Interprocedural Reachability for Flat Integer Programs

Pierre Ganty, Radu Iosif

Interprocedural Reachability for Flat Integer Programs

Pierre Ganty[1] and Radu Iosif[2]

[1]IMDEA Software Institute [2] CNRS/VERIMAG, Grenoble, France

Abstract. We study programs with integer data, procedure calls and arbitrary call graphs. We show that, whenever the guards and updates are given by octagonal relations, the reachability problem along control flow paths within some language $w_1^* \ldots w_d^*$ over program statements is decidable in NEXPTIME. To achieve this upper bound, we combine a program transformation into the same class of programs but without procedures, with an NP-completeness result for the reachability problem of procedure-less programs. Besides the program, the expression $w_1^* \ldots w_d^*$ is also mapped onto an expression of a similar form but this time over the transformed program statements. Several arguments involving context-free grammars and their generative process enable us to give tight bounds on the size of the resulting expression. The currently existing gap between NP-hard and NEXPTIME can be closed to NP-complete when a certain parameter of the analysis is assumed to be constant.

1 Introduction

This paper studies the complexity of the reachability problem for a class of programs featuring procedures and local/global variables ranging over integers. In general, the reachability problem for this class is undecidable [24]. Thus, we focus on a special case of the reachability problem which restricts both the class of input programs and the set of executions considered. The class of input programs is restricted by considering that all updates to the integer variables $\mathbf{x}$ are defined by *octagonal constraints*, that are conjunctions of atoms of the form $\pm x \pm y \leqslant c$, with $x, y \in \mathbf{x} \cup \mathbf{x}'$, where $\mathbf{x}'$ denote the future values of the program variables. The reachability problem is restricted by limiting the search to program executions conforming to a regular expression of the form $w_1^* \ldots w_d^*$ where the w_i's are finite sequences of program statements.

We call this problem *flat-octagonal reachability* (fo-reachability, for short). Concretely, given: (i) a program $\mathcal{P}$ with procedures and local/global variables, whose statements are specified by octagonal constraints, and (ii) a bounded expression $\mathbf{b} = w_1^* \ldots w_d^*$, where w_i's are sequences of statements of $\mathcal{P}$, the fo-reachability problem $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ asks: can $\mathcal{P}$ run to completion by executing a sequence of program statements $w \in \mathbf{b}$? Studying the complexity of this problem provides the theoretical foundations for implementing efficient decision procedures, of practical interest in areas of software verification, such as bug-finding [10], or counterexample-guided abstraction refinement [15,14].

Our starting point is the decidability of the fo-reachability problem in the absence of procedures. Recently, the precise complexity of this problem was coined to NP-complete [7]. However, this result leaves open the problem of dealing with procedures and local variables, let alone when the graph of procedure calls has cycles, such as in the example of Fig. 1 (a). Pinning down the complexity of the fo-reachability problem in presence of (possibly recursive) procedures, with local variables ranging over integers, is the challenge we address here.

The decision procedure we propose in this paper reduces $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$, from a program $\mathcal{P}$ with arbitrary call graphs, to procedure-less programs as follows:

1. we apply a source-to-source transformation returning a procedure-less program $\mathcal{Q}$, with statements also defined by octagonal relations, such that $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ is equivalent to the unrestricted reachability problem for $\mathcal{Q}$, when no particular bounded expression is supplied.
2. we compute a bounded expression $\Gamma_{\mathbf{b}}$ over the statements of $\mathcal{Q}$, such that $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ is equivalent to $\mathrm{REACH}_{fo}(\mathcal{Q}, \Gamma_{\mathbf{b}})$.

The above reduction allows us to conclude that the fo-reachability problem for programs with arbitrary call graphs is decidable and in NEXPTIME. Naturally, the NP-hard lower bound [7] for the fo-reachability problem of procedure-less programs holds in our setting as well. Despite our best efforts, we did not close the complexity gap yet. However we pinned down a natural parameter, called *index*, related to programs with arbitrary call graphs, such that, when setting this parameter to a fixed constant (like 3 in 3-SAT), the complexity of the resulting fo-reachability problem for programs with arbitrary call graphs becomes NP-complete. Indeed, when the index is fixed, the aforementioned reduction computing $\mathrm{REACH}_{fo}(\mathcal{Q}, \Gamma_{\mathbf{b}})$ runs in polynomial time. Then the NP decision procedure for the fo-reachability of procedure-less programs [7] shows the rest.

The index parameter is better understood in the context of formal languages. The control flow of procedural programs is captured precisely by the language of a context-free grammar. A k-index ($k > 0$) underapproximation of this language is obtained by filtering out the derivations containing a sentential form with $k+1$ occurrences of nonterminals. The key to our results is a toolbox of language theoretic constructions of independent interest that enables to reason about the structure of context-free derivations generating words into $\mathbf{b} = w_1^* \dots w_d^*$, that is, words of the form $w_1^{i_1} \dots w_d^{i_d}$ for some integers $i_1, \dots, i_d \geqslant 0$.

To properly introduce the reader to our result, we briefly recall the important features of our source-to-source transformation through an illustrative example. We apply first our program transformation [11] to the program $\mathcal{P}$ shown in Fig. 1 (a). The call graph of this program consists of a single state `P` with a self-loop. The output program $\mathcal{Q}$ given Fig. 1 (e), has no procedures and it can thus be analyzed using any existing intra-procedural tool [6,4]. The relation between the variables x and z of the input program can be inferred from the analysis of the output program. For instance, the input-output relation of the program $\mathcal{P}$ is defined by $z' = 2x$, which matches the precondition $z_O = 2x_I$ of the program $\mathcal{Q}$. Consequently, any assertion such as "*there exists a value $n > 0$ such that*

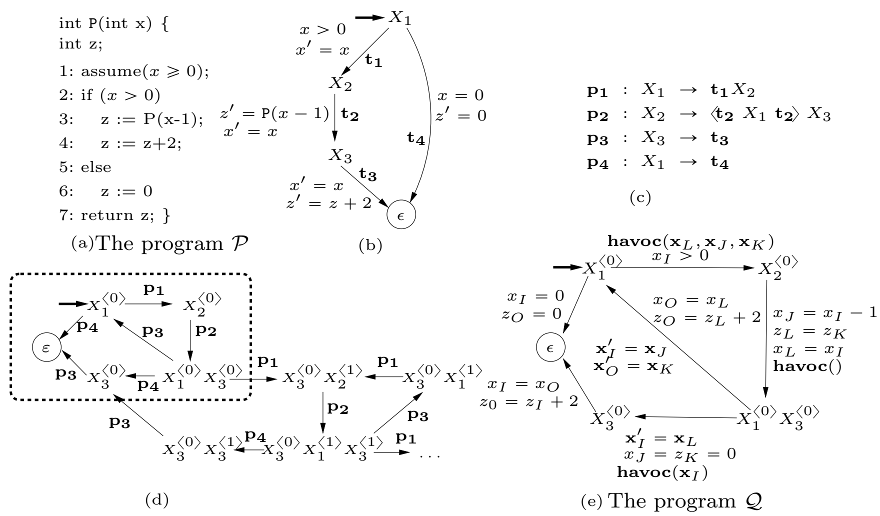

Fig. 1: $\mathbf{x}_I = \{x_I, z_I\}$ ($\mathbf{x}_O = \{x_O, z_O\}$) are for the input (output) values of x and z, respectively. $\mathbf{x}_{J,K,L}$ provide extra copies. $\mathbf{havoc}(\mathbf{y})$ stands for $\bigwedge_{x \in \mathbf{x}_{I,O,J,K,L} \setminus \mathbf{y}} x' = x$, and $\mathbf{x}'_\alpha = \mathbf{x}_\beta$ for $\bigwedge_{x \in \mathbf{x}} x'_\alpha = x_\beta$.

$\mathtt{P}(n) < n$" can be phrased as: "*there exist values $n < m$ such that $\mathcal{Q}(n, m)$ reaches its final state*". While the former can be encoded by a reachability problem on $\mathcal{P}$, by adding an extra conditional statement, the latter is an equivalent reachability problem for $\mathcal{Q}$.

For the sake of clarity, we give several representations of the input program $\mathcal{P}$ that we assume the reader is familiar with including the text of the program in Fig. 1 (a) and the corresponding control flow graph in Fig. 1 (b).

In this paper, the formal model we use for programs is based on context-free grammars. The grammar for $\mathcal{P}$ is given at Fig. 1 (c). The rôle of the grammar is to define the set of *interprocedurally valid* paths in the control-flow graph of the program $\mathcal{P}$. Every edge in the control-flow graph matches one or two symbols from the finite alphabet $\{\mathbf{t_1}, \langle\mathbf{t_2}, \mathbf{t_2}\rangle, \mathbf{t_3}, \mathbf{t_4}\}$, where $\langle\mathbf{t_2}$ and $\mathbf{t_2}\rangle$ denote the call and return, respectively. The set of nonterminals is $\{X_1, X_2, X_3, X_4\}$. Each edge in the graph translates to a production rule in the grammar, labeled $\mathbf{p_1}$ to $\mathbf{p_4}$. For instance, the call edge $X_2 \xrightarrow{\mathbf{t_2}} X_3$ becomes $X_2 \to \langle\mathbf{t_2} X_1 \mathbf{t_2}\rangle X_3$. The language of the grammar of Fig. 1 (c) (with axiom X_1) is the set $L = \{(\mathbf{t_1}\langle\mathbf{t_2})^n\, \mathbf{t_4}\, (\mathbf{t_2}\rangle\mathbf{t_3})^n \mid n \in \mathbb{N}\}$ of interprocedurally valid paths in the control-flow graph. Observe that L is included in the language of the regular expression $\mathbf{b} = (\mathbf{t_1}\langle\mathbf{t_2})^*\, \mathbf{t_4}^*\, (\mathbf{t_2}\rangle\mathbf{t_3})^*$.

Our program transformation is based on the observation that the semantics of $\mathcal{P}$ can be precisely defined on the set of *derivations* of the associated grammar. In principle, one can always represent this set of derivations as a possibly infinite automaton (Fig. 1 (d)), whose states are sequences of nonterminals annotated with priorities (called ranks)[1], and whose transitions are labeled with production rules. Each finite path in this automaton, starting from $X_1^{\langle 0 \rangle}$, defines a valid prefix of a derivation. Since $L \subseteq \mathbf{b}$, Luker [20] shows that it is sufficient to keep a finite sub-automaton, enclosed with a dashed box in Fig. 1 (d), in which each state consists of a finite number of ranked nonterminals (in our case at most 2).

Finally, we label the edges of this finite automaton with octagonal constraints that capture the semantics of the relations labeling the control-flow graph from Fig. 1 (b). We give here a brief explanation for the labeling of the finite automaton in Fig. 1 (e), in other words, the output program $\mathcal{Q}$ (see [11] for more details). The idea is to compute, for each production rule $\mathbf{p_i}$, a relation $\rho_i(\mathbf{x}_I, \mathbf{x}_O)$, based on the constraints associated with the symbols occurring in $\mathbf{p_i}$ (labels from Fig. 1 (b)). For instance, in the transition $X_2^{\langle 0 \rangle} \xrightarrow{\mathbf{p_2}} X_1^{\langle 0 \rangle} X_3^{\langle 0 \rangle}$, the auxiliary variables store intermediate results of the computation of $\mathbf{p_2}$ as follows: $[\mathbf{x}_I]\ \langle\mathbf{t_2}\ [\mathbf{x}_J]\ X_1\ [\mathbf{x}_K]\ \mathbf{t_2}\rangle\ [\mathbf{x}_L]\ X_3\ [\mathbf{x}_O]$. The guard of the transition can be understood by noticing that $\langle\mathbf{t_2}$ gives rise to the constraint $x_J = x_I - 1$, $\mathbf{t_2}\rangle$ to $z_L = z_K$, $x_I = x_L$ corresponds to the frame condition of the call, and $\mathbf{havoc}()$ copies all current values of $\mathbf{x}_{I,J,K,L,O}$ to the future ones. It is worth pointing out that the constraints labeling the transitions of the program $\mathcal{Q}$ are necessarily octagonal if the statements of $\mathcal{P}$ are defined by octagonal constraints.

An intra-procedural analysis of the program $\mathcal{Q}$ in Fig. 1 (e) infers the precondition $x_I \geqslant 0 \wedge z_O = 2x_I$ which coincides with the input/output relation of the recursive program $\mathcal{P}$ in Fig. 1 (a), i.e. $x \geqslant 0 \wedge z' = 2x$. The original query $\exists n > 0\colon \mathtt{P}(n) < n$ translates thus into the satisfiability of the formula $x_I > 0 \wedge z_O = 2x_I \wedge x_I < z_O$, which is clearly false.

The paper is organised as follows: basic definitions are given Section 2, Section 3 defines the fo-reachability problem, Section 4 presents an alternative program semantics based on derivations and introduces subsets of derivations which are sufficient to decide reachability, Section 5 starts with on overview of our decision procedure and our main complexity results and continues with the key steps of our algorithms. The appendix contains all the missing details.

2 Preliminaries

Let Σ be a finite nonempty set of symbols, called an *alphabet*. We denote by Σ^* the set of finite words over Σ which includes ε, the empty word. The concatenation of two words $u, v \in \Sigma^*$ is denoted by $u \cdot v$ or $u\,v$. Given a word $w \in \Sigma^*$, let $|w|$ denote its length and let $(w)_i$ with $1 \leqslant i \leqslant |w|$ be the ith symbol of w. Given $w \in \Sigma^*$ and $\Theta \subseteq \Sigma$, we write $w{\downarrow_\Theta}$ for the word obtained by deleting from w all symbols not in Θ, and sometimes we write $w{\downarrow_a}$ for $w{\downarrow_{\{a\}}}$. A *bounded expression* $\mathbf{b}$ over alphabet Σ is a regular expression of the form $w_1^* \dots w_d^*$, where $w_1, \dots, w_d \in \Sigma^*$ are nonempty words and its size is given by $|\mathbf{b}| = \sum_{i=1}^{d} |w_i|$. We use $\mathbf{b}$ to denote both the bounded expression and its language. We call a language L *bounded* when $L \subseteq \mathbf{b}$ for some bounded expression $\mathbf{b}$.

[1] The precise definition and use of ranks will be explained in Section 4.

A *grammar* is a tuple $G = \langle \Xi, \Sigma, \Delta \rangle$ where Ξ is a finite nonempty set of *nonterminals*, Σ is an alphabet of *terminals*, such that $\Xi \cap \Sigma = \emptyset$, and $\Delta \subseteq \Xi \times (\Sigma \cup \Xi)^*$ is a finite set of *productions*. For a production $(X, w) \in \Delta$, often conveniently noted $X \to w$, we define its *size* as $|(X, w)| = |w| + 1$, and $|G| = \sum_{p \in \Delta} |p|$ defines the size of G.

Given two words $u, v \in (\Sigma \cup \Xi)^*$, a production $(X, w) \in \Delta$ and a position $1 \leqslant j \leqslant |u|$, we define a *step* $u \xRightarrow{(X,w)/j}_G v$ if and only if $(u)_j = X$ and $v = (u)_1 \cdots (u)_{j-1}\, w\, (u)_{j+1} \cdots (u)_{|u|}$. We omit (X, w) or j above the arrow when clear from the context. A *control word* is a finite word $\gamma \in \Delta^*$ over the alphabet of productions. A *step sequence* $u \xRightarrow{\gamma}_G v$ is a sequence $u = w_0 \xRightarrow{(\gamma)_1}_G w_1 \ldots w_{n-1} \xRightarrow{(\gamma)_n}_G w_n = v$ where $n = |\gamma|$. If $u \in \Xi$ is a nonterminal and $v \in \Sigma^*$ is a word without nonterminals, we call the step sequence $u \xRightarrow{\gamma}_G v$ a *derivation*. When the control word γ is not important, we write $u \Rightarrow^*_G v$ instead of $u \xRightarrow{\gamma}_G v$, and we chose to omit the grammar G when clear from the context.

Given a nonterminal $X \in \Xi$ and $Y \in \Xi \cup \{\varepsilon\}$, i.e. Y is either a nonterminal or the empty word, we define the set $L_{X,Y}(G) = \{u\, v \in \Sigma^* \mid X \Rightarrow^* u\, Y\, v\}$. The set $L_{X,\varepsilon}(G)$ is called the *language* of G produced by X, and is denoted $L_X(G)$ in the following. For a set $\Gamma \subseteq \Delta^*$ of control words (also called a *control set*), we denote by $\hat{L}_{X,Y}(\Gamma, G) = \{u\, v \in \Sigma^* \mid \exists \gamma \in \Gamma \colon X \xRightarrow{\gamma} u\, Y\, v\}$ the language generated by G using only control words from Γ. We also write $\hat{L}_X(\Gamma, G)$ for $\hat{L}_{X,\varepsilon}(\Gamma, G)$.

Let $\mathbf{x}$ denote a nonempty finite set of integer variables, and $\mathbf{x}' = \{x' \mid x \in \mathbf{x}\}$. A *valuation* of $\mathbf{x}$ is a function $\nu : \mathbf{x} \to \mathbb{Z}$. The set of all such valuations is denoted by $\mathbb{Z}^{\mathbf{x}}$. A formula $\phi(\mathbf{x}, \mathbf{x}')$ is evaluated with respect to two valuations $\nu, \nu' \in \mathbb{Z}^{\mathbf{x}}$, by replacing each occurrence of $x \in \mathbf{x}$ with $\nu(x)$ and each occurrence of $x' \in \mathbf{x}'$ with $\nu'(x)$. We write $(\nu, \nu') \models \phi$ when the formula obtained from these replacements is valid. A formula $\phi_R(\mathbf{x}, \mathbf{x}')$ *defines* a relation $R \subseteq \mathbb{Z}^{\mathbf{x}} \times \mathbb{Z}^{\mathbf{x}}$ whenever for all $\nu, \nu' \in \mathbb{Z}^{\mathbf{x}}$, we have $(\nu, \nu') \in R$ iff $(\nu, \nu') \models \phi_R$. The composition of two relations $R_1, R_2 \subseteq \mathbb{Z}^{\mathbf{x}} \times \mathbb{Z}^{\mathbf{x}}$ defined by formulae $\varphi_1(\mathbf{x}, \mathbf{x}')$ and $\varphi_2(\mathbf{x}, \mathbf{x}')$, respectively, is the relation $R_1 \circ R_2 \subseteq \mathbb{Z}^{\mathbf{x}} \times \mathbb{Z}^{\mathbf{x}}$, defined by $\exists \mathbf{y} \,.\, \varphi_1(\mathbf{x}, \mathbf{y}) \wedge \varphi_2(\mathbf{y}, \mathbf{x}')$. For a finite set S, we denote its cardinality by $\|S\|$.

3 Interprocedural Flat Octogonal Reachability

In this section we define formally the class of programs and reachability problems considered. An *octagonal relation* $R \subseteq \mathbb{Z}^{\mathbf{x}} \times \mathbb{Z}^{\mathbf{x}}$ is a relation defined by a finite conjunction of constraints of the form $\pm x \pm y \leqslant c$, where $x, y \in \mathbf{x} \cup \mathbf{x}'$ and $c \in \mathbb{Z}$. The set of octagonal relations over the variables in $\mathbf{x}$ and $\mathbf{x}'$ is denoted as $\mathrm{Oct}(\mathbf{x}, \mathbf{x}')$. The *size* of an octagonal relation R, denoted $|R|$ is the size of the binary encoding of the smallest octagonal constraint defining R.

