Noname manuscript No.
(will be inserted by the editor)

Parallel Complexity of CFL-Reachability Problem: Tractable Cases

Ekaterina Shemetova · Alexander Okhotin · Semyon Grigorev

Received: date / Accepted: date

Abstract Whereas it has been shown that context-free language (CFL) reachability problem is P-complete, there are some subclasses of context-free languages, for which CFL-reachability lies in NC complexity class. We show two common classes which generalize known examples of such tractable subclasses: bounded-oscillation languages and context-free languages with a poly-slender storage languages. Polynomiality of the rational indices of languages in these classes is proved. Polynomial time algorithm for deciding whether a given PDA has poly-slender storage language is given. Closure properties of tractable subclasses in terms of polynomial rational index are investigated.

Keywords CFL-reachability · parallel complexity · graphs · regular languages · context-free languages · context-free path queries

1 Introduction

The context-free language (CFL) reachability problem for a context-free grammar G and directed edge-labelled graph D consists of determining for pairs of nodes v and u whether v can reach u via a path labelled by a string in $L(G)$. It is an important problem underlying some fundamential static code analysis like inter-procedural data flow analysis, inter-procedural slicing and

E. Shemetova
St. Petersburg Academic University, ul. Khlopina, 8, Saint Petersburg 194021, Russia, and JetBrains Research
E-mail: katyacyfra@gmail.com

A. Okhotin
St. Petersburg State University, 7/9 Universitetskaya nab., Saint Petersburg 199034, Russia,

S. Grigorev
St. Petersburg State University, 7/9 Universitetskaya nab., Saint Petersburg 199034, Russia, and JetBrains Research
E-mail: s.v.grigoriev@spbu.ru

other [5, 21, 27], and graph database query evaluation [2, 12, 14, 36]. CFL-reachability is a kind of graph reachability problem with path constraints given by context-free languages. Unlike context-free language recognition, which is in NC (when context-free grammar is fixed), CFL-reachability is P-complete [26, 35]. Practically, it means that there is no efficient parallel algorithm for solving this problem (unless P $\neq$ NC).

While problem is not parallelizable in general, it is useful to develop more efficient parallel solutions for a fixed classes of context-free languages. For example, there are context-free languages which admit more efficient parallel algorithms in comparision with the general case of context-free recognition [16, 17, 23]. The same holds for CFL-reachability problem: there are some examples of context-free languages, for which CFL-reachability problem lies in NL complexity class (for example, linear and one-counter languages) [15, 19, 28].

CFL-reachability problem has long been known to be P-complete [11]. A parallel complexity of this problem is studied by both static code analysis [26, 27] and database communities [1, 32, 35]. First investigations of such type were made in terms of Datalog queries, because Datalog query can be represented via context-free grammar, while database can be considered as a graph. Important decidability result is obtained in [7]: deciding whether a query is in NC or P-complete, is undecidable. Ulman and Van Gelder in [32] introduce a notion of a *polynomial fringe property* and shows that Datalog queries having this property are in NC. A query π has the *polynomial fringe property* if and only if there is a polynomial p such that, for each regular language R recognized by an automoton with n states, $L(\pi) \cap R$ is either empty or contains a word shorter than $p(n)$. It is undecidable whether an elementary chain rule program has the polynomial fringe property. Their important results can be reinterpreted in terms of CFL-reachability as follows:

1. CFL-reachability for linear languages and piecewise linear languages (each rule of the grammar has at most one recursive subgoal) and for arbitrary graphs is in NC, hence these languges have polynomial fringe property
2. The same holds for D_1 (the Dyck language on one kind of parentheses) and its GSM-mappings (one-counter languages)
3. CFL-reachability for D_2 (the Dyck language on two kinds of parentheses) is P-complete.

Last result is important because any CFL can be represented as a D_2, so it is the direct consequence of P-competeness of CLF-reachability in general. Also, using the fact that D_2 is included in many interesting subclasses of context-free languages, such as visibly pushdown languages [23], simply deterministic languages (which is LL(1)) we can state that CFL-reachability for these languages is P-complete. Afrati et al. in [1] investigate parallel complexity of Datalog simple chain queries and presents Polynomial Stack Lemma which will be discussed in detail in Section 4.

The definition of polynomial fringe property coinsides with the notion of so called *rational index*: for a context-free language having the polynomial

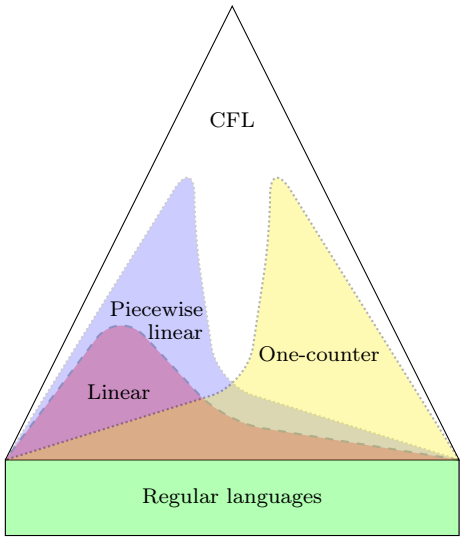

Fig. 1 The hierarchy of languages, for which CFL-reachability problem is in NC .

rational index is the same as having polynomial fringe property. More precisely, rational index $\rho_L(n)$ is a function, which denotes maximum length of the shortest word in $L(\pi) \cap R$ for arbitrary R recognized by n state automaton. The notion of rational index was introduced in [4] as a complexity measure for context-free languages and was investigated independently from the polynomial fringe property. Particulary, it have been proved that the rational index of D_1 is in $O(n^2)$ [6] and the rational index of languages generating all context-free languages (example of such language is D_2) belongs to $exp(\Theta(n^2/\ln n))$ [24]. An example of non-generating language with exponential rational index is given in [28]. Also it have been shown that for every algebraic number γ the language with the rational index in $\Theta(n^\gamma)$ exists [25].

