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

(Parallel) Efficient Subclasses Of CFL-Reachability Queries

Ekaterina Shemetova · Semyon Grigorev

Received: date / Accepted: date

Abstract The context-free language (CFL) reachability problem for a context-free grammar G and directed edge-labelled graph D consists of determining for every pair of nodes v and u whether v can reach u via a path labelled by string in $L(G)$.

Inspite the

Keywords CFL-reachability · Complexity · Boolean circuit

1 Introduction

The context-free language (CFL) reachability problem is an important problem underlying some fundamential static code analysis [19, 15] and graph database query evaluation [11][23][10].

Unlike context-free language recognition, which is in NC, CFL-reachability is P-complete [22][18]. Practically, it means that we don't have an efficient parallel algorithm for solving this problem (unless $P \neq NC$). But want some cases... . Simple cases (examples, trees etc)... Our focus is on the...
This paper is organized as follows.

2 Preliminaries

Context-free languages Chomsky normal form, PDA, regular languages
Substitution closure of a class of languages L is the least class containing all

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

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

substitutions of languages from L to the languages from L.
Rational transduction, Generator, rational dominator

CFL-reachability The general formulation of CFL-reachability problem can be stated as follows. $G = (\Sigma, N, P)$ — G_A — grammar G with fixed start nonterminal A
$l(\pi)$ denotes a string which is obtained from concatenation of edge labels of the path π.
Connections with LGAP(L-Reachability = regular relializability = intersection of context-free language and a regular language)

Circuits and complexity classes

General complexity of CFL-reachability It is well known that CFL-reachability problem is P-complete [22].

3 Boolean circuit for CFL-reachability

In this section we build generalization of classical Boolean circuit for context-free recognition from the Brent-Goldschlager-Rytter algorithm [5, 21] in order to evaluate CFL-reachability problem.

3.1 Circuit description

The idea of the Brent-Goldschlager-Rytter parallel algorithm is based on replacement of ordinary parse trees with a equivalent system of logical dependencies. We can adapt this system for graph input as follows (context-free grammar is in the Chomsky normal form):

1. $\frac{i \xrightarrow{a} j}{A(i,j)}$ — if edge from the node i to the node j has label "a" and $A \to a \in P$, then there is a parse tree with nonterminal A at the root deriving a path from i to j.

2. $\frac{B(i,j)}{\substack{A \\ C}(i,j::z)}$ — creating a gap on the right: if a parse tree with the nonterminal B at the root deriving a path from i to j exists and $A \to BC \in P$, then a parse tree with nonterminal A at the root containing a smaller subtree with the nonterminal C deriving path from j to z ("gap"), where $1 \leq z \leq n$, may exist.

3. $\frac{C(j,z)}{\substack{A \\ B}(i::j,z)}$ — creating a gap on the left: if a parse tree with the nonterminal C at the root deriving a path from j to z exists and $A \to BC \in P$, then a parse tree with nonterminal A at the root containing a smaller subtree with the nonterminal B deriving a path from i to j ("gap"), where $1 \leq i \leq n$, may exist.

4. $\frac{B(i,j), \frac{A}{B}(i::j,z)}{A(i,z)}$ — filling the gap: if a parse tree with the nonterminal A at the root deriving a path from i to z, which cointains the gap represented by a node labelled B deriving a path from i to j and a parse tree with the nonterminal B at the root deriving a path from i to j exist, then the whole parse tree with the nonterminal A at the root deriving a path from i to z exists.

5. $\frac{\frac{A}{E}(i,j::w,z), \frac{E}{D}(j,u::v,w)}{\frac{A}{D}(i,u::v,z)}$ — combination of conditional propositions: if there is a parse tree with the nonterminal A at the root deriving a path from i to z with the gap represented by a node labelled E deriving a path from j to w and there is a parse tree with the nonterminal E at the root deriving a path from j to w, which contains gap represented by some node D, then there is a parse tree with the nonterminal A at the root deriving a path from i to z, which contains the gap represented by the above mentioned node D.