An *octagonal program* is a tuple $\mathcal{P} = \langle G, I, [.] \rangle$, where G is a grammar $G = \langle \Xi, \Sigma, \Delta \rangle$, $I \in \Xi$ is an *initial* location, and $[.] : L_I(G) \to \mathrm{Oct}(\mathbf{x}, \mathbf{x}')$ is a mapping of the words produced by the grammar G, starting with the initial location I, to octagonal relations. The alphabet Σ contains a symbol t for each *internal* program statement (that is not a call to a procedure) and two symbols $\langle t, t\rangle$ for each *call* statement t. The grammar G has three kinds of productions: (*i*) (X, t) if t is a statement leading from X to a return location, (*ii*) $(X, t\,Y)$ if t leads from X to Y, and (*iii*) $(X, \langle t\, Y\, t\rangle Z)$ if t is a call statement, Y is the initial location of the callee, and Z is the continuation of the call. Through several program transformations, we may generate another grammar with other kinds of productions. The only property we need for our results is that every grammar G with we deal with has each of its productions (X, w) satisfying: $|w{\downarrow_{\Sigma}}| \leqslant 2$ and $|w{\downarrow_{\Xi}}| \leqslant 2$ where Σ and Ξ are the terminals and nonterminals of G, respectively. Each edge t that is not a call has an associated octagonal relation $\rho_t \in \mathrm{Oct}(\mathbf{x}, \mathbf{x}')$ and each matching pair $\langle t, t\rangle$ has an associated *frame condition* $\phi_t \in \mathrm{Oct}(\mathbf{x}, \mathbf{x}')$, which equates the values of the local variables, that are not updated by the call, to their future values. The size of an octagonal program $\mathcal{P} = \langle G, I, [.]\rangle$, with $G = \langle \Xi, \Sigma, \Delta\rangle$, is the sum of the sizes of all octagonal relations labeling the productions of G, formally $|\mathcal{P}| = \sum_{(X,\,t)\in\Delta} |\rho_t| + \sum_{(X,\,tY)\in\Delta} |\rho_t| + \sum_{(X,\ \langle t\ Y\ t\rangle\ Z)\in\Delta} (|\rho_{\langle t}| + |\rho_{t\rangle}| + |\phi_t|)$.

For example, the program in Fig. 1 (a,b) is represented by the grammar in Fig. 1 (c). The terminals are mapped to octagonal relations as: $\rho_{\mathbf{t_1}} \equiv x > 0 \wedge x' = x$, $\rho_{\langle\mathbf{t_2}} \equiv x' = x-1$, $\rho_{\mathbf{t_2}\rangle} \equiv z' = z$, $\rho_{\mathbf{t_3}} \equiv x' = x \wedge z' = z+2$ and $\rho_{\mathbf{t_4}} \equiv x = 0 \wedge z' = 0$. The frame condition is $\phi_{\mathbf{t_2}} \equiv x' = x$, as only z is updated by the call $z' = \mathtt{P}(x-1)$.

Word-based semantics. For each word $w \in L_I(G)$, each occurrence of a terminal $\langle t$ in w is matched by an occurrence of $t\rangle$, and the matching positions are nested[2]. The semantics of the word $[w]$ is an octagonal relation defined inductively[3] on the structure of w: (*i*) $[t] = \rho_t$, (*ii*) $[t \cdot v] = \rho_t \circ [v]$, and (*iii*) $[\langle t \cdot u \cdot t\rangle \cdot v] = \big(\big(\rho_{\langle t} \circ [u] \circ \rho_{t\rangle}\big) \cap \phi_t\big) \circ [v]$, for all $t, \langle t, t\rangle \in \Sigma$ such that $\langle t$ and $t\rangle$ match. For instance, the semantics of the word $w = \mathbf{t_1}\langle\mathbf{t_2}\mathbf{t_4}\mathbf{t_2}\rangle\mathbf{t_3} \in L_{X_1}(G)$, for the grammar G given in Fig. 1 (c), is $[w] \equiv x = 1 \wedge z' = 2$. Observe that this word defines the effect of an execution of the program in Fig. 1 (a) where the function `P` is called twice—the first call is a top-level call, and the second is a recursive call (line 3).

Reachability problem. The semantics of a program $\mathcal{P} = \langle G, I, [.]\rangle$ is defined as $[\mathcal{P}] = \bigcup_{w\in L_I(G)} [w]$. Consider, in addition, a bounded expression $\mathbf{b}$, we define $[\mathcal{P}]_{\mathbf{b}} = \bigcup_{w\in L_I(G)\cap\mathbf{b}} [w]$. The problem asking whether $[\mathcal{P}]_{\mathbf{b}} \neq \varnothing$ for a pair $\mathcal{P}, \mathbf{b}$ is called the *flat-octagonal reachability problem.* We use $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ to denote a particular instance.

4 Index-bounded depth-first derivations

In this section, we give an alternate but equivalent program semantics based on derivations. Although simple, the word semantics is defined using a nesting relation that pairs the positions of a word labeled with matching symbols $\langle t$ and $t\rangle$. In contrast, the derivation-based semantics just needs the control word.

[2] A relation $\rightsquigarrow \subseteq \{1, \ldots, |w|\} \times \{1, \ldots, |w|\}$ is said to be nested [2] when no two pairs $i \rightsquigarrow j$ and $i' \rightsquigarrow j'$ cross each other, as in $i < i' \leqslant j < j'$.

[3] Octagonal relations are closed under intersections and compositions [23].

To define our derivation based semantics, we first define structured subsets of derivations namely the depth-first and bounded-index derivations. The reason is two-fold: (a) the correctness proof of our program transformation [11] returning the procedure-less program $\mathcal{Q}$ depends on bounded-index depth-first derivations, and (b) in the reduction of the $\text{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ problem to that of $\text{REACH}_{fo}(\mathcal{Q}, \Gamma_{\mathbf{b}})$, the computation of $\Gamma_{\mathbf{b}}$ depends on the fact that the control structure of $\mathcal{Q}$ stems from a finite automaton recognizing bounded-index depth-first derivations. Key results for our decision procedure are those of Luker [20,21] who, intuitively, shows that if $L_X(G) \subseteq \mathbf{b}$ then it is sufficient to consider depth-first derivations in which no step contains more than k simultaneous occurrences of nonterminals, for some $k > 0$ (Theorem 1).

Depth-first derivations. It is well-known that a derivation can be associated a unique parse tree. A derivation is said to be *depth-first* if it corresponds to a depth-first traversal of the corresponding parse tree. More precisely, given a step sequence $w_0 \xRightarrow{(X_0,v_0)/j_0} w_1 \dots w_{n-1} \xRightarrow{(X_{n-1},v_{n-1})/j_{n-1}} w_n$, and two integers m and i such that $0 \leqslant m < n$ and $1 \leqslant i \leqslant |w_m|$ define $f_m(i)$ to be the index ℓ of the first word w_ℓ of the step sequence in which the particular occurrence of $(w_m)_i$ appears. A step sequence is *depth-first* [21] iff for all m, $0 \leqslant m < n$:

$$f_m(j_m) = \max\{f_m(i) \mid 1 \leqslant i \leqslant |w_m| \text{ and } (w_m)_i \in \Xi\} \enspace .$$

For example, $X \xRightarrow{(X,YY)/1} YY \xRightarrow{(Y,Z)/2} YZ \xRightarrow{(Z,a)/2} Ya$ is depth-first, whereas $X \xRightarrow{(X,YY)/1} YY \xRightarrow{(Y,Z)/2} YZ \xRightarrow{(Y,Z)/1} ZZ$ is not. We have $f_2(1) = 1$ because $(w_2)_1 = Y$ first appeared at w_1, $f_2(2) = 2$ because $(w_2)_2 = Z$ first appeared at w_2, $j_2 = 1$ and $f_2(2) \nleqslant f_2(j_2)$ since $2 \nleqslant 1$. We denote by $u \underset{\mathbf{df}}{\overset{\gamma}{\Rightarrow}} w$ a depth-first step sequence and call it depth-first derivation when $u \in \Xi$ and $w \in \Sigma^*$.

Depth-first derivation-based semantics. In previous work [11], we defined the semantics of a procedural program based on the control word of the derivation instead of the produced words. We briefly recall this definition here. Given a depth-first derivation $X \underset{\mathbf{df}}{\overset{\gamma}{\Rightarrow}} w$, the relation $[\gamma] \subseteq \mathbb{Z}^{\mathbf{x}} \times \mathbb{Z}^{\mathbf{x}}$ is defined inductively on γ as follows: (i) $[(X,t)] = \rho_t$, (ii) $[(X, t\,Y) \cdot \gamma'] = \rho_t \circ [\gamma']$ where $Y \underset{\mathbf{df}}{\overset{\gamma'}{\Rightarrow}} w'$, and ($iii$) $[(X, \langle t\, Y\, t \rangle\, Z) \cdot \gamma' \cdot \gamma''] = [(X, \langle t\, Y\, t \rangle\, Z) \cdot \gamma'' \cdot \gamma'] = \big((\rho_{\langle t} \circ [\gamma'] \circ \rho_{t\rangle}) \cap \phi_t\big) \circ [\gamma'']$ where $Y \underset{\mathbf{df}}{\overset{\gamma'}{\Rightarrow}} w'$ and $Z \underset{\mathbf{df}}{\overset{\gamma''}{\Rightarrow}} w''$. We showed [11, Lemma 2] that, whenever $X \underset{\mathbf{df}}{\overset{\gamma}{\Rightarrow}} w$, we have $[w] \neq \varnothing$ iff $[\gamma] \neq \varnothing$.

Index-bounded derivations. A step $u \Rightarrow v$ is said to be k-index ($k > 0$) iff neither u nor v contains $k+1$ occurrences of nonterminals, i.e. $|u{\downarrow_\Xi}| \leqslant k$ and $|v{\downarrow_\Xi}| \leqslant k$. We denote by $u \underset{(k)}{\overset{\gamma}{\Rightarrow}} v$ a k-index step sequence and by $u \underset{\mathbf{df}(k)}{\overset{\gamma}{\Longrightarrow}} v$ a step sequence which is both depth-first and k-index. For $X \in \Xi$, $Y \in \Xi \cup \{\varepsilon\}$ and $k > 0$, we define the *k-index language* $L^{(k)}_{X,Y}(G) = \{u\,v \in \Sigma^* \mid \exists \gamma \in \Delta^* \colon X \underset{(k)}{\overset{\gamma}{\Rightarrow}} u\,Y\,v\}$, the *$k$-index depth-first control set* $\Gamma^{\mathbf{df}(k)}_{X,Y}(G) = \{\gamma \in \Delta^* \mid \exists u, v \in \Sigma^* \colon X \underset{\mathbf{df}(k)}{\overset{\gamma}{\Longrightarrow}} u\,Y\,v\}$. We write $L^{(k)}_X(G)$ and $\Gamma^{\mathbf{df}(k)}_X(G)$ when $Y = \varepsilon$, and drop G from the previous notations, when the grammar is clear from the context. For instance, for the grammar in Fig. 1 (c), we have $L^{(2)}_{X_1}(G) = \{(\mathbf{t_1}\langle\!\mathbf{t_2})^n\,\mathbf{t_4}\,(\mathbf{t_2}\!\rangle\mathbf{t_3})^n \mid n \in \mathbb{N}\} = L_{X_1}(G)$ and $\Gamma^{\mathbf{df}(2)}_{X_1} = (\mathbf{p_1p_2p_3})^* \,(\mathbf{p_4} \cup \mathbf{p_1p_2p_4p_3})$.

Theorem 1 (Lemma 2 [21], Theorem 1 [20]). *Given a grammar $G = \langle \Xi, \Sigma, \Delta\rangle$ and $X \in \Xi$:*

- *for all $w \in \Sigma^*$, $X \underset{(k)}{\Longrightarrow}^* w$ if and only if $X \underset{\mathbf{df}(k)}{\Longrightarrow}^* w$;*
- *if $L_X(G) \subseteq \mathbf{b}$ for a bounded expression $\mathbf{b}$ over Σ then $L_X(G) = L^{(K)}_X(G)$ where $K = O(|G|)$.*

The introduction of the notion of index naturally calls for an index dependent semantics and an index dependent reachability problem. As we will see later, we have tight complexity results when it comes to the index dependent reachability problem. Given $k > 0$, let $[\mathcal{P}]^{(k)} = \bigcup_{w \in L^{(k)}_I(G)} [w]$ and let $[\mathcal{P}]^{(k)}_{\mathbf{b}} = \bigcup_{w \in L^{(k)}_I(G) \cap \mathbf{b}} [w]$. Thus we define, for a constant k not part of the input, the problem $\mathrm{REACH}^{(k)}_{fo}(\mathcal{P}, \mathbf{b})$, which asks whether $[\mathcal{P}]^{(k)}_{\mathbf{b}} \neq \varnothing$.

Finite representations of bounded-index depth-first control sets. It is known that the set of k-index depth-first derivations of a grammar G is recognizable by a finite automaton [21, Lemma 5]. Below we give a formal definition of this automaton, that will be used to produce bounded control sets for covering the language of G. Moreover, we provide an upper bound on its size, which will be used to prove an upper bound for the time to compute this set (Section 5).

Given $k > 0$ and a grammar $G = \langle \Xi, \Sigma, \Delta\rangle$, we define a labeled graph $A^{\mathbf{df}(k)}_G$ such that its paths defines the set of k-index depth-first step sequences of G. To define the vertices and edges of this graph, we introduce the notion of ranked words, where the rank plays the same rôle as the value $f_m(i)$ defined previously. The advantage of ranks is that only k of them are needed for k-index depth-first derivations whereas the set of $f_m(i)$ values grows with the length of derivations. Since we restrict ourselves to k-index depth-first derivations, we thus only need k ranks, from 0 to $k-1$. The rank based definition of depth-first derivations can be found in Appendix B.1.

For a d-dimensional vector $\mathbf{v} \in \mathbb{N}^d$, we write $(\mathbf{v})_i$ for its ith element ($1 \leqslant i \leqslant d$). A vector $\boldsymbol{v} \in \mathbb{N}^d$ is said to be *contiguous* if $\{(\boldsymbol{v})_1, \ldots, (\boldsymbol{v})_d\} = \{0, \ldots, k\}$, for some $k \geqslant 0$. Given an alphabet Σ define the ranked alphabet $\Sigma^{\mathbb{N}}$ to be the set $\{\sigma^{\langle i\rangle} \mid \sigma \in \Sigma, i \in \mathbb{N}\}$. A ranked word is a word over a ranked alphabet. Given a word w of length n and an n-dimensional vector $\boldsymbol{\alpha} \in \mathbb{N}^n$, the *ranked word* $w^{\boldsymbol{\alpha}}$ is the sequence ${(w)_1}^{\langle(\boldsymbol{\alpha})_1\rangle} \ldots {(w)_n}^{\langle(\boldsymbol{\alpha})_n\rangle}$, in which the ith element of $\boldsymbol{\alpha}$ annotates the ith symbol of w. We also denote $w^{\langle\!\langle c\rangle\!\rangle} = {(w)_1}^{\langle c\rangle} \ldots {(w)_{|w|}}^{\langle c\rangle}$ as a shorthand. Let $A^{\mathbf{df}(k)}_G = \langle Q, \Delta, \rightarrow\rangle$ be the following labeled graph, where:

$$Q = \{w^{\boldsymbol{\alpha}} \mid w \in \Xi^*, |w| \leqslant k, \boldsymbol{\alpha} \in \mathbb{N}^{|w|} \text{ is contiguous}, (\boldsymbol{\alpha})_1 \leqslant \cdots \leqslant (\boldsymbol{\alpha})_{|w|}\}$$

is the set of vertices, the edges are labeled by the set Δ of productions of G, and the edge relation is defined next. For all vertices $q, q' \in Q$ and labels $(X, w) \in \Delta$, we have $q \xrightarrow{(X,w)} q'$ if and only if

- $q = u\,X^{\langle i\rangle}\,v$ for some u, v, where i is the maximum rank in q, and

– $q' = u\,v\,(w{\downarrow_{\Xi}})^{\langle\!\langle i' \rangle\!\rangle}$, where $|u\,v\,(w{\downarrow_{\Xi}})^{\langle\!\langle i' \rangle\!\rangle}| \leqslant k$ and $i' = \begin{cases} 0 & \text{if } u\,v = \varepsilon \\ i & \text{else if } (u\,v){\downarrow_{\Xi^{\langle i \rangle}}} = \varepsilon \\ i+1 & \text{else} \end{cases}$

We denote by $|A_G^{\mathbf{df}(k)}| = \|Q\|$ the size (number of vertices) of $A_G^{\mathbf{df}(k)}$. In the following, we omit the subscript G from $A_G^{\mathbf{df}(k)}$, when the grammar is clear from the context. For example, the graph $A^{\mathbf{df}(2)}$ for the grammar from Fig. 1 (c), is the subgraph of Fig. 1 (d) enclosed in a dashed line.

Lemma 1. *Given $G = \langle \Xi, \Sigma, \Delta \rangle$, and $k > 0$, for each $X \in \Xi$, $Y \in \Xi \cup \{\varepsilon\}$ and $\gamma \in \Delta^*$, we have $\gamma \in \Gamma_{X,Y}^{\mathbf{df}(k)}(G)$ if and only if $X^{\langle 0 \rangle} \xrightarrow{\gamma} Y^{\langle 0 \rangle}$ is a path in $A_G^{\mathbf{df}(k)}$. Moreover, we have $|A_G^{\mathbf{df}(k)}| = |G|^{\mathcal{O}(k)}$.*

5 A Decision Procedure for $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$

In this section we describe a decision procedure for the problem $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ where $\mathcal{P} = \langle G, I, [.] \rangle$ is an octagonal program, whose underlying grammar is $G = \langle \Xi, \Sigma, \Delta \rangle$, and $\mathbf{b} = w_1^* \ldots w_d^*$ is a bounded expression over Σ. The procedure follows the roadmap described next.

First, we compute, in time polynomial in the sizes of $\mathcal{P}$ and $\mathbf{b}$, a set of programs $\{\mathcal{P}_i = \langle G^{\cap}, X_i, [.] \rangle\}_{i=1}^{\ell}$, such that $L_I(G) \cap \mathbf{b} = \bigcup_{i=1}^{\ell} L_{X_i}(G^{\cap})$, which implies $[\mathcal{P}]_{\mathbf{b}} = \bigcup_{i=1}^{\ell} [\mathcal{P}_i]$. The grammar $G^{\cap}$ is an automata-theoretic product between the grammar G and the bounded expression $\mathbf{b}$. For space reasons, the formal definition of $G^{\cap}$ is deferred to Appendix A, and we refer the reader to Example 1. Deciding $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ reduces thus to deciding several instances $\{\mathrm{REACH}_{fo}(\mathcal{P}_i, \mathbf{b})\}_{i=1}^{\ell}$ of the fo-reachability problem.

