq n$. Let L denote such a vector, then $\tau_\ell(L)$ stores the information w.r.t. $\mathcal{A}_\ell$. However, it is necessary to model that input and output can be on divergent paths. This is done by distinguishing two cases. In case that input and output are on the same path, $\tau_\ell(L)$ is the state of $\mathcal{A}_\ell$ on the combined input sequence and output sequence. In case that the output is mapped to a divergent path, $\tau_\ell(L)$ is a set of states of $\mathcal{A}_\ell$ that is obtained by combining all possible input sequences with the produced output sequence. Now we are ready to formally define the graph $\mathcal{G}$:

- From a vertex v of the form $(p, \{L_1, \dots, L_m\})$, where p is a state of $\mathcal{D}$ and each L_j is a vector of states resp. sets of states over $\mathcal{A}_1, \dots, \mathcal{A}_n$, the following edges exist:
  - $v \to (v, f)$ if there is a transition $(p, f, p_1 \dots, p_i) \in \Delta_{\mathcal{D}}$

    (edges for every possible input symbol)
- From a vertex (v, f) with $f \in \Sigma_i$ the following edges exist:
  - $((p, \{L_1, \dots, L_m\}), f) \overset{o_1,\dots,o_m}{\to} [(p_1, Q_1), \dots, (p_i, Q_i)]$ if the following conditions hold:
    * $(p, f, p_1 \dots, p_i) \in \Delta_{\mathcal{D}}$,
    * $o_j \in \Gamma(X_i)$ for each $1 \leq j \leq m$, and

      (for each of the L_j an output o_j consisting of one output symbol and directions to continue is chosen)
    * the sets $Q_1, \dots, Q_i$ are constructed as follows: for $o_j = g(x_{j_1}, \dots, x_{j_r})$, add L^j_k to Q_{j_k} for each $1 \leq k \leq r$, where L^j_k is build up as follows:
      · if $\tau_\ell(L_j) \in Q_{\mathcal{A}_\ell}$, say $q \in Q_{\mathcal{A}_\ell}$, and there is $(q, (f, g), q_1, \dots, q_{max\{rk(f), rk(g)\}}) \in \Delta_{\mathcal{A}_\ell}$,

      $$\text{then } \tau_\ell(L^j_k) = \begin{cases} q_k & \text{if } j_k = k, \quad \text{(input and output continue in the same direction)} \\ \{q_k\} & \text{otherwise. (input and output continue in divergent directions)} \end{cases}$$

      (input and output are at the same position; the corresponding transition in $\mathcal{A}_\ell$ is applied, for $1 \leq \ell \leq n$)

      · if $\tau_\ell(L_j) \in 2^{Q_{\mathcal{A}_\ell}}$, set $\tau_\ell(L^j_k)$ to $\emptyset$ and for each $q \in \tau_\ell(L_j)$ and each $f' \in \Sigma$ such that there is $(q, (f', g), q_1, \dots, q_{max\{rk(f'), rk(g)\}}) \in \Delta_{\mathcal{A}_\ell}$, add q_k to $\tau_\ell(L^j_k)$.

(input and output are on divergent paths; all possibly applicable transitions in $\mathcal{A}_\ell$ are applied and the reachable states are collected, for $1 \leq \ell \leq n$)

- From $[v_1, \dots, v_i]$ an edge to v_j exists for all $1 \leq j \leq i$. (edges to all directions)
- The initial vertex is $(p_0, \{L\})$, where $L = [q_0^{\mathcal{A}_1}, \dots, q_0^{\mathcal{A}_n}]$ and p_0 is the initial state of $\mathcal{D}$.

As we can see, storing the resulting state vectors in a set bounds the number of output choices that have to be made in a vertex.

Now that we have defined $\mathcal{G}$, we consider the unfolding $\mathcal{H}$ of $\mathcal{G}$ which is a regular infinite tree. Consequently, each vertex of $\mathcal{H}$ is associated with a labeled path, interpreted as an input sequence π, and additionally it is associated with a bounded number of labeled paths, interpreted as output sequences produced by a transducer while reading the input sequence π. Note that different vertices of $\mathcal{H}$ may represent the same input sequence, but differ in the associated output sequences. This is a regular infinite tree that has the desired property, namely, each input sequence together with a (sufficiently large) number of possible output sequences is represented in the tree.