Using logical dependencies above, elements of the circuit can be described with two types of elements: $x_{A,i,j}$, which is true when a parse tree with the nonterminal A at the root deriving a path from i to j exists, and $y_{A,i,j,B,k,l}$ for conditional propositions, which is true when there is a parse tree with the nonterminal A at the root deriving a path from i to j containg a hole instead of a subtree with the nonterminal B at the root deriving a path from k to l.

Therefore the circuit contains the following gates:

- *Input*: input gates, one gate for every edge of the input graph and alphabet symbol; evaluates to true if the corresponding edge has an appropriate label
- internal AND-gates for evaluating the combinations of conditional propositions and filling the gaps
- internal OR-gates for creating gaps: conditional proposition $y_{A,i,j,E,k,l}$ can be obtained by taking the gap from the right subtree (if $x_{B,i,z}$ is true and $y_{C,z,j,E,k,l}$ is true) or by taking the gap from the left subtree ($y_{B,i,z,E,k,l}$ is true and $x_{C,z,j}$ is true) for rule $A \to BC \in P$. Notice that if start and end nodes of the tree with the gap coincide with start and end nodes of the gap ($i = k, z = l$ for the left case, $z = k, j = l$ for the right case), then the gap just added to the left or right side of another tree.
- *Output*: output gates, one gate for every pair of graph nodes and nonterminal; evaluates to true if appropriate parse tree exists.

3.2 Bounds on the circuit size and depth

Circuit size

Lemma 1 *Let $G = (\Sigma, N, P)$ be a context-free grammar in Chomky normal form and let n be a number of nodes in the input graph. Then the circuit described in 3.1 has $O(n^6)$ elements.*

Proof Let's count circuit gates by type. The number of input gates is $|\Sigma|n^2$ and the number of output gates is $|N|n^2$. Also there are $|N|^2n^5$ OR-gates for creating the gaps and $|N|^2n^4$ AND-gates for filling gaps. It is easy to see that the maximal number of elements is needed for combinations of conditional propositions: there are $|N|^3n^6$ such elements ($|N|^2n^4 \times |N|n^2$). Therefore the circuit contains at most $O(n^6)$ elements.

Circuit depth Before we analyze the depth of the circuit we shall prove the following statement.

Lemma 2 *Let $G = (\Sigma, N, P)$ be a context-free grammar in Chomky normal form. Then every parse tree with k leaves contains a middle node ("critical" node) that spans over more than $\frac{1}{3}k$ and at most $\frac{2}{3}k$ leaves.*

Proof Because the grammar is in the Chomsky normal form, every node of a parse tree has two children. Going from the root to leaves, we can choose the largest of two subtrees at each node while the current subtree has more than $\frac{2}{3}k$ leaves. Obviously, the first node that has at most $\frac{2}{3}k$ leaves is bound to have more than $\frac{1}{3}k$ leaves due to binary branching.

Lemma 3 *Let l be a length of the shortest path from node i to node j labelled by string from $L(G_A)$. Then proposition $A(i, j)$ or $\frac{A}{D}(i, u :: v, j)$ has a proof of height $O(\log l)$.*

Proof Proof by induction on l.
Induction hypotesis: every proposition with no more than $\frac{2}{3}l$ leaves has a proof of logarithmic height ($O(\log l)$).

At first, consider the proposition $A(i, j)$. By Lemma 2, a corresponding tree has a critical node E with two children B and C (for $E \to BC \in P$). Then we can write the proof tree as follows.

$$\dfrac{\dfrac{B(u,l) \qquad C(l,v)}{E(u,v)} \qquad \frac{A}{E}(i, u :: v, j)}{A(i,j)}$$

Because E is the critical node, each of the propositions $B(u, l), C(l, v)$, $\frac{A}{E}(i, u :: v, j)$ has no more than $\frac{2}{3}l$ leaves and therefore has a proof of logarithmic height by inductive hypothesis.