Example 1. Let us consider the bounded expression $\mathbf{b} = (ac)^*\,(ab)^*\,(db)^*$. Consider the grammar $G^{\mathbf{b}}$ with the following productions: $\mathrm{Q}_1^{(1)} \to a\,\mathrm{Q}_2^{(1)} \mid \varepsilon$, $\mathrm{Q}_1^{(2)} \to a\,\mathrm{Q}_2^{(2)} \mid \varepsilon$, $\mathrm{Q}_1^{(3)} \to d\,\mathrm{Q}_2^{(3)} \mid \varepsilon$, $\mathrm{Q}_2^{(1)} \to c\,\mathrm{Q}_1^{(1)} \mid c\,\mathrm{Q}_1^{(2)} \mid c\,\mathrm{Q}_1^{(3)}$, $\mathrm{Q}_2^{(2)} \to b\,\mathrm{Q}_1^{(2)} \mid b\,\mathrm{Q}_1^{(3)}$, $\mathrm{Q}_2^{(3)} \to b\,\mathrm{Q}_1^{(3)}$. It is easy to check that $\mathbf{b} = \bigcup_{i=1}^{3} L_{\mathrm{Q}_1^{(i)}}(G^{\mathbf{b}})$. Let $G = \langle \{X, Y, Z, T\}, \{a, b, c, d\}, \Delta \rangle$ where $\Delta = \{X \to aY,\ Y \to Zb,\ Z \to cT,\ Z \to \varepsilon,\ T \to Xd\}$, i.e. we have $L_X(G) = \{(ac)^n\,ab\,(db)^n \mid n \in \mathbb{N}\}$. The following productions define a grammar $G^{\cap}$:

$$
\begin{array}{ll}
[\mathrm{Q}_1^{(j)} X \mathrm{Q}_1^{(3)}] \xrightarrow{p_1} a\,[\mathrm{Q}_2^{(j)} Y \mathrm{Q}_1^{(3)}] & [\mathrm{Q}_2^{(1)} Y \mathrm{Q}_1^{(3)}] \xrightarrow{p_2} [\mathrm{Q}_2^{(1)} Z \mathrm{Q}_2^{(3)}]\,b \\
[\mathrm{Q}_2^{(1)} Z \mathrm{Q}_2^{(3)}] \xrightarrow{p_3} c\,[\mathrm{Q}_1^{(j)} T \mathrm{Q}_2^{(3)}] & [\mathrm{Q}_2^{(2)} Z \mathrm{Q}_2^{(2)}] \xrightarrow{p_4} \varepsilon \\
[\mathrm{Q}_1^{(j)} T \mathrm{Q}_2^{(3)}] \xrightarrow{p_5} [\mathrm{Q}_1^{(j)} X \mathrm{Q}_1^{(3)}]\,d \text{ , for } j = 1, 2 & [\mathrm{Q}_1^{(2)} X \mathrm{Q}_1^{(3)}] \xrightarrow{p_6} a\,[\mathrm{Q}_2^{(2)} Y \mathrm{Q}_1^{(3)}] \\
[\mathrm{Q}_2^{(2)} Y \mathrm{Q}_1^{(3)}] \xrightarrow{p_7} [\mathrm{Q}_2^{(2)} Z \mathrm{Q}_2^{(2)}]\,b &
\end{array}
$$

One can check $L_X(G) = L_X(G) \cap \mathbf{b} = L_{[\mathrm{Q}_1^{(1)} X \mathrm{Q}_1^{(3)}]}(G^{\cap}) \cup L_{[\mathrm{Q}_1^{(2)} X \mathrm{Q}_1^{(3)}]}(G^{\cap})$. ■

A bounded expression $\mathbf{b} = w_1^* \ldots w_d^*$ over alphabet Σ is said to be *d-letter-bounded* (or simply letter-bounded, when d is not important) when $|w_i| = 1$, for all $i = 1, \ldots, d$. A letter-bounded expression $\widetilde{\mathbf{b}}$ is *strict* if all its symbols are distinct. A language $L \subseteq \Sigma^*$ is (strict, letter-) bounded iff $L \subseteq \mathbf{b}$, for some (strict, letter-) bounded expression $\mathbf{b}$.

Second, we reduce the problem from $\mathbf{b} = w_1^* \ldots w_d^*$ to the strict letter-bounded case $\widetilde{\mathbf{b}} = a_1^* \ldots a_d^*$, by building a grammar $G^{\bowtie}$, with the same nonterminals as $G^{\cap}$, such that, for each $i = 1, \ldots, \ell$ (*i*) $L_{X_i}(G^{\bowtie}) \subseteq \widetilde{\mathbf{b}}$, (*ii*) $w_1^{i_1} \ldots w_d^{i_d} \in L_{X_i}^{(k)}(G^{\cap})$ iff $a_1^{i_1} \ldots a_d^{i_d} \in L_{X_i}^{(k)}(G^{\bowtie})$, for all $k > 0$ (*iii*) from each control set Γ that covers the language $L_{X_i}^{(k)}(G^{\bowtie}) \subseteq \hat{L}_{X_i}(\Gamma, G^{\bowtie})$ for some $k > 0$, one can compute, in polynomial time, a control set $\widetilde{\Gamma}$ that covers the language $L_{X_i}^{(k)}(G^{\cap}) \subseteq \hat{L}_{X_i}(\widetilde{\Gamma}, G^{\cap})$.

Example 2 (contd. from Example 1). Let $\mathcal{A} = \{a_1, a_2, a_3\}$, $\widetilde{\mathbf{b}} = a_1^* a_2^* a_3^*$ and $h \colon \mathcal{A} \to \Sigma^*$ be the homomorphism given by $h(a_1) = ac, h(a_2) = ab$ and $h(a_3) = db$. The grammar $G^{\bowtie}$ results from deleting a's and d's in $G^{\cap}$ and replacing b in p_2 by a_3, b in p_7 by a_2 and c by a_1. Then, it is easy to check that $h^{-1}(L_X(G)) \cap \widetilde{\mathbf{b}} = L_{[\mathrm{Q}_1^{(1)} X \mathrm{Q}_1^{(3)}]}(G^{\bowtie}) \cup L_{[\mathrm{Q}_1^{(2)} X \mathrm{Q}_1^{(3)}]}(G^{\bowtie}) = \{a_1^n\, a_2\, a_3^n \mid n \in \mathbb{N}\}$. ∎

Third, for the strict letter-bounded grammar $G^{\bowtie}$, we compute a control set $\Gamma \subseteq (\Delta^{\bowtie})^*$ using the result of Theorem 3, which yields a set of bounded expressions $\mathcal{S}_{\widetilde{\mathbf{b}}} = \{\Gamma_{i,1}, \ldots, \Gamma_{i,m_i}\}$, such that $L_{X_i}^{(k)}(G^{\bowtie}) \subseteq \bigcup_{j=1}^{m_i} \hat{L}_{X_i}(\Gamma_{i,j} \cap \Gamma_{X_i}^{\mathbf{df}(k+1)}, G^{\bowtie})$. By applying the aforementioned transformation (*iii*) from Γ to $\widetilde{\Gamma}$, we obtain that $L_{X_i}^{(k)}(G^{\cap}) \subseteq \bigcup_{j=1}^{m_i} \hat{L}_{X_i}(\widetilde{\Gamma}_{i,j} \cap \Gamma_{X_i}^{\mathbf{df}(k+1)}, G^{\cap})$. Theorem 1 allows to effectively compute value $K > 0$ such that $L_{X_i}(G^{\cap}) = L_{X_i}^{(K)}(G^{\cap})$, for all $i = 1, \ldots, \ell$. Thus we obtain[4] $L_{X_i}(G^{\cap}) = \bigcup_{j=1}^{m_i} \hat{L}_{X_i}(\widetilde{\Gamma}_{i,j} \cap \Gamma_{X_i}^{\mathbf{df}(K+1)}, G^{\cap})$, for all $i = 1, \ldots, \ell$.

The final step consists in building a finite automaton $A^{\mathbf{df}(K+1)}$ that recognizes the control set $\Gamma_{X_i}^{\mathbf{df}(K+1)}$ (Lemma 1). This yields a procedure-less program $\mathcal{Q}$, whose control structure is given by $A^{\mathbf{df}(K+1)}$, and whose labels are given by the semantics of control words. We recall that, for every word $w \in L_{X_i}(G^{\cap})$ there exists a control word $\gamma \in \Gamma_{X_i}^{\mathbf{df}(K+1)}$ such that $[w] \neq \emptyset$ iff $[\gamma] \neq \emptyset$. We have thus reduced each of the instances $\{\mathrm{REACH}_{fo}(\mathcal{P}_i, \mathbf{b})\}_{i=1}^{\ell}$ of the fo-reachability problem to a set of instances $\{\mathrm{REACH}_{fo}(\mathcal{Q}, \widetilde{\Gamma}_{i,j}) \mid 1 \leqslant i \leqslant \ell,\ 1 \leqslant j \leqslant m_i\}$. The latter problem, for procedure-less programs, is decidable in NPTIME [7]. Next is our main result whose proof is in Appendix B.6.

Theorem 2. *Let $\mathcal{P} = \langle G, I, [.] \rangle$ be an octagonal program, where $G = \langle \Xi, \Sigma, \Delta \rangle$ is a grammar, and $\mathbf{b}$ is a bounded expression over Σ. Then the problem* $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ *is decidable in* NEXPTIME, *with a* NP*-hard lower bound. If, moreover, k is a constant,* $\mathrm{REACH}_{fo}^{(k)}(\mathcal{P}, \mathbf{b})$ *is* NP*-complete.*

The rest of this section describes the construction of the control sets $\mathcal{S}_{\widetilde{\mathbf{b}}}$ and gives upper bounds on the time needed for this computation. We use the following ingredients: (*i*) Algorithm 1 for building bounded control sets for s-letter bounded languages, where $s \geqslant 0$ is a constant (in our case, at most 2) (Section 5.1), and (*ii*) a decomposition of k-index depth-first derivations, that distinguishes between

[4] Because $L_{X_i}(G^{\cap}) \subseteq L_{X_i}^{(K)}(G^{\cap}) \subseteq \bigcup_{j=1}^{m_i} \hat{L}_{X_i}(\widetilde{\Gamma}_{i,j} \cap \Gamma_{X_i}^{\mathbf{df}(k+1)}, G^{\cap}) \subseteq L_{X_i}(G^{\cap})$.

a prefix producing a word from the 2-letter bounded expression $a_1^* a_d^*$, and a suffix producing two words included in bounded expressions strictly smaller than $\widetilde{\mathbf{b}}$ (Section 5.2). The decomposition enables the generalization from s-letter bounded languages where s is a constant to arbitrary letter bounded languages. In particular, the required set of bounded expressions $\mathcal{S}_{\widetilde{\mathbf{b}}}$ is built inductively over the structure of this decomposition, applying at each step Algorithm 1 which computes bounded control sets for 2-letter bounded languages. The main algorithm (Algorithm 2) returns a finite set $\mathcal{S}_{\widetilde{\mathbf{b}}}$ of bounded expressions $\{\Gamma_1, \ldots, \Gamma_m\}$. Below we abuse notation and write $\bigcup \mathcal{S}_{\widetilde{\mathbf{b}}}$ for $\bigcup_{i=1}^m \Gamma_i$. The time needed to build each bounded expression $\Gamma_i \in \mathcal{S}_{\widetilde{\mathbf{b}}}$ is $|G|^{\mathcal{O}(k)}$ and does not depend of $|\widetilde{\mathbf{b}}| = d$, whereas the time needed to build the entire set $\mathcal{S}_{\widetilde{\mathbf{b}}}$ is $|G|^{\mathcal{O}(k)+d}$. These arguments come in handy when deriving an upper bound on the (non-deterministic) time complexity of the fo-reachability problem for programs with arbitrary call graphs. A non-deterministic version of Algorithm 2 that choses one set $\Gamma_i \in \mathcal{S}_{\widetilde{\mathbf{b}}}$, instead of building the whole set $\mathcal{S}_{\widetilde{\mathbf{b}}}$, is used to establish the upper bounds for the $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ and $\mathrm{REACH}_{fo}^{(k)}(\mathcal{P}, \mathbf{b})$ problems in the proof of Theorem 2.

5.1 Constant s-Letter Bounded Languages

Here we define an algorithm for building bounded control sets that are sufficient for covering a s-letter bounded language $L_X(G) \subseteq a_1^* \ldots a_s^*$, when $s \geqslant 0$ is a constant[5], i.e. not part of the input of the algorithm. In the following, we consider the labeled graph $A^{\mathbf{df}(k)} = \langle Q, \Delta, \rightarrow \rangle$, whose paths correspond to the k-index depth-first step sequences of G (Lemma 1). Recall that the number of vertices in this graph is $|A^{\mathbf{df}(k)}| \leqslant |G|^{2k}$.

Given $q, q' \in Q$, we denote by $\Pi(q, q')$ the set of paths with source q and destination q'. For a path π, we denote by $\omega(\pi) \in \Delta^*$ the sequence of edge labels on π. A path π is a *cycle* if its endpoints coincide. Furthermore, the path is said to be an *elementary cycle* if it contains no other cycle than itself. Finally, π is acyclic if it contains no cycle. The word *induced* by a path in $A^{\mathbf{df}(k)}$ is the sequence of terminal symbols generated by the productions fired along that path. Observe that, since $L_X(G) \subseteq a_1^* \ldots a_s^*$, any word induced by a subpath of some path $\pi \in \Pi(X^{\langle 0 \rangle}, \varepsilon)$ is necessarily of the form $a_1^{i_1} \ldots a_s^{i_s}$, for some $i_1, \ldots, i_s \geqslant 0$.

Algorithm 1 describes the effective construction of a bounded expression Γ over the productions of G using the sets of elementary cycles of $A^{\mathbf{df}(k)}$. The crux is to find, for each vertex q of $A^{\mathbf{df}(k)}$, a subset C_q of elementary cycles having q at the endpoints, such that the set of words induced by C_q is that of the entire set of elementary cycles having q at endpoints. Since the only vertex occurring more than once in an elementary cycle ρ is the endpoint q, we have that $|\rho|$ is at most the number of vertices $|A^{\mathbf{df}(k)}|$, and each production rule generates at most 2 terminal symbols, hence no word induced by a elementary cycle is longer than $2|A^{\mathbf{df}(k)}| \leqslant 2|G|^{2k}$. The number of words $a_1^{i_1} \ldots a_s^{i_s}$ induced by elementary

[5] In our case $s = 0, 1, 2$, but the construction can be generalized to any constant $s \geqslant 0$.

cycles with endpoints q is thus bounded by the number of nonnegative solutions of the inequality $x_1 + \cdots + x_s \leqslant 2|G|^{2k}$, which, in turn, is of the order of $|G|^{\mathcal{O}(k)}$. So for each vector $\boldsymbol{v} \in \mathbb{N}^s$ such that $(\boldsymbol{v})_1 + \cdots + (\boldsymbol{v})_s \leqslant 2|G|^{2k}$, it suffices to include in C_q only one elementary cycle inducing the word $a_1^{(\boldsymbol{v})_1} \dots a_s^{(\boldsymbol{v})_s}$. Thus it is sufficient to consider sets C_q of cardinality $\|C_q\| = |G|^{\mathcal{O}(k)}$, for all $q \in Q$.

Lines (2–5) of Algorithm 1 build a graph $\mathcal{H}$ with vertices $\langle q, a_1^{i_1} \dots a_s^{i_s}\rangle$, where $q \in Q$ is a vertex of $A^{\mathbf{df}(k)}$ and $i_1, \dots, i_s$ a solution to the above inequality (line 2), hence $\mathcal{H}$ is a finite and computable graph. There is an edge between two vertices $\langle q, a_1^{i_1} \dots a_s^{i_s}\rangle$ and $\langle q', a_1^{j_1} \dots a_s^{j_s}\rangle$ in $\mathcal{H}$ if and only if $q \xrightarrow{p} q'$ in $A^{\mathbf{df}(k)}$ and $a_\ell^{j_\ell} = a_\ell^{i_\ell} \cdot (p{\downarrow_{a_\ell}})$ for every ℓ, that is j_ℓ is the sum of i_ℓ and the number of occurrences of a_ℓ produced by p (which is precisely captured by the word $p{\downarrow_{a_\ell}}$) (line 4). The sets C_q are computed by applying the Dijkstra's single source shortest path algorithm[6] to the graph $\mathcal{H}$ (line 7) and retrieving in C_q the paths $\langle q, \varepsilon\rangle \to^* \langle q, a_1^{i_1} \dots a_s^{i_s}\rangle$, such that $i_1 + \cdots + i_s \leqslant 2|G|^{2k}$ (line 9).

For a finite set of words $S = \{u_1, \dots, u_h\}$, the function $\textsc{Concat}(S)$ returns the bounded expression $u_1^* \dots u_h^*$. Algorithm 1 uses this function to build a bounded expression Γ that covers all words induced by paths from $\Pi(X^{\langle 0\rangle}, \varepsilon)$. This construction relies on the following argument: for each $\pi \in \Pi(X^{\langle 0\rangle}, \varepsilon)$, there exists another path $\pi' \in \Pi(X^{\langle 0\rangle}, \varepsilon)$, such that their induced words coincide, and, moreover, π' can be factorizedas $\varsigma_1 \cdot \theta_1 \cdots \varsigma_\ell \cdot \theta_\ell \cdot \varsigma_{\ell+1}$, where $\varsigma_1 \in \Pi(X^{\langle 0\rangle}, q_1)$, $\varsigma_{\ell+1} \in \Pi(q_\ell, \varepsilon)$ and $\varsigma_j \in \Pi(q_{j-1}, q_j)$ for each $1 < j \leqslant \ell$ are acyclic paths, $\theta_1, \dots, \theta_\ell$ are elementary cycles with endpoints $q_1, \dots, q_\ell$, respectively, and $\ell \leqslant |A^{\mathbf{df}(k)}|$. Thus we can cover each segment ς_i by a bounded expression $C = \textsc{Concat}(\Delta)^{|G|^{2k}-1}$ (line 13), and each segment θ_j by the bounded expression $B_0 = \textsc{Concat}(\{\omega(\pi) \mid \pi \in C_{q_j}\})$ (line 10), yielding the required expression Γ. The following lemma proves the correctness of Algorithm 1 and gives an upper bound on its runtime.

Lemma 2. *Let $G = \langle \Xi, \mathcal{A}, \Delta\rangle$ be a grammar and $a_1^* \dots a_s^*$ is a strict s-letter-bounded expression over $\mathcal{A}$, where $s \geqslant 0$ is a constant. Then, for each $k > 0$ there exists a bounded expression Γ over Δ such that, for all $X \in \Xi$ and $Y \in \Xi \cup \{\varepsilon\}$, we have $L^{(k)}_{X,Y}(G) = \hat{L}_{X,Y}(\Gamma \cap \Gamma^{\mathbf{df}(k)}_{X,Y}, G)$, provided that $L_{X,Y}(G) \subseteq a_1^* \dots a_s^*$. Moreover, Γ is computable in time $|G|^{\mathcal{O}(k)}$.*

5.2 The General Case

The key to the general case is a lemma decomposing derivations.

Decomposition Lemma. Our construction of a bounded control set that covers a strict letter-bounded context-free language $L_X(G) \subseteq a_1^* \dots a_d^*$ is by induction on $d \geqslant 1$, and is inspired by a decomposition of the derivations in G, given by Ginsburg [12, Chapter 5.3, Lemma 5.3.3]. Because his decomposition is oblivious to the index or the depth-first policy, it is too weak for our needs. Therefore, we give first a stronger decomposition result for k-index depth-first derivations.

[6] We consider all edges to be of weight 1.