CFL-reachability problem can be observed as an intersection non-emptiness problem for context-free language (pushdown automaton) and regular language (finite automoton), because labelled graph is a special kind of nondeterministic finite automata. Complexity of such problem is studied in [8, 29, 33].

For surveys investigating computational complexity of the language reachability for different variants of languages (regular, context-free, context-sensitive) and graphs (acyclic graphs, trees, grid graphs) we refer reader to [3, 15, 19].

Our focus is on investigating the parallel complexity of CFL-reachability. Especially we are interested in generalization of "easy" subclasses and discovering new examples of context-free languages, for which CFL-reachability is in NC. Effective subclasses can be useful in practice, because general problem is not tractable [20]. Also it is natural to ask which properties of such subclasses implies parallel effectiveness. A rational index and a fringe property are known as examples of such properties, but there is a lack of common characteristics in terms of pushdown automata for subclasses with the good rational index and

fringe property. In other words, why some languages have polynomial rational indices? What is the difference between them and other subclasses of context free-languages?

Hierarchy of previously founded examples of context-free languages, for which CFL-reachability problem is in NC, is presented on Figure 1. Linear (and piecewise linear) languages and one-counter languages form distinct families of context-free languages, but both have polynomial rational indices (polynomial fringe property). Notice that one reason for this is that they are defined by natural restrictions on their pushdown automata (PDA). Restricting a PDA such that the height of its stack is only allowed to increase and then to decrease, thus performing only one turn, leads to the definition of one-turn PDAs. It is known that these PDAs can be characterized by linear grammars. One-counter languages are exactly languages recognized by PDA with only one stack symbol. But some of the restrictions on the pushdown store cannot be simulated by others and this is exactly the case of linear and one-counter languages. Our main idea to obtain generalizations of known tractable classes via investigation of restrictions on PDA store.

Our contributions. Our results can be summarized as follows:

- We show that CFL-reachability problem for bounded-oscillation languages, which were introduced in [9], is in NC. This class generalizes the case of linear languages.
- We introduce a new subclass of context-free languages — context-free languages with a poly-slender pushdown store language. These languages are the natural generalization of one-counter languages and CFL-reachability problem for them is in NC. Decidability of deciding poly-slenderness of pushdown store language is proved and polynomial time algorithm are given. Also we obtain polynomial time algorithm for deciding where a given finite automaton recognize poly-slender rational language.
- Closure properties of the languages with polynomial rational indices are investigated, particulary it is shown that family of the languages with polynomial rational indices is a full AFL.

Organization. This paper is organized as follows. In Section 2, we present the basic definitions from formal languages used in this paper. In Section 3 we study bounded-oscillation languages and show that CFL-reachability problem for them is in NC. In Section 4 we introduce context-free languages with a poly-slender pushdown store language and investigate some of their properties. In Section 5 our focus is on closure properties of the languages with polynomial rational indices. We conclude in Section 6 with a summary of our findings.

2 Preliminaries

Formal languages A *context-free grammar* is a triple $G = (\Sigma, N, P)$, where Σ is a finite set of alphabet symbols, N is a set of nonterminal symbols, P is a set of production rules. $L(G)$ is a context-free language generated by

context-free grammar G. Notice that usually context-free grammar is defined with a fixed start nonterminal. G_A denotes a context-free grammar with fixed start nonterminal $A \in N$, $L(G_A)$ is a language generated by a grammar G_A, respectively. We use the notation $A \stackrel{*}{\Rightarrow} w$ to denote that the string $w \in \Sigma^*$ can be derived from a non-terminal A by sequence of applying the production rules from P. A *parse tree* is an entity which represents the structure of the derivation of a terminal string from some nonterminal.

A grammar G is said to be is in the *Chomsky normal form*, if all production rules of P are of the form: $A \to BC$ or $A \to a$, where $A, B, C \in N$ and $a \in \Sigma$.

Notice that the rules of the form $A \to \varepsilon$, where ε denotes an empty string, are omitted. This is because it is trivial to find an empty string in the case of graph input.

Context-free languages are accepted by pushdown automata (PDA). *Pushdown automaton* is a 7-tuple $M = (Q, \Sigma, \Gamma, \delta, q_0, Z, F)$, where Q is a finite set of states, Σ is a input alphabet, Γ is a finite set which is called the stack alphabet, δ is a finite subset of $Q \times (\Sigma \cap \{\varepsilon\}) \times \Gamma \times Q \times \Gamma^*$, $q_0 \in Q$ is the start state, $Z \in \Gamma$ is the initial stack symbol and $F \subseteq Q$ is the set of accepting states.

Some operations on languages will be mentioned during this paper.