There are two cases for the second proposition $\frac{A}{D}(i, u :: v, j)$.

1. *A subtree with the critical node at the root contains the hole $u :: v$.*
   Let E be a label of the critical node. Then a corresponding subtree splits the path from the vertex i to the vertex u onto two paths: $i \to k$ and $k \to u$ and the path from the vertex v to the vertex j onto $v \to m$ and $m \to j$ respectively (as illustrated in Figure 1 (left)). The next proof tree represents the proof of the proposition $\frac{A}{D}(i, u :: v, j)$ in this case.

$$\dfrac{\frac{A}{E}(i, k :: m, j) \qquad \frac{E}{D}(k, u :: v, m)}{\frac{A}{E}(i, u :: v, j)}$$

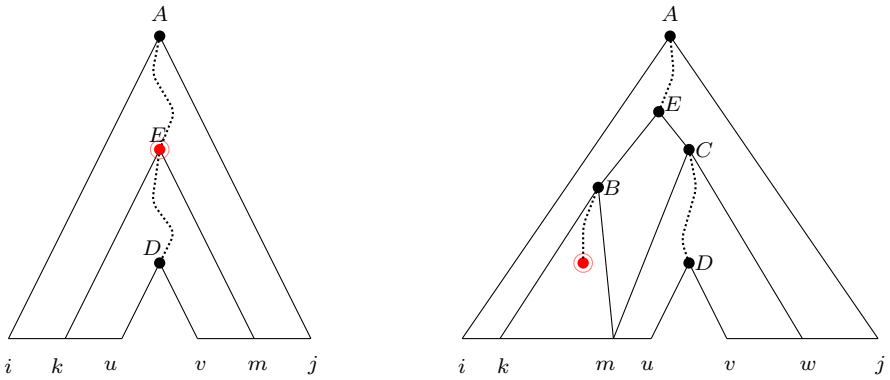

Fig. 1 Cases for conditional proposition $\frac{A}{D}(i, u :: v, j)$: (left) a subtree with the critical node at the root contains the hole; (right) a subtree with the critical node at the root contains only leaves corresponding to the path $i \to u$.

Conditional proposition $\frac{A}{E}(i, k :: m, j)$ and $\frac{E}{D}(k, u :: v, m)$ have no more than $\frac{2}{3}l$ leaves, so $\frac{A}{E}(i, u :: v, j)$ has a proof of logarithmic height.

2. *A subtree with the critical node at the root contains only leaves corresponding to the path from the vertex i to the vertex u in the input graph.*
   This case is illustrated in Figure 1 (right). Consider the biggest subtree which contains a subtree with the critical node at the root and has leaves corresponding only to the path from the vertex i to the vertex u of the input graph. Let B be a label at the root of this tree and $k \to m$ be a corresponding path in the input graph. Let E be a parent node of node B and let C be a second child of E (for $E \to BC \in P$). A parse tree with the root labelled by C has some leaves corresponding to the path $v \to j$, so let w be a vertex which splits this path on two. Now we are able to write the proof of $\frac{A}{E}(i, u :: v, j)$.

$$\dfrac{\dfrac{\dfrac{}{B(k,m)}}{\frac{E}{C}(k, m :: w)} \quad \frac{C}{D}(m, u :: v, w)}{\frac{E}{D}(k, u :: v, w)} \quad \frac{A}{E}(i, k :: w, j)}{\frac{A}{D}(i, u :: v, j)}$$

   The subtree with the root labelled B obviously contains at least $\frac{2}{3}l$ leaves, so each of the other propositions $\frac{C}{D}(m, u :: v, w)$ and $\frac{A}{E}(i, k :: w, j)$ has no more then $\frac{2}{3}l$ leaves. By induction hypothesis they have a proof of logarithmic height. Thus, proposition $\frac{A}{D}(i, u :: v, j)$ can be proved via 4 steps using propositions with the proofs of logarithmic depth.
   The case when the subtree with the critical node at the root contains only leaves corresponding to the path from the vertex v to the vertex j in the input graph is held symmetrically.