Algorithm 1 Control Sets for the Case of Constant Size Bounded Expressions

input A grammar $G = \langle \Xi, \mathcal{A}, \Delta \rangle$,
a strict s-letter-bounded expression $a_1^* \ldots a_s^*$ over $\mathcal{A}$, where $s \geqslant 0$ is a fixed constant, and $k > 0$

output a bounded expression Γ over Δ such that $L_{X,Y}^{(k)}(G) = \hat{L}_{X,Y}(\Gamma \cap \Gamma_{X,Y}^{\mathbf{df}(k)}, G)$ for all $X \in \Xi$ and $Y \in \Xi \cup \{\varepsilon\}$, such that $L_{X,Y}(G) \subseteq a_1^* \ldots a_s^*$

function $\textsc{ConstantBoundedControlSet}(G, a_1^* \ldots a_s^*, k)$
  $Val \leftarrow \{a_1^{k_1} \cdots a_s^{k_s} \mid \sum_{j=1}^{s} k_j \leqslant 2|G|^{2k}\}$
  $V \leftarrow Q \times Val$ $\quad\triangleright$ Q are the vertices of $A^{\mathbf{df}(k)}$, considering $\|Q\| \leqslant |G|^{2k}$ suffices
  $\delta \leftarrow \{\langle q, a_1^{i_1} \ldots a_s^{i_s}\rangle \xrightarrow{p} \langle q', a_1^{j_1} \ldots a_s^{j_s}\rangle \mid q \xrightarrow{p} q' \text{ in } A^{\mathbf{df}(k)},\ \forall \ell \in \{1, \ldots, s\}.\ a_\ell^{j_\ell} = a_\ell^{i_\ell} \cdot (p\downarrow_{a_\ell})\}$
  $\mathcal{H} \leftarrow \langle V, \Delta, \delta \rangle$
  $B_0 \leftarrow \varepsilon$
  $\textsc{DijkstraShortestPaths}(\mathcal{H})$
  for $q \in Q$ **do**
    $C_q \leftarrow \bigcup_{w \in Val} \textsc{GetShortestPath}(\mathcal{H}, \langle q, \varepsilon \rangle, \langle q, w \rangle)$
    $B_0 \leftarrow B_0 \cdot \textsc{Concat}(\{\omega(\pi) \mid \pi \in C_q\})$
  $C \leftarrow \varepsilon$
  for $i = 1 \ldots |G|^{2k} - 1$ **do**
    $C \leftarrow C \cdot \textsc{Concat}(\Delta)$
  $\Gamma \leftarrow \varepsilon$
  for $i = 1 \ldots |G|^{2k}$ **do**
    $\Gamma \leftarrow \Gamma \cdot C \cdot B_0$
  $\Gamma \leftarrow \Gamma \cdot C \cdot B_0 \cdot C$
  return Γ

Without loss of generality, the decomposition lemma assumes the bounded expression covering $L_X(G)$ to be *minimal*: a strict letter-bounded expression $\widetilde{\mathbf{b}}$ is *minimal* for a language L iff $L \subseteq \widetilde{\mathbf{b}}$ and for every subexpression $\mathbf{b}'$, resulting from deleting some a_i^* from $\widetilde{\mathbf{b}}$, we have $L \nsubseteq \mathbf{b}'$. Clearly, each strict letter-bounded language has a unique minimal expression.

Basically, for every k-index depth-first derivation with control word γ, its productions can be rearranged into a $(k+1)$-index depth-first derivation, consisting of a prefix $\gamma^\sharp$ producing a word in $a_1^*\, a_d^*$, then a production (X_i, w) followed by two control words γ' and γ'' that produce words contained within two bounded expressions $a_\ell^* \ldots a_m^*$ and $a_m^* \ldots a_r^*$, respectively, where $\max(m - \ell, r - m) < d - 1$ (Lemma 3). Let us first define the partition $(\Xi_{\widehat{1..d}}, \Xi_{\widetilde{1..d}})$ of Ξ, as follows:

$$Y \in \Xi_{\widehat{1..d}} \Leftrightarrow L_Y(G) \cap (a_1 \cdot \mathcal{A}^*) \neq \varnothing \ \text{ and } \ L_Y(G) \cap (\mathcal{A}^* \cdot a_d) \neq \varnothing \ .$$

Naturally, define $\Xi_{\widetilde{1..d}} = \Xi \backslash \Xi_{\widehat{1..d}}$. Since the bounded expression $a_1^* \ldots a_d^*$ is, by assumption, minimal for $L_X(G)$, then a_1 occurs in some word of $L_X(G)$ and a_d occurs in some word of $L_X(G)$. Thus it is always the case that $\Xi_{\widehat{1..d}} \neq \varnothing$, since $X \in \Xi_{\widehat{1..d}}$. The partition of nonterminals into $\Xi_{\widehat{1..d}}$ and $\Xi_{\widetilde{1..d}}$ induces a decomposition of the grammar G. First, let $G^\sharp = \langle \Xi, \mathcal{A}, \Delta^\sharp \rangle$, where:

$$\Delta^\sharp = \{(X_j, w) \in \Delta \mid X_j \in \Xi_{\widetilde{1..d}}\} \cup \{(X_j, u\, X_r\, v) \in \Delta \mid X_j, X_r \in \Xi_{\widehat{1..d}}\} \ .$$

Then, for each production $(X_i, w) \in \Delta$ such that $X_i \in \Xi_{\widehat{1..d}}$ and $w \in (\Xi_{\widetilde{1..d}} \cup \mathcal{A})^*$, we define the grammar $G_{i,w} = \langle \Xi, \mathcal{A}, \Delta_{i,w} \rangle$, where:

$$\Delta_{i,w} = \{(X_j, v) \in \Delta \mid X_j \in \Xi_{\widetilde{1..d}}\} \cup \{(X_i, w)\} \ .$$

The decomposition of derivations is formalized by the following lemma:

Lemma 3. *Given a grammar $G = \langle \Xi, \mathcal{A}, \Delta \rangle$, a nonterminal $X \in \Xi$ such that $L_X(G) \subseteq a_1^* \ldots a_d^*$ for some $d \geqslant 3$, and $k > 0$, for every derivation $X \xRightarrow[\mathbf{df}(k)]{\gamma}_G w$, there exists a production* $\mathrm{p} = (X_i, a\, y\, b\, z) \in \Delta$ *with $X_i \in \Xi_{\widehat{1..d}}$, $a, b \in \mathcal{A} \cup \{\varepsilon\}$ and $y, z \in \Xi_{\widetilde{1..d}} \cup \{\varepsilon\}$, and control words $\gamma^\sharp \in (\Delta^\sharp)^*$, $\gamma_y, \gamma_z \in (\Delta_{i,aybz})^*$, such that $\gamma^\sharp\, \mathrm{p}\, \gamma_y\, \gamma_z$ is a permutation of γ and:*

1. *$X \xRightarrow[\mathbf{df}(k+1)]{\gamma^\sharp}_{G^\sharp} u\, X_i\, v$ is a step sequence in $G^\sharp$ with $u, v \in \mathcal{A}^*$;*
2. *$y \xRightarrow[\mathbf{df}(k_y)]{\gamma_y}_{G_{i,aybz}} u_y$ and $z \xRightarrow[\mathbf{df}(k_z)]{\gamma_z}_{G_{i,aybz}} u_z$ are (possibly empty) derivations in $G_{i,aybz}$ ($u_y, u_z \in \mathcal{A}^*$), for some integers $k_y, k_z > 0$, such that* $\max(k_y, k_z) \leqslant k$ *and* $\min(k_y, k_z) \leqslant k - 1$*;*
3. *$X \xRightarrow[\mathbf{df}(k+1)]{\gamma^\sharp\, \mathrm{p}\, \gamma_y\, \gamma_z}_G w$ if $y \xRightarrow[\mathbf{df}(k-1)]{\gamma_y}_{G_{i,aybz}} u_y$, and $X \xRightarrow[\mathbf{df}(k+1)]{\gamma^\sharp\, \mathrm{p}\, \gamma_z\, \gamma_y}_G w$ if $z \xRightarrow[\mathbf{df}(k-1)]{\gamma_z}_{G_{i,aybz}} u_z$;*
4. *$L_{X,X_i}(G^\sharp) \subseteq a_1^* a_d^*$;*
5. *$L_y(G_{i,aybz}) \subseteq a_\ell^* \ldots a_m^*$ if $y \in \Xi_{\widetilde{1..d}}$, and $L_z(G_{i,aybz}) \subseteq a_m^* \ldots a_r^*$ if $z \in \Xi_{\widetilde{1..d}}$, for some integers $1 \leqslant \ell \leqslant m \leqslant r \leqslant d$, such that* $\max(m - \ell, r - m) < d - 1$.

Let us now turn to the general case, in which the size of the strict letter-bounded expression $\widetilde{\mathbf{b}} = a_1^* \ldots a_d^*$ is not constant, i.e. d is part of the input of the algorithm. The output of Algorithm 2 is a finite set of bounded expressions $\mathcal{S}_{\widetilde{\mathbf{b}}}$ such that $L_X^{(k)}(G) \subseteq \hat{L}_X(\bigcup \mathcal{S}_{\widetilde{\mathbf{b}}} \cap \mathit{\Gamma}_X^{\mathbf{df}(k+1)}, G)$. The construction of the set $\mathcal{S}_{\widetilde{\mathbf{b}}}$ by Algorithm 2 (function LetterBoundedControlSet) follows the structure of the decomposition of control words given by Lemma 3. For every k-index depth-first derivation with control word γ, its productions can be rearranged into a $(k+1)$-index depth-first derivation, consisting of (i) a prefix $\gamma^\sharp$ producing a word in $a_1^*\, a_d^*$, then (ii) a *pivot* production (X_i, w) followed by two words γ' and γ'' such that: (iii) γ' and γ'' produce words included in two bounded expressions $a_\ell^* \ldots a_m^*$ and $a_m^* \ldots a_r^*$, respectively, where $\max(m - \ell, r - m) < d - 1$. The algorithm follows this decomposition and builds bounded expressions $\mathit{\Gamma}^\sharp$, $(X_i, w)^*$, and the sets $\mathcal{S}'$ and $\mathcal{S}''$ with the goal of capturing $\gamma^\sharp$, (X_i, w), γ and γ'', respectively, for all the control words such as γ. Because $\gamma^\sharp$ produces a word from $a_1^*\, a_d^*$, the bounded expression $\mathit{\Gamma}^\sharp$ is built calling ConstantBoundedControlSet (line 9). Since γ' and γ'' produce words within two sub-expressions of $a_1^* \ldots a_d^*$ with as many as $d - 2$ letters, these cases are handled by two recursive calls to LetterBoundedControlSet (lines 16 and 19).

Theorem 3. *Given a grammar $G = \langle \Xi, \mathcal{A}, \Delta \rangle$, and $X \in \Xi$, such that $L_X(G) \subseteq \widetilde{\mathbf{b}}$, where $\widetilde{\mathbf{b}}$ is the minimal strict d-letter bounded expression for $L_X(G)$, for each $k > 0$, there exists a finite set of bounded expressions $\mathcal{S}_{\widetilde{\mathbf{b}}}$ over Δ such that $L_X^{(k)}(G) \subseteq \hat{L}_X(\bigcup \mathcal{S}_{\widetilde{\mathbf{b}}} \cap \mathit{\Gamma}_X^{\mathbf{df}(k+1)}, G)$. Moreover, $\mathcal{S}_{\widetilde{\mathbf{b}}}$ can be constructed in time $|G|^{\mathcal{O}(k)+d}$ and each $\mathit{\Gamma} \in \mathcal{S}_{\widetilde{\mathbf{b}}}$ can be constructed in time $|G|^{\mathcal{O}(k)}$.*

The next lemma shows that the worst-case exponential blowup in the value k is unavoidable.

Algorithm 2 Control Sets for Letter-Bounded Grammars

input A grammar $G = \langle \Xi, \mathcal{A}, \Delta \rangle$, a nonterminal $X \in \Xi$,
a strict d-letter-bounded expression $\widetilde{\mathbf{b}}$ over $\mathcal{A}$, such that $L_X(G) \subseteq \widetilde{\mathbf{b}}$, and $k > 0$
output a set $\mathcal{S}_{\tilde{\mathbf{b}}}$ of bounded expressions over Δ, such that $L_X^{(k)}(G) \subseteq \hat{L}_X(\bigcup \mathcal{S}_{\tilde{\mathbf{b}}} \cap \Gamma_X^{\mathbf{df}(k+1)}, G)$

function LetterBoundedControlSet$(G_0, X_0, a_{i_1}^* \dots a_{i_d}^*, k)$
  match G_0 **with** $\langle \Xi, \mathcal{A}, \Delta_0 \rangle$
  $a_{j_1}^* \cdots a_{j_s}^* \leftarrow$ minimizeExpression$(G_0, X_0, a_{i_1}^* \cdots a_{i_d}^*)$ ▷ $\{j_1, \dots, j_s\} \subseteq \{i_1, \dots, i_d\}$
  if $|a_{j_1}^* \dots a_{j_s}^*| \leqslant 2$ **then**
    return $\{$ConstantBoundedControlSet$(G_0, a_{j_1}^* \dots a_{j_s}^*, k)\}$
  $(\Xi_{\overset{\frown}{j_1 \cdot\cdot j_s}}, \Xi_{\overset{\smile}{j_1 \cdot\cdot j_s}}) \leftarrow$ partitionNonterminals$(G_0, a_{j_1}^*\ a_{j_s}^*)$
  $\Delta^\sharp \leftarrow \{(X_j, w) \in \Delta_0 \mid X_j \in \Xi_{\overset{\smile}{j_1 \cdot\cdot j_s}}\} \cup \{(X_j, u\, X_r\, v) \in \Delta_0 \mid X_j, X_r \in \Xi_{\overset{\frown}{j_1 \cdot\cdot j_s}}\}$
  $G^\sharp \leftarrow \langle \Xi, \mathcal{A}, \Delta^\sharp \rangle$
  $\Gamma^\sharp \leftarrow$ ConstantBoundedControlSet$(G^\sharp, a_{j_1}^*\ a_{j_s}^*, k+1)$
  $\mathcal{S}_{\tilde{\mathbf{b}}} \leftarrow \varnothing$
  for $(X_i, aybz) \in \Delta_0$ **such that** $X_i \in \Xi_{\overset{\frown}{j_1 \cdot\cdot j_s}}$, $a, b \in \mathcal{A} \cup \{\varepsilon\}$ and $y, z \in \Xi_{\overset{\smile}{j_1 \cdot\cdot j_s}} \cup \{\varepsilon\}$ **do**
    if $L_{X_0, X_i}(G^\sharp) \subseteq a_{j_1}^*\ a_{j_s}^*$ **then**
      $\Delta_{i,aybz} \leftarrow \{(X_j, v) \in \Delta \mid X_j \in \Xi_{\overset{\smile}{j_1 \cdot\cdot j_s}}\} \cup \{(X_i, a\, y\, b\, z)\}$
      $G_{i,aybz} \leftarrow \langle \Xi, \mathcal{A}, \Delta_{i,aybz} \rangle$
      if $y \in \Xi$ **then**
        $\mathcal{S}' \leftarrow$ LetterBoundedControlSet$(G_{i,aybz}, y, a_{j_1}^* \cdots a_{j_s}^*, k)$
      else $\mathcal{S}' \leftarrow \varnothing$ ▷ $y = \varepsilon$ in this case
      if $z \in \Xi$ **then**
        $\mathcal{S}'' \leftarrow$ LetterBoundedControlSet$(G_{i,aybz}, z, a_{j_1}^* \cdots a_{j_s}^*, k)$
      else $\mathcal{S}'' \leftarrow \varnothing$ ▷ $z = \varepsilon$ in this case
      $\mathcal{S}_{\tilde{\mathbf{b}}} \leftarrow \mathcal{S} \cup \bigcup_{\Gamma \in \mathcal{S}' \cup \mathcal{S}''} \Gamma^\sharp \cdot (X_i, a\, y\, b\, z)^* \cdot \Gamma$
  return $\mathcal{S}_{\tilde{\mathbf{b}}}$

function minimizeExpression$(G, X, a_{i_1}^* \dots a_{i_d}^*)$
  $\mathtt{expr} \leftarrow \varepsilon$
  for $\ell = 1, \dots, d$ **do**
    if $L_X(G) \cap (\mathcal{A}^* \cdot a_{i_\ell} \cdot \mathcal{A}^*) \neq \varnothing$ **then**
      $\mathtt{expr} \leftarrow \mathtt{expr} \cdot a_{i_\ell}^*$
  return $\mathtt{expr}$

function partitionNonterminals$(G, a_{j_1}^*\ a_{j_s}^*)$
  match G **with** $\langle \Xi, \mathcal{A}, \Delta \rangle$
  $\mathtt{vars} \leftarrow \varnothing$
  for $Y \in \Xi$ **do**
    if $L_Y(G) \cap a_{j_1}\, \mathcal{A}^* \neq \varnothing \wedge L_Y(G) \cap \mathcal{A}^*\, a_{j_s} \neq \varnothing$ **then**
      $\mathtt{vars} \leftarrow \mathtt{vars} \cup \{Y\}$
  return $(\mathtt{vars}, \Xi \setminus \mathtt{vars})$

Lemma 4. *For every $k > 0$ there exists a grammar $G = \langle \Xi, \Sigma, \Delta \rangle$ and $X \in \Xi$ such that $|G| = \mathcal{O}(k)$ and every bounded expression Γ, such that $L_X(G) = \hat{L}_X(\Gamma \cap \Gamma_X^{\mathbf{df}(k+1)}, G)$ has length $|\Gamma| \geqslant 2^{k-1}$.*

6 Related Work

The programs we have studied feature unbounded control (the call stack) and unbounded data (the integer variables). The decidability and complexity of the reachability problem for such programs pose challenging research questions. A long standing and still open one is the decidability of the reachability problem for programs where variables behave like Petri net counters and control paths are taken in a context-free language. A lower bound exists [17] but decidability remains open. Atig and Ganty [3] showed decidability when the context-free language is of bounded index. The complexity of reachability was settled for branching VASS by Lazic and Schmitz [18]. When variables updates/guards are given by gap-order constraints, reachability is decidable [1,25]. It is in PSPACE when the set of control paths is regular [8]. More general updates and guard (like octagons) immediately leads to undecidability. This explains the restriction to bounded control sets. Demri *et al.* [9] studied the case of updates/guards of the form $\sum_{i=1}^{n} a_i \cdot x_i + b \leqslant 0 \wedge \mathbf{x}' = \mathbf{x} + c$. They show that LTL is NP-complete on for bounded regular control sets, hence reachability is in NP. Godoy and Tiwari [13] studied the invariant checking problem for a class of procedural programs where all executions conform to a bounded expression, among other restrictions.

References

1. P. A. Abdulla, M. F. Atig, G. Delzanno, and A. Podelski. Push-down automata with gap-order constraints. In *FSEN '13*, volume 8161 of *LNCS*, page 199–216. Springer, 2013.
2. R. Alur and P. Madhusudan. Adding nesting structure to words. *J. ACM*, 56(3):16:1–16:43, 2009.
3. M. F. Atig and P. Ganty. Approximating petri net reachability along context-free traces. In *FSTTCS '11*, volume 13 of *LIPIcs*, pages 152–163. Schloss Dagstuhl, 2011.
4. S. Bardin, A. Finkel, J. Leroux, and L. Petrucci. FAST: Fast acceleration of symbolic transition systems. In *CAV '03*, volume 2725 of *LNCS*, pages 118–121. Springer, 2003.
5. A. Bouajjani, J. Esparza, A. Finkel, O. Maler, P. Rossmanith, B. Willems, and P. Wolper. An efficient automata approach to some problems on context-free grammars. *Information Processing Letters*, 74:221–227, 2000.
6. M. Bozga, R. Iosif, and F. Konečný. Fast acceleration of ultimately periodic relations. In *CAV '10*, volume 6174 of *LNCS*, pages 227–242. Springer, 2010.
7. M. Bozga, R. Iosif, and F. Konečný. Safety problems are np-complete for flat integer programs with octagonal loops. In *VMCAI '14*, volume 8318 of *LNCS*, 2014.
8. L. Bozzelli and S. Pinchinat. Verification of gap-order constraint abstractions of counter systems. *Theoretical Computer Science*, 523:1–36, 2014.