Our goal is to construct a parity tree automaton, whose tree language is non-empty if, and only if, R has a uniformization by a deterministic top-down tree transducer without delay. For an introduction to parity tree automata, see e.g. [9]. The idea is to annotate $\mathcal{H}$ with an output strategy s. The strategy selects for each node of the form (v, f) with $f \in \Sigma$ one child, i.e., s fixes an output choice. Let $\mathcal{H}^\frown s$ denote the tree $\mathcal{H}$ with annotations encoding s. Given $\mathcal{H}^\frown s$ and some input tree $t \in \mathit{dom}(R)$, then we can identify a unique output tree, that a DTDT would produce while reading t using the output choices defined by s. For an input tree t, let $s(t)$ denote the corresponding output tree. The strategy s corresponds to a uniformizer if for all $t \in \mathit{dom}(R)$ holds that $(t, s(t)) \in R$. The following lemma shows that the set of trees $\mathcal{H}^\frown s$ such that s corresponds to a uniformizer is a regular set of trees.

Lemma 16 *There exists a parity tree automaton $\mathcal{C}$ that accepts exactly those trees $\mathcal{H}^\frown s$ such that $(t, s(t)) \in R$ for all $t \in \mathit{dom}(R)$.*

Proof. We show that there exists a parity tree automaton that accepts exactly those trees $\mathcal{H}^\frown s$ for which s is a strategy such that there exists an input tree $t \in \mathit{dom}(R)$ and it holds $(t, s(t)) \notin R$. Then, by closure properties of regular tree languages, we can obtain a parity tree automaton that accepts exactly those trees $\mathcal{H}^\frown s$ such that $(t, s(t)) \in R$ for all $t \in \mathit{dom}(R)$. That is, $\mathcal{H}^\frown s$ is accepted if, and only if, s corresponds to a uniformizer without delay.

A parity tree automaton, say $\bar{\mathcal{C}}$, that accepts exactly those trees $\mathcal{H}^\frown s$ for which there is $t \in \mathit{dom}(R)$ such that $(t, s(t)) \notin R$, i.e., $(t, s(t)) \notin R_i$ for all i, is constructed as follows. Recall that $\mathit{dom}(R)$ is the union of pairwise disjoint domains. This means, in order to show that s does not correspond to a uniformizer, it is sufficient to show for some i that there is $t \in \mathit{dom}(R_i)$ such that $(t, s(t)) \notin R_i$. Hence, $\bar{\mathcal{C}}$ first guesses an i for which there is a tree $t \in \mathit{dom}(R_i)$ such that $\mathcal{A}_i$ does not accept $t \otimes s(t)$. Since $\mathcal{A}_i$ is deterministic, we can prove the existence of such a tree by guessing a path where the deterministic run of $\mathcal{A}_i$ on the combined input and output specified by s fails.

However, since s may be chosen such that the paths of input and output sequence diverge, to show that the run fails, $\bar{\mathcal{C}}$ operates as follows. The parity tree automaton wants to simulate the run of $\mathcal{A}_i$ on the output choices defined by s that lead to a position where the run fails. Note that although s fixes the output choices, more than one output sequence might be associated with the same input sequence. Thus, $\bar{\mathcal{C}}$ has to guess the input sequence that leads to the output choices fixed by s causing a non-accepting run, and also, it has to pick the output sequence among the associated output sequences that will be used to simulate a run of $\mathcal{A}_i$. For this purpose, $\bar{\mathcal{C}}$ guesses an input sequence and follows the vertices that correspond to choosing this labeled path in $\mathcal{H}^\frown s$. As long as the path of the guessed input sequence is the same path as the output sequence under consideration, $\bar{\mathcal{C}}$ simulates the run of $\mathcal{A}_i$ on the combined input and output on that path. The idea is explained a bit more formally below. Let x resp. y denote the input resp. output sequence up to last position where both share the same path, let $\mathit{path}(x) = \mathit{path}(y) = u$. Thus, $\bar{\mathcal{C}}$ is in a vertex in $\mathcal{H}^\frown s$ that corresponds to the input sequence x and output sequence y and has simulated the run of $\mathcal{A}_i$ on $x \otimes y$. Now, we describe how we proceed if input and output diverge, i.e., $\bar{\mathcal{C}}$ continues the input sequence in direction i and the associated output sequence under consideration is continued in direction j with $i \neq j$. Let x' resp. y' denote the continuation of the input resp. output sequence. Furthermore, let $path(x') = iv$ and $path(y') = jv'$. Then, in addition to following x' in $\mathcal{H}^\frown s$, $\bar{\mathcal{C}}$ chooses the input labels along ujv' beginning from uj. Let x'' denote the resulting labeled path with $path(x'') = jv'$. The labels have to be chosen such that both xx' and xx'' can be part of a tree from $dom(R_i)$. Let $\mathcal{A}_i : q_0^{\mathcal{A}_i} \xrightarrow{x \otimes y}_j q$, that is, q is the result of the simulation of the run of $\mathcal{A}_i$ so far. In the process of following x' in $\mathcal{H}^\frown s$, $\bar{\mathcal{C}}$ simulates the run of $\mathcal{A}_i$ on $x'' \otimes y'$ starting from q. Then, $\bar{\mathcal{C}}$ accepts if the run fails, i.e., $\mathcal{A}_i : q_0^{\mathcal{A}_i} \xrightarrow{x \otimes y}_j q \xrightarrow{x'' \otimes y'} Q_{\mathcal{A}_i} \setminus F_{\mathcal{A}_i}$. This means, there exists some $t \in dom(R_i) \cap T_\Sigma^{xx'} \cap T_\Sigma^{xx''}$ such that $(t, s(t)) \notin R_i$, and consequently, $(t, s(t)) \notin R$.