Using lemmas above, we can conclude the following.

Corollary 1 *Let $G = (\Sigma, N, P)$ be a context-free grammar in Chomky normal form and let $\mathcal{L}$ be the length of the maximum in all nonterminals and all pairs of graph nodes shortest path labelled by a string from $L(G)$. Then there is a uniform family of circuits for solving CFL-reachability problem on the graphs with n nodes, which are of depth $O(\log n \log \mathcal{L})$ and contain $O(n^6)$ elements.*

The depth and therefore the efficiency of the circuit depends on the value of parameter $\mathcal{L}$. We investigate bounds for $\mathcal{L}$ in the next two sections.

4 Efficient subclasses of context-free languages

In this section our focus is on estimating the value of $\mathcal{L}$ for some fixed context-free languages and arbitrary input graphs.

4.1 General case

Hellings in [11] gave the worst-case lower and upper bounds on the $\mathcal{L}$ parameter.

Theorem 1 (Hellings) *Let $G = (\Sigma, N, P)$ be a context-free grammar and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. In the worst case, we have $n^2 2^{|N|} \leq \mathcal{L} \leq 2^{|N|n^2-1}$.*

Using the bounds above and Corollary 1, we can get the maximum depth of our circuit in the worst case: $O(\log n \log \mathcal{L}) = O(\log n \log 2^{|N|n^2-1}) = $
$= O(|N|n^2 \log n)$. The circuit can have a depth which is polynomial on the input length in case of arbitrary context-free grammar. It is not surprising because of the nature of P-completeness: we don't have an efficient parallel algorithm or circuit with a polylogarithmic on the input length depth for CFL-reachability problem because it is P-complete problem (unless $P \neq NC$).

But we also have a polynomial lower bound on $\mathcal{L}$, which give us the effective circuit with depth $O(\log^2 n)$ in some cases. Our next goal is to investigate important subclasses of context-free languages, and find those for which CFL-reachability is in the class NC.

4.2 Rational index

It is known that the value of $\mathcal{L}$ is different for various types of context-free languages, in particular it can be bounded from above by the polynomial or exponential function on the number of graph nodes [6, 16, 17]. Boasson et al. in [3] introduced definition of such function called *rational index*. 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.

It is easy to see that in the case of fixed context-free language and arbitrary graph, $\mathcal{L}$ is some kind of the rational index: we can represent arbitrary graphs with n nodes via nondeterministic automations with at most n states, then $\mathcal{L}$ is exactly rational index. So, we have the following corollary.

Corollary 2 *Let L be a context-free language on an alphabet Σ, which has a polynomially bounded from above rational index, and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. Then, CFL-reachability problem for L and D is in NC^2.*

For example, language $L = \{a^n b^m, n \neq m\}$ has polynomial rational index $\rho_L(n) = 2n - 1$ [17], so CFL-reachability problem for L and arbitrary directed labelled graph with n nodes is in NC^2.

4.3 Linear, metalinear and superlinear languages

A *linear language* is a language generated by some *linear grammar*. A *linear grammar* is a context-free grammar that has at most one nonterminal in the right hand side of each of its productions. The linear grammar $G = (\Sigma, N, P)$ is said to be in the Chomsky normal form, when all of its production rules are of the form: $A \to aB$, or $A \to Ba$, or $A \to a$, where $B \in N$ and $a \in \Sigma$.

Before we consider the value of $\mathcal{L}$ for linear grammars, we shall prove the following.

Lemma 4 *Let $G = (\Sigma, N, P)$ be a context-free grammar, $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes and π be the maximum in all nonterminals and all pairs of graph nodes shortest path labelled by a string from $L(G)$. Then a height of a parse tree for $l(\pi)$ does not exceed $|N|n^2$.*

Proof Assume that the parse tree for $l(\pi)$ 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 $l(\pi)$ 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 $l(\pi')$ which is shorter than $l(\pi)$. But $l(\pi)$ is the shortest, then we have a contradiction.