A *homomorphism* is a function $h : \Sigma^* \to \Delta^*$ defined as follows:

- $h(\varepsilon) = \varepsilon$ and for $a \in \Sigma$, $h(a)$ is any string in Δ^*,
- for $a = a_1 a_2 ... a_k \in \Sigma^*$ $(k \geq 2)$, $h(a) = h(a_1)h(a_2)...h(a_k)$.

Given a homomorphism $h : \Sigma^* \to \Delta^*$ and a language L define $h(L) = \{h(w) \mid w \in L\} \subseteq \Delta^*$.
Insertion of a class of a language K into a language L is a language $L' = \{uxv \mid x \in K, u, v \in L\}$

A *full trio* is a family of languges is closed under arbitrary homomorphism and intersection with regular language. A *full AFL (abstract family of languages)* is a full trio closed under union, concatenation and the Kleene plus.

A *regular language* is a language that can be expressed with a regular expression or a deterministic or non-deterministic finite automata. An *nondeterministic finite automata* (NFA) is represented by a 5-tuple, $(Q, \Sigma, \Delta, q_0, F)$, where Q is a finite set of states, Σ is a finite set of input symbols, $\Delta : Q \times \Sigma \to 2^{|Q|}$ is a transition function, $q_0 \in Q$ is a start state, $F \subseteq Q$ is a set of accepting (final) states. *Deterministic fininite automaton* is NFA with the following restrictions: each of its transitions is uniquely determined by its source state and input symbol, and reading an input symbol is required for each state transition.

For a language L over an alphabet Σ, its rational index ρ_L is a function defined as follows:

$$\rho_L(n) = \max\{\min\{|w| : w \in L \cap K\}, K \in Rat_n, L \cap K \neq \emptyset\}, \qquad (1)$$

where $|w|$ is the length of a word w and Rat_n denotes the set of regular languages on an alphabet Σ, recognized by a finite nondeterministic automation

with at most n states. $w(\rho_L(n))$ denotes word w, whose length is the value of rational index of L.

Context-free language reachability A *directed labelled graph* is a triple $D = (Q, \Sigma, \delta)$, where Q is a finite set of nodes, Σ is a finite set of alphabet symbols, and $\delta \subseteq Q \times \Sigma \times Q$ is a finite set of labeled edges.

Let $i\pi j$ denote a unique path between nodes i and j of the input graph and $l(\pi)$ denote a unique string which is obtained from concatenation of edge labels along the path π. Then the general formulation of CFL-reachability can be stated as follows.

Definition 1 Let $G = (\Sigma, N, P)$ be a context-free grammar and $D = (Q, \Sigma, \delta)$ be a directed labelled graph. Given two nodes i and j we say that j is *reachable* from i if there exists a path $i\pi j$, such that $l(i\pi j) \in L(G)$.

When this problem is restricted to some language L (not necessary context-free), it is called *L-reachability*. There are four varieties of CFL-reachability problems: all-pairs problem, single-source problem, single-target problem and single-source/single-target problem [27]. In this paper we consider all-pairs problem. The *all-pairs problem* is to determine all pairs of nodes i and j such that j is reachable from i. Notice that our definition of context-free grammar doesn't include a start nonterminal: one may be interested in reachability for every $L(G_A)$, where $A \in N$. In other words, we need to find all triples (A, i, j) for which j is reachable from i via path labelled from a string from $L(G_A)$.

3 Bounded-oscillation languages

Bounded-oscillation languages were introduced in [9]. Just like one-counter and linear languages, it is defined by restriction on the pushdown automata. This restriction is based on the notion of *oscillation*, a special measure of how variable is the stack height over time. Oscillation is defined using a hierarchy of *harmonics*. Let $\bar{a}$ be a *push*-move and a be a *pop*-move. Then PDA run can be described by well-nested word of $\bar{a}$-s and a-s. Order 1 harmonic h_1 is $\bar{a}a$ $\bar{a}a$ (*push pop push pop*), order 2 harmonic h_2 is $\bar{a}$ <order 1 harmonic> a $\bar{a}$ <order 1 harmonic> a, so $(i+1)$-st harmonic is $\bar{a}$ h_i a $\bar{a}$ h_i a.

PDA run r is *k-oscillating* if the harmonic of order k is the greatest harmonic that is contained in r. *bounded-oscillation languages* are languages accepted by pushdown automata restricted to k-oscillating runs. It is important that the problem whether a given CFL is a bounded-oscillation language is undecidable [9].

Example 1 Consider Figure 2. It shows how the stack height changes during the run of PDA. Corresponding well-nested word is $\bar{a}$ $\bar{a}$ $\bar{a}a$ $\bar{a}a$ a $\bar{a}a$ a. The greatest harmonic in this word is 1 order harmonic: $\mathbf{\bar{a}}$ $\bar{a}$ $\mathbf{\bar{a}a}$ $\mathbf{\bar{a}a}$ a $\bar{a}a$ $\mathbf{a}$, therefore oscillation of the run is 1.

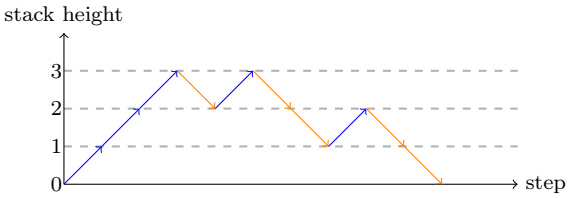

Fig. 2 Stack heights during the run of PDA.