It follows that $\bar{\mathcal{C}}$ accepts exactly those trees $\mathcal{H}^\frown s$ for which there is $t \in dom(R)$ such that $(t, s(t)) \notin R$. A formal construction of $\bar{\mathcal{C}}$ is omitted. The desired parity tree automaton $\mathcal{C}$ can then be obtained from $\bar{\mathcal{C}}$ using complementation and intersection. □

The next lemma shows that the uniformization problem posed in this section reduces to deciding the emptiness problem for $\mathcal{C}$.

Lemma 17 *The tree language $T(\mathcal{C})$ is non-empty if, and only if, R has a uniformization by a deterministic top-down tree transducer without delay.*

Proof. If the tree language $T(\mathcal{C})$ is non-empty, then there exists at least one regular tree $\mathcal{H}^\frown s$ that is accepted by $\mathcal{C}$. From such an $\mathcal{H}^\frown s$ we can construct a DTDT without delay that uniformizes R as follows. Given a vertex from $\mathcal{H}^\frown s$ of the form (p, Q) with $L \in Q$, then (p, L) is used as a state of the uniformizer. The transition rules have to be constructed such that they correspond to the output choices specified by s. This is, if $((p, \{L_1, \ldots, L_m\}), f) \overset{o_1,\ldots,o_m}{\rightarrow} [(p_1, Q_1), \ldots, (p_i, Q_i)]$ is selected by s in $\mathcal{H}^\frown s$, then for each $1 \leq j \leq m$ we add the rule $(p, L_j)(f(x_1, \ldots, x_i)) \to g((p_{j_1}, L_1^j)(x_{j_1}), \ldots, (p_{j_r}, L_r^j)(x_{j_r}))$ to the transitions, where $o_j = g(x_{j_1}, \ldots, x_{j_r})$ and $L_1^j, \ldots, L_r^j$ as described in the construction of $\mathcal{G}$ above.

Conversely, assume that R has a uniformization by a DTDT $\mathcal{T}$ without delay. We have to show that $T(\mathcal{C})$ is non-empty. Therefore, it is sufficient to define s in correspondence to $\mathcal{T}$, clearly $\mathcal{C}$ then accepts $\mathcal{H}^\frown s$.
□

As a consequence of the above lemma, Theorem 15 follows directly.

4.2. General non-deterministic top-down specifications

In the last section, we considered the special case of non-deterministic tree relations that can be obtained by union of deterministic ↓TA-recognizable relations with disjoint domains and had little restriction on the uniformizer. In this section, we consider general non-deterministic tree relations and restrict the uniformizer more strongly. We are looking for a uniformization by a deterministic top-down transducer that corresponds to relabeling (with bounded delay) the input tree, more formally, in a run of the transducer the read input and correspondingly produced output are always on the same path, i.e., there is no switching and no copying of subtrees, and furthermore the distance between read input symbol and produced output is bounded. We refer to this kind of DTDTs as linear DTDTs without switching.

