FO Model Checking on Nested Pushdown Trees

Alexander Kartzow

TU Darmstadt, Fachbereich Mathematik, Schlossgartenstr. 7, 64289 Darmstadt

Abstract. Nested Pushdown Trees are unfoldings of pushdown graphs with an additional jump-relation. These graphs are closely related to collapsible pushdown graphs. They enjoy decidable μ-calculus model checking while monadic second-order logic is undecidable on this class. We show that nested pushdown trees are tree-automatic structures, whence first-order model checking is decidable. Furthermore, we prove that it is in 2-EXPSPACE using pumping arguments on runs of pushdown systems. For these arguments we also develop a Gaifman style argument for graphs of small diameter.

1 Introduction

Nested pushdown trees were introduced in [1] as an expansion of trees generated by pushdown systems with nested jump-edges. They were proposed for software verification as jump-edges may be used to reason about matching pairs of calls and returns in a program. Another approach to software verification checks pushdown trees (without jump-edges) against specifications given by automata or μ-calculus formulas. But these methods even lack the ability to express that every call has a matching return. Alur et al. showed that nested pushdown trees are tame structures with respect to the μ-calculus, in the sense that μ-calculus model checking on nested pushdown trees is decidable. On the other hand they proved the undecidability of monadic second-order logic on nested pushdown trees. These results make nested pushdown trees an interesting class from a model theoretic point of view because there are few natural classes that separate μ-calculus and monadic second-order logic with respect to model checking. In fact, the author knows of only one similar result, namely, for the class of collapsible pushdown graphs [7]. The hierarchy of collapsible pushdown graphs forms an extension of the hierarchy of higher-order pushdown graphs by using a new operation called collapse. There is a close relation between nested pushdown trees and collapsible pushdown graphs: the former are first-order interpretable in collapsible pushdown graphs of order two. [1] In this sense, jump-edges form a very weak form of collapse-edges. For both classes nothing is known so far about the decidability of first-order model checking. In

[1] As the proof of this claim is unpublished, we give an idea: A node in a nested pushdown tree is a run, i.e., a list of pairs of states and stacks. Push the state onto the stack. This list of stacks can be seen as a level 2 stack and every edge in the nested tree can then be simulated by up to four operations of the collapsible pushdown system.

the following we are going to settle the problem for nested pushdown trees with the positive answer that first-order model checking for nested pushdown trees is in 2-EXPSPACE. Furthermore, we show that nested pushdown trees are tree-automatic. The notion of tree-automatic structures was developed in [2] and generalises the concept of automatic structures to the tree case. These are (usually) infinite structures that allow a finite representation by tree-automata. Due to the good algorithmic behaviour of tree-automata the class of tree-automatic structures has nice properties, e.g., first-order model checking is decidable. But in general even automatic structures, and hence also tree-automatic structures, have non-elementary lower bounds for FO model checking [4]. Nevertheless we can show that model checking on nested pushdown trees is elementary by using pumping techniques for pushdown systems.

Here is an outline of the paper. In Section 2 we present an Ehrenfeucht-Fraïssé-game argument for the equivalence of certain structures with parameters for first-order logic up to a fixed quantifier rank. This argument is a form of locality argument on structures of small diameter, despite the fact that small diameters normally prohibit the use of locality arguments. We use local isomorphisms on subgraphs which are nicely embedded into the full graph. Later, this is a main tool in our pumping arguments. Section 3.1 provides the definition of nested pushdown trees and Section 3.2 contains the proof that these structures are tree-automatic. In order to show that first-order model checking on nested pushdown trees is in 2-EXPSPACE (Section 3.4), we develop pumping arguments on nested pushdown trees in 3.3.

2 A Gaifman Style Lemma on Graphs of Small Diameter

In this section we present a game argument showing that certain tuples of a given graph have the same $\simeq_\rho$-type, where $\simeq_\rho$ is equivalence for first-order formulas up to quantifier rank ρ. This argument forms the back-bone of the transformations we are going to use on tuples in a nested pushdown tree. It is a kind of Gaifman-locality argument for certain graphs with possibly small diameter. The crucial property of these graphs is that there are some generic edges that make the diameter small in the sense that a lot of vertices are connected to the same vertex, but when these edges are removed the diameter becomes large. Therefore, on the graph with these generic edges removed we can apply Gaifman-like arguments in order to establish partial isomorphisms and $\simeq_\rho$-equivalence. As disjoint but isomorphic neighbourhoods in such a graph have generic edges to the same vertices (in the full graph) moving a tuple from one neighbourhood to the other does not change the $\simeq_\rho$-type of the tuple.

We use the following definitions and notation. By FO we denote *first-order logic* and we write FO_ρ for the restriction of FO to formulas of quantifier rank up to ρ. We write $\bar{a} = a_1, a_2 \dots, a_n \in A$ for a tuple of elements from a set A. For structures $\mathfrak{A}$ and $\mathfrak{B}$ with n parameters $\bar{a} \in A^n$ and $\bar{b} \in B^n$ we write $\mathfrak{A}, \bar{a} \simeq_\rho \mathfrak{B}, \bar{b}$ for the fact that $\mathfrak{A}, \models \varphi(\bar{a})$ if and only if $\mathfrak{B} \models \varphi(\bar{b})$ for all $\varphi \in \mathrm{FO}_\rho$. For some structure $G = (V, E_1, E_2, \dots, E_n)$ with binary relations $E_1, E_2, \dots, E_n$ and sets $A, B \subseteq V$ we say that A and B *touch* if $A \cap B \neq \emptyset$ or there are $a \in A$, $b \in B$ such that $(a,b) \in E_i$ or $(b,a) \in E_i$ for some $i \leq n$. For a tuple $\bar{a} \in A$ we define inductively the *l-neighbourhood* of $\bar{a}$ with respect to A setting $A_0(\bar{a}) := \{a_i \in \bar{a}\}$, and

$$A_{l+1}(\bar{a}) := A_l(\bar{a}) \cup \{b \in A : \text{there are } i \leq n \text{ and } c \in A_l(\bar{a}) \text{ s.t. } (b,c) \in E_i \text{ or } (c,b) \in E_i\} \ .$$

We write $N_l(\bar{a})$ for the l-neighbourhood with respect to the whole universe V.

We say that A and B are isomorphic over $C \subseteq V$ and write $A \simeq_C B$ if there is some isomorphism $\varphi : G{\restriction}A \simeq G{\restriction}B$ such that for all $a \in A$ and $c \in C$

$$(a,c) \in E_i \text{ iff } (\varphi(a),c) \in E_i \qquad \text{and} \qquad (c,a) \in E_i \text{ iff } (c,\varphi(a)) \in E_i \ .$$

Lemma 1. *Let $G = (V, E_1, E_2, \dots, E_n)$ be some structure, $A, B \subseteq V$ not touching and let $\varphi : A \simeq B$ be an isomorphism of the induced subgraphs. Let $\bar{a} \in A$ and $\bar{c} \in C := G \setminus \left(A_{2^\rho}(\bar{a}) \cup B_{2^\rho}(\varphi(\bar{a}))\right)$.*

$$\varphi{\restriction}A_{2^\rho - 1}(\bar{a}) : A_{2^\rho - 1}(\bar{a}) \simeq_C B_{2^\rho - 1}(\varphi(\bar{a})) \quad \textit{implies} \quad G, \bar{a}, \varphi(\bar{a}), \bar{c} \simeq_\rho G, \varphi(\bar{a}), \bar{a}, \bar{c} \ .$$

Proof. We prove the claim by induction on ρ using Ehrenfeucht-Fraïssé-Game-terminology. By symmetry, we may assume that Spoiler extends the left-hand side, i.e., extending $\bar{a}, \varphi(\bar{a}), \bar{c}$ by some $d \in V$. The general idea is that Spoiler either chooses an element in $A \cup B$ that is close to $\bar{a}$ or $\varphi(\bar{a})$ and Duplicator responds with applying the isomorphism φ. Otherwise, Duplicator just responds choosing the same element as Spoiler.
Local case: if $d \in A_{2^{\rho-1}}(\bar{a})$ set $a' := d$ and if $d \in \varphi(A_{2^{\rho-1}}(\bar{a}))$ set $a' := \varphi^{-1}(d)$. Then we set $\bar{a}' := \bar{a}, a'$.