Oscillation of the run is closely related with the *dimension* of the corresponding parse tree. For each node v in a tree T a dimension $dim(v)$ is inductively defined as follows:

- If v is a leaf, then $dim(v) = 0$
- If v is an internal node with k children $v_1, v_2, ..., v_k$ for $k \geq 1$, then

$$dim(v) = \begin{cases} \max_{i \in \{1...k\}} dim(v_i) & \text{if there is a unique maximum} \\ \max_{i \in \{1...k\}} dim(v_i) + 1 & \text{otherwise} \end{cases} \quad (2)$$

Dimension of a parse tree T $dim(T)$ is a dimention of it's root. It is observable from the definition that dimension of a tree T is the height of the largest perfect binary tree, which can be obtained from T by contracting edges and accordingly identifying vertices. A tree with dimension $dim(T) = 2$ is illustrated in Figure 3.

It is known that the dimension of parse trees and the oscillation defined on PDA runs are in linear relationship.

Lemma 1 (Ganty, Valput) *Let a grammar $G = (\Sigma, N, P)$ be in Chomsky normal form and let T be a parse tree of G. We have that $osc(T) - 1 \leq dim(T) \leq 2osc(T)$.*

It is not hard to see that if $L(G)$ is k-bounded-oscillation language, parse trees of G always have a bounded dimension. Before we consider the value of the rational index for k-bounded-oscillation languages, we need to prove the following.

Lemma 2 *Let $G = (\Sigma, N, P)$ be a context-free grammar, $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. Then a height of a parse tree for $w(\rho_{L(G)})$ does not exceed $|N|n^2$.*

Proof Assume that the parse tree for $w(\rho_{L(G)})$ has a height of more than $|N|n^2$. There are $|N|n^2$ unique labels (A, i, j) for nodes of the parse tree, so according to the pigeonhole principle, the parse tree for $w(\rho_{L(G)})$ contains at least one subtree T with label (A, i, j) at the root, which has a subtree T' with the same label. Then we can change T with T' and get a new string w' which is shorter than $w(\rho_{L(G)})$. But $w(\rho_{L(G)})$ is the shortest, then we have a contradiction.

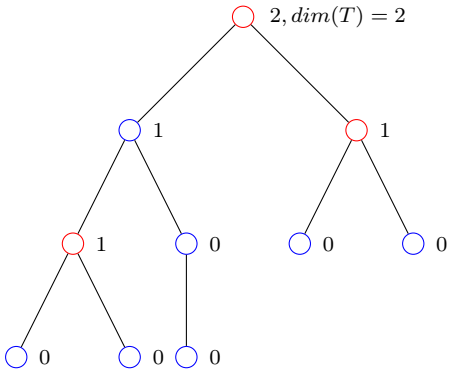

Fig. 3 A tree T with $dim(T) = 2$.

From Lemma 2 we have that rational index of linear languages is in $O(n^2)$.

Theorem 1 *Let L be a k-bounded-oscillation language with grammar $G = (\Sigma, N, P)$ in Chomsky normal form and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. Then $\rho_{L(G)}$ is in $O((|N|n^2)^{k/2})$.*

Proof Proof by induction on dimension $dim(T)$ (if oscillation is bounded by k, then dimension value is at most $2k$).
Basis. $dim = 1$.
Consider the worst-case tree T with the dimension $dim(T) = 1$. The root of the tree has the same dimension and has two children (because the grammar in Chomsky normal form). There are two cases: first, when both of child nodes have dimension equal to 0, then the tree has only two leaves and second, when one of children has a dimension 1, and the second child has a dimension equal to 0. For the second case we can recursively construct a tree of maximum height $|N|n^2$ (Lemma 2). Every internal node of such tree has two children, one of wich has dimension equal to 0 and therefore has only one leaf. This tree is exactly the worst-case tree for linear grammar in Chomsky normal form, so the number of leaves in such tree is $O(h) = O(|N|n^2)$, where h is the height of the tree. Thanks to Lemma 1, the maximum value of oscillation for $dim(T) = 1$ is 2, so $\mathcal{L}$ is no more than $O(h^{k/2}) = O((|N|n^2)^{k/2}) = O(|N|n^2)$.
Inductive step. $dim = d + 1$.
Assume $\mathcal{L}$ is no more than $O(h^d)$ for every d, where h is the height of the tree. We have two cases for the root node with dimension equal to $d+1$: 1) both of children have a dimension equal to d, then by proposition the tree of heght h has no more than $O(h^d)$ leaves; 2) one of children has a dimension $d+1$, and the second child v has a dimension $dim(v) \leq d$. Again, a tree of maximum height with the maximum number of leaves can be constructed recursively: each node of such tree has two children u and v with dimension $d + 1$ and d respectively (the more value of dimension of the node, the more leaves in the corresponding tree). By proposition we have no more than $(h-1)^d+(h-2)^d+(h-3)^d+...+1 =$

$O(h^{d+1})$ leaves, so proposition holds for $dim = d + 1$. Finally, we have $\mathcal{L}$ is no more than $O(h^d) = O((|N|n^2)^{k/2})$ for every k.