We obtain the following result.

Theorem 18 *Given $k > 0$, it is decidable whether an N↓TA-recognizable relation has a uniformization by a linear deterministic top-down tree transducer without switching in which the delay is bounded by k.*

In the following we will see that this problem reduces to deciding the winner in a safety game. Recall the game $\mathcal{G}_{\mathcal{A}}^k$ presented in Section 3.1, there it is modeled that input and output are on same path and the delay between input and output is bounded by k. This corresponds to the restrictions we put on the uniformizer in this section. Let R be a tree-automatic specification recognized by an N↓TA $\mathcal{A}$. The game $\mathcal{G}_{\mathcal{A}}^k$ can also be defined for N↓TAs. It is easy to see that finding a winning strategy in $\mathcal{G}_{\mathcal{A}}^k$ corresponds to a uniformizer in which the delay is bounded by k for R. However, the converse is not true in general if $\mathcal{A}$ is an N↓TA as opposed to a D↓TA as in Section 3. We will show that the use of guidable tree automata [10] for the specifications solves this problem. The concept of guidable tree automata is that another tree automaton can act as a guide, meaning that a tree automaton $\mathcal{B}$ can guide a tree automaton $\mathcal{A}$ if an accepting run of $\mathcal{B}$ on a tree t can be translated deterministically into an accepting run of $\mathcal{A}$ on t.

Formally, an N↓TA $\mathcal{A}$ can be *guided by* an N↓TA $\mathcal{B}$ if there is a mapping $g : Q_\mathcal{A} \times \Delta_\mathcal{B} \to \Delta_\mathcal{A}$ such that $g(q,(p,a,p_1,\dots,p_i)) = (q,a,q_1,\dots,q_i)$ for some $q_1,\dots,q_i \in Q_\mathcal{A}$, and for every accepting run ρ of $\mathcal{B}$ over a tree t, $g(\rho)$ is an accepting run of $\mathcal{A}$ over t, where $g(\rho) = \rho'$ is the unique run such that $\rho'(\varepsilon) = q_0^\mathcal{A}$, and for all $u \in dom_t : (\rho'(u), val_t(u), \rho'(u1), \dots, \rho'(ui)) = g\left(\rho'(u), (\rho(u), val_t(u), \rho(u1), \dots, \rho(ui))\right)$. An N↓TA $\mathcal{A}$ is called *guidable* if it can be guided by every N↓TA $\mathcal{B}$ such that $T(\mathcal{B}) \subseteq T(\mathcal{A})$.

The following lemma shows that from the existence of a winning strategy in the game a uniformizer can be obtained and vice versa. The key idea of the proof is, given a guidable automaton for the specification, to turn the uniformizer into a guide for the specification automaton in order to construct a winning strategy in the safety game.

Lemma 19 *Let R be recognized by a guidable N↓TA $\mathcal{A}$. R has a uniformization by a linear DTDT without switching in which the delay is bounded by k if, and only if, Out has a winning strategy in $\mathcal{G}^k_\mathcal{A}$.*

Proof. Assume Out has a winning strategy in $\mathcal{G}^k_\mathcal{A}$, then there is also a positional winning strategy. A uniformizer can be constructed as described in the proof of Lemma 5.

For the other direction, assume R is uniformized by a linear DTDT $\mathcal{T}$ without switching in which the delay is bounded by k. That is, for each reachable configuration (t, t', φ) of $\mathcal{T}$ holds for each $u \in D_{t'}$ that either $u \sqsubseteq \varphi(u)$ or $\varphi(u) \sqsubseteq u$ and the absolute value of $|u| - |\varphi(u)|$ is bounded by k.

In order to construct a winning strategy in $\mathcal{G}^k_\mathcal{A}$, we first construct an N↓TA $\mathcal{B}$ that recognizes $R(\mathcal{T})$ and then use $\mathcal{B}$ to guide $\mathcal{A}$. The relation $R(\mathcal{T})$ is generally not D↓TA-recognizable, thus we have to construct an N↓TA that recognizes $R(\mathcal{T})$. However, from a run of $\mathcal{T}$ on a tree t, it will be possible to deterministically construct an accepting run of $\mathcal{B}$ on $t \otimes \mathcal{T}(t)$, which then can be used to guide $\mathcal{A}$. In a sense, $\mathcal{T}$ serves as a guide for $\mathcal{B}$, and $\mathcal{B}$ serves as a guide for $\mathcal{A}$.