As $A_{2^{\rho-1}}(\bar{a}') \subseteq A_{2^\rho}(\bar{a})$, we have $\bar{c} \in C' := G \setminus \left(A_{2^{\rho-1}}(\bar{a}') \cup \varphi(A_{2^{\rho-1}}(\bar{a}'))\right)$. Since A and B do not touch and $C' = C \cup D$ for $D \subseteq \left(A \setminus A_{2^{\rho-1}}(\bar{a}')) \cup (B \setminus B_{2^{\rho-1}}(\varphi(\bar{a}'))\right)$ we get $A_{2^{\rho-1}-1}(\bar{a}') \simeq_{C'} \varphi(A_{2^{\rho-1}-1}(\bar{a}'))$. Hence, we obtain by induction hypothesis

$$G, \bar{a}', \varphi(\bar{a}'), \bar{c} \simeq_{\rho-1} G, \varphi(\bar{a}'), \bar{a}', \bar{c}$$

Nonlocal case: otherwise, $d \in C' := G \setminus \left(A_{2^{\rho-1}}(\bar{a}) \cup \varphi(A_{2^{\rho-1}}(\bar{a}))\right)$ and we set $\bar{c}' := \bar{c}_\rho, d$. Note that $A_{2^{\rho-1}-1}(\bar{a}) \simeq_{C'} \varphi(A_{2^{\rho-1}-1}(\bar{a}))$ as A and B are not touching and the distance of elements in $A_{2^{\rho-1}-1}(\bar{a})$ and elements in $C' \cap A$ is at least 2. Hence, by induction hypothesis

$$G, \bar{a}, \varphi(\bar{a}), \bar{c}' \simeq_{\rho-1} G, \varphi(\bar{a}), \bar{a}, \bar{c}' \ . \qquad \square$$

3 Nested Pushdown Trees

Nested pushdown trees are generated by pushdown systems in the following way. We unfold the configuration graph of a pushdown system and we add a *jump relation* that connects every push- with the corresponding pop-operations.

After formally introducing nested pushdown trees, we show that this class of structures is tree-automatic. This already implies that FO model checking for nested pushdown trees is decidable. But it does not yield an elementary bound for the complexity since the model checking for tree-automatic structures is in general non-elementary [4].

We give a separate argument that yields an elementary bound. This argument is based on pumping techniques. In Section 3.3 we present these techniques which shorten long runs but preserve their $\simeq_\rho$-type in the nested pushdown tree. Due to this result, we only have to inspect finitely many short runs in order to find witnesses for existential quantifications. Section 3.4 shows that this search may be done in 2-EXPSPACE.

3.1 Definition

Definition 1 (Pushdown System). *A tuple $P = (Q, \Sigma, \Delta, (q_0, \bot))$ with a finite set of states Q, a finite set of stack symbols Σ, an initial configuration $(q_0, \bot) \in Q \times \Sigma$ and a transition relation $\Delta \subseteq Q \times \Sigma \times Q \times \left\{\mathrm{pop}, \mathrm{id}, \mathrm{push}_\sigma \textit{ for each } \sigma \in \Sigma\right\}$ is called a* pushdown system.

Definition 2. *A* run *r of P is a function $r : \{0, 1, 2, \dots, n\} \to Q \times \Sigma^*$ such that for all $i < n$ there is some $(q, \sigma, p, op) \in \Delta$ and some $w_i \in \Sigma^*$ such that $r(i) = (q, w_i\sigma)$ and $r(i+1) = (p, op(w_i\sigma))$, where $\mathrm{pop}(w_i\sigma) = w_i$, $\mathrm{id}(w_i\sigma) = w_i\sigma$, and $\mathrm{push}_\tau(w_i\sigma) = w_i\sigma\tau$. We call r a run from $r(0)$ to $r(n)$. We say that the* length *of r is* $\mathrm{length}(r) := n$.

For runs r and r' of length n and m, respectively, such that $r(n) = r'(0)$ we call

$$s : \{0, 1, \dots, n+m\} \to Q \times \Sigma^* \qquad s(i) := \begin{cases} r(i) & \text{if } i \leq n \\ r'(i-n) & \text{otherwise} \end{cases}$$

the composition of r and r'. *We also say that s* decomposes into r and r'.

Note that a run does not necessarily start in the initial configuration $(q_0, \bot)$ of the pushdown system P. The next definition summarises some useful notation about runs.

Definition 3. *Let r be a run of a pushdown system $P = (Q, \Sigma, \Delta, (q_0, \bot))$ and let w, v be words over Σ.*

- *If r has length n, then* $\mathrm{last}(r) := r(n)$.
- *By $w \leq v$ we mean that w is a prefix of v.*
- *For $r(i) = (q, v)$, we set $\mathrm{Stck}(r(i)) := v$. We write $|r(i)|$ for $|v|$ and $w \leq r(i)$ if $w \leq v$.*
- *We say that r is w-*prefixed *if $w \leq r(i)$ for all $i \in \mathrm{dom}(r)$.*
- *We set* $\max(r) := \max\{|r(i)| : i \in \mathrm{dom}(r)\}$.

Remark 1 (Prefix Replacement). Let r be a w-prefixed run of some pushdown system P for some word $w \in \Sigma^*$. If $w' \in \Sigma^*$ ends with the same letter as w then the function

$$r[w/w'] : \mathrm{dom}(r) \to Q \times \Sigma^*$$
$$r[w/w'](i) := (q_i, w'w_i) \quad \text{if } r(i) = (q_i, ww_i)$$

is a run of P, where ww_i denotes the usual concatenation of the words w and w_i.

Definition 4 (Nested Pushdown Tree (NPT)). *Let $P = (Q, \Sigma, \Delta, (q_0, \bot))$ be a pushdown system. Then the* nested pushdown tree *generated by P is* $\mathrm{NPT}(P) := (R, \rightarrow, \hookrightarrow)$ *where $(R, \rightarrow)$ is the unfolding of the configuration graph of P, i.e., R is the set of all runs of P starting at the configuration $q_0, \bot$. For two runs $r_1, r_2 \in R$, we have $r_1 \rightarrow r_2$ if r_2 extends r_1 by exactly one configuration. The binary relation $\hookrightarrow$ is called* jump relation *and is defined as follows: let $r_1, r_2 \in R$ and* $\mathrm{last}(r_1) = (q, w) \in Q \times \Sigma^*$. *Then $r_1 \hookrightarrow r_2$ if r_1 is an initial segment of r_2,* $\mathrm{last}(r_2) = (q', w)$ *for some $q' \in Q$ and w is a proper prefix of all stacks between* $\mathrm{last}(r_1)$ *and* $\mathrm{last}(r_2)$, *i.e., $w < r_2(i)$ for all* $\mathrm{length}(r_1) < i < \mathrm{length}(r_2)$.