As we can see from the proof above, the family of linear languages is included in the family of bounded-oscillation languages. The reason is that the family of bounded-oscillation languages generalizes the family of languages accepted by finite-turn pushdown automata [9]. It is interesting that for arbitrary CFL, particularly for D_2 , the value of osciilation is not constant-bounded: it depends on the length of input and does not exeed $O(\log n)$ for the input of length n [13, 34].

Example 2 (Superlinear languages.)
Let $G = (\Sigma, N, P)$ be a context-free grammar. G is *superlinear* if all productions of P satisfy these conditions:

1. there is a subset $N_L \subseteq N$ such that every $A \in N_L$ has only linear productions $A \to aB$ or $A \to Ba$, where $B \in N_L$ and $a \in \Sigma$.
2. if $A \in N \setminus N_L$, then A can have non-linear productions of the form $A \to BC$ where $B \in N_L$ and $C \in N$, or linear productions of the form $A \to \alpha B \mid B\alpha \mid \alpha$ for $B \in N_L$, $\alpha \in \Sigma^*$.

From the grammar G it is observable that its parse trees have dimension at most 2. From Theorem 1, if dimension of all parse trees is bounded by some k then the rational index of such language is polynomial, so CFL-reachability problem for superlinear languages and arbitrary graph is in NC.

4 Languages with poly-slender storage languages

In previous section restriction in terms of variability of stack height of PDA was described. But it is not the case for D_1, which is a k-oscillating CFL for no k. In this section another kind of stack restriction is considered — poly-slenderness of a pushdown storage language as a measure of how variable is stack contents along accepting computations of PDA.

For PDA M, its *pushdown store language* $P(M)$ consists of all words occurring on the pushdown store along accepting computations of M. It is well-known that store language of any PDA is a regular language. D_1 is a one-counter language, so its pushdown store language is obviously Z^*Z_0, where Z is a single pushdown symbol and Z_0 is a bottom-of-pushdown symbol Z_0.

Afrati et. al in [1] give a notion of *polynomial stack property* and show that if a PDA has polynomial stack property, then corresponding query has polynomial fringe property (and hence, lies in NC complexity class). PDA has polynomial stack property iff the largest possible number of different contents of the same height k along the accepting computations of PDA M is bounded by polynomial $O(k^d)$ for $d \geq 0$. For, example PDA for D_1 has polynomial stack property, because there is the only one possible variant of contents for every height of stack.

Notice that for PDA polynomial stack property is equivalent to having *poly-slender* pushdown store language (or storge language with polynomial density). Density of a language shows the number of words of length n in language. Language $L \subseteq \Sigma^*$ is called *poly-slender language (or language with polynomial density)* if function $|L \cap \Sigma^n|$ is bounded by $O(n^k)$ for some $k \geq 0$. For example, the language Z^*Z_0 is of polynomial density (even of a constant density), whereas the language $(Z_1 + Z_2)^* Z_0$ is of exponential density.

Thus we can reformulate The Polynomial Stack Lemma from [1] as follows:

Lemma 3 *Let L be a context-free language and M be a PDA recognising it. If pushdown storage language $P(M)$ is a poly-slender regular language, then CFL-reachability problem for L and arbitrary graph is in NC complexity class.*

While deciding whether a query has the polynomial fringe property is undecidable [32], it is decidable in polynomial time whether a given PDA has a poly-slender storage language and hence polynomial stack property.

At first we show how to use Ufnarovskii's criterion for the growth of directed graphs [31] to decide in polynomial time whether a given NFA recognizes poly-slender language.

Let $G = (V, E)$ be a directed graph, then *growth function of graph* $r_G(n)$ defines the number of all paths of length at most n in G. Vertex is called *cyclic* if it occurs in some cycle, *double cyclic* if at least two distinct cycles pass through it (these cycles should form distinct graphs, not paths). Graph G is *cyclically simple* if it has no doubly cyclic vertices (hence, any distinct cycles have no common vertices).

Lemma 4 (Ufnarovskii's criterion.) *For a directed graph G growth function $r_G(n)$ is either polynomial or exponential. More precisely:*

- *$r_G(n)$ is exponential iff G has a double cyclic vertex*
- *$r_G(n)$ is polynomial iff G is cyclically simple*

Theorem 2 *Given an NFA F it is decidable in polynomial time whether F recognizes poly-slender language.*

Proof Consider a directed graph $G = (V, E)$ corresponding to the given NFA F. Cyclicical simplicity of G can be easily checked by running DFS from each vertex of the graph. Then using Ufnarovskii's criterion (Lemma 4), one defines is $r_G(n)$ exponential or polynomial. F recognizes poly-slender language if number of all accepting computatation paths of length n is bounded by some polynomial $O(n^k)$. Let $a_G(n)$ be the number of accepting paths of length at most n in F. It is left to show that if $r_G(n)$ is polynomial (exponential) then $a_G(n)$ is polynomial (exponential).
It is easy to see that $a_G(n) \leq r_G(n)$, because $r_G(n)$ counts all paths in graph and not every path is accepting. Clearly, if $r_G(n)$ is polynomial then $a_G(n)$ is polynomial. Every non-accepting path is a subpath of an accepting path (otherwise it can be deleted from NFA). An accepting path of length n has at most $\binom{n+1}{2} - 1 = \frac{(n+1)n}{2} - 1$ distinct subpaths. The length of any subpath

Fig. 4 NFAs for languages Z^*Z_0 (left) and $(Z_1 + Z_2)^*Z_0$ (right).

is bounded by n. So we have $r_G(n) \leq a_G(n)(\frac{(n+1)n}{2} - 1)n \leq (n^3 + n^2)a_G(n)$. Thus if $r_G(n)$ is exponential then $a_G(n)$ is exponential. This completes the proof.