9. S. Demri, A. K. Dhar, and A. Sangnier. Taming past ltl and flat counter systems. In *IJCAR '12*, volume 7364 of *LNCS*, page 179–193. Springer, 2012.
10. J. Esparza and P. Ganty. Complexity of pattern-based verification for multithreaded programs. In *POPL '11*, pages 499–510. ACM Press, 2011.
11. P. Ganty, R. Iosif, and F. Konečný. Underapproximation of procedure summaries for integer programs. In *TACAS '13*, volume 7795 of *LNCS*, pages 247–261. Springer, 2013.
12. S. Ginsburg. *The Mathematical Theory of Context-Free Languages*. McGraw-Hill, Inc., New York, NY, USA, 1966.
13. G. Godoy and A. Tiwari. Invariant checking for programs with procedure calls. In *SAS '09*, volume 5673 of *LNCS*, pages 326–342. Springer, 2009.
14. H. Hojjat, R. Iosif, F. Konečný, V. Kuncak, and P. Rümmer. Accelerating interpolants. In *ATVA '12*, 2012.
15. D. Kroening, M. Lewis, and G. Weissenbacher. Under-approximating loops in C programs for fast counterexample detection. In *CAV '13*, LNCS, pages 381–396. Springer, 2013.
16. M. Lange and H. Leiß. To CNF or not to CNF? An efficient yet presentable version of the CYK algorithm. *Informatica Didactica*, 8, 2008-2010.
17. R. Lazic. The reachability problem for vector addition systems with a stack is not elementary. RP '12, 2012.
18. R. Lazic and S. Schmitz. Non-elementary complexities for branching VASS, MELL, and extensions. In *CSL-LICS '14*. ACM, 2014.
19. A. W. Lin. *Model Checking Infinite-State Systems: Generic and Specific Approaches*. PhD thesis, School of Informatics, University of Edinburgh, August 2010.
20. M. Luker. A family of languages having only finite-index grammars. *Inf. and Control*, 39(1):14–18, 1978.
21. M. Luker. Control sets on grammars using depth-first derivations. *Math. Systems Theory*, 13:349–359, 1980.
22. M. Luttenberger and M. Schlund. An extension of parikh's theorem beyond idempotence. *CoRR*, abs/1112.2864, 2011.
23. A. Miné. The octagon abstract domain. *Higher-Order and Symbolic Computation*, 19(1):31–100, 2006.
24. M. Minsky. *Computation: Finite and Infinite Machines*. Prentice-Hall, 1967.
25. P. Z. Revesz. A closed-form evaluation for datalog queries with integer (gap)-order constraints. *Theoretical Computer Science*, 116(1):117 – 149, 1993.
26. A. Schrijver. *Theory of Linear and Integer Programming*. Wiley, 1998.

Appendix

The appendix is divided in two parts. Appendix A contains easy results about context-free languages and have been included for the sake of being self-contained. They are variations of classical constructions so as to take into account index and depth-first policy. To keep proofs concise, we assume that the grammars are in 2-normal form (2NF for short). A grammar is in 2NF if all its productions (X, w) satisfy $|w| \leqslant 2$. Any grammar G can be converted into an equivalent 2NF grammar H, such that $|H| = O(|G|)$, in time $O(|G|^2)$ [16]. Note that 2NF is a special case of the general form we assumed where each production (X, w) is such that w contains at most 2 terminals and 2 nonterminals. Appendix B contains the rest of the proofs about the combinatorial properties of derivations.

A From Bounded to Letter-bounded Languages

It is well-known that the intersection between a context-free and a regular language is context-free. Below we define the grammar that generates the intersection between the language of a given grammar $G = \langle \Xi, \Sigma, \Delta \rangle$ and a regular language given by a bounded expression $\mathbf{b} = w_1^* \ldots w_d^*$ over Σ where ℓ_i denotes the length of each w_i. Let $G^{\mathbf{b}} = \langle \Xi^{\mathbf{b}}, \Sigma, \Delta^{\mathbf{b}} \rangle$ be the grammar generating the regular language of $\mathbf{b}$, where:

$$\begin{aligned}
\Xi^{\mathbf{b}} &= \left\{ \mathrm{Q}_r^{(s)} \mid 1 \leqslant s \leqslant d \ \wedge \ 1 \leqslant r \leqslant \ell_s \right\} \\
\Delta^{\mathbf{b}} &= \left\{ \mathrm{Q}_i^{(s)} \to (w_s)_i \, \mathrm{Q}_{i+1}^{(s)} \mid 1 \leqslant s \leqslant d \ \wedge \ 1 \leqslant i < \ell_s \right\} \cup \\
&\quad \left\{ \mathrm{Q}_{\ell_s}^{(s)} \to (w_s)_{\ell_s} \, \mathrm{Q}_1^{(s')} \mid 1 \leqslant s \leqslant s' \leqslant d \right\} \cup \\
&\quad \left\{ \mathrm{Q}_1^{(s)} \to \varepsilon \mid 1 \leqslant s \leqslant d \right\} \ .
\end{aligned}$$

It is routine to check that $\{w \mid \mathrm{Q}_1^{(i)} \Rightarrow^* w \text{ for some } 1 \leqslant i \leqslant d\} = \mathbf{b}$. Moreover, notice that the number of nonterminals in $G^{\mathbf{b}}$ equals the size of $\mathbf{b}$, i.e. $\|\Xi^{\mathbf{b}}\| = |\mathbf{b}|$.

Remark 1. Note that when $\mathbf{b}$ is letter-bounded ($\mathbf{b} = a_1^* \ldots a_d^*$), the grammar $G_1^{\mathbf{b}} = (\Xi_1^{\mathbf{b}}, \Sigma, \Delta_1^{\mathbf{b}})$ generating is given by:

$$\begin{aligned}
\Xi_1^{\mathbf{b}} &= \left\{ \mathrm{Q}^{(s)} \mid 1 \leqslant s \leqslant d \right\} \cup \{\mathrm{Q}_{sink}\} \\
\Delta_1^{\mathbf{b}} &= \left\{ \mathrm{Q}^{(s)} \to a_{s'} \, \mathrm{Q}^{(s')} \mid 1 \leqslant s \leqslant s' \leqslant d \right\} \cup \\
&\quad \left\{ \mathrm{Q}^{(s)} \to b \, \mathrm{Q}_{sink} \mid b \in \Sigma \backslash \{a_s, a_{s+1}, \ldots, a_d\} \right\} \cup \\
&\quad \left\{ \mathrm{Q}^{(s)} \to \varepsilon \mid 1 \leqslant s \leqslant d \right\} \cup \\
&\quad \{ \mathrm{Q}_{sink} \to b \, \mathrm{Q}_{sink} \mid b \in \Sigma \}
\end{aligned}$$

is such that $L_{\mathrm{Q}^{(1)}}(G_1^{\mathbf{b}}) = \mathbf{b}$. Furthermore, $G_1^{\mathbf{b}}$ is complete—all terminals can be produced from all nonterminals—and it is deterministic when $\mathbf{b}$ is strict. Then a grammar $\overline{G^{\mathbf{b}}_1}$, such that $L_{\mathrm{Q}^{(1)}}(\overline{G^{\mathbf{b}}_1}) = \Sigma^*\backslash\mathbf{b}$, can be computed in time $\mathcal{O}(|G^{\mathbf{b}}_1|)$, by replacing each production $\mathrm{Q}^{(s)} \to \varepsilon$, $1 \leqslant s \leqslant d$, with $\mathrm{Q}_{sink} \to \varepsilon$.

Given $G^{\mathbf{b}}$, and a grammar $G = (\Xi, \Sigma, \Delta)$ in 2NF and $X \in \Xi$, our goal is to define a grammar $G^{\cap} = \langle \Xi^{\cap}, \Sigma, \Delta^{\cap}\rangle$ that produces the language $L_X(G) \cap L(\mathbf{b})$, for some $X \in \Xi$. The definition of $G^{\cap} = \langle \Xi^{\cap}, \Sigma, \Delta^{\cap}\rangle$ follows:

- $\Xi^{\cap} = \left\{[\mathrm{Q}^{(r)}_s X \mathrm{Q}^{(u)}_v] \mid X \in \Xi \wedge \mathrm{Q}^{(r)}_s \in \Xi^{\mathbf{b}} \wedge \mathrm{Q}^{(u)}_v \in \Xi^{\mathbf{b}} \wedge r \leqslant u\right\}$
- $\Delta^{\cap}$ is defined as follows:
  - for every production $X \to w \in \Delta$ where $w \in \Sigma^*$, $\Delta^{\cap}$ has a production
  $$[\mathrm{Q}^{(r)}_s X \mathrm{Q}^{(u)}_v] \to w \qquad\qquad \text{if } \mathrm{Q}^{(r)}_s \Rightarrow^* w\, \mathrm{Q}^{(u)}_v \ ; \tag{1}$$
  - for every production $X \to Y \in \Delta$, where $Y \in \Xi$, $\Delta^{\cap}$ has a production
  $$[\mathrm{Q}^{(r)}_s X \mathrm{Q}^{(u)}_v] \to [\mathrm{Q}^{(r)}_s Y \mathrm{Q}^{(u)}_v] \ ; \tag{2}$$
  - for every production $X \to a\,Y \in \Delta$, where $a \in \Sigma$ and $Y \in \Xi$, $\Delta^{\cap}$ has a production
  $$[\mathrm{Q}^{(r)}_s X \mathrm{Q}^{(u)}_v] \to a\,[\mathrm{Q}^{(x)}_y Y \mathrm{Q}^{(u)}_v] \qquad \text{if } \mathrm{Q}^{(r)}_s \to a\, \mathrm{Q}^{(x)}_y \in \Delta^{\mathbf{b}} \ ; \tag{3}$$
  - for every production $X \to Y\,a \in \Delta$, where $Y \in \Xi$ and $a \in \Sigma$, $\Delta^{\cap}$ has a production
  $$[\mathrm{Q}^{(r)}_s X \mathrm{Q}^{(u)}_v] \to [\mathrm{Q}^{(r)}_s Y \mathrm{Q}^{(x)}_y]\,a \qquad \text{if } \mathrm{Q}^{(x)}_y \to a\, \mathrm{Q}^{(u)}_v \in \Delta^{\mathbf{b}} \ ; \tag{4}$$
  - for every production $X \to Y\,Z \in \Delta$, $\Delta^{\cap}$ has a production
  $$[\mathrm{Q}^{(r)}_s X \mathrm{Q}^{(u)}_v] \to [\mathrm{Q}^{(r)}_s Y \mathrm{Q}^{(x)}_y]\,[\mathrm{Q}^{(x)}_y Z \mathrm{Q}^{(u)}_v] \ ; \tag{5}$$
  - $\Delta^{\cap}$ has no other production.

Let $\zeta \colon \Xi^{\cap} \to \Xi$ be the function that "strips" every nonterminal $[\mathrm{Q}^{(s)}_r X \mathrm{Q}^{(u)}_v] \in \Xi^{\cap}$ of the nonterminals from $\Xi^{\mathbf{b}}$, i.e. $\zeta([\mathrm{Q}^{(s)}_r X \mathrm{Q}^{(u)}_v]) = X$. In the following, we abuse notation and extend the ζ function to symbols from $\Sigma \cup \Xi^{\cap}$, by defining $\zeta(a) = a$, for each $a \in \Sigma$, and further to words $w \in (\Sigma \cup \Xi^{\cap})^*$ as $\zeta(w) = \zeta((w)_1) \cdots \zeta((w)_{|w|})$. Finally, for a production $p = (X, w) \in \Delta^{\cap}$, we define $\zeta(p) = (\zeta(X), \zeta(w))$, and for a control word $\gamma \in (\Delta^{\cap})^*$, we write $\zeta(\gamma)$ for $\zeta((\gamma)_1) \cdots \zeta((\gamma)_{|\gamma|})$.

Lemma 5. *Given a grammar $G = \langle \Xi, \Sigma, \Delta\rangle$ and a grammar $G^{\mathbf{b}} = \langle \Xi^{\mathbf{b}}, \Sigma, \Delta^{\mathbf{b}}\rangle$ generating $\mathbf{b}$, for every $X \in \Xi$, $\mathrm{Q}^{(r)}_s, \mathrm{Q}^{(u)}_v \in \Xi^{\mathbf{b}}$, $w \in \Sigma^*$, and every $k > 0$, we have:*

(i) for every $\gamma \in (\Delta^{\cap})^*$, $[\mathrm{Q}^{(r)}_s X \mathrm{Q}^{(u)}_v] \xRightarrow[\mathbf{df}(k)]{\gamma}{}^{*}\, w$ only if $X \xRightarrow[\mathbf{df}(k)]{\zeta(\gamma)} w$ and $\mathrm{Q}^{(r)}_s \Rightarrow^*_{G^{\mathbf{b}}} w\, \mathrm{Q}^{(u)}_v$

(ii) for every $\delta \in \Delta^*$, $X \xRightarrow[\mathbf{df}(k)]{\delta} w$ and $\mathrm{Q}^{(r)}_s \Rightarrow^*_{G^{\mathbf{b}}} w\, \mathrm{Q}^{(u)}_v$ only if $[\mathrm{Q}^{(r)}_s X \mathrm{Q}^{(u)}_v] \xRightarrow[\mathbf{df}(k)]{\gamma}{}^{*}\, w$, for some $\gamma \in \zeta^{-1}(\delta)$.

Consequently, we have $\bigcup_{1\leqslant s\leqslant x\leqslant d} L_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^\cap) = L_X(G) \cap \mathbf{b}$.

Proof. (i) By induction on $|\gamma| > 0$. For the base case $|\gamma| = 1$—γ is the production $([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to w) \in \Delta^\cap$ with $w \in \Sigma^*$—by case (1) of the definition of $\Delta^\cap$, we have $\mathrm{Q}_s^{(r)} \Rightarrow^*_{G^{\mathbf{b}}} w\,\mathrm{Q}_v^{(u)}$ and there exists a production $X \to w \in \Delta$. Since, moreover, $\zeta([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to w) = (X \to w)$, we have that $X \xRightarrow[\mathbf{df}(1)]{\zeta(\gamma)} w$ in G.

For the induction step $|\gamma| > 1$, we have $\gamma = ([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to \tau) \cdot \gamma'$, for some production $[\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to \tau \in \Delta^\cap$, and a word $\tau \in (\Sigma \cup \Xi^\cap)^*$ of length $|\tau| \leqslant 2$. We distinguish four cases, based on the structure of τ:

1. if $\tau = [\mathrm{Q}_s^{(r)} Y \mathrm{Q}_v^{(u)}]$ then $\tau \xRightarrow[\mathbf{df}(k)]{\gamma'} w$ is a derivation of $G^\cap$. By the induction hypothesis, we obtain that $\mathrm{Q}_s^{(r)} \Rightarrow^*_{G^{\mathbf{b}}} w\,\mathrm{Q}_v^{(u)}$ and $Y \xRightarrow[\mathbf{df}(k)]{\zeta(\gamma')} w$ is a derivation of G. But $X \to Y \in \Delta$—case (2) of the definition of $\Delta^\cap$—hence $\zeta(\gamma) = (X \to Y) \cdot \zeta(\gamma')$ and $X \xRightarrow[\mathbf{df}(k)]{\zeta(\gamma)} w$ is a derivation of G.
2. if $\tau = a\,[\mathrm{Q}_y^{(x)} Y \mathrm{Q}_v^{(u)}]$ then $w = a \cdot w'$ and $G^\cap$ has derivation $[\mathrm{Q}_y^{(x)} Y \mathrm{Q}_v^{(u)}] \xRightarrow[\mathbf{df}(k)]{\gamma'} w'$. By the induction hypothesis, we obtain $\mathrm{Q}_y^{(x)} \Rightarrow^*_{G^{\mathbf{b}}} w'\mathrm{Q}_v^{(u)}$ and G has a derivation $Y \xRightarrow[\mathbf{df}(k)]{\zeta(\gamma')} w'$. By the case (3) of the definition of $\Delta^\cap$, we have $\mathrm{Q}_s^{(r)} \to a\,\mathrm{Q}_y^{(x)} \in \Delta^{\mathbf{b}}$ and $\zeta([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to \tau) = (X \to aY) \in \Delta$. Thus $\mathrm{Q}_s^{(r)} \Rightarrow^*_{G^{\mathbf{b}}} w\,\mathrm{Q}_v^{(u)}$ and $X \xRightarrow[\mathbf{df}(k)]{\zeta(\gamma)} w$, since $\zeta(\gamma) = (X \to aY) \cdot \zeta(\gamma')$.
3. the case $\tau = [\mathrm{Q}_s^{(r)} Y \mathrm{Q}_y^{(x)}]\,a$ is symmetric, using the case (4) of the definition of $\Delta^\cap$.
4. if $\tau = [\mathrm{Q}_s^{(r)} Y \mathrm{Q}_y^{(x)}]\,[\mathrm{Q}_y^{(x)} Z \mathrm{Q}_v^{(u)}]$ then, by Lemma 7, there exist words $w_1, w_2 \in \Sigma^*$ such that $w = w_1 w_2$ and either one of the following applies:
   (a) $[\mathrm{Q}_s^{(r)} Y \mathrm{Q}_y^{(x)}] \xRightarrow[\mathbf{df}(k-1)]{\gamma_1} w_1$, $[\mathrm{Q}_y^{(x)} Z \mathrm{Q}_v^{(u)}] \xRightarrow[\mathbf{df}(k)]{\gamma_2} w_2$ and $\gamma' = \gamma_1\,\gamma_2$, or
   (b) $[\mathrm{Q}_s^{(r)} Y \mathrm{Q}_y^{(x)}] \xRightarrow[\mathbf{df}(k)]{\gamma_1} w_1$, $[\mathrm{Q}_y^{(x)} Z \mathrm{Q}_v^{(u)}] \xRightarrow[\mathbf{df}(k-1)]{\gamma_2} w_2$ and $\gamma' = \gamma_2\,\gamma_1$.

   We consider the first case only, the second being symmetric. Since $|\gamma_1| < |\gamma|$ and $|\gamma_2| < |\gamma|$, we apply the induction hypothesis and find out that $\mathrm{Q}_s^{(r)} \Rightarrow^*_{G^{\mathbf{b}}} w_1\,\mathrm{Q}_y^{(x)}$, $\mathrm{Q}_y^{(x)} \Rightarrow^*_{G^{\mathbf{b}}} w_2\,\mathrm{Q}_v^{(u)}$, and G has derivations $Y \xRightarrow[\mathbf{df}(k-1)]{\zeta(\gamma_1)} w_1$ and $Z \xRightarrow[\mathbf{df}(k)]{\zeta(\gamma_2)} w_2$. Then $\mathrm{Q}_s^{(r)} \Rightarrow^*_{G^{\mathbf{b}}} w_1\,w_2\,\mathrm{Q}_v^{(u)}$ where $w_1\,w_2 = w$. By case (5) of the definition of $\Delta^\cap$, Δ has a production $(X \to YZ) = \zeta([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to \tau)$. Since $\gamma' = \gamma_1\,\gamma_2$, then $\zeta(\gamma) = (X \to YZ) \cdot \zeta(\gamma_1)\,\zeta(\gamma_2)$, and G has a k-index depth-first derivation $X \xRightarrow[\mathbf{df}(k)]{\zeta(\gamma)} w$.

(ii) By induction on $|\delta| > 0$. For the base case $|\delta| = 1$, we have $\delta = (X \to w) \in \Delta$. By the case (1) from the definition of $\Delta^\cap$, $G^\cap$ has a rule $[\mathrm{Q}_r^{(s)} X \mathrm{Q}_v^{(u)}] \to w$ and, since, moreover, $\zeta([\mathrm{Q}_r^{(s)} X \mathrm{Q}_v^{(u)}] \to w) = \delta$, we have $\gamma = ([\mathrm{Q}_r^{(s)} X \mathrm{Q}_v^{(u)}] \to w)$.