Theorem 2 *Let $G = (\Sigma, N, P)$ be a linear grammar and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. Then the length of the maximum in all nonterminals and all pairs of graph nodes shortest path labelled by a string from $L(G)$ ($\mathcal{L}$) is $O(|N|n^2)$.*

Proof From Lemma 4, a height of a parse tree is no more then $|N|n^2$. Let's construct a parse tree of such height for linear grammar in the Chomsky normal form. Every rule for linear grammar has at most one nonterminal in the right

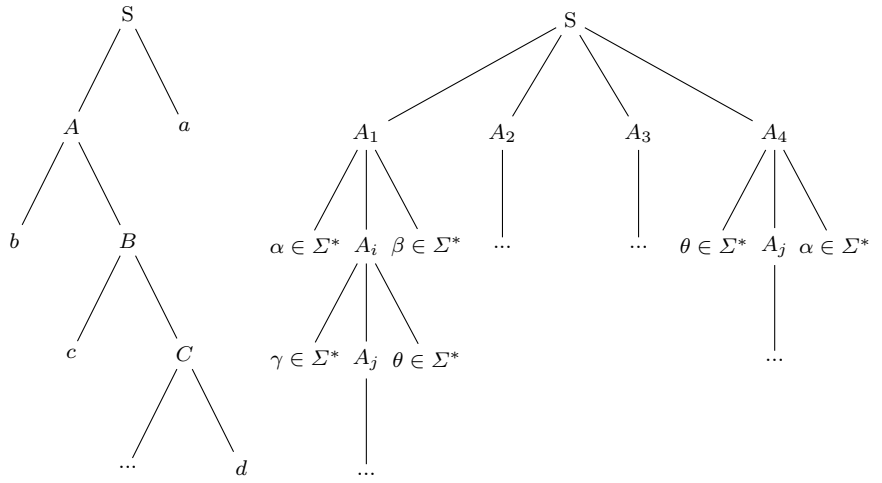

Fig. 2 A parse tree for linear grammar in the Chomsky normal form (left) and for metalinear grammar with width $m = 4$ (right).

hand side of each of its productions, so if the grammar is in the Chomsky normal form, then there is the only one terminal symbol at the every level of the parse tree (as illustrated in Figure 2 (left)). The tree has no more then $|N|n^2$ levels, so the length of any shortest path labelled by a string from $L(G)$ is no more than $O(|N|n^2)$.

Combining Theorem 2 and Corollary 1 we are able to conclude the following.

Corollary 3 *Let $G = (\Sigma, N, P)$ be a linear grammar and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. Then CFL-reachability problem for G and D is in NC^2.*

There are natural extensions of linear languages: *metalinear* and *superlinear* languages. Just like linear languages, these classes of context-free languages are known to be easy to parse [12]. We will show that they are "easy" for our circuit too.

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. $S \to A_1 A_2 ... A_m$, 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 $max\{m \mid S \to A_1 A_2 ... A_m\}$. Metalinear languages of width 1 are obviously linear languages.

Lemma 5 *Let $G = (\Sigma, N, P)$ be a metalinear grammar, where G_S has the width m, k be a length of the longest sequence of terminal symbols in the right-hand-side of rules and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes.*

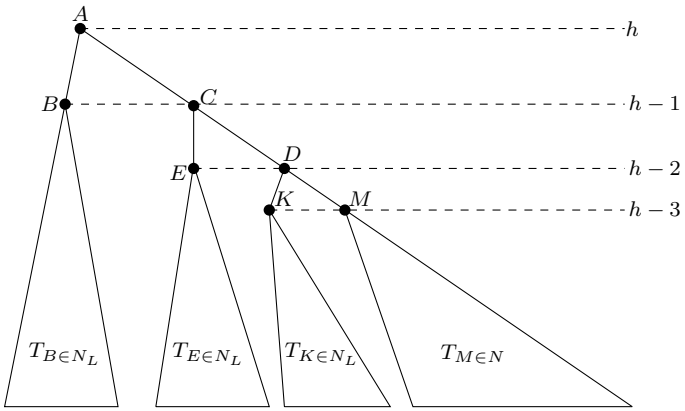

Fig. 3 A worst-case parse tree for superlinear grammar.