3.2 NPT are tree-automatic

We start with the notion of a tree-automatic structure which was introduced in [2]. A *tree* is a finite, prefix closed subset of $\{0, 1\}^*$, where ε represents the root and we assume the successors at each vertex to be ordered. For a finite set Σ, a Σ-labelled tree is a map $c : T \rightarrow \Sigma$ for some tree T. The *convolution* of two Σ-labelled trees c_1 and c_2 is defined as $c_1 \otimes c_2 : \mathrm{dom}(t_1) \cup \mathrm{dom}(t_2) \rightarrow (\Sigma \cup \{\square\})^2$, where $\square$ represents undefined elements, and

$$(c_1 \otimes c_2)(t) = \begin{cases} (c_1(t), c_2(t)) & \text{if } t \in \mathrm{dom}(c_1) \cap \mathrm{dom}(c_2) \ , \\ (c_1(t), \square) & \text{if } t \in \mathrm{dom}(c_1) \setminus \mathrm{dom}(c_2) \ , \\ (\square, c_2(t)) & \text{if } t \in \mathrm{dom}(c_2) \setminus \mathrm{dom}(c_1) \ . \end{cases}$$

A *tree-automaton* is a tuple $A = (Q, \Sigma, \Delta, F)$ where Q is a finite set of states, $\Sigma \subseteq Q$ a finite set of labels, $\Delta \subseteq Q^2 \times \Sigma \times Q$ the transition relation, and $F \subseteq Q$ the set of final states. A run of A on a Σ-labelled tree $c : T \rightarrow \Sigma$ is a function $r : T \rightarrow Q$ such that for each leaf $l \in T$ we have $r(l) = c(l)$ and for inner nodes $n \in T$ we have $r(n) = q$ if there is some $(q_0, q_1, \sigma, q) \in \Delta$ such that $r(ni) = q_i$ for $i \in \{0, 1\}$ and $c(n) = \sigma$. A run r is accepting if $r(0) \in F$. Note that we require $\Sigma \subseteq Q$ as A has no special initial state but starts at every leaf of the tree initialised with the label of this leaf.

A structure $\mathfrak{B} = (B, E_1, E_2, \ldots, E_n)$ with binary relations E_i is *tree-automatic* if there are automata $A_B, A_{E_1}, A_{E_2}, \ldots, A_{E_n}$ such that

1. A_B accepts a set C of Σ-labelled trees.
2. There is a bijection $f : C \rightarrow B$
3. for $c_1, c_2 \in C$, the automaton A_{E_i} accepts $c_1 \otimes c_2$ if and only if $(f(c_1), f(c_2)) \in E_i$.

Theorem 1. *Nested pushdown trees are tree-automatic.*[2]

By the decidability of the FO model checking for arbitrary tree-automatic structures [2] we obtain that FO model checking on nested pushdown trees is decidable.

For the proof of the theorem, we use the fact that, for every context-free grammar G, there is a tree-automaton A which accepts exactly the derivation trees of G. In order to prove tree-automaticity of a NPT, it therefore suffices to

[2] I thank Dietrich Kuske for proposing a useful coding of runs in trees.

give a context free grammar of runs of a pushdown system P (starting at the initial configuration) and to provide grammars generating all pairs of derivations of runs that are connected by $\to$ or $\hookrightarrow$, respectively.

Let $P = (Q, \Sigma, \Delta, (q_0, \bot))$ be a pushdown system. The following context-free grammar generates all runs of P which start at the initial configuration $q_0, \bot$. The terminal symbols are the transitions of P, i.e., $T := \Delta$. We use the non-terminal symbols $N := \{X_{(q,\sigma)} : q \in Q, \sigma \in \Sigma\} \cup \{C^p_{(q,\sigma)} : q, p \in Q, \sigma \in \Sigma\}$. The idea of the coding is the following. A non-terminal $X_{(q,\sigma)}$ generates a subrun starting from (q, σ) and a $C^p_{(q,\sigma)}$ generates a subrun starting at (q, σ), ending at (p, σ) and in between this element σ is never removed from the stack. Note that such a subrun may be extended by prefixing some push_σ- and postfixing some pop-operation that deletes this symbol σ again. For $q, p, r, s \in Q$ and $\sigma, \tau \in \Sigma$, the productions are

$$\begin{aligned}
X_{(q,\sigma)} \to{}& (q,\sigma,p,\text{id}) \;\big|\; (q,\sigma,p,\text{id})X_{(p,\sigma)} \;\big|\; (q,\sigma,p,\text{push}_\tau) \;\big|\; (q,\sigma,p,\text{push}_\tau)X_{(p,\tau)}\\
&\big|\; (q,\sigma,p,\text{push}_\tau)C^r_{(p,\tau)}(r,\tau,s,\text{pop}) \;\big|\; (q,\sigma,p,\text{push}_\tau)C^r_{(p,\tau)}(r,\tau,s,\text{pop})X_{(s,\sigma)}\\
&\big|\; (q,\sigma,p,\text{push}_\tau)(p,\tau,r,\text{pop}) \;\big|\; (q,\sigma,p,\text{push}_\tau)(p,\tau,r,\text{pop})X_{(r,\sigma)}\\
\text{and } C^p_{(q,\sigma)} \to{}& (q,\sigma,r,\text{id})C^p_{(r,\sigma)} \;\big|\; (q,\sigma,p,\text{id}) \;\big|\; (q,\sigma,r,\text{push}_\tau)C^s_{(r,\tau)}(s,\tau,u,\text{pop})C^p_{(u,\sigma)}\\
&\big|\; (q,\sigma,s,\text{push}_\tau)(s,\tau,u,\text{pop})C^p_{(u,\sigma)} \;\big|\; (q,\sigma,r,\text{push}_\tau)C^s_{(r,\tau)}(s,\tau,p,\text{pop})\\
&\big|\; (q,\sigma,r,\text{push}_\tau)(r,\tau,p,\text{pop})
\end{aligned}$$

Note that for every run r of P starting in $(q_0, \bot)$ there is a unique derivation tree starting from $X_{(q_0,\bot)}$ and the leaves of this derivation tree – read from left to right – are the transitions of r. Vice versa, every derivation tree codes a valid run.

As a next step we show that the set of convolutions of the derivation trees of runs r_1, r_2 such that r_2 extends r_1 by exactly one transition may also be defined via some context free grammar. Note that if a run r_2 extends another run r_1 by a push_σ- or id-transition, the derivation trees only differ in the subtree that starts at the end of the unique longest path that is labelled by non-terminals $X_{(q,\sigma)}$ (where q and σ may vary along the path). The coding of r_2 contains an isomorphic copy of this subtree in the coding of r_1, and extends this subtree by a new rightmost successor with label $X_{(q,\sigma)}$ for some $q \in Q$ and $\sigma \in \Sigma$ and this new rightmost successor has a successor itself which is labelled by the last transition of r_2. The case of a pop-transition is a bit more involved as the subrun between this pop-operation and the corresponding push_σ-operation is derived from a $C^p_{(q,\sigma)}$ symbol in the derivation of r_2, while it is derived from $X_{(q,\sigma)}$ in the derivation of r_1. But in fact, for both derivations, the form of this subtree is the same and the terminal symbols coincide. The only difference is that the non-terminals of the form $X_{(r,\tau)}$ in the derivation of r_1 are replaced by $C^s_{(r,\tau)}$ for some $s \in Q$ in the derivation of r_2.