For the induction step $|\delta| > 1$, we have $\delta = (X \to \tau) \cdot \delta'$. We distinguish four cases, based on the structure of τ:

1. if $\tau = Y$, for some $Y \in \Xi$, by the induction hypothesis, $G^\cap$ has a derivation $[\mathrm{Q}_r^{(s)} Y \mathrm{Q}_v^{(u)}] \xRightarrow[\mathbf{df}(k)]{\gamma'} w$, for some $\gamma' \in \zeta^{-1}(\delta')$. Since $\mathrm{Q}_s^{(r)} \Rightarrow^*_{G^\mathbf{b}} w\,\mathrm{Q}_v^{(u)}$ —by case (2) of the definition of $\Delta^\cap$— $G^\cap$ has a production $p = ([\mathrm{Q}_r^{(s)} X \mathrm{Q}_v^{(u)}] \to [\mathrm{Q}_r^{(s)} Y \mathrm{Q}_v^{(u)}])$. We define $\gamma = p \cdot \gamma'$. It is immediate to check that $\zeta(\gamma) = \delta$.
2. if $\tau = a\,Y$, for some $a \in \Sigma$ and $Y \in \Xi$, then $w = a \cdot w'$. Hence $\mathrm{Q}_s^{(r)} \Rightarrow_{G^\mathbf{b}} a\,\mathrm{Q}_y^{(x)}$, $\mathrm{Q}_y^{(x)} \Rightarrow^*_{G^\mathbf{b}} w'\,\mathrm{Q}_v^{(u)}$ and G has a derivation $Y \xRightarrow[\mathbf{df}(k)]{\delta'} w'$. By the induction hypothesis, $G^\cap$ has a derivation $[\mathrm{Q}_y^{(x)} Y \mathrm{Q}_v^{(u)}] \xRightarrow[\mathbf{df}(k)]{\gamma'} w'$, for some $\gamma' \in \zeta^{-1}(\delta')$. By the case (3) of the definition of $\Delta^\cap$, there exists a production $p = ([\mathrm{Q}_r^{(s)} X \mathrm{Q}_v^{(u)}] \to aY) \in \Delta^\cap$. We define $\gamma = p \cdot \gamma'$. It is immediate to check that $\zeta(\gamma) = \delta$, hence $[\mathrm{Q}_r^{(s)} X \mathrm{Q}_v^{(u)}] \xRightarrow[\mathbf{df}(k)]{\gamma} w$.
3. the case $\tau = Y\,a$, for some $Y \in \Xi$ and $a \in \Sigma$, is symmetrical.
4. if $\tau = Y\,Z$, for some $Y, Z \in \Xi$, then, by Lemma 7, there exist words $w_1, w_2 \in \Sigma^*$ such that $w = w_1\,w_2$ and either one of the following cases applies:
   (a) $Y \xRightarrow[\mathbf{df}(k-1)]{\delta_1} w_1$, $Z \xRightarrow[\mathbf{df}(k)]{\delta_2} w_2$ and $\delta' = \delta_1\,\delta_2$, or
   (b) $Y \xRightarrow[\mathbf{df}(k)]{\delta_1} w_1$, $Z \xRightarrow[\mathbf{df}(k-1)]{\delta_2} w_2$ and $\delta' = \delta_2\,\delta_1$.

   Moreover, we have $\mathrm{Q}_s^{(r)} \Rightarrow^*_{G^\mathbf{b}} w_1\,\mathrm{Q}_y^{(x)}$ and $\mathrm{Q}_y^{(x)} \Rightarrow^*_{G^\mathbf{b}} w_2\,\mathrm{Q}_v^{(u)}$, for some $\mathrm{Q}_y^{(x)} \in \Xi^\mathbf{b}$. We consider the first case only, the second being symmetric. Since $|\delta_1| < |\delta|$ and $|\delta_2| < |\delta|$ we apply the induction hypothesis and find two control words $\gamma_1 \in \zeta^{-1}(\delta_1)$ and $\gamma_2 \in \zeta^{-1}(\delta_2)$ such that $G^\cap$ has derivations $[\mathrm{Q}_s^{(r)} Y \mathrm{Q}_y^{(x)}] \xRightarrow[\mathbf{df}(k-1)]{\gamma_1} w_1$ and $[\mathrm{Q}_y^{(x)} Z \mathrm{Q}_v^{(u)}] \xRightarrow[\mathbf{df}(k)]{\gamma_2} w_2$. By case (5) of the definition of $\Delta^\cap$, $G^\cap$ has a production $p = ([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to [\mathrm{Q}_s^{(r)} Y \mathrm{Q}_y^{(x)}][\mathrm{Q}_y^{(x)} Z \mathrm{Q}_v^{(u)}])$. Since $\delta' = \delta_1\,\delta_2$, we define $\gamma = p\,\gamma_1\,\gamma_2$. It is immediate to check that $\zeta(\gamma) = \delta$ and $[\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \xRightarrow[\mathbf{df}(k)]{\gamma} w$. □

In the rest of this section, for a given bounded expression $\mathbf{b} = w_1^* \ldots w_d^*$ over Σ, we associate the strict d-letter-bounded expression $\widetilde{\mathbf{b}} = a_1^* \ldots a_d^*$ over an alphabet $\mathcal{A}$, disjoint from Σ, i.e. $\mathcal{A} \cap \Sigma = \varnothing$, and a homomorphism $h\colon \mathcal{A} \to \Sigma^*$ mapping as follows: $a_i \mapsto w_i$, for all $1 \leqslant i \leqslant d$. The next step is to define a grammar $G^\bowtie = \langle \Xi^\bowtie, \mathcal{A}, \Delta^\bowtie \rangle$, such that $\Xi^\bowtie = \Xi^\cap$ and, for all $X \in \Xi, 1 \leqslant s \leqslant x \leqslant d$:

$$h^{-1}(L_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^\cap)) \cap \widetilde{\mathbf{b}} = L_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^\bowtie)\ .$$

The grammar $G^\bowtie$ is defined from $G^\cap$, by the following modification of the productions from $\Delta^\cap$, defined by a function $\iota\colon \Delta^\cap \mapsto \Delta^\bowtie$:

- $\iota([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to w) = [\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to z$ where
  1. if $|w| = 0$ then $z = \varepsilon$.

2. if $|w| = 1$ then we have $\mathrm{Q}_s^{(r)} \Rightarrow_{G^\mathbf{b}} w\,\mathrm{Q}_v^{(u)}$ and we let $z = a_r$ if $v = 1$ else $z = \varepsilon$.
3. if $|w| = 2$ then we have $\mathrm{Q}_s^{(r)} \Rightarrow_{G^\mathbf{b}} (w)_1\,\mathrm{Q}_x^{(y)} \Rightarrow_{G^\mathbf{b}} (w)_1(w)_2\,\mathrm{Q}_v^{(u)}$ for some x, y. Define the word $z = z' \cdot z''$ of length at most 2 such that $z' = a_r$ if $x = 1$; else $z' = \varepsilon$ and $z'' = a_y$ if $v = 1$ else $z'' = \varepsilon$.

– $\iota([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to b\,[\mathrm{Q}_y^{(x)} Y \mathrm{Q}_v^{(u)}]) = [\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to c\,[\mathrm{Q}_y^{(x)} Y \mathrm{Q}_v^{(u)}]$ where $c = a_r$ if $y = 1$; else $c = \varepsilon$.
– $\iota([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to [\mathrm{Q}_s^{(r)} Y \mathrm{Q}_y^{(x)}]\,b) = [\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to [\mathrm{Q}_s^{(r)} Y \mathrm{Q}_y^{(x)}]\,c$ where $c = a_x$ if $v = 1$; else $c = \varepsilon$.
– $\iota(p) = p$ otherwise.

Let $\Delta^{\bowtie} = \{\iota(p) \mid p \in \Delta^{\cap}\}$. In addition, for every control word $\gamma \in (\Delta^{\cap})^*$ of length n, let $\iota(\gamma) = \iota((\gamma)_1) \cdots \iota((\gamma)_n) \in \Delta^{\bowtie}$. A consequence of the following proposition is that the inverse relation $\iota^{-1} \subseteq \Delta^{\bowtie} \times \Delta^{\cap}$ is a total function.

Proposition 1. *For each production $p \in \Delta^{\bowtie}$, the set $\iota^{-1}(p)$ is a singleton.*

Proof. By case split, based on the type of the production $p \in \Delta^{\bowtie}$. Since $G^{\bowtie}$ is in 2NF we have:

– if $p = ([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to a)$ then $\iota^{-1}(p) = \{[\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to w\}$, where $\mathrm{Q}_s^{(r)} \Rightarrow^*_{G^\mathbf{b}} w\,\mathrm{Q}_v^{(u)}$ is the shortest step sequence of $G^\mathbf{b}$ between $\mathrm{Q}_s^{(r)}$ and $\mathrm{Q}_v^{(u)}$ which is unique by $G^\mathbf{b}$ and produces $w \in \Sigma^*$.
– if $p = ([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to [\mathrm{Q}_y^{(x)} Y \mathrm{Q}_t^{(z)}])$, then either one of the cases below must hold:
  - *(i)* $\mathrm{Q}_u^{(v)} = \mathrm{Q}_z^{(t)}$ and $\mathrm{Q}_s^{(r)} \Rightarrow_{G^\mathbf{b}} b\,\mathrm{Q}_y^{(x)}$, for some $y \neq 1$. In this case b is uniquely determined by $\mathrm{Q}_s^{(r)}$ and $\mathrm{Q}_y^{(x)}$, thus we get $\iota^{-1}(p) = \{[\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to b\,[\mathrm{Q}_y^{(x)} Y \mathrm{Q}_t^{(z)}]\}$.
  - *(ii)* $\mathrm{Q}_s^{(r)} = \mathrm{Q}_y^{(x)}$ and $\mathrm{Q}_t^{(z)} \Rightarrow_{G^\mathbf{b}} b\,\mathrm{Q}_v^{(u)}$, for some $t \neq \ell_z$. In this case we get, symmetrically, $\iota^{-1}(p) = \{[\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to [\mathrm{Q}_y^{(x)} Y \mathrm{Q}_t^{(z)}]\,b\}$.
  - *(iii)* $\mathrm{Q}_u^{(v)} = \mathrm{Q}_z^{(t)}$ and $\mathrm{Q}_s^{(r)} = \mathrm{Q}_y^{(x)}$. Then $\iota^{-1}(p) = \{p\}$.
– if $p = ([\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to a_r\,[\mathrm{Q}_y^{(x)} Y \mathrm{Q}_v^{(u)}])$ for some $a_r \in \mathcal{A}$, hence $y = 1$ (respectively, $[\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to [\mathrm{Q}_s^{(r)} Y \mathrm{Q}_y^{(x)}]\,a_r$ hence $v = 1$) and then the only possibility is $\iota^{-1}(p) = \{[\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to (w_r)_{\ell_r}\,[\mathrm{Q}_y^{(x)} Y \mathrm{Q}_v^{(u)}]\}$ (respectively, $[\mathrm{Q}_s^{(r)} X \mathrm{Q}_v^{(u)}] \to [\mathrm{Q}_s^{(r)} Y \mathrm{Q}_y^{(x)}]\,(w_r)_{\ell_r}$).
– if $p = ([\mathrm{Q}_s^{(r)} X \mathrm{Q}_y^{(x)}] \to [\mathrm{Q}_s^{(r)} Y \mathrm{Q}_v^{(u)}]\,[\mathrm{Q}_v^{(u)} Z \mathrm{Q}_y^{(x)}])$ then $\iota^{-1}(p) = \{p\}$. □

Lemma 6. *Given a grammar $G = \langle \Xi, \Sigma, \Delta \rangle$ and a bounded expression $\mathbf{b} = w_1^* \ldots w_d^*$ over Σ, for every $X \in \Xi$, every $1 \leqslant s \leqslant x \leqslant d$ and every $k > 0$, the following hold:*

1. $L^{(k)}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\bowtie}) = h^{-1}(L^{(k)}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\cap})) \cap \widetilde{\mathbf{b}}$,
2. *for each control set $\Gamma \subseteq (\Delta^{\bowtie})^*$, such that $L^{(k)}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\bowtie}) \subseteq \hat{L}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(\Gamma, G^{\bowtie})$, we have $L^{(k)}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\cap}) \subseteq \hat{L}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(\iota^{-1}(\Gamma), G^{\cap})$,*
3. *$G^{\bowtie}$ is computable in time $\mathcal{O}(|\mathbf{b}|^3 \cdot |G|)$.*

Proof. We start by proving the following facts:

Fact 1. *For all $X \in \Xi$ and $1 \leqslant s \leqslant x \leqslant d$, we have $L_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\bowtie}) \subseteq \widetilde{\mathbf{b}}$.*

Proof. Let $\tilde{w} \in L_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\bowtie})$. We have $[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}] \overset{\gamma}{\Rightarrow} \tilde{w}$ is a derivation of $G^{\bowtie}$ for some control word γ over $\Delta^{\bowtie}$. By contradiction, assume $\tilde{w} \notin \widetilde{\mathbf{b}}$, that is there exist p, p' such that $p < p'$ and $(\tilde{w})_p = a_j$ and $(\tilde{w})_{p'} = a_i$ with $i < j$. The definition of ι shows that there exists $w \in L_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\cap})$ such that $[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}] \xRightarrow{\iota^{-1}(\gamma)} w$ in $G^{\cap}$, hence that $w \in \mathbf{b}$ since $L_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\cap}) \subseteq \mathbf{b}$, and finally that $\mathrm{Q}_1^{(s)} \Rightarrow^*_{G^{\mathbf{b}}} w\, \mathrm{Q}_1^{(x)}$. Now, the mapping ι is defined such that a production in its image produces a a_r when, in the underlying $G^{\mathbf{b}}$, either control moves forward from $\mathrm{Q}_s^{(r)}$ to $\mathrm{Q}_1^{(u)}$, e.g. $[\mathrm{Q}_s^{(r)} X \mathrm{Q}_y^{(x)}] \to a_r\, [\mathrm{Q}_1^{(u)} Y \mathrm{Q}_y^{(x)}]$ or control moves backward form $\mathrm{Q}_1^{(u)}$ to $\mathrm{Q}_s^{(r)}$, e.g. $[\mathrm{Q}_y^{(x)} X \mathrm{Q}_1^{(u)}] \to [\mathrm{Q}_y^{(x)} Y \mathrm{Q}_s^{(r)}]\, a_r$. Therefore, by the previous assumption on $\tilde{w}$ where a_j occurs before a_i, we have that a production of $\mathrm{Q}_{\ell_j}^{(j)} \to (w_j)_{\ell_j}\, \mathrm{Q}_1^{(u)}$ for some $u \geqslant j$ and then a production of $\mathrm{Q}_{\ell_i}^{(i)} \to (w_i)_{\ell_i}\, \mathrm{Q}_1^{(u')}$ for some $u' \geqslant i$ necessarily occurs in that order in $\iota^{-1}(\gamma)$. But this is a contradiction because $j > i$ and the definition of $G^{\mathbf{b}}$ prohibits control to move from $\mathrm{Q}_{p_j}^{(j)}$ to $\mathrm{Q}_{p_i}^{(i)}$ for any p_i, p_j. □

Fact 2. *For all $X \in \Xi$, $1 \leqslant s \leqslant x \leqslant d$, $\gamma \in (\Delta^{\cap})^*$, $k > 0$ and $i_1, \ldots, i_d \in \mathbb{N}$:*

$$[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}] \underset{(k)}{\overset{\gamma}{\Longrightarrow}} w_1^{i_1} \ldots w_d^{i_d} \text{ in } G^{\cap} \text{ if and only if } [\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}] \underset{(k)}{\overset{\iota(\gamma)}{\Longrightarrow}} a_1^{i_1} \ldots a_d^{i_d} \text{ in } G^{\bowtie}\ .$$

Proof. By induction on $|\gamma| > 0$, and case analysis on the right-hand side of $(\gamma)_1$. □

(1) "$\subseteq$" Let $\tilde{w} \in L^{(k)}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\bowtie})$. By Fact 1, we have that $\tilde{w} \in \widetilde{\mathbf{b}}$. It remains to show that $\tilde{w} \in h^{-1}(L_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\cap}))$, i.e. that $h(\tilde{w}) \in L_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\cap})$, which follows by Fact 2. "$\supseteq$" Let $\tilde{w} \in h^{-1}(L^{(k)}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\cap})) \cap \widetilde{\mathbf{b}}$ be a word, hence $\tilde{w} = a_1^{i_1} \ldots a_d^{i_d}$ for some $i_1, \ldots, i_d \in \mathbb{N}$. Then $h(\tilde{w}) \in L^{(k)}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\cap})$ by Fact 2 and we are done.

(2) Let $w = w_1^{i_1} \ldots w_d^{i_d} \in L^{(k)}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\cap})$ be a word. Then $G^{\cap}$ has a derivation $[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}] \underset{(k)}{\Longrightarrow}^* w$. By Fact 2, also $G^{\bowtie}$ has a derivation $[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}] \underset{(k)}{\Longrightarrow}^* a_1^{i_1} \ldots a_d^{i_d}$. By the hypothesis $L^{(k)}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(G^{\bowtie}) \subseteq \hat{L}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(\Gamma, G^{\bowtie})$, there exists a control word $\gamma \in \Gamma$ such that $[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}] \overset{\gamma}{\Rightarrow} a_1^{i_1} \ldots a_d^{i_d}$ in $G^{\bowtie}$, and by Fact 2, we have $[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}] \xRightarrow{\iota^{-1}(\gamma)} w_1^{i_1} \ldots w_d^{i_d}$ in $G^{\cap}$. Hence $w \in \hat{L}_{[\mathrm{Q}_1^{(s)} X \mathrm{Q}_1^{(x)}]}(\iota^{-1}(\Gamma), G^{\cap})$.

(3) Given that each production $p^{\bowtie} \in \Delta^{\bowtie}$ is the image of a production $p^{\cap} \in \Delta^{\cap}$ via ι, we have $|p^{\bowtie}| = |\iota(p^{\cap})| \leqslant |p^{\cap}|$. Hence $|G^{\bowtie}| \leqslant |G^{\cap}|$. Now, each production

$p^\cap \in \Delta^\cap$ corresponds to a production p of G, such that the nonterminals occurring on both sides of p are decorated with at most 3 nonterminals from $\Xi^{\mathbf{b}}$. Since $\|\Xi^{\mathbf{b}}\| = |\mathbf{b}|$, we obtain that, for each production p of G, $G^\cap$ has at most $|\mathbf{b}|^3$ productions of size $|p|$. Hence $|G^{\bowtie}| \leqslant |G^\cap| \leqslant |\mathbf{b}|^3 \cdot |G|$, and $G^{\bowtie}$ can be constructed in time $|\mathbf{b}|^3 \cdot |G|$. □

Remark 2. Given $G = \langle \Xi, \mathcal{A}, \Delta \rangle$, $X \in \Xi$, and a strict d-letter-bounded expression $\widetilde{\mathbf{b}} = a_1^* \ldots a_d^*$, the check $L_X(G) \subseteq \widetilde{\mathbf{b}}$ can be decided in time $\mathcal{O}(|\widetilde{\mathbf{b}}| \cdot |G|)$, by building a grammar $\overline{G_1^{\mathbf{b}}}$ such that $L_{\mathrm{Q}^{(1)}}(\overline{G_1^{\mathbf{b}}}) = \Sigma^* \backslash \widetilde{\mathbf{b}}$ (see Remark 1) and checking $L_X(G) \cap L_{\mathrm{Q}^{(1)}}(\overline{G_1^{\mathbf{b}}}) \stackrel{?}{=} \emptyset$. A similar argument shows that queries $L_X(G) \cap (\mathcal{A}^* \cdot a_s \cdot \mathcal{A}^*) \stackrel{?}{=} \emptyset$, $1 \leqslant s \leqslant d$, can be answered in time $\mathcal{O}(|G|)$ [5, Section 5].