Example 3 (Ufnarovskii's criterion and a polynomial density) NFAs recognising poly-slender language Z^*Z_0 and a regular language with an exponential density $(Z_1 + Z_2)^*Z_0$ are presented on Figure 4. The graph of the first automotion is cyclically simple and the graph of the second automotion has a double cycled vertex q_0.
It is well-known result that regular language is poly-slender if and only if it can be represented as a finite union of the regular expressions of the form $xy_1^*z_1...y_t^*z_t$, where $t \geq 0$ and $x, y_1, z_1, ..., y_t, z_t$ are words in Σ^* [30]. NFA for every such expression is clearly cyclically simple.

Using the fact that for a given PDA M, NFA for $P(M)$ can be constructed in deterministic polynomial time (size of NFA is quadratic in the number of states and linear in the number of pushdown symbols of M) [10, 22] and Theorem 2 we immediatly deduce the following.

Corollary 1 *Given a pushdown automotion M, it can be decided whether $P(M)$ is poly-slender in polynomial time.*

5 Closure properties of languages with polynomial rational indices

Given a context-free language L having polynomial rational index it is interesting to find which language operations preserve this property. Boasson et al. [4] give following useful relations for polynomial indices of two languages L and L':

- $\rho_{L\cup L'} \leq \max(\rho_L, \rho_{L'})$
- $\rho_{LL'} \leq \rho_L + \rho_{L'}$
- $\rho_{L\cap R}(n) \leq \rho_L(nm)$, where R is a regular language recognised by an m state automaton
- $\rho_{h(L)}(n) \leq \rho_L(n)$ and $\rho_{h^{-1}(L)}(n) < n(\rho_L(n) + 1)$, where $h : \Sigma^* \to \Delta^*$ is a homomorphism.

Theorem 3 *Context-free languages with polynomial rational indices are closed under intersection and quotient with a regular language, union, reversal, substitution, concatenation, Kleene plus, prefix and insertion of a regular language.*

Proof At first, from the relations above it is easy observable that the family of context-free languages with polynomial rational indices is a full trio. Every full trio is closed under prefix and quotient with regular languages. Obviously, CFLs with polynomial rational indices languages are closed under reversal.
Next we show that context-free languages with polynomial rational indices are closed under Kleene plus (we consider this variation on the Kleene star operation because empty strings are omitted in the definition of CFL-reachability problem).
Kleene star. Let L be a language with polynomial rational index. Consider language L^+ with the start nonterminal S. By definition of the Kleene plus operation, a rightmost derivation from S generates a sequence of one or more start nonterminals A from L_{G_A}, each of which generates some string in L. Let D be a directed labelled graph with n nodes. Suppose there are nodes u and v in D such that:

1. v is not L_{G_A}-reachable from u and
2. v is L^+-reachable from u

Then v is reachable from u via concatenation of words in L_{G_A}. Consider the longest shortest path $u\pi_1 v$ between u and v. It can be obtained by joining $(A, u, i), (A, i, j), ..., (A, w, v)$ into (S, u, v). If L has the polynomial rational index, then for every triple (A, i, j) corresponding shortest path $i\pi j$ has at most polynomial length. There are no more than $O(n)$ such triples in concatenation because there are no repetitions of the same node in the sequence of start and end nodes of triples (otherwise $u\pi_1 v$ is not the shortest path, for example, path $u \to i \to k \to l \to i \to j \to v$ can be replaced with shorter path $u \to i \to j \to v$), so $u\pi_1 v$ has at most polynomial length. In other words, we have $\rho_{L^+}(n) \leq n(\rho_L(n))$.
Family of languages with polynomial rational indices is a full trio closed under union, concatenation and Kleene star, therefore it is a full ALF. Full AFLs is known to be closed under substitution.
Insertion of a regular language. To prove closure under insertion of a regular language, the following PDA can be constructed. Let L be a context-free language with polynomial rational index and let M be a PDA recognising L. New PDA M' for insertion of a regular language R recognised by finite automation F into L can be obtained as follows: duplicate all states in M, initial state is placed in first set and final states reside in the second set. Every state in the first set has its own copy of outgoing arcs of the initial state of F. Every ingoing arc of final state of F is connected directly to every state of the second set of M. In other words, every state from the first set is the inial state of F and every state from the second set is a final state of F. All acs from F are labelled the same way as they are labelled in F. Consider intersection of M'

and arbitrary finite automotion F' with n states (to deal with ε-labelled arcs, loops with ε can be added to every state of finite automotion). The longest non-empty shortest path $i\pi j$ in the intersection of M' and F' consists of three sub-paths: $i\pi_1 k$, $k\pi_2 m$ and $m\pi_3 j$, where $i\pi_1 k$ ($m\pi_3 j$) is the shortest paths in the intersection of F' and first (second) set of states of M respectively, and $k\pi_2 m$ is the shortest path in the intersection of F' and F. $k\pi_2 m$ has at most polynomial length because all regular languges have polynomial fringe property, $i\pi_1 k$ and $m\pi_3 j$ have polynomial length because M is a PDA for language with rational index, therefore $i\pi j$ has polynomial length.