We use the following notation. For some terminal or non-terminal a we write a^2 as an abbreviation for the pair (a, a) and we write $Z^p_{(q,\sigma)}$ for the pair $(X_{(q,\sigma)}, C^p_{(q,\sigma)})$.

The productions are

$$\begin{aligned}
(X_{(q,\sigma)})^2 \to& a_1^2 a_2^2 \dots a_n^2 (X_{(p,\tau)})^2 \quad \text{for every } X_{(q,\sigma)} \to a_1 a_2 \dots a_n X_{(p,\tau)} \\
(C^p_{(q,\sigma)})^2 \to& a_1^2 a_2^2 \dots a_n^2 \quad \text{for every } C^p_{(q,\sigma)} \to a_1 a_2 \dots a_n \\
(X_{(q,\sigma)})^2 \to& (q,\sigma,p,\mathrm{id})^2\big(\square, X_{(p,\sigma)}\big) \;\big|\; (q,\sigma,p,\mathrm{push}_\tau)^2\big(\square, X_{(p,\tau)}\big) \\
&\big|\; (q,\sigma,p,\mathrm{push}_\tau)^2(C^r_{(p,\tau)})^2(r,\tau,s,\mathrm{pop})^2\big(\square, X_{(s,\sigma)}\big) \\
&\big|\; (q,\sigma,p,\mathrm{push}_\tau)^2 Z^r_{(p,\tau)}\big(\square,(r,\tau,s,\mathrm{pop})\big) \;\big|\; (q,\sigma,p,\mathrm{push}_\tau)^2\big(\square,(p,\tau,s,\mathrm{pop})\big) \\
Z^p_{(q,\sigma)} \to& (q,\sigma,r,\mathrm{id})^2 Z^p_{(r,\sigma)} \;\big|\; (q,\sigma,p,\mathrm{id})^2 \;\big|\; (q,\sigma,r,\mathrm{push}_\tau)^2(C^s_{(r,\tau)})^2(s,\tau,u,\mathrm{pop})^2 Z^p_{(u,\sigma)} \\
&\big|\; (q,\sigma,s,\mathrm{push}_\tau)^2(s,\tau,u,\mathrm{pop})^2 Z^p_{(u,\sigma)} \;\big|\; (q,\sigma,r,\mathrm{push}_\tau)^2(C^s_{(r,\tau)})^2(s,\tau,p,\mathrm{pop})^2 \\
&\big|\; (q,\sigma,r,\mathrm{push}_\tau)^2(r,\tau,p,\mathrm{pop})^2 \\
(\square, X_{(q,\sigma)}) \to& \big(\square,(q,\sigma,p,\mathrm{id})\big) \;\big|\; \big(\square,(q,\sigma,p,\mathrm{push}_\tau)\big)
\end{aligned}$$

Analogously to the $\to$-case, we can give a grammar for runs r_1, r_2 such that $r_1 \hookrightarrow r_2$. If $r_1 \hookrightarrow r_2$, then r_1 is an initial segment of r_2. Thus, the derivation of r_2 contains that of r_1. It extends the derivation of r_1 by a derivation of the form $(q,\sigma,p,\mathrm{push}_\tau)C^r_{(p,\tau)}(r,\tau,s,\mathrm{pop})$. The following productions describe this:

$$\begin{aligned}
(X_{(q,\sigma)})^2 \to& a_1^2 a_2^2 \dots a_n^2 (X_{(p,\tau)})^2 \quad \text{for every } X_{(q,\sigma)} \to a_1 a_2 \dots a_n X_{(p,\tau)} \\
(C^p_{(q,\sigma)})^2 \to& a_1^2 a_2^2 \dots a_n^2 \quad \text{for every } C^p_{(q,\sigma)} \to a_1 a_2 \dots a_n \\
(X_{(q,\sigma)})^2 \to& (q,\sigma,p,\mathrm{id})^2\big(\square, X_{(p,\sigma)}\big) \;\big|\; (q,\sigma,p,\mathrm{push}_\tau)^2\big(\square, X_{(p,\tau)}\big) \\
&\big|\; (q,\sigma,p,\mathrm{push}_\tau)^2(C^r_{(p,\tau)})^2(r,\tau,s,\mathrm{pop})^2\big(\square, X_{(s,\sigma)}\big) \\
\big(\square, X_{(p,\sigma)}\big) \to& \big(\square,(q,\sigma,p,\mathrm{push}_\tau)\big)\big(\square,(p,\tau,r,\mathrm{pop})\big) \\
&\big|\; \big(\square,(q,\sigma,p,\mathrm{push}_\tau)\big)(\square, C^s_{(r,\tau)})\big(\square,(p,\tau,r,\mathrm{pop})\big)
\end{aligned}$$

The productions of $\big(\square, C^p_{(q,\sigma)}\big)$ are exactly as for $C^p_{(q,\sigma)}$ in the second component with first component always marked $\square$, i.e., the first run is already finished and the second run extends the first one by some "closed" subrun, i.e., a subrun that starts and ends with the same stack content.

3.3 $\simeq_\rho$-Pumping on NPT

In this section we present several pumping lemmas on runs of a pushdown system P. The aim is to show that for every run of a pushdown system there is another one of bounded length which represents a node with the same $\simeq_\rho$-type in the NPT generated by P. We use these lemmas later to prove an elementary bound for the complexity of FO model checking on nested pushdown trees. As every $\simeq_\rho$-type has a witness of bounded length, a model checking algorithm for an FO$_\rho$-formula only has to check runs of bounded length in order to find a witness for an existential quantification.

We bound the length of a run in three steps. The first one reduces the size of the last stack of a run, the second one reduces the size of the maximal stack passed along the run and the last one gives us basically a bound on the number of occurrences of every given stack along the run. We will see that these conditions are sufficient for bounding its length.

We start with a general observation about the structure of runs that are related by some edge. We will use this lemma in several of our pumping lemmas.

Lemma 2. *Let $r = r_1 \circ r_2 \circ r_3$ be a run of a pushdown system P, $w \in \Sigma^*$ and $\sigma \in \Sigma$ such that r_2 is w-prefixed, r_3 is $(w\sigma)$-prefixed and* $\mathrm{Stck}(\mathrm{last}(r_1)) = w$. *If $r * s$ for $* \in \{\hookrightarrow, \hookleftarrow, \rightarrow, \leftarrow\}$ then $s = r_1 \circ r'_2$ for some w-prefixed run r'_2.*

Proof. As $w\sigma \leq \mathrm{last}(r)$ we have $w \leq \mathrm{last}(s)$. Hence, the only non-trivial case is $s \hookrightarrow r$. By definition of $\hookrightarrow$, we have $w\sigma \leq r(i)$ for all $i \in \mathrm{dom}(r) \setminus \mathrm{dom}(s)$ and s is an initial segment of r. Thus, r_1 is an initial segment of s. □

Now we can state our first pumping lemma, that reduces the size of the last configuration of a given run, while preserving its $\simeq_\rho$-type.