B Other proofs

Lemma 7. *Given $G = \langle \Xi, \Sigma, \Delta \rangle$ and a k-index depth-first step sequence $X\,Y \xRightarrow[\mathbf{df}(k)]{\gamma} w$, for two nonterminals $X, Y \in \Xi$, $w \in \Sigma^*$, and $\gamma \in \Delta^*$. There exist $w_1, w_2 \in \Sigma^*$ such that $w_1\,w_2 = w$, and $\gamma_1, \gamma_2 \in \Delta^*$ such that either one of the following holds:*

1. *$X \xRightarrow[\mathbf{df}(k-1)]{\gamma_1} w_1$ and $Y \xRightarrow[\mathbf{df}(k)]{\gamma_2} w_2$ and $\gamma = \gamma_1\,\gamma_2$, or*
2. *$X \xRightarrow[\mathbf{df}(k)]{\gamma_1} w_1$ and $Y \xRightarrow[\mathbf{df}(k-1)]{\gamma_2} w_2$ and $\gamma = \gamma_2\,\gamma_1$.*

Proof. The step sequence $X\,Y \xRightarrow[\mathbf{df}(k)]{\gamma} w$ has one of two possible forms, by the definition of a depth-first sequence:

- $X\,Y \xRightarrow[\mathbf{df}(k)]{\gamma_1} w_1\,Y \xRightarrow[\mathbf{df}(k)]{\gamma_2} w_1\,w_2$, or
- $X\,Y \xRightarrow[\mathbf{df}(k)]{\gamma_2} X\,w_2 \xRightarrow[\mathbf{df}(k)]{\gamma_1} w_1\,w_2$,

for some words $w_1, w_2 \in \Sigma^*$ and control words $\gamma_1, \gamma_2 \in \Delta^*$. Let us consider the first case, the second being symmetric. Since $X\,Y \xRightarrow[\mathbf{df}(k)]{\gamma_1} w_1\,Y$ is a k-index step sequence, the sequence $X \xRightarrow{\gamma_1} w_1$ obtained by erasing the Y nonterminal from the last position in all steps of the sequence, is of index $k-1$, i.e. $X \xRightarrow[\mathbf{df}(k-1)]{\gamma_1} w_1$. Also, since $w_1\,Y \xRightarrow[\mathbf{df}(k)]{\gamma_2} w_1\,w_2$, we obtain $Y \xRightarrow[\mathbf{df}(k)]{\gamma_2} w_2$, by erasing the first $|w_1|$ symbols in all steps of the sequence. Clearly, in this case we have $\gamma = \gamma_1\,\gamma_2$. □

B.1 Proof of Lemma 1

First, we formally define the notion of depth-first derivations by annotating symbols occurring in every step with a positive integer called the *rank*. Intuitively, the rank assigns a priority between symbols in a word. For a set S of symbols (e.g. the terminals and nonterminals) and a set $I \subseteq \mathbb{N}$, we define $S^I = \{s^{\langle i \rangle} \mid s \in S,\, i \in I\}$ and call S^I a *ranked alphabet*. We also sometimes write $S^{\langle i \rangle}$ when I is a singleton. A *ranked word* (r-word) is a word over a ranked alphabet. Given a word w of length n and an n-dimensional vector $\boldsymbol{\alpha} \in \mathbb{N}^n$, the *ranked word* $w^{\boldsymbol{\alpha}}$ is the sequence $(w)_1{}^{\langle(\boldsymbol{\alpha})_1\rangle} \ldots (w)_n{}^{\langle(\boldsymbol{\alpha})_n\rangle}$, in which the ith element of $\boldsymbol{\alpha}$ annotates the ith symbol of w. We also denote $w^{\langle\!\langle c\rangle\!\rangle} = (w)_1{}^{\langle c\rangle} \ldots (w)_{|w|}{}^{\langle c\rangle}$ as a shorthand. Let $G = \langle \Xi, \Sigma, \Delta \rangle$ be a grammar and $u \xRightarrow{(Z,w)/j} v$ be a step, for a vector $\boldsymbol{\alpha} \in \mathbb{N}^{|u|}$, we define the *ranked step* (r-step) $u^{\boldsymbol{\alpha}} \xRightarrow{(Z,w)/j} v^{\boldsymbol{\beta}}$ if and only if $(u)_j = Z$ and

$$v^{\boldsymbol{\beta}} = (u^{\boldsymbol{\alpha}})_1 \cdots (u^{\boldsymbol{\alpha}})_{j-1}\, w^{\langle\!\langle m+1\rangle\!\rangle}\, (u^{\boldsymbol{\alpha}})_{j+1} \cdots (u^{\boldsymbol{\alpha}})_{|u|}$$

where each symbol in w has rank $m+1$ and

$$m = \max\left(\{(\boldsymbol{\alpha})_i \mid \exists i\colon 1 \leqslant i \leqslant |u|, i \neq j, (u)_i \in \Xi\} \cup \{-1\}\right)$$

is the maximum among the ranks of the nonterminals in $u^{\boldsymbol{\alpha}}$, with position j omitted[7]. An r-step is said to be *depth-first*, denoted $u^{\boldsymbol{\alpha}} \underset{\mathbf{df}}{\Rightarrow} v^{\boldsymbol{\beta}}$ iff the rank of the nonterminal at position j where the rule applies is maximal, i.e. $(\boldsymbol{\alpha})_j = m$. For instance the transition labelled $\mathbf{p_2}$ in Fig. 1 (d) is a depth-first r-step. A r-step sequence is said to be depth-first if all of its r-steps are depth-first. Finally, an unranked step sequence $w_0 \xRightarrow{(\gamma)_1} w_1 \ldots w_{n-1} \xRightarrow{(\gamma)_n} w_n$ is said to be depth-first, written $w_0 \underset{\mathbf{df}}{\overset{\gamma}{\Rightarrow}} w_n$, iff there exist vectors $\boldsymbol{\alpha}_1 \in \mathbb{N}^{|w_1|}, \ldots, \boldsymbol{\alpha}_n \in \mathbb{N}^{|w_n|}$ such that $w_0^{\langle\!\langle 0\rangle\!\rangle} \underset{\mathbf{df}}{\xRightarrow{(\gamma)_1}} w_1^{\boldsymbol{\alpha}_1} \ldots w_{n-1}^{\boldsymbol{\alpha}_{n-1}} \underset{\mathbf{df}}{\xRightarrow{(\gamma)_n}} w_n^{\boldsymbol{\alpha}_n}$ holds.

Let $\Upsilon^{(k)} = \{w^{\boldsymbol{\alpha}} \mid \exists u^{\boldsymbol{\beta}}\colon u^{\boldsymbol{\beta}} = (w^{\boldsymbol{\alpha}}){\downarrow_{\Xi^{\mathbb{N}}}}, |u^{\boldsymbol{\beta}}| \leqslant k, \boldsymbol{\beta} \text{ is contiguous}, \max_i(\boldsymbol{\beta})_i \leqslant k-1\}$ be the set of r-words such that when deleting ranked terminals, the resulting word is no longer than k and has ranks between 0 and $k-1$. It is routine to check that $\Upsilon^{(k)}$ is closed for the relation $\underset{\mathbf{df}(k)}{\Longrightarrow}$. For a r-word $w^{\boldsymbol{\alpha}} \in \Upsilon^{(k)}$, let $[w^{\boldsymbol{\alpha}}]$ be the r-word $(w^{\boldsymbol{\alpha}}{\downarrow_{\Xi^{\langle 0\rangle}}})\,(w^{\boldsymbol{\alpha}}{\downarrow_{\Xi^{\langle 1\rangle}}}) \ldots (w^{\boldsymbol{\alpha}}{\downarrow_{\Xi^{\langle k\rangle}}})$. Intuitively, $[w^{\boldsymbol{\alpha}}]$ projects out the terminals of w, and orders the remaining nonterminals in the increasing order of their ranks. For instance, $\left[a^{\langle 1\rangle} Y^{\langle 1\rangle} Z^{\langle 0\rangle}\right] = Z^{\langle 0\rangle} Y^{\langle 1\rangle}$. The $[.]$ operator is naturally lifted from r-words to sets of r-words. Recall that we define the set Q of states of $A^{\mathbf{df}(k)} = (Q, \Delta, \rightarrow)$ as $Q = \{w^{\boldsymbol{\alpha}} \mid w \in \Xi^*, |w| \leqslant k, \boldsymbol{\alpha} \text{ is contiguous}, (\boldsymbol{\alpha})_1 \leqslant \cdots \leqslant (\boldsymbol{\alpha})_{|w|}\}$. It is routine to check that $\left[\Upsilon^{(k)}\right] = Q$ holds. Now let us consider $\rightarrow$ which we defined as follows. Let $q, q' \in Q$, $(X, w) \in \Delta$ we have $q \xrightarrow{(X,w)} q'$ iff

- $q = u\, X^{\langle i\rangle}\, v$ for some u, v and where i is the maximum rank in q, and
- $q' = u\, v\, (w{\downarrow_{\Xi}})^{\langle\!\langle i'\rangle\!\rangle}$ where $|u\, v\, (w{\downarrow_{\Xi}})^{\langle\!\langle i'\rangle\!\rangle}| \leqslant k$ and $i' = \begin{cases} 0 & \text{if } u\, v = \varepsilon \\ i & \text{else if } (u\, v){\downarrow_{\Xi^{\langle i\rangle}}} = \varepsilon \\ i+1 & \text{else} \end{cases}$

As $q \in Q$, we find that $q \in \left[\Upsilon^{(k)}\right]$. Furthermore, it is an easy exercise to show that $q \xrightarrow{(X,w)} q'$ iff there exists $w^{\boldsymbol{\eta}} \in \Upsilon^{(k)}$ such that $q \underset{\mathbf{df}(k)}{\xRightarrow{(X,w)}} w^{\boldsymbol{\eta}}$ and $[w^{\boldsymbol{\eta}}] = q'$. It follows that, we can equivalently write $A_G^{\mathbf{df}(k)} = \left\langle \left[\Upsilon^{(k)}\right], \Delta, \rightarrow \right\rangle$ for the labeled graph the edge relation, is defined as: $u^{\boldsymbol{\alpha}} \xrightarrow{p} v^{\boldsymbol{\beta}}$ iff $\exists w^{\boldsymbol{\eta}} \in \Upsilon^{(k)}.\, u^{\boldsymbol{\alpha}} \underset{\mathbf{df}(k)}{\xRightarrow{p}} w^{\boldsymbol{\eta}} \wedge v^{\boldsymbol{\beta}} = [w^{\boldsymbol{\eta}}]$.

[7] If $Z = (u)_j$ is the only non-terminal in u, we have $m + 1 = -1 + 1 = 0$.

Proof (of Lemma 1). "$\Rightarrow$" We shall prove the following more general statement. Let $u^{\boldsymbol{\alpha}} \xRightarrow[\mathbf{df}(k)]{\gamma} w^{\boldsymbol{\beta}}$ where $u^{\boldsymbol{\alpha}} \in \Upsilon^{(k)}$ be a k-index depth-first r-step sequence. By induction on $|\gamma| \geqslant 0$, we show the existence of a path $[u^{\boldsymbol{\alpha}}] \xrightarrow{\gamma} [w^{\boldsymbol{\beta}}]$ in $A^{\mathbf{df}(k)}$. For the base case $|\gamma| = 0$, we have $u^{\boldsymbol{\alpha}} = w^{\boldsymbol{\beta}}$ which yields $[u^{\boldsymbol{\alpha}}] = [w^{\boldsymbol{\beta}}]$ and since $u^{\boldsymbol{\alpha}} \in \Upsilon^{(k)}$ the hypothesis shows that $u^{\boldsymbol{\alpha}}, w^{\boldsymbol{\beta}} \in \Upsilon^{(k)}$, hence that $[u^{\boldsymbol{\alpha}}], [w^{\boldsymbol{\beta}}] \in [\Upsilon^{(k)}]$ and we are done. For the induction step $|\gamma| > 0$, let $v^{\boldsymbol{\eta}} \xRightarrow[\mathbf{df}(k)]{p} w^{\boldsymbol{\beta}}$ be the last step of the sequence, for some $p \in \Delta$, i.e. $\gamma = \sigma \cdot p$ with $\sigma \in \Delta^*$. By the induction hypothesis, $A^{\mathbf{df}(k)}$ has a path $[u^{\boldsymbol{\alpha}}] \xrightarrow{\sigma} [v^{\boldsymbol{\eta}}]$. Since $[v^{\boldsymbol{\eta}}], [w^{\boldsymbol{\beta}}] \in [\Upsilon^{(k)}]$ and $v^{\boldsymbol{\eta}} \xRightarrow[\mathbf{df}(k)]{p} w^{\boldsymbol{\beta}}$, we have that $[v^{\boldsymbol{\eta}}] \xrightarrow{p} [w^{\boldsymbol{\beta}}]$ by definition of $\rightarrow$, hence we obtain a path $[u^{\boldsymbol{\alpha}}] \xrightarrow{\gamma} [w^{\boldsymbol{\beta}}]$.

"$\Leftarrow$" We prove a more general statement. Let $U \xrightarrow{\gamma} W$ be a path in $A_G^{\mathbf{df}(k)}$, for some words $U, W \in [\Upsilon^{(k)}]$. We show by induction on $|\gamma|$ that there exist r-words $u^{\boldsymbol{\alpha}}, w^{\boldsymbol{\beta}} \in \Upsilon^{(k)}$, such that $[u^{\boldsymbol{\alpha}}] = U$, $[w^{\boldsymbol{\beta}}] = W$, and $u^{\boldsymbol{\alpha}} \xRightarrow[\mathbf{df}(k)]{\gamma} w^{\boldsymbol{\alpha}}$. The base case $|\gamma| = 0$ is trivial, because $U = W$ and since $U \in [\Upsilon^{(k)}]$ then there exists $u^{\boldsymbol{\alpha}} \in \Upsilon^{(k)}$ such that $[u^{\boldsymbol{\alpha}}] = U = W$ and we are done. For the induction step $|\gamma| > 0$, let $\gamma = \sigma \cdot p$, for some production $p \in \Delta$ and $\sigma \in \Delta^*$. By the induction hypothesis, there exist r-words $u^{\boldsymbol{\alpha}}, v^{\boldsymbol{\eta}} \in \Upsilon^{(k)}$ such that $U = [u^{\boldsymbol{\alpha}}] \xrightarrow{\sigma} [v^{\boldsymbol{\eta}}] \xrightarrow{p} W$ is a path in $A^{\mathbf{df}(k)}$, and $u^{\boldsymbol{\alpha}} \xRightarrow[\mathbf{df}(k)]{\sigma} v^{\boldsymbol{\eta}}$ is a k-index r-step sequence. The definition of the edge relation in $A^{\mathbf{df}(k)}$ and $[v^{\boldsymbol{\eta}}] \xrightarrow{p} w$ shows that $v^{\boldsymbol{\eta}} \xRightarrow[\mathbf{df}(k)]{p} w^{\boldsymbol{\beta}}$ for some $w^{\boldsymbol{\beta}} \in \Upsilon^{(k)}$ such that $[w^{\boldsymbol{\beta}}] = W$.

For the upper bound on the size of $A^{\mathbf{df}(k)}$, recall that each vertex of $A^{\mathbf{df}(k)}$ is a ranked word of length at most k, consisting of non-terminals only, with ranks in the interval $[0, k-1]$. Moreover, the productions of G do not produce more than 2 nonterminals at a time. Hence, in every vertex of $A^{\mathbf{df}(k)}$, at most 2 positions carry the same rank. Since the length of each vertex in Q is at most k and, for each $i \in [0, k-1]$, there are at most $\|\Xi\|^2$ choices of nonterminals with rank i, we have $|A_G^{\mathbf{df}(k)}| \leqslant \|\Xi\|^{2k} \leqslant |G|^{2k}$. □

B.2 Proof of Lemma 2

When $L_{X,Y}(G) \subseteq \widetilde{\mathbf{b}}$, because $\widetilde{\mathbf{b}} = a_1^* \dots a_s^*$ is a strict s-letter-bounded expression with s a fixed constant, for every step sequence $X \xRightarrow{\gamma}_G u\,Y\,v$, we have $u\,v = \gamma\!\downarrow_{a_1} \dots \gamma\!\downarrow_{a_s}$. Also remark that $u\,v = a_1^{(\boldsymbol{v})_1} \dots a_s^{(\boldsymbol{v})_s}$ for some $\boldsymbol{v} \in \mathbb{N}^s$, hence that $(\boldsymbol{v})_\ell = |\gamma\!\downarrow_{a_\ell}|$ for each $\ell = 1, \dots, s$. For convenience, given $\gamma \in \Delta^*$, we denote $\gamma\!\downarrow\!\downarrow_{\widetilde{\mathbf{b}}} = \gamma\!\downarrow_{a_1} \dots \gamma\!\downarrow_{a_s}$.

We recall the definition of the labeled graph $A^{\mathbf{df}(k)} = \langle Q, \Delta, \rightarrow \rangle$ whose number of vertices we denote by N. Due to the form of the productions in G, we can safely restrict Q to r-words with at most 2 nonterminals having the same rank, hence $N \leqslant |G|^{2k}$. We define $\Omega(q)$ is the set of elementary cycles with $q \in Q$ as endpoints.

Proposition 2. *Let $G = \langle \Xi, \Sigma, \Delta \rangle$ be a grammar, $X \in \Xi$ be a nonterminal and $\widetilde{\mathbf{b}} = a_1^* \ldots a_s^*$ be a strict s-letter bounded expression, for some $s \geqslant 0$. For any two vertices $q, q' \in Q$ of $A^{\mathbf{df}(k)}$, and any path $\pi \in \Pi(q, q')$, there exists a path $\pi' \in \Pi(q, q')$ such that $|\pi| = |\pi'|$, $\omega(\pi)\downarrow\downarrow_{\widetilde{\mathbf{b}}} = \omega(\pi')\downarrow\downarrow_{\widetilde{\mathbf{b}}}$ and π' is of the form $\varsigma_1 \cdot \theta_1 \cdots \varsigma_\ell \cdot \theta_\ell \cdot \varsigma_{\ell+1}$, where $\varsigma_1 \in \Pi(q, q_1)$, $\varsigma_{\ell+1} \in \Pi(q_\ell, q')$ and $\varsigma_j \in \Pi(q_{j-1}, q_j)$, for each $1 < j \leqslant \ell$, are acyclic paths, $\theta_1 \in (\Omega(q_1))^*, \ldots, \theta_\ell \in (\Omega(q_\ell))^*$ are cycles, and $\ell \leqslant \|Q\|$.*

Proof. The proof goes along the lines of that of Lemma 7.3.2 in Lin's PhD thesis [19]. This proof is carried on graphs labeled with integer tuples, and addition, instead of concatenation. Since the only property of integer tuple addition, used in the proof of [19, Lemma 7.3.2], is commutativity, it suffices to observe that $\omega(\pi)\downarrow\downarrow_{\widetilde{\mathbf{b}}} = \omega(\pi')\downarrow\downarrow_{\widetilde{\mathbf{b}}}$, whenever $\omega(\pi)$ is a permutation of $\omega(\pi')$. □

Proof (of Lemma 2). Given two step sequences $X \stackrel{\gamma}{\Rightarrow}_G u\,Y\,v$, $X \stackrel{\gamma'}{\Rightarrow}_G u'\,Y\,v'$, the following are equivalent:

- $|\gamma\downarrow_{a_\ell}| = |\gamma'\downarrow_{a_\ell}|$ for all $\ell = 1, \ldots, s$,
- $\gamma\downarrow\downarrow_{\widetilde{\mathbf{b}}} = \gamma'\downarrow\downarrow_{\widetilde{\mathbf{b}}}$,
- $u\,v = u'\,v'$.