We now present the construction of $\mathcal{B} = (Q_\mathcal{B}, \Sigma_\perp \times \Gamma_\perp, q_0^\mathcal{B}, \Delta_\mathcal{B})$ given $\mathcal{T} = (Q_\mathcal{T}, \Sigma, \Gamma, q_0^\mathcal{T}, \Delta_\mathcal{T})$, where M is the maximal height h of a right-hand side of a rule in $\Delta_\mathcal{T}$. We define the set of states $Q_\mathcal{B}$ as $Q_\mathcal{T} \cup Q_A \cup Q_D \cup Q_T$ with $q_0^\mathcal{T}$ as initial state, where

a) $Q_A := \{p^t \mid p \in Q_\mathcal{T}$ and $t \in T_{\Gamma \cup Q_\mathcal{T}}(X)$ with $h(t) \le M + k\}$ is the set of states indicating that output is ahead,

b) $Q_D := D \cup \{(p,z)^t \mid (p,z) \in D$ and $t \in T_{\Gamma \cup Q_\mathcal{T}}(X)$ with $h(t) \le M\}$, where $D := \{(p,z) \mid p \in Q_\mathcal{T}$ and $z \in (\Sigma \mathrm{dir}_\Sigma Q_\mathcal{T})^* \Sigma$ with $|z| < 3k\}$, is the set of states indicating that there is output delay,

c) $Q_T := \{p_t \mid t \in T_\Gamma$ with $h(t) \le M\}$ is the set of states that are used to recognize $T_\Sigma \times \{t\}$ for each $p_t \in Q_T$.

Now we present the construction of $\Delta_\mathcal{B}$, for that we make an observation about the possible configurations of $\mathcal{T}$. Let (t, t', φ) be a configuration of $\mathcal{T}$ such that there is delay, for example $\varphi(u) = v$, $v \sqsubset u$ and $vi \sqsubseteq u$, i.e., output is ahead. Assume in the next computation step output is produced, in order to satisfy the restriction that input and output have to be on the same path, the right-hand side of the applied rule has to be of the form $w[p(x_i)]$, where w is a 1-context. Otherwise, for some output of the form $w'[\dots, p(x_i), \dots, q(x_j), \dots]$, we would obtain a successor configuration (t, t'', φ') with $\varphi'(vj) = u'$ and $vj \not\sqsubseteq u \sqsubset u'$, i.e., input and output are no longer on the same path. In case that there is output delay, i.e., $\varphi(u) = v$, $u \sqsubset v$ and $ui \sqsubseteq v$, in a computation the next rule that is applied which produces output has to be of the form $g[t_1, \dots, t_{i-1}, w[\dots], t_{i+1}, \dots, t_n]$ with $t_1, \dots, t_{i-1}, t_{i+1}, \dots, t_n \in T_\Gamma$. Thereby, a successor configuration (t, t'', φ') is reached with $\varphi(u') = v'$ such that $u \sqsubset ui \sqsubseteq u'$ and $v \sqsubset v'$, i.e., input and output are on the same path.

First, we describe how $\Delta_\mathcal{B}$ is constructed w.r.t. a configuration of $\mathcal{T}$ such that input and output position are the same (in d)), or the output position is ahead of the input position (in e)). Since the output can be ahead at most k symbols, the part that is ahead is stored in the states.

d) For $p \in Q_\mathcal{T}$ and $p(f(x_1,\dots,x_i)) \to g(w_1[p_1(x_1)],\dots,w_i[p_i(x_i)],t_{i+1},\dots,t_n) \in \Delta_\mathcal{T}$, we add

$$(p,(f,g),q_1,\dots,q_i,p_{t_{i+1}}\dots,p_{t_n}) \text{ to } \Delta_\mathcal{B},$$

where $q_j = p_j$ if $w_j = \circ$, otherwise $q_j = p_j^{w_j[x_j]}$ for all $1 \le j \le i$, and

e) for $p^{w[\dots]} \in Q_A$ and $p(f(x_1,\dots,x_i)) \to w''[p'(x_j)] \in \Delta_\mathcal{T}$ such that $w[\dots]$ is of the form $g(t_1,\dots,t_{j-1},w'[\dots], t_{j+1},\dots,t_n)$, we add

$$(p^{w[\dots]},(f,g),p_{t_1},\dots,p_{t_{j-1}},p'^{w'[w''[p'(x_j)]]},p_{t_{j+1}}\dots,p_{t_n}) \text{ to } \Delta_\mathcal{B}.$$