Then the length of the maximum in all nonterminals and all pairs of graph nodes shortest path labelled by a string from $L(G)$ ($\mathcal{L}$) is $O(mk|N|n^2)$.

Proof Consider the highest parse tree for metalinear grammar. By Lemma 4, it's height is no more then $|N|n^2$. Obviously, if the root of the tree is labelled by some $A_i \in N - \{S\}$, value of $\mathcal{L}$ in this case is the same as value of $\mathcal{L}$ for the case of linear grammar — $O(|N|n^2)$ for constant value of k. If the root of the tree is labelled by S, the tree looks like as illustrated in Figure 2 (right). It is easy to see that there are no more than km terminal symbols on every level of such tree. Thus, we have $\mathcal{L} = O(mk|N|n^2)$.

Applying Lemma 5 and Corollary 1, we deduce with the corollary.

Corollary 4 *Let $G = (\Sigma, N, P)$ be a metalinear grammar and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. Then CFL-reachability problem for G and D is in NC^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 $A \in N \backslash N_L$, 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 $B \in N_L$, $\alpha \in \Sigma^*$.

It can be seen from conditions above, that superlinear grammars contain the metalinear grammars.

Lemma 6 *Let $G = (\Sigma, N, P)$ be a superlinear grammar and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. Then the length of the maximum in all nonterminals and all pairs of graph nodes shortest path labelled by a string from $L(G)$ ($\mathcal{L}$) is $O(|N^2|n^4)$.*

Proof Let's construct a parse tree of maximum height h from the root to leaves. Let A be a nonterminal at the root of the parse tree T_A. We have two cases to consider:

1. $A \in N_L$ or $A \in N \setminus N_L$ and A has linear productions: then the tree with A at the root is the same tree as parse tree for linear grammar and it has $O(h)$ leaves.
2. $A \in N \setminus N_L$ and $A \to BC \in P$: then number of leaves in the tree $le(T_A)$ can be expressed with the following reccurent formula: $le(T_A) = le(T_B) + le(T_C) = h(T_B) + le(T_C) = h(T_A) - 1 + le(T_C)$.

In the worst case, we have to add tree with i leaves on every i-th level of the parse tree (counting from $h-1$ to 0) (see Figure 6). By Lemma 4, $h \leq |N|n^2$, so in the worst case the highest tree will have $|N|n^2 - 1 + |N|n^2 - 2... + 1 = O(|N^2|n^4)$ leaves.

Combining Lemma 6 and Corollary 1 we are able to conclude the following.

Corollary 5 *Let $G = (\Sigma, N, P)$ be a superlinear grammar and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. Then CFL-reachability problem for G and D is in NC^2.*

4.4 Dyck languages

Dyck language over Σ_k is the set of well-balanced words over alphabet $\Sigma_k = \{(_1, (_2, ..., (_k,)_1,)_2, ...,)_k\}$. Dyck language can be defined by the following rules: $S \to SS \mid S \to \varepsilon \mid S \to (_1S)_1 \mid S \to (_2S)_2 \mid ... \mid S \to (_kS)_k$, where S is the start symbol, and ε is the empty string. We will denote the Dyck language over Σ_i by D_i.