Lemma 3 (First $\simeq_\rho$-Pumping Lemma). *Let $\bar{r} = r_1, r_2, \dots, r_m \in \mathrm{NPT}(P)$ and $r \in \mathrm{NPT}(P)$ such that*

$$|\mathrm{last}(r)| > |\mathrm{last}(r_i)| + (2 + 2^{\rho+1})|Q| \cdot |\Sigma| + 2^\rho + 1 \quad \textit{for all } i \leq m \ .$$

There is an $s \in \mathrm{NPT}(P)$ such that $|\mathrm{last}(s)| < |\mathrm{last}(r)|$ and $\mathrm{NPT}(P), \bar{r}, r \simeq_\rho \mathrm{NPT}(P), \bar{r}, s$.

Proof. Because of the length of $v := \mathrm{Stck}(\mathrm{last}(r))$ there are $w_1 < w_2 \leq v$ and decompositions of r as $r = r_{w_1} \circ s_{w_1} = r_{w_2} \circ s_{w_2}$ such that

1. $s_{w_i}(0) = (w_i, q)$ for some $q \in Q$ and all $i \in \{1, 2\}$;
2. s_{w_i} is w_i-prefixed;
3. $|w_1| > |\mathrm{last}(r_i)|$ and $|\mathrm{last}(r)| > |w_2| + 2^\rho$;
4. w_1 and w_2 end with the same letter $\sigma \in \Sigma$;
5. $|w_2| - |w_1| > 1 + 2^{\rho+1}$.

Then $s := r_{w_1} \circ s_{w_2}[w_2/w_1]$ is well defined by Remark 1. Note that $N_{2^\rho}(r)$ and $N_{2^\rho}(s)$ do not touch because $|\mathrm{last}(r)| - |\mathrm{last}(s)| = |w_2| - |w_1| > 1 + 2^{\rho+1}$ and for runs connected by a path of length $2 \cdot 2^\rho + 1$ the height of their last stacks does not differ by more than $2 \cdot 2^\rho + 1$. Furthermore, due to 3., Lemma 2, and Remark 1, it follows that for all $r' \in N_{2^\rho}(r)$ we have $r' = r_{w_2} \circ r'_{w_2}$ for some w_2-prefixed r'_{w_2} and the function $\varphi : r' \mapsto r_{w_1} \circ r'_{w_2}[w_2/w_1]$ is an embedding of $N_{2^\rho}(r)$ into $N_{2^\rho}(s)$. For the same reasons, $\varphi^{-1} : r_{w_1} \circ r'_{w_1} \mapsto r_{w_2} \circ r'_{w_1}[w_1/w_2]$ for a w_1-prefixed run r'_{w_1} forms an embedding of $N_{2^\rho}(s)$ into $N_{2^\rho}(r)$. Finally, as $|\mathrm{last}(r)| > |\mathrm{last}(s)| \geq |w_1| > |\mathrm{last}(r_i)| + 2^\rho$, again by Lemma 2, r_i cannot be in the 2^ρ-neighbourhood of r and s. Hence, we may apply Lemma 1. □

Now we are going to prove a second $\simeq_\rho$-type preserving pumping lemma that preserves the last configuration of a run r, but reduces $\max(r)$.

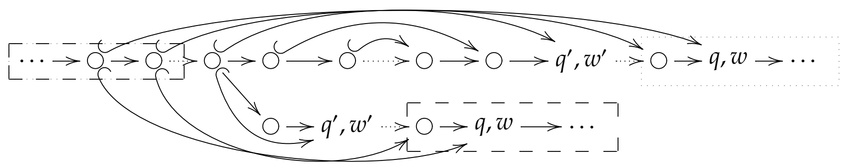

Fig. 1. Second pumping lemma: we replace the upper q, w by the lower one. The dotted / dashed boxes mark the neighbourhood of the upper / lower q, w

Lemma 4 (Second $\simeq_\rho$-Pumping Lemma). *Let* $\bar{r} = r_1, r_2, \ldots, r_m \in \mathrm{NPT}(P)$ *and* $r \in \mathrm{NPT}(P)$ *such that* $\max(r) > \max(r_i) + |Q|^2|\Sigma| + 1$ *for all* $1 \leq i \leq m$, *and such that* $\max(r) > |\mathrm{last}(r)| + |Q|^2|\Sigma| + 2^\rho + 1$. *Then there is some* $s \in \mathrm{NPT}(P)$ *such that* $\mathrm{last}(s) = \mathrm{last}(r)$, $\max(s) < \max(r)$, *and* $\bar{r}, r \simeq_\rho \bar{r}, s$.

Proof. We eliminate the last occurrence of a stack of length $\max(r)$ in r. For this purpose, let $i \in \mathrm{dom}(r)$ be maximal with $\mathrm{Stck}(r(i)) = w$ for $w \in \Sigma^*$ with $|w| > |Q^2||\Sigma| + 2^\rho + 1 + |\mathrm{last}(r)|$. Then for all $\mathrm{last}(r) \leq v \leq w$, the run r decomposes as $r = r_v \circ s_v \circ t_v$ such that $i \in \mathrm{dom}(r_v \circ s_v)$, s_v is v-prefixed, $s_v(0) = (q_v, v)$, $\mathrm{last}(s_v) = (p_v, v)$ for some $q_v, p_v \in Q$, and $|t(i)| < |v|$ for all $1 \leq i \leq \mathrm{length}(t)$. Then there are $v_1 < v_2 \leq w$ with

1. $\max(r_i) < |v_1|$
2. $|v_2| > |v_1| > |\mathrm{last}(r)| + 2^\rho$
3. $q_{v_1} = q_{v_2}, p_{v_1} = p_{v_2}$
4. the last letter of v_1 and v_2 is the same $\sigma \in \Sigma$.

Then we set $s'_{v_2} := s_{v_2}[v_2/v_1]$. Note that $s := r_{v_1} \circ s'_{v_2} \circ t_{v_1}$ is a well defined run. We use Lemma 1 to show that $\bar{r}, r \simeq_\rho \bar{r}, s$. We set

$$A := \{t \in N_{2^\rho}(r) : t = r_{v_1} \circ s_{v_1} \circ t', t' \text{ run}\} \quad B := \{t \in N_{2^\rho}(s) : t = r_{v_1} \circ s'_{v_2} \circ t', t' \text{ run}\} \ .$$

Note that $r_i \notin A \cup B$ as for all $t \in A \cup B$, we have $\max(t) \geq |v_1| > \max(r_i)$. From Lemma 2 it follows that for each run t' such that $r_{v_1} \circ s_{v_1} \circ t' \in N_{2^\rho}(r)$ or $r_{v_1} \circ s'_{v_2} \circ t' \in N_{2^\rho}(s)$ we have $|t'(i)| < |v_1|$ for $1 \leq i \leq \mathrm{length}(t')$. Hence, for $j := \mathrm{length}(r_{v_1} \circ s_{v_1})$ and every $t \in A$ we have $\mathrm{Stck}(t(j)) = v_1$, while for all $t \in B$ we have $|t(j)| < |v_1|$ as $\mathrm{length}(s_{v_1}) > \mathrm{length}(s'_{v_2})$. Thus, for all $a \in A$ and $b \in B$ the runs a and b disagree on a proper prefix of both elements, whence A and B cannot touch.