Secondly, we describe how $\Delta_\mathcal{B}$ is constructed w.r.t. a configuration of $\mathcal{T}$ such that input and output position are the same and $\mathcal{T}$ delays the output (in f)), or the output is already delayed (in g)) and $\mathcal{T}$ might delay further. In both cases, $\mathcal{T}$ reads deterministically through the input and after at most k steps produces output. These different possibilities are encoded in the states, $\Delta_\mathcal{B}$ allows us to guess a possibility (in f) resp. g)) and verify the correctness of the guess (in h)).

f) For $p \in Q_\mathcal{T}$, and $p(f(\dots)) \to p_1(x_j), p_1(f_1(\dots)) \to p_2(x_{j_1}),\dots,p_{n-1}(f_{n-1}(\dots)) \to p_n(x_{j_{n-1}}) \in \Delta_\mathcal{T}$, and $p_n(f_n(\dots)) \to g(t_1,\dots,t_{j-1},w[\dots],t_{j+1},\dots,t_m) \in \Delta_\mathcal{T}$, we add

$$(p,(f,g),p_{t_1},\dots,p_{t_{j-1}},(p_1,z)^{w[\dots]},p_{t_{j+1}},\dots,p_{t_n}) \text{ to } \Delta_\mathcal{B},$$

where $z = f_1 j_1 p_2 f_2 j_2 p_3 \dots p_n f_n$,

g) for $(p, fjp' \cdot y \cdot p_1 f_1) \in Q_D$, $p_1(f_1(\dots)) \to p_2(x_{j_1}),\dots,p_{n-1}(f_{n-1}(\dots)) \to p_n(x_{j_{n-1}}) \in \Delta_\mathcal{T}$, and $p_n(f_n(\dots)) \to g(t_1,\dots,t_{j-1},w[\dots],t_{j+1},\dots,t_m) \in \Delta_\mathcal{T}$, we add

$$((p,fjp'\cdot y \cdot p_1 f_1),(f,g),p_{t_1},\dots,p_{t_{j-1}},(p',yz)^{w[\dots]},p_{t_{j+1}},\dots,p_{t_n}) \text{ to } \Delta_\mathcal{B},$$

where $z = p_1 f_1 j_1 p_2 f_2 j_2 p_3 \dots p_n f_n$.

Note that the constructions from f) and g) cause the non-determinism of $\mathcal{B}$.

h) For $(p,fjp'y)^{w[\dots]} \in Q_D$ such that $w[\dots]$ is of the form $g(t_1,\dots,t_{j-1},w'[\dots],t_{j+1},\dots,t_n)$, we add

$$((p,fjp'y)^{w[\dots]},(f,g),p_{t_1},\dots,p_{t_{j-1}},(p',y)^{w'[\dots]},p_{t_{j+1}}\dots,p_{t_n} \text{ to } \Delta_\mathcal{B}.$$

This construction yields finitely many rules for $\Delta_\mathcal{B}$, because the delay of $\mathcal{T}$ is bounded by k. It follows directly from the construction that $T(\mathcal{B})$ recognizes $R(\mathcal{T})$.