Dyck languages play an important role in formal languages theory. The famous Chomsky-Schützenberger theorem states that every context-free language L can be represented via a regular language R and a Dyck language D_n, which are combined by means of an intersection and a homomorphism h.

$$L = h(D_n \cap R) \tag{2}$$

Because every context-free language can be represented with Dyck language, it is natural to expect that CFL-reachability is P-complete for Dyck languages. P-completeness for the similiar problem LGAP in case of D_2 was proved in [8, 13, 20]. We will show that $\measuredangle$ is exponential for D_k where $k \geq 2$, so our circuit is not effective in that case.

Lemma 7 (Pierre) *The rational indexes of generators of context-free languages belongs to $exp(\Theta(n^2/\ln n))$.*

Theorem 3 *Let L be a Dyck language over $k \geq 2$ brackets and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. Then, $\measuredangle$ belongs to $exp(\Theta(n^2/\ln n))$.*

Proof By definition, generator is a context-free language which can transform into any context-free language through a rational transduction [16]. Notice that for $k > 2$ there is a homomorphism g such that $D_k = g^{-1}(D_2)$ can be constructed. Thanks to the Chomsky-Schützenberger theorem, we have the following equation.

$$L = h(g^{-1}(D_2) \cap R) \tag{3}$$

Thus, every context-free language can be generated by D_k, where $k \geq 2$. By 7 we can conclude that $\mathcal{L}$ for D_k belongs to $exp(\Theta(n^2/\ln n))$.

We remark that strict upper bound on $\mathcal{L}$ for arbitrary context-free grammar is obtained from Lemma 7, so it answers to the corresponding open question stated in [11].

It is known that D_1 is a generator of *one-counter languages* [9]. *One-counter* languages are the languages recognized by *one-counter automata* — pushdown automata with a single stack symbol. For example, $L = \{w\#w' \mid |w| \neq |w'|\}$ is one-counter language.

Lemma 8 (Deleage, Pierre) *The rational index of D_1 is in $O(n^2)$.*

Theorem 4 *Let L be a one-counter language, and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. Then, CFL-reachability problem for L and D is in NC^2.*

Proof Because D_1 is a generator of one-counter languages [9], D_1 rationally dominates any one-counter language. If L rationally dominates L', then there exists an integer c such that for every $n \in \mathbb{N}$ $\rho_{L'(n)} < cn(\rho_L(cn)+1)$ [3]. Thus, the rational index of any one-counter language doesn't exceed $O(n^3)$, and by Corollary 1, the depth of the circuit for one-counter languages is $O(\log^2 n)$.

Notice that there is an effective sequential algorithm for CFL-reachability for D_1 based on matrix multiplication [4]. We believe that using Theorem 4 it can be extended for working with one-counter languages.

4.5 Greibach languages and substitution closures

Consider families of linear and one-counter languages. We can see that they are "easy" for CFL-reachability (Theorem 4, Corollary 3). One reason for this is that they are defined by natural restrictions on grammars or pushdown automatons (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 [14]. One-counter languages are recognized by PDA with only one stack symbol. The same restrictions holds for substitution closures of these languages, because substitution corresponds to a nesting of the pushdown stack [7]. It is easily observed that substitution closure of polinomially-bounded $\mathcal{L}$ language yields polinomially-bounded $\mathcal{L}$ language.

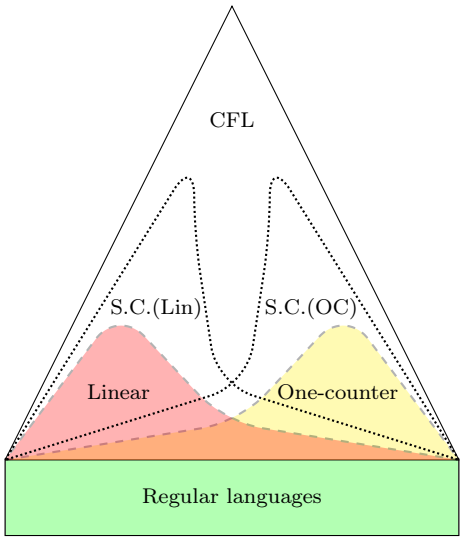

Fig. 4 A hierarchy of linear languages (Lin), one-counter languages (OC) and their substitution closures (S.C).