Now we claim that there is an isomorphism of the induced subgraphs $\varphi : A_{2^\rho}(r) \simeq B_{2^\rho}(s)$, given by $r_{v_1} \circ s_{v_1} \circ t' \mapsto r_{v_1} \circ s'_{v_2} \circ t'$. For this note that for any two runs t', t'' and for $* \in \{\rightarrow, \leftarrow, \hookrightarrow, \hookleftarrow\}$ we have

$$(r_{v_1} \circ s_{v_1} \circ t') * (r_{v_1} \circ s_{v_1} \circ t'') \quad \text{iff} \quad t' * t'' \quad \text{iff} \quad (r_{v_1} \circ s'_{v_2} \circ t') * (r_{v_1} \circ s'_{v_2} \circ t'') \ .$$

In order to apply the game argument, we finally have to show that edges between $A_{2^\rho - 1}(r)$ and $\mathrm{NPT}(P) \setminus A_{2^\rho}(r)$ are preserved under φ. Assume that $a \in A_{2^\rho-1}(r)$ and $c \in \mathrm{NPT}(P) \setminus \left(A_{2^\rho}(r) \cup B_{2^\rho}(s)\right)$. Note that $a \rightarrow c$ or $a \hookrightarrow c$ implies that a is a subrun of c and thus $c \in A_{2^\rho}(r)$ by definition of A. Assume that $c \rightarrow a$. then $|\mathrm{last}(c)| \leq |\mathrm{last}(r) + 2^\rho| < |v_1|$. Hence $c \neq r_{v_1} \circ s_{v_1}$. But as $r_{v_1} \circ s_{v_1}$ is a

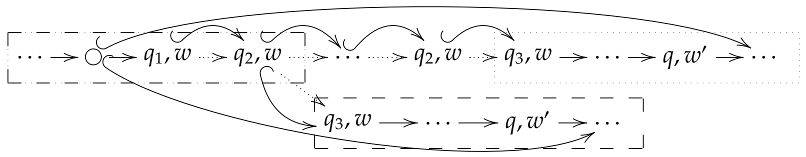

Fig. 2. Third pumping lemma: we replace the upper q, w' by the lower one. The dotted / dashed boxes mark the neighbourhood of the upper / lower q, w'

proper initial segment of a, this results in $c \in A_{2^\rho}(r)$. Thus, if $c \in \mathrm{NPT}(P) \setminus A_{2^\rho}(r)$ is connected to a then $c \hookrightarrow a$ and c is an initial segment of $r_{v_1} \circ s_{v_1}$. But as the last stack of a and c agree and $|\mathrm{last}(a)| < |v_1|$ then c is an initial segment of r_{v_1}. Thus, $c \hookrightarrow \varphi(a)$ as s_{v_1} and s'_{v_2} are both v_1-prefixed and $\mathrm{last}(a) = \mathrm{last}(\varphi(a)) < v_1$.

Now we found an $\simeq_\rho$-equivalent run s that is shorter than r. Iterating this process leads eventually to some run s with the desired properties □

Now we state our last pumping lemma, which decreases the number of occurrences of a given stack in a run r without affecting its $\simeq_\rho$-type. In order to do this we have to define what it means for a given stack w to occur often in a run r. We are going to count the occurrences of w as a stack in a w-prefixed subrun of r. Afterwards, we will see that bounding this number and $\max(r)$ is a sufficient condition to bound the total number of occurrences of a stack w in r.

Definition 5. *Let r be a run of the pushdown system $P = (Q, \Sigma, \Delta, (q_0, \bot))$ of length n. The number of occurrences of w in r is denoted $|r|_w := \left|\{i \in \mathbb{N} : \mathrm{Stck}(r(i)) = w\}\right|$. We set $\Xi(r) := \max\left\{|s|_w : w \in \Sigma^* \text{ and } s \text{ is a } w\text{-prefixed subrun of } r\right\}$.*

Lemma 5 (Third $\simeq_\rho$-Pumping Lemma). *Let $\bar{r} \in \mathrm{NPT}(P)$ such that $\Xi(r_i) \leq B$ for all $r_i \in \bar{r}$ and some $B \in \mathbb{N}$. For $r \in \mathrm{NPT}(P)$, there is some $s \in \mathrm{NPT}(P)$ such that* $\max(s) \leq \max(r), \mathrm{last}(s) = \mathrm{last}(r), \Xi(s) \leq B + (2^{\rho+1} + 2)|Q| + 2^\rho + 1,$ *and* $\bar{r}, r \simeq_\rho \bar{r}, s$.

Figure 2 gives an idea of the proof which is similar to that of the Lemma 4.

3.4 FO model checking on NPT is in 2-EXPSPACE

Using the three pumping lemmas we can now establish a "dynamic small witness property" for NPT: given the length of the runs representing parameters in a formula of quantifier rank ρ, we can bound the length of the run representing a witness for the first existential quantification occurring in the formula, if there is some witness for this quantification at all. The crucial point is that a bound on $\max(r)$ and a bound on $\Xi(r)$ yield a bound on the length of r:

Lemma 6. *Let $P = (Q, \Sigma, \Delta, (q_0, \bot))$ be a pushdown system and r a run of P such that* $\max(r) = h$ *and* $\Xi(r) = b$, *then* $\mathrm{length}(r) \leq \frac{b^{h+2} - b}{b-1}$.

Proof. Let $m_h := b$. For every $w \in \Sigma^h$ and some w-prefixed subrun s of r we have $\mathrm{length}(s) \leq m_h$ as the height of all stacks in s is h, whence all elements in s have stack w.

Now assume that every subrun t of r which is w-prefixed for some $w \in \Sigma^{n+1}$ has $\mathrm{length}(t) \leq m_{n+1}$. Let $w \in \Sigma^n$ be an arbitrary word and let s be a maximal w-prefixed subrun of r. Then there are $0 = e_1 < e_2 < \ldots < e_f < e_{f+1} = \mathrm{length}(s) + 1$ such that for $0 \leq i \leq f$ we have $\mathrm{Stck}(s(e_i)) = w$ and s restricted to (e_i, e_{i+1}) is w_i-prefixed for some $w_i \in \Sigma^{n+1}$. We have $f \leq b$ due to $\Xi(s) \leq \Xi(r) \leq b$. By assumption we get $\mathrm{length}(s) \leq (1 + m_{n+1})b$. Note that r is ε-prefixed, hence

$$\mathrm{length}(r) \leq m_0 = b + bm_1 = b + b^2 + b^2 m_2 = \ldots = m_h \sum_{i=0}^{h} b^i = \frac{b^{h+2} - b}{b - 1}. \quad \square$$

In the following we define our notion of a small run. Let $P = (Q, \Sigma, \Delta, (q_0, \bot))$ be a pushdown system. For $j \leq k \in \mathbb{N}$ we say that some $r \in \mathrm{NPT}(P)$ is (j,k)-*small* if

$$|\mathrm{last}(r)| \leq 6|P|^2 j 2^k, \qquad \max(r) \leq 8|P|^3 j 2^k, \qquad \text{and } \Xi(r) \leq 6|P| j 2^k \ .$$

Lemma 7. *Let $P = (Q, \Sigma, \Delta, (q_0, \bot))$ be a pushdown system, $\bar{r} = r_1, r_2, \ldots, r_i \in \mathrm{NPT}(P)$ and $i \leq k \in \mathbb{N}$. Then there are $\bar{r}' = r'_1, r'_2, \ldots, r'_i \in \mathrm{NPT}(P)$ such that every r'_j is (j,k)-small and $\bar{r} \simeq_{k-i} \bar{r}'$.*

The proof is by induction on i using the pumping lemmas.