In a play, let (q,y) be the reached vertex after a sequence of moves ending by a move of In. These moves describe a labeled path $xiy \in \text{Path}_\Sigma$ with $x,y \in \text{Path}_\Sigma$, and $i \in \text{dir}_\Sigma$, and let $path(xi) = u$ and $path(xiy) = v$. We will choose the strategy of Out such that the following properties are satisfied. There is a $t \in T_\Sigma^{xiy}$ such that there is a run $\rho_\mathcal{A}$ of $\mathcal{A}$ on $t \otimes \mathcal{T}(t)$ that yields $\rho_\mathcal{A}(u) = q$. Further, there is a configuration (t,t',φ) of $\mathcal{T}$ with $\varphi(u') = v$ and either $u = u'$ or $v \sqsubset u'$. Also, there is a run $\rho_\mathcal{B}$ of $\mathcal{B}$ on $t \otimes \mathcal{T}(t)$ with the following properties. First, if $y \in \Sigma$ and $val_{t'}(u') = p$, then it holds that $\rho_\mathcal{B}(u) = p$ in case that $u = u'$, or $\rho_\mathcal{B}(u) = p^{w[p'(x_i)]}$ in case that $v \sqsubset u'$, where $\mathcal{T}(t)[\circ/u']|_v = w[\circ]$. Secondly, if $||y|| > 1$, let $y = f_1 j_1 \dots f_{n-1} j_{n-1} f_n$, then $u = u'$ and it holds that $\rho_\mathcal{B}(u) = p_1$, or $\rho_\mathcal{B}(u) = (p_1,z)$, where $z = f_1 j_1 p_2 f_2 j_2 p_3 \dots p_n f_n$ and $p_1 \dots p_n$ is the sequence of states that $\mathcal{T}$ is in while reading from u to v.

We are now ready to define the strategy inductively as follows. We distinguish between two possibilities. Either $\mathcal{T}$ produces no output in the next step, then Out also delays, or $\mathcal{T}$ produces output. In the latter case, the next move of Out will be chosen based on the state of $\mathcal{B}$ and the produced output of $\mathcal{T}$. First, we consider the case $y = f \in \Sigma$. Then, by induction hypothesis, we have $\rho_\mathcal{B}(u) = p$ or $\rho_\mathcal{B}(u) = p^{w[p'(x_i)]}$. From the construction of $\mathcal{B}$, it follows that there is exactly one applicable rule r in $\Delta_\mathcal{B}$ that matches $\rho_\mathcal{B}(u)$ and input symbol f. Recall that $\rho_{\mathcal{A}}(u) = q$, we now apply g to (q, r) and obtain a rule r' in $\Delta_{\mathcal{A}}$. Out chooses the edge corresponding to r' which exists as $(\mathcal{B}, g)$ guides $\mathcal{A}$.

Secondly, we consider $||y|| > 1$. We only present the case where the run $\rho_{\mathcal{B}}$ of $\mathcal{B}$ on $t \otimes \mathcal{T}(t)$ is constructed such that $\rho_{\mathcal{B}}(u) \in Q_{\mathcal{T}}$ holds, the other case is analogous. Let $y = fjf_1j_1 \dots f_{n-1}j_{n-1}f_n$, let $\rho_{\mathcal{B}}(u) = p$ and let $p_n(f_n(x_1, \dots, x_i)) \to g(t_1, \dots, t_{j-1}, w[\dots], t_{j+1}, \dots, t_m)$ be the rule of $\mathcal{T}$ that is applied to produce output after reading from u to v without producing output via the state sequence $pp_1 \dots p_n$. In order to successfully continue the run of $\mathcal{B}$ on $t \otimes \mathcal{T}(t)$, we set $z = f_1j_1p_2f_2j_2p_3 \dots p_nf_n$, and we have to choose the rule r of the form $(p, (f, g), p_{t_1}, \dots, p_{t_{j-1}}, (p_1, z)^{w[\dots]}, p_{t_{j+1}}, \dots, p_{t_n}) \in \Delta_{\mathcal{B}}$ which is guaranteed to exist by construction of $\mathcal{B}$. Out chooses the edge corresponding to $g(q, r)$ as next move. For the part of $\rho_{\mathcal{B}}$ where the rest of the output $w[\dots]$ from $(p_1, z)^{w[\dots]}$ is concerned, the applicable transitions of $\mathcal{B}$ are deterministically determined. For this part, Out can make a sequence of moves by choosing the edges corresponding to the application of g to $\rho_{\mathcal{B}}$.

Clearly, in the next vertex reached after a move of In the induction hypothesis is also valid. The defined strategy is a winning strategy for Out, because an accepting run of $\mathcal{T}$ directly corresponds to an accepting run of $\mathcal{B}$ which can be translated to an accepting run of $\mathcal{A}$ when applying g. Thus, Out never reaches a vertex without outgoing edges. □

As a consequence of Lemma 19 and the fact that a winning strategy for Out in $\mathcal{G}^k_{\mathcal{A}}$ can effectively be computed, together with the fact that for each automatic tree relation a guidable N↓TA $\mathcal{A}$ can effectively be constructed, see [10], we immediately obtain Theorem 18.