But some of the restrictions on the pushdown store or on the grammar cannot be simulated by others. It is exactly the case of linear and one-counter languages: they and their iterated substitution closures are distinct families of context-free languages [2]. Hierarchy of linear languages, one-counter languages and their substitution closures is illustrated in Figure 4.

Family of *Greibach languages* is the substitution closure of linear and one-counter languages. It is a large strict subfamily of the family of context-free languages: it does not contain the language D_2 [1]. Because it is the substitution closure of two polinomially-bounded $\mathcal{L}$ languages, we have the following corollary.

Corollary 6 *Let L be a Greibach language and $D = (V, E, \Sigma)$ be a directed labelled graph with n nodes. Then CFL-reachability problem for L and D is in NC^2.*

5 Efficient subclasses of graphs

5.1 Directed acyclic graphs and trees

5.2 ??? Bounded regular languages

6 Conclusions and open problems

Metalinear grammars used in DNA computing? — check it. For every algebraic number α bigger than 1, a Greibach language whose rational index is in

n^{α} can be found **??**. However some context- free languages with polynomial rational indexes are not Greibach languages **??**. Add to open — how can we describe all easy CFL? Is it possible? Seems like not.
Do only stack restictions imply easyness? (bounded stack — regular languages).

References

1. Autebert JM, Berstel J, Boasson L (1997) Context-Free Languages and Pushdown Automata, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 111–174
2. Beauquier J, Berstel J (1981) More about the geography of context-free languages. Information and Control 49(2):91 – 108
3. Boasson L, Courcelle B, Nivat M (1981) The rational index: a complexity measure for languages. SIAM Journal on Computing 10(2):284–296
4. Bradford PG (2018) Efficient exact paths for dyck and semi-dyck labeled path reachability. CoRR abs/1802.05239, 1802.05239
5. Brent R, M Goldschlager L (1983) A parallel algorithm for context-free parsing
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. Ginsburg S (1975) Algebraic and Automata-Theoretic Properties of Formal Languages. Elsevier Science Inc., New York, NY, USA
8. Greenlaw R, Hoover HJ, Ruzzo WL (1995) Limits to Parallel Computation: P-completeness Theory. Oxford University Press, Inc., New York, NY, USA
9. Greibach SA (1967) An infinite hierarchy of context-free languages. In: 8th Annual Symposium on Switching and Automata Theory (SWAT 1967), pp 32–36, DOI 10.1109/FOCS.1967.5
10. 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
11. Hellings J (2015) Path results for context-free grammar queries on graphs. CoRR abs/1502.02242, 1502.02242
12. Jayaram R, Saha B (2017) Approximating language edit distance beyond fast matrix multiplication: Ultralinear grammars are where parsing becomes hard! In: ICALP
13. Komarath B, Sarma J, Sunil KS (2017) On the complexity of l-reachability. CoRR abs/1701.03255, URL http://arxiv.org/abs/1701.03255, 1701.03255
14. Kutrib M, Malcher A (2007) Finite turns and the regular closure of linear context-free languages. Discrete Applied Mathematics 155(16):2152 – 2164

15. 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
16. 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
17. 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
18. 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
19. Reps TW (1997) Program analysis via graph reachability. Information & Software Technology 40:701–726
20. Rubtsov AA, Vyalyi MN (2015) Regular realizability problems and context-free languages. CoRR abs/1503.00295, 1503.00295
21. Rytter W (1985) On the recognition of context-free languages. In: Skowron A (ed) Computation Theory, Springer Berlin Heidelberg, Berlin, Heidelberg, pp 318–325
22. Yannakakis M (1990) Graph-theoretic methods in database theory. pp 230–242, DOI 10.1145/298514.298576
23. Zhang X, Feng Z, Wang X, Rao G (2015) Context-free path queries on RDF graphs. CoRR abs/1506.00743, 1506.00743