With the bounds on the length of runs we can do FO model checking by brute force inspection of short runs. In order to check for an existential witness we only have to test all runs of bounded length. The bound depends on the number of parameters chosen before and on the size of the formula which we check. This means for a fixed quantifier in some formula φ we only have to check a finite initial part of the nested pushdown tree under consideration. Thus, we can give an alternating algorithm for FO model checking on NPT that works similar to the FO model checking algorithm on finite structures explained in [6].

Theorem 2. *The structure complexity of* FO *model checking on* NPT *is in* EXPSPACE*, while its expression and combined complexity are in* 2-EXPSPACE*.*

Proof. We assume that the i-th quantifier with respect to quantifier depth binds x_i. The algorithm ModelCheck (see next page), decides $\mathrm{NPT}(P) \models \varphi$. Due to Lemma 7, a straightforward induction shows that ModelCheck is correct. We analyse the space that this algorithm uses. Due to Lemma 6 an (i,k)-small run r has bounded length and we can store it as a list of $\exp(O(i|P|^4 k \exp(k)))$ many transitions. Thus, we need $\exp(O(i|P|^4 k \exp(k))) \log(P)$ space for storing one run. Additionally, we need space for checking whether such a list of transitions forms a valid run and for checking the atomic type of the runs. We can do this by simulation of P. The size of the stack is bounded by the size of the runs. Thus, the alternating algorithm ModelCheck is in

$$\mathrm{ASPACE}\Big(|\varphi| \log(|P|) \exp(O(|P|^4 |\varphi|^2 \exp(|\varphi|)))\Big) \subseteq \mathrm{ASPACE}\Big(\exp(O(|P|^4 \exp(2|\varphi|)))\Big) \ .$$

As the number of alternations is bounded by $|\varphi|$, we see by [5](Theorem 4.2) that FO model checking for NPT is in $\mathrm{DSPACE}\Big(\exp(O(|P|^4 \exp(2|\varphi|)))\Big)$.

```
Algorithm: ModelCheck($P, \alpha, \varphi$)
Input: pushdown system $P$, $\varphi \in \mathrm{FO}_\rho$, an assignment $\alpha : \mathrm{free}(\varphi) \to \mathrm{NPT}(P)$ such
       that $n = |\mathrm{dom}(\alpha)|$ and $\alpha(x_j)$ is $(j, \rho + n)$-small for each $j \leq n$
if $\varphi$ is an atom or negated atom then
    if $\mathrm{NPT}(P) \models \varphi[\alpha]$ then accept else reject;
if $\varphi = \varphi_1 \vee \varphi_2$ then guess $i \in \{1, 2\}$, and ModelCheck($P, \alpha, \varphi_i$);
if $\varphi = \varphi_1 \wedge \varphi_2$ then universally choose $i \in \{1, 2\}$, and ModelCheck($P, \alpha, \varphi_i$);
if $\varphi = \exists x_i \varphi_1$ then
    guess an $(i, k + n)$-small $a$ of $\mathrm{NPT}(P)$ and ModelCheck($P, \alpha[x_i \mapsto a], \varphi_1$);
if $\varphi = \forall x_i \varphi_1$ then
    universally choose an $(i, k + n)$-small $a$ of $\mathrm{NPT}(P)$ and
    ModelCheck($P, \alpha[x_i \mapsto a], \varphi_1$);
```

Algorithm 1: ModelCheck used in the proof of Theorem 2.

4 Conclusions

By tree-automaticity as well as pumping techniques we showed decidability of the FO model checking on NPT. Both approaches are transferable to some extent to the case of collapsible pushdown graphs. The tree-automaticity argument applies at least to level 2 of the hierarchy of collapsible pushdown automata. [3] But for arguments in the spirit of generation growth [4] combined with a result about counting abilities of higher-order pushdown systems[3], one obtains level 5 collapsible pushdown systems that are not tree-automatic. This raises the question of a characterisation of all tree-automatic collapsible pushdown graphs, especially for levels 3 and 4. Another open problem is effective FO model checking on collapsible pushdown graphs and whether pumping techniques lead to effective model checking algorithms on these graphs.

References

1. Rajeev Alur, Swarat Chaudhuri, and P. Madhusudan. Languages of nested trees. In *In Proc. 18th International Conference on Computer-Aided Verification, LNCS 4144*, pages 329–342. Springer, 2006.
2. Achim Blumensath. Automatic structures. diploma thesis, RWTH Aachen, 1999.
3. Achim Blumensath. On the structure of graphs in the caucal hierarchy. *Theor. Comput. Sci.*, 400(1-3):19–45, 2008.
4. Achim Blumensath and Erich Grädel. Automatic structures. In *Proc. 15th IEEE Symp. on Logic in Computer Science*, pages 51–62. IEEE Computer Society Press, 2000.
5. Ashok K. Chandra, Dexter C. Kozen, and Larry J. Stockmeyer. Alternation. *J. ACM*, 28(1):114–133, 1981.
6. E. Grädel. Finite model theory and descriptive complexity. In *Finite Model Theory and Its Applications*, pages 125–230. Springer-Verlag, 2007.
7. M. Hague, A. S. Murawski, C.-H. L. Ong, and O. Serre. Collapsible pushdown automata and recursion schemes. In *LICS '08: Proceedings of the 2008 23rd Annual IEEE Symposium on Logic in Computer Science*, pages 452–461, 2008.

[3] We obtained this result recently and hope to publish it soon.

Appendix with Omitted Proofs

A.1 Proof of Lemma 5

We start with an auxiliary lemma. Given runs r and s which are close to each other in some nested pushdown tree, the following lemma shows that the number of occurrences of a stack w in w-prefixed subruns of r and s have to be similar.

Lemma 8. *Let $r = r_1 \circ s_1 \circ u_1$ be a run such that s_1 is w-prefixed for some $w \in \Sigma^*$ and either* $\mathrm{length}(u_1) = 0$ *or $u_1(1) < w$. If $r \hookrightarrow s$, then $s = r_1 \circ s'_1 \circ u'_1$ where s'_1 is w-prefixed and* $\mathrm{length}(u'_1) = 0$ *or $u'_1(1) < w$ and $|s'_1|_w - |s_1|_w \in \{0, 1\}$.*

Proof. If $u_1(1) < w$, then $s = r_1 \circ s_1 \circ u_1 \circ u'$. Otherwise if $\mathrm{length}(u_1) = 0$, then $s = r_1 \circ s_1 \circ u'$ such that the last stacks of u' and s_1 agree, hence $w \leq \mathrm{last}(u') < u'(i)$ for all $1 \leq i < \mathrm{length}(u')$. Thus, $|s_1 \circ u'|_w \leq |s_1|_w + 1$. □

Note that the lemma also applies to runs r, s with $r \rightarrow s$ as in this case $|\mathrm{dom}(s) \setminus \mathrm{dom}(r)| = 1$, whence the number of occurrences of w can differ at most by one. Using this, we can prove our last pumping lemma, which bounds $\Xi(r)$. The proof relies on the fact that we find subruns r_1, r_2 with $\mathrm{last}(r_1) = \mathrm{last}(r_2)$ and $|r_1|_w$ much smaller than $|r_2|_w$ for some word w.