5. Conclusion

In this paper, we considered the synthesis of deterministic top-down tree transducers from tree automatic specifications. We have shown that is decidable whether a deterministic top-down specification can be realized by a top-down tree transducer under the restriction that the transducer is not required to validate the input, meaning that a transducer implementing a uniformization can behave arbitrarily on invalid inputs. If uniformization is possible, our decision procedure yields a top-down tree transducer that realizes the specification.

We have seen that the presented decision procedure concerning uniformization without input validation cannot be transferred directly to decide the problem corresponding to the classical uniformization question (with input validation). The reason for this is that in the employed transducer model it is not possible to verify the input without producing output.

The synthesis problem for general nondeterministic tree automatic specifications remains open. However, we have solved two special cases, namely for unions of deterministic specifications with disjoint domains and uniformizers without delay, as well as for all nondeterministic specifications and uniformizers of bounded delay in which input and output always remain on the same path.

In future work, we would like to develop methods for solving the general case of nondeterministic specifications (and the full class of uniformizers). Furthermore, we would also like to consider nondeterministic top-down tree transducers as specification formalism. For this class of specifications, the synthesis problem in its full generality is undecidable, because it already is for the restriction to words (synthesis of sequential transducers from rational relations [5]). We would like to identify interesting decidable restrictions of this problem.

References

[1] A. Church, Logic, arithmetic and automata, in: Proceedings of the international congress of mathematicians, 1962, pp. 23–35.

[2] J. Büchi, L. Landweber, Solving sequential conditions by finite-state strategies, Transactions of the American Mathematical Society.

[3] F. Hosch, L. Landweber, Finite delay solutions for sequential conditions., in: ICALP, 1972, pp. 45–60.

[4] M. Holtmann, Ł. Kaiser, W. Thomas, Degrees of lookahead in regular infinite games, in: Foundations of Software Science and Computational Structures, Springer, 2010, pp. 252–266.

[5] A. Carayol, C. Löding, Uniformization in Automata Theory, To appear in: Logic, Methodology and Philosophy of Science. Proceedings of the Fourteenth International congress. P. Schroeder-Heister, G. Heinzmann, W. Hodges, P. Edouard Bour, eds., London: College Publications (2012).

[6] H. Comon, M. Dauchet, R. Gilleron, C. Löding, F. Jacquemard, D. Lugiez, S. Tison, M. Tommasi, Tree Automata Techniques and Applications, release October, 12th 2007 (2007).
URL `http://www.grappa.univ-lille3.fr/tata`

[7] T. Milo, D. Suciu, V. Vianu, Typechecking for xml transformers, J. Comput. Syst. Sci. 66 (1) (2003) 66–97.

[8] J. Engelfriet, On tree transducers for partial functions, Inf. Process. Lett. 7 (4) (1978) 170–172.

[9] W. Thomas, Automata on infinite objects, in: Handbook of Theoretical Computer Science, Volume B: Formal Models and Sematics (B), 1990, pp. 133–192.

[10] C. Löding, Logic and automata over infinite trees, habilitation Thesis, RWTH Aachen, Germany (2009).

[11] C. Löding, S. Winter, Synthesis of deterministic top-down tree transducers from automatic tree relations, in: Proceedings Fifth International Symposium on Games, Automata, Logics and Formal Verification, GandALF 2014, Verona, Italy, September 10-12, 2014., 2014, pp. 88–101. `doi:10.4204/EPTCS.161.10`.

[12] F. Gèsceg, M. Steinby, Tree automata, Akademiai Kiado.

[13] E. Grädel, W. Thomas, T. Wilke (Eds.), Automata, Logics, and Infinite Games, no. 2500 in Lecture Notes in Compter Science, Springer, 2002.

[14] E. Grädel, Finite Model Theory and Descriptive Complexity, in: Finite Model Theory and Its Applications, Springer, 2007, pp. 125–230.

[15] F. Ramsey, On a problem of formal logic, Proceedings of the London Mathematical Society 2 (1) (1930) 264.

[16] R. Diestel, Graph Theory, 2nd Edition, Vol. 173 of Graduate texts in mathematics, Springer, 2000.

[17] S. Winter, Uniformization of Automaton Definable Tree Relations, masterthesis, RWTH Aachen, Germany (2013).