Using closure properties, it is easier to find new subclasses of context-free languages for which CFL-reachability problem is in NC.

Example 4 (Metalinear languages.)
Let $G = (\Sigma, N, P, S)$ be a context-free grammar. G is *meatalinear* if all productions of P are of the following forms:

1. $S \to A_1 A_2 ... A_k$, where $A_i \in N - \{S\}$
2. $A \to u$, where $A \in N - \{S\}$ and $u \in (\Sigma^*((N - \{S\}) \cup \varepsilon)\Sigma^*)$

The width of a metalinear grammar is $max\{k \mid S \to A_1 A_2 ... A_k\}$. Metalinear languages of width 1 are obviously linear languages. It is easy to see that every metalinear language is a concatenation of k linear languages. Linear languages have polynomial rational index, CFLs with polynomial rational index are closed under concatenation, so metalinear languages have polynomial rational index.

6 Conclusions and open problems

We have obtained two large classes for which CFL-reachability problem is in NC. The first is the class of bounded-oscillation languages, which generalizes linear languages. The second class is context-free languages with a poly-slender pushdown store languages, which is generalization of one-counter languages. Recall that regular languages have polynomial fringe property (and are accerted by PDA whith a bounded stack height), also It is known that L-reachibility for regular languages is in $NL \subseteq NC^2$ [19, 35]. Thereby it has been demonstrated that some natural restrictions on the pushdown storage implies polynomial rational index for corresponding context-free language: low variability of stack height during the PDA run (bounded-oscillation PDA) and low variety of stack contents (languages with a poly-slender pushdown store languages).

It will be intersting to know is there another kind of stack restiction which implies polynomial rational index? Or is there a context-free language which does not belong to any mentioned class and is not obtained by applying language operation on them? Are there any other properties (except polynomial rational index) which implies parallel effectiveness of language? For example there is a Datalog query, which doesn't have a polynomial fringe property but

its evaluation is in NC [18]. On the other hand, does any strict property of a context-free language, which means P-completeness of CFL-reachability for it exist? Of course, such property should be undecidable.

We considered CFL-reachability problem for a fixed context-free languages and arbitrary graphs. What tractable cases can be obtained for a fixed graphs and arbitrary context-free language? The known and trivial examples are acyclic graphs and trees. Can we have more complicated classes of graphs for which CFL-problem is in NC? Interesting algebraic properties of such graphs (NFA) are given in [8], but there is a lack of descriptions of these properties in terms of graphs (NFA).

References

1. Afrati F, Papadimitriou C (1987) The parallel complexity of simple chain queries. In: Proceedings of the Sixth ACM SIGACT-SIGMOD-SIGART Symposium on Principles of Database Systems, ACM, New York, NY, USA, PODS '87, pp 210–213, DOI 10.1145/28659.28682, URL http://doi.acm.org/10.1145/28659.28682
2. Azimov R, Grigorev S (2018) Context-free path querying by matrix multiplication. In: Proceedings of the 1st ACM SIGMOD Joint International Workshop on Graph Data Management Experiences & Systems (GRADES) and Network Data Analytics (NDA), ACM, New York, NY, USA, GRADES-NDA '18, pp 5:1–5:10, DOI 10.1145/3210259.3210264, URL http://doi.acm.org/10.1145/3210259.3210264
3. Barrett C, Jacob R, Marathe M (2000) Formal-language-constrained path problems. SIAM J Comput 30:809–837, DOI 10.1137/S0097539798337716
4. Boasson L, Courcelle B, Nivat M (1981) The rational index: a complexity measure for languages. SIAM Journal on Computing 10(2):284–296
5. Chatterjee K, Choudhary B, Pavlogiannis A (2017) Optimal dyck reachability for data-dependence and alias analysis. Proc ACM Program Lang 2(POPL):30:1–30:30, DOI 10.1145/3158118, URL http://doi.acm.org/10.1145/3158118
6. Deleage JL, Pierre L (1986) The rational index of the dyck language $D_1{}^*$. Theoretical Computer Science 47:335 – 343, DOI 10.1016/0304-3975(86)90158-1
7. Gaifman H, Mairson H, Sagiv Y, Vardi M (1987) Undecidable optimization problems for database logic programs. vol 40, pp 106–115, DOI 10.1145/174130.174142
8. Ganardi M, Hucke D, König D, Lohrey M (2016) Circuit evaluation for finite semirings. 1602.04560
9. Ganty P, Valput D (2016) Bounded-oscillation pushdown automata. Electronic Proceedings in Theoretical Computer Science 226:178–197, DOI 10.4204/eptcs.226.13, URL http://dx.doi.org/10.4204/EPTCS.226.13
10. Geffert V, Malcher A, Meckel K, Mereghetti C, Palano B (2013) A direct construction of finite state automata for pushdown store languages. In:

Jurgensen H, Reis R (eds) Descriptional Complexity of Formal Systems, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 90–101
11. Greenlaw R, Hoover HJ, Ruzzo WL (1995) Limits to Parallel Computation: P-completeness Theory. Oxford University Press, Inc., New York, NY, USA
12. Grigorev S, Ragozina A (2017) Context-free path querying with structural representation of result. In: Proceedings of the 13th Central & Eastern European Software Engineering Conference in Russia, ACM, New York, NY, USA, CEE-SECR '17, pp 10:1–10:7, DOI 10.1145/3166094.3166104, URL http://doi.acm.org/10.1145/3166094.3166104
13. Gundermann T (1985) A lower bound on the oscillation complexity of context-free languages. In: Fundamentals of Computation Theory, FCT '85, Cottbus, GDR, September 9-13, 1985, pp 159–166, DOI 10.1007/BFb0028800, URL https://doi.org/10.1007/BFb0028800
14. Hellings J (2015) Path results for context-free grammar queries on graphs. CoRR abs/1502.02242, 1502.02242
15. Holzer M, Kutrib M, Leiter U (2011) Nodes connected by path languages. In: Mauri G, Leporati A (eds) Developments in Language Theory, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 276–287
16. Ibarra OH, Jiang T, Ravikumar B (1988) Some subclasses of context-free languages in nc1. Information Processing Letters 29(3):111 – 117, DOI https://doi.org/10.1016/0020-0190(88)90047-6, URL http://www.sciencedirect.com/science/article/pii/0020019088900476
17. Ibarra OH, Jiang T, Chang JH, Ravikumar B (1991) Some classes of languages in nc1. Information and Computation 90(1):86 – 106, DOI https://doi.org/10.1016/0890-5401(91)90061-6, URL http://www.sciencedirect.com/science/article/pii/0890540191900616
18. Kanellakis PC (1986) Logic programming and parallel complexity. In: Ausiello G, Atzeni P (eds) ICDT '86, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 1–30
19. Komarath B, Sarma J, Sunil KS (2014) On the complexity of l-reachability. In: Jürgensen H, Karhumäki J, Okhotin A (eds) Descriptional Complexity of Formal Systems, Springer International Publishing, Cham, pp 258–269
20. Kuijpers J, Fletcher G, Yakovets N, Lindaaker T (2019) An experimental study of context-free path query evaluation methods. In: Proceedings of the 31st International Conference on Scientific and Statistical Database Management, ACM, New York, NY, USA, SSDBM '19, pp 121–132, DOI 10.1145/3335783.3335791, URL http://doi.acm.org/10.1145/3335783.3335791
21. Lu Y, Shang L, Xie X, Xue J (2013) An incremental points-to analysis with cfl-reachability. In: Jhala R, De Bosschere K (eds) Compiler Construction, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 61–81
22. Malcher A, Meckel K, Mereghetti C, Palano B (2012) Descriptional complexity of pushdown store languages. J Autom Lang Comb 17(2):225âĂŞ244

23. Okhotin A, Salomaa K (2014) Complexity of input-driven pushdown automata. SIGACT News 45:47–67
24. Pierre L (1992) Rational indexes of generators of the cone of context-free languages. Theoretical Computer Science 95(2):279 – 305, DOI 10.1016/0304-3975(92)90269-L
25. Pierre L, Farinone JM (1990) Context-free languages with rational index in $\theta(n^\gamma)$ for algebraic numbers γ. RAIRO - Theoretical Informatics and Applications - Informatique Théorique et Applications 24(3):275–322
26. Reps T (1996) On the sequential nature of interprocedural program-analysis problems. Acta Inf 33(5):739–757, DOI 10.1007/BF03036473, URL http://dx.doi.org/10.1007/BF03036473
27. Reps TW (1997) Program analysis via graph reachability. Information & Software Technology 40:701–726
28. Rubtsov AA, Vyalyi MN (2015) Regular realizability problems and context-free languages. CoRR abs/1503.00295, 1503.00295
29. Swernofsky J, Wehar M (2015) On the complexity of intersecting regular, context-free, and tree languages. In: Halldórsson MM, Iwama K, Kobayashi N, Speckmann B (eds) Automata, Languages, and Programming, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 414–426
30. Szilard A, Yu S, Zhang K, Shallit J (1992) Characterizing regular languages with polynomial densities. In: Havel IM, Koubek V (eds) Mathematical Foundations of Computer Science 1992, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 494–503
31. Ufnarovskii VA (1982) A growth criterion for graphs and algebras defined by words. Mathematical notes of the Academy of Sciences of the USSR 31(3):238–241, DOI 10.1007/BF01145476, URL https://doi.org/10.1007/BF01145476
32. Ullman JD, Van Gelder A (1986) Parallel complexity of logical query programs. In: 27th Annual Symposium on Foundations of Computer Science (sfcs 1986), pp 438–454, DOI 10.1109/SFCS.1986.40
33. Vyalyi MN (2012) On complexity of regular realizability problems. ArXiv abs/1211.0606
34. Wechsung G (1979) The oscillation complexity and a hierarchy of context-free languages. In: Fundamentals of Computation Theory, FCT 1979, Proceedings of the Conference on Algebraic, Arthmetic, and Categorial Methods in Computation Theory, Berlin/Wendisch-Rietz, Germany, September 17-21, 1979., pp 508–515
35. Yannakakis M (1990) Graph-theoretic methods in database theory. pp 230–242, DOI 10.1145/298514.298576
36. Zhang X, Feng Z, Wang X, Rao G (2015) Context-free path queries on RDF graphs. CoRR abs/1506.00743, 1506.00743