Proof (of third $\simeq_\rho$-pumping lemma). Assume $\Xi(r)$ is to big because $w \in \Sigma^*$ occurs to often. We decompose r as $r = r_1 \circ s_1 \circ t_1 \circ u_1$ such that s_1 starts and ends at the same configuration, i.e., $s_1(0) = \mathrm{last}(s_1) = (q, w)$ for a word $w \in \Sigma^*$ and a $q \in Q$. Furthermore, we choose the decomposition such that $s_1 \circ t_1$ is maximal w-prefixed, $|s_1|_w \geq 2^{\rho+1} + 2$, and $|s_1 t_1|_w = |s_1|_w + 2^\rho$. Set $s := r_1 \circ t_1 \circ u_1$. We claim that $r \simeq_\rho s$. The proof uses Lemma 1. Let

$$\begin{aligned}
A :=& \{t \in \mathrm{NPT}(P) : t = r_1 \circ s_1 \circ t, t \ \ w\text{-prefixed}, |t|_w \in [0, 2^{\rho+1}]\} \\
&\cup \{t \in \mathrm{NPT}(P) : t = r_1 \circ s_1 \circ t \circ u, t \ \ w\text{-prefixed}, |t|_w \in [0, 2^{\rho+1}], u(1) < w\} \\
B :=& \{t \in \mathrm{NPT}(P) : t = r_1 \circ t, t \ \ w\text{-prefixed}, |t|_w \in [0, 2^{\rho+1}]\} \\
&\cup \{t \in \mathrm{NPT}(P) : t = r_1 \circ t \circ u, t \ \ w\text{-prefixed}, |t|_w \in [0, 2^{\rho+1}], u(1) < w\} \ .
\end{aligned}$$

First note that $\bar{r} \in \mathrm{NPT}(P) \setminus (A \cup B)$ due to $\Xi(r_i) + 2^\rho < \Xi(s) < \Xi(r)$. Furthermore, A and B do not touch: for $a \in A, b \in B$ assume $b = r_1 \circ t' \circ u'$. Then the greatest common initial segment of a and b is an initial segment of $r_1 \circ t$ which is a proper initial segment of a and b. Thus there cannot be any edge between a and b. Otherwise $b = r_1 \circ t'$ and $\mathrm{last}(b) \leq w$ with $|t|_w \leq 2^{\rho+1}$. Then $a = r_1 \circ t' \circ t_a$ for some w-prefixed t_a and by Lemma 8 $|t' \circ t_a|_w \leq 2^{\rho+1} + 1$. On the other hand $a = r_1 \circ s_1 \circ t'$ with $|s_1 \circ t'|_w \geq |s_1|_w \geq 2^{\rho+1} + 2$. Thus, by contradiction A and B do not touch.

The map $\varphi : A \rightarrow B, r_1 \circ s_1 \circ t \mapsto r_1 \circ t$ is an isomorphism. For $* \in \{\rightarrow, \leftarrow, \hookrightarrow, \hookleftarrow\}$,

$$\begin{aligned}
& (r_1 \circ s_1 \circ t) * (r_1 \circ s_1 \circ t') \\
\text{iff} \quad & t * t' \\
\text{iff} \quad & (r_1 \circ t) * (r_1 \circ t') \ .
\end{aligned}$$

In order to apply Lemma 1, we have to show the preservation of edges between $A_{2^\rho-1}(r)$ and $\mathrm{NPT}(P) \setminus A_{2^\rho}(r)$. For this note that from Lemma 8 it follows that $a \in A_k(r)$ implies $a = r_1 \circ s_1 \circ t \circ u$ or $a = r_1 \circ s_1 \circ t$ for some w-prefixed t and $|t|_w \in [2^\rho - k, 2^\rho + k]$ and $u(1) < w$. Thus, for $a \in A_{2^\rho-1}(r), c \in \mathrm{NPT}(P) \setminus A_{2^\rho}(r)$ with $a * c$ for $* \in \{\rightarrow, \leftarrow, \hookrightarrow, \hookleftarrow\}$ Lemma 8 implies that $c \hookrightarrow a = r_1 \circ s_1 \circ t \circ u$, c is an initial segment of r_1, and $\mathrm{last}(a) < w$. But this is only the case if $c \hookrightarrow \varphi(a) = r_1 \circ t \circ u$ as s_1 is w-prefixed and $\mathrm{last}(a) < w$. Hence, the game argument shows that $\bar{r}, r \simeq_\rho \bar{r}, s$. Iteration of this process proves the lemma. □

A.2 Proof of Lemma 7

Proof. We prove the claim by induction on i. For $i = 0$ the claim is trivially true. Assume the claim is true for some $i \in \mathbb{N}$ and some $k \geq i + 1$. Let $\bar{r} = r_1, r_2, \dots, r_{i+1} \in \mathrm{NPT}(P)$, then there are $r'_1, r'_2, \dots, r'_i \in \mathrm{NPT}(P)$ such that $r_1, r_2, \dots, r_i \simeq_{k-i} r'_1, r'_2, \dots, r'_i$ and r'_j is (j,k)-small for all $1 \leq j \leq i$. Because of the $\simeq_{k-i}$-equivalence there is some element $a \in \mathrm{NPT}(P)$ such that $r_1, r_2, \dots, r_i, r_{i+1} \simeq_{k-j-1} r'_1, r'_2, \dots, r'_i, a$. The first $\simeq_n$-pumping lemma shows that we can choose this a in such a way that

$$|\mathrm{last}(a)| \leq 6|P|^2 i 2^k + |Q||\Sigma|(2 + 2^{(k-(i+1))+1}) + 2^{(k-(i+1))} + 1 \leq 6|P|^2(i+1)2^k \ .$$

Due to the second $\simeq_n$-pumping lemma, there is some $b \in \mathrm{NPT}(P)$ such that

$$\begin{aligned} &r'_1, r'_2, \dots, r'_i, a \simeq_{k-j-1} r'_1, r'_2, \dots, r'_i, b \ , \\ &\mathrm{last}(b) = \mathrm{last}(a) \ \text{, and} \\ &\max(b) \leq 8|P|^3 i 2^k + |Q|^2|\Sigma| + 1 \leq 8|P|^3(i+1)2^k \ . \end{aligned}$$

Now we may apply the third $\simeq_n$-pumping lemma and find some $c \in \mathrm{NPT}(P)$ such that

$$\begin{aligned} r'_1, r'_2, \dots, r'_i, b &\simeq_{k-j-1} r'_1, r'_2, \dots, r'_i, c \ , \\ \mathrm{last}(c) &= \mathrm{last}(b) \ , \\ \max(c) &\leq \max(b) \ \text{, and} \\ \Xi(c) &\leq 6|P| i 2^k + (2^{k-(i+1)+1} + 2)|Q| + 2^{k-(i+1)} + 1 \leq 6|P|(i+1)2^k \ . \end{aligned}$$

□