Since $L_{X,Y}(G) \subseteq \widetilde{\mathbf{b}}$ where $\widetilde{\mathbf{b}}$ is a strict s-letter bounded expression, for every $\pi \in \Omega(q)$ the induced word $a_1^{k_1} \ldots a_s^{k_s} = \omega(\pi)\downarrow\downarrow_{\widetilde{\mathbf{b}}}$ is such that: $\sum_{j=1}^{s} k_j \leqslant 2N$, i.e. each production in Δ issues at most 2 symbols from $\{a_1, \ldots, a_s\}$, and each elementary cycle is of length at most N. The nonnegative solutions of the inequation $\sum_{j=1}^{s} k_j \leqslant 2N$ are solutions to the equation $\sum_{j=1}^{s} k_j + y = 2N$, for a nonnegative slack variable $y \geqslant 0$. Since the number of nonnegative solutions to the latter equation[8] is $\binom{s+2N}{s}$, we have:

$$\|\{\omega(\pi)\downarrow\downarrow_{\widetilde{\mathbf{b}}} \mid \pi \in \Omega(q)\}\| = \binom{s+2N}{s} = \mathcal{O}(N^s) \ . \tag{6}$$

For each vertex q, we are interested in a set $C_q \subseteq \Omega(q)$ such that $\|C_q\| = \mathcal{O}(N^s)$ and, moreover, for each $\pi \in \Omega(q)$ there exists $\pi' \in C_q$ such that $\omega(\pi)\downarrow\downarrow_{\widetilde{\mathbf{b}}} = \omega(\pi')\downarrow\downarrow_{\widetilde{\mathbf{b}}}$ when $\Pi(X^{\langle 0 \rangle}, q) \neq \varnothing$ and $\Pi(q, Y^{\langle 0 \rangle}) \neq \varnothing$ holds.

For now we assume we have computed such sets $\{C_q\}_{q \in Q}$ (their effective computation will be described later). We are now ready to define the bounded expression $\Gamma_{\widetilde{\mathbf{b}}}$. Given a finite set $\Gamma = \{\gamma_1, \ldots, \gamma_n\} \subseteq \Delta^*$ of control words indexed following some total ordering (e.g. we assume a total order $\prec$ on $\Xi \cup \mathcal{A}$, and define $(X_1, w_1) \prec_\Delta (X_2, w_2) \Leftrightarrow X_1 \cdot w_1 \prec^{lex} X_2 \cdot w_2$ in the lexicographical extension of $\prec$, then extend $\prec_\Delta$ to a lexicographical order $\prec_\Delta^{lex}$ on control words), we define the bounded expression: $concat(\Gamma) = \gamma_1^* \cdots \gamma_n^*$. Let $Q = \{q_1, \ldots, q_N\}$ be the set of vertices of $A^{\mathbf{df}(k)}$, taken in some order. We define the set $\{B_i\}_{i \geqslant 0}$ of bounded

[8] The number of nonnegative solutions of an equation $n = x_1 + \cdots + x_m$ is $\binom{m+n-1}{m-1}$.

expressions as follows:

$$B_0 = concat(\{\omega(\pi) \mid \pi \in C_{q_1}\}) \cdots concat(\{\omega(\pi) \mid \pi \in C_{q_N}\})$$
$$B_1 = concat(\Delta)^{N-1} \cdot B_0 \cdot concat(\Delta)^{N-1}$$
$$B_i = concat(\Delta)^{N-1} \cdot B_0 \cdot B_{i-1}, \text{ for all } i \geqslant 2$$

Finally, let:

$$\Gamma_{\widetilde{\mathbf{b}}} = B_N \enspace .$$

Let us now prove the language inclusion.

It follows from Theorem 1, that $L^{(k)}_{X,Y}(G) = \hat{L}_{X,Y}(\Gamma^{\mathbf{df}(k)}_{X,Y}, G)$ for every $X \in \Xi$, $Y \in \Xi \cup \{\varepsilon\}$ and $k > 0$. Hence we trivially have $\hat{L}_{X,Y}(\Gamma_{\widetilde{\mathbf{b}}} \cap \Gamma^{\mathbf{df}(k)}_{X,Y}, G) \subseteq \hat{L}_{X,Y}(\Gamma^{\mathbf{df}(k)}_{X,Y}, G) = L^{(k)}_{X,Y}(G)$. For the contrapositive $L^{(k)}_{X,Y}(G) \subseteq \hat{L}_{X,Y}(\Gamma_{\widetilde{\mathbf{b}}} \cap \Gamma^{\mathbf{df}(k)}_{X,Y}, G)$, it suffices to show the following: given a k-index depth first step sequence $X \xRightarrow[\mathbf{df}(k)]{\gamma} u\,Y\,v$, there exists a control word $\gamma' \in \Gamma_{\widetilde{\mathbf{b}}}$ such that $X \xRightarrow[\mathbf{df}(k)]{\gamma'} u'\,Y\,v'$ and $u\,v = u'\,v'$.

Because Lemma 1 shows that each path $\pi \in \Pi(X^{\langle 0\rangle}, Y^{\langle 0\rangle})$ corresponds to a control word $\omega(\pi)$ such that $X \xRightarrow[\mathbf{df}(k)]{\omega(\pi)} u\,Y\,v$, and because $L^{(k)}_{X,Y}(G) \subseteq \widetilde{\mathbf{b}}$ where $\widetilde{\mathbf{b}}$ is a strict s-letter bounded expression, it suffices to show that exists a path $\rho \in \Pi(X^{\langle 0\rangle}, Y^{\langle 0\rangle})$ such that $\omega(\rho) \in \Gamma_{\widetilde{\mathbf{b}}}$ and $\omega(\pi){\downarrow\downarrow}_{\widetilde{\mathbf{b}}} = \omega(\rho){\downarrow\downarrow}_{\widetilde{\mathbf{b}}}$. We apply the result from Prop. 2 which shows that there exists a path $\rho \in \Pi(X^{\langle 0\rangle}, Y^{\langle 0\rangle})$, such that $|\rho| = |\pi|$, $\omega(\rho){\downarrow\downarrow}_{\widetilde{\mathbf{b}}} = \omega(\pi){\downarrow\downarrow}_{\widetilde{\mathbf{b}}}$ and ρ is of the form $\varsigma_1 \cdot \theta_1 \cdots \varsigma_\ell \cdot \theta_\ell \cdot \varsigma_{\ell+1}$, where $\varsigma_1 \in \Pi(X^{\langle 0\rangle}, q_{i_1})$, $\varsigma_{\ell+1} \in \Pi(q_{i_\ell}, Y^{\langle 0\rangle})$, and $\varsigma_j \in \Pi(q_{i_{j-1}}, q_{i_j})$ for each $1 < j \leqslant \ell$ are acyclic paths, $\theta_1 \in (\Omega(q_{i_1}))^*, \ldots, \theta_\ell \in (\Omega(q_{i_\ell}))^*$ are cycles, $q_{i_1}, \ldots, q_{i_\ell}$ are vertices, and $\ell \leqslant \|Q\|$. Hence we conclude that

- $\omega(\varsigma_j) \in concat(\Delta)^{N-1}$, for all $1 \leqslant j \leqslant \ell + 1$,
- for each cycle $\theta_j \in (\Omega(q_{i_j}))^*$, consisting of a concatenation of several elementary cycles $\theta_j^1, \ldots, \theta_j^{\ell_j} \in \Omega(q_{i_j})$, the cycle θ_j^{lex} obtained by a lexicographic reordering of $\theta_j^1, \ldots, \theta_j^{\ell_j}$ (based on the lexicographic order of their value in Δ^*) belongs to B_0, for all $1 \leqslant j \leqslant \ell$. Second, it is easy to see that the words produced by θ_j and θ_j^{lex} are the same, since the order of productions labeling θ_j (θ_j^{lex}) is not important.

Let π' be the path $\varsigma_1 \cdot \theta_1^{lex} \cdots \varsigma_\ell \cdot \theta_\ell^{lex} \cdot \varsigma_{\ell+1}$. By Prop. 2, we have that $\omega(\pi){\downarrow\downarrow}_{\widetilde{\mathbf{b}}} = \omega(\pi'){\downarrow\downarrow}_{\widetilde{\mathbf{b}}}$. Moreover, $\omega(\pi') \in B_N = \Gamma_{\widetilde{\mathbf{b}}}$. Since $X \xRightarrow[\mathbf{df}(k)]{\omega(\pi)} u\,Y\,v$ and $X \xRightarrow[\mathbf{df}(k)]{\omega(\pi')} u'\,Y\,v'$ are step sequences of G, the previous equality implies $u\,v = u'\,v'$.

Concerning the time needed to construct the bounded expression $\Gamma_{\widetilde{\mathbf{b}}}$, the main ingredient in the previous, is the definition of the sets of cycles $\{C_q\}_{q \in Q}$, such that $\|C_q\| = \mathcal{O}(N^s)$ and, moreover, for each $\pi \in \Omega(q)$ there exists $\pi' \in C_q$ such that $\omega(\pi){\downarrow\downarrow}_{\widetilde{\mathbf{b}}} = \omega(\pi'){\downarrow\downarrow}_{\widetilde{\mathbf{b}}}$ when $\Pi(X^{\langle 0\rangle}, q) \neq \varnothing$ and $\Pi(q, Y^{\langle 0\rangle}) \neq \varnothing$ holds. Below we describe the construction of such sets.

Define $Val = \{a_1^{\ell_1} \dots a_s^{\ell_s} \in \widetilde{\mathbf{b}} \mid \sum_{j=1}^{s} \ell_j \leqslant 2N\}$. Using previous arguments (i.e. equation (6)), it is routine to check that $\|Val\| = \mathcal{O}(N^s)$. Consider the labeled graph $\mathcal{H} = \langle V, \Delta, \rightarrow \rangle$, defined upon $A^{\mathbf{df}(k)}$, where:

- $V = Q \times Val$, and
- $\langle q', a_1^{i_1} \dots a_s^{i_s} \rangle \xrightarrow{(Z,z)} \langle q'', a_1^{j_1} \dots a_s^{j_s} \rangle$ iff $q' \xrightarrow{(Z,z)} q''$ and $a_\ell^{j_\ell} = a_\ell^{i_\ell} \cdot z{\downarrow_{a_\ell}}$ for each ℓ

First, observe that the number of vertices in this graph is $\|V\| \leqslant N^{2k} \cdot \binom{s+2N}{s} = |G|^{\mathcal{O}(k)}$. Second, it is routine to check (by induction on the length of a path) that given a path $\pi \in \Pi_{\mathcal{H}}(\langle q, \varepsilon \rangle, \langle q, a_1^{i_1} \dots a_s^{i_s} \rangle)$ for some $i_1, \dots, i_s \in \mathbb{N}$ we have $\omega(\pi){\downarrow\downarrow_{\widetilde{\mathbf{b}}}} = a_1^{i_1} \dots a_s^{i_s}$. Next, for each $q \in Q$ define the set $\mathcal{P}_q$ of paths of $\mathcal{H}$ consisting for each $a_1^{i_1} \dots a_s^{i_s} \in Val$ of a single path (one with the least number of edges) from $\langle q, \varepsilon \rangle$ to $\langle q, a_1^{i_1} \dots a_s^{i_s} \rangle$. By definition of Val, we have that $\|\mathcal{P}_q\| = \|Val\| = \mathcal{O}(N^s)$ and, moreover, for each $\rho \in \Omega(q)$ (ρ is a path of $A^{\mathbf{df}(k)}$) there exists a path $\pi \in \mathcal{P}_q$ such that $\omega(\rho){\downarrow\downarrow_{\widetilde{\mathbf{b}}}} = \omega(\pi){\downarrow\downarrow_{\widetilde{\mathbf{b}}}} = a_1^{i_1} \dots a_s^{i_s}$ where $\langle q, \varepsilon \rangle$ and $\langle q, a_1^{i_1} \dots a_s^{i_s} \rangle$ are the endpoints of π.

Hence, we define C_q to be the set of cycles in $A^{\mathbf{df}(k)}$ corresponding to the paths in $\mathcal{P}_q$. The latter can be computed applying Dijkstra's single source shortest path algorithm on $\mathcal{H}$, with source vertex $\langle q, \varepsilon \rangle$, and assuming that the distance between adjacent vertices is always 1. The running time of the Dijkstra's algorithm is $\mathcal{O}(\|V\|^2) = |G|^{\mathcal{O}(k)}$. Upon termination, one can reconstruct a shortest path π from $\langle q, \varepsilon \rangle$ to each vertex $\langle q, a_1^{i_1} \dots a_s^{i_s} \rangle$, and add the corresponding cycle of $A^{\mathbf{df}(k)}$ to C_q. Since there are at most $|G|^{\mathcal{O}(k)}$ vertices $\langle q, a_1^{i_1} \dots a_s^{i_s} \rangle$ in V, and building a shortest path for each such vertex takes at most $|G|^{\mathcal{O}(k)}$ time, we can populate the set C_q in time $|G|^{\mathcal{O}(k)}$. Once the sets C_q are built, it remains to compute the bounded expressions $concat(\{\omega(\pi) \mid \pi \in C_q\})$, $concat(\Delta)^{N-1}$ and $B_0, \dots, B_N$. As shown below, they are all computable in time $|G|^{\mathcal{O}(k)}$.

Algorithm 1 gives the construction of $\Gamma_{\widetilde{\mathbf{b}}}$. An upper bound on the time needed for building $\Gamma_{\widetilde{\mathbf{b}}}$ can be derived by a close analysis of the running time of Algorithm 1. The input to the algorithm is a grammar G, a strict s-letter bounded expression $\widetilde{\mathbf{b}}$ and an integer $k > 0$. First (lines 2–5) the algorithm builds the $\mathcal{H}$ graph, which takes time $|G|^{\mathcal{O}(k)}$. The loop on (lines 8–10) computes, for each vertex $q \in Q$, and each s-dimensional vector $\boldsymbol{v} \in Val$, an elementary path from $\langle q, \varepsilon \rangle$ to $\langle q, a_1^{(\boldsymbol{v})_1} \dots a_s^{(\boldsymbol{v})_s} \rangle$ in $\mathcal{H}$. For each q, this set is kept in a variable C_q (line 9). The variable B_0 at the end of the loop contains the expression $concat(\{\omega(\pi) \mid \pi \in \mathcal{P}_{q_1}\}) \cdots concat(\{\omega(\pi) \mid \pi \in \mathcal{P}_{q_N}\})$, Since both $\|Q\| = |G|^{\mathcal{O}(k)}$ and $\|Val\| = |G|^{\mathcal{O}(k)}$, the loop at (lines 8–10) takes time $|G|^{\mathcal{O}(k)}$ as well.

The remaining part of the algorithm computes first an over-approximation of $concat(\Delta)^{N-1}$ (lines 11–13) in the variable C—observe that the algorithm computes $concat(\Delta)^{|G|^{2k}} - 1$ instead of $concat(\Delta)^{N-1}$. Finally, the control set $\Gamma_{\widetilde{\mathbf{b}}}$ with the needed property is produced by $|G|^{2k} \geqslant N$ repeated concatenations of the bounded expression $C \cdot B_0$, at lines (15–16). Since both loops take time at most $|G|^{2k}$, we conclude that Algorithm 1 runs in time $|G|^{\mathcal{O}(k)}$. $\square$

B.3 Proof of Lemma 3

A grammar G is said to be *reduced* for X iff $L_{X,Y}(G) \neq \varnothing$ and $L_Y(G) \neq \varnothing$, for every $Y \in \Xi$, $X \neq Y$. A grammar can be reduced in polynomial time, by eliminating unreachable and unproductive nonterminals [12, Lemma 1.4.4].

Proof (of Lemma 3). We start by proving a series of five facts.

(i) First, no production of G has the form (Y, v), where $Y \in \Xi_{\overset{\smile}{\mathrm{1..d}}}$ and v contains a symbol of $\Xi_{\overset{\frown}{\mathrm{1..d}}}$. By contradiction, assume such a production exists where $Z \in \Xi_{\overset{\frown}{\mathrm{1..d}}}$ is a nonterminal occurring in v. Because $Z \in \Xi_{\overset{\frown}{\mathrm{1..d}}}$, a_1 occurs in some word of $L_Z(G)$ and a_d occurs in some word of $L_Z(G)$. On the other hand, we have that either no word of $L_Y(G)$ contains a_1 or no word of $L_Y(G)$ contains a_d, since $Y \in \Xi_{\overset{\smile}{\mathrm{1..d}}}$. Because G is reduced, we have $\{u \mid v \Rightarrow^* u\} \neq \varnothing$. We reach a contradiction, since $\{u \mid Y \xRightarrow{(Y,v)} v \Rightarrow^* u\}$ contains a word in which a_1 occurs and a word in which a_d occurs, because Z occurs in v.

(ii) Define $Q(u, v)$ to be the following proposition:

$$\left\{u' \in (\Xi \cup \mathcal{A})^* \mid u \Rightarrow^* u'\right\} \subseteq (\{a_1\} \cup \Xi_{\overset{\smile}{\mathrm{1..d}}})^*$$

and

$$\left\{v' \in (\Xi \cup \mathcal{A})^* \mid v \Rightarrow^* v'\right\} \subseteq (\{a_d\} \cup \Xi_{\overset{\smile}{\mathrm{1..d}}})^* \ .$$

We show that $Q(u, v)$ holds if $X_i \Rightarrow^* u\, X_j\, v$ with $X_i, X_j \in \Xi_{\overset{\frown}{\mathrm{1..d}}}$. By contradiction, assume that there exists u' such that $u \Rightarrow^* u'$ and $u' \notin (\{a_1\} \cup \Xi_{\overset{\smile}{\mathrm{1..d}}})^*$ (a similar argument holds for v). Then either (a) u' contains a symbol a_ℓ, for $\ell > 1$ or (b) u' contains a nonterminal $Z \in \Xi_{\overset{\frown}{\mathrm{1..d}}}$. Because G is reduced, we have $\{u' \mid u \Rightarrow^* u'\} \neq \varnothing$. In either case (a) or (b), there exists a step sequence $u' \Rightarrow^* u_1\, a_\ell\, u_2 \in \mathcal{A}^*$ such that $\ell > 1$. Since $X_j \in \Xi_{\overset{\frown}{\mathrm{1..d}}}$, we have that $X_j\, v \Rightarrow^* a_1\, u_3 \in \mathcal{A}^*$, hence that $X_i \Rightarrow^* u_1\, a_\ell\, u_2\, a_1\, u_3$ and finally that $L_X(G) \nsubseteq \widetilde{\mathbf{b}}$, since G is reduced, a contradiction.

(iii) For every step sequence $X_j \Rightarrow^* x$, where $X_j \in \Xi_{\overset{\frown}{\mathrm{1..d}}}$, x cannot be of the form $u_1\, X_d\, u_2\, X_e\, u_3$ where $X_d, X_e \in \Xi_{\overset{\frown}{\mathrm{1..d}}}$. In fact, take the decomposition $u = u_1$ and $v = u_2\, X_e\, u_3$ (the case $u = u_1\, X_d\, u_2$ and $v = u_3$ yields the same result). Because (ii) applies, we find that $Q(u, v)$ holds but $v \notin (\{a_d\} \cup \mathcal{A} \cup \Xi_{\overset{\smile}{\mathrm{1..d}}})^*$, hence a contradiction.

(iv) If $X \overset{\gamma}{\Rightarrow}_G u\, X_i\, v$ is a step sequence of G, for some $X_i \in \Xi_{\overset{\frown}{\mathrm{1..d}}}$, $\gamma \in \Delta^*$ then $X \overset{\gamma}{\Rightarrow}_{G^\sharp} u\, X_i\, v$ is also a step sequence of $G^\sharp$. The proof goes by induction on $n = |\gamma|$. Let $X = w_0 \xRightarrow{(\gamma)_1}_G w_1 \cdots w_{n-1} \xRightarrow{(\gamma)_n}_G w_n = u\, X_i\, v$. If $n = 0$ then $\gamma = \varepsilon$, $X = X_i \in \Xi_{\overset{\frown}{\mathrm{1..d}}}$ and $u = v = \varepsilon$, which trivially yields a step sequence of $G^\sharp$. For the inductive case, because of (i) we find that, necessarily, $(w_{n-1})_\ell \in \Xi_{\overset{\frown}{\mathrm{1..d}}}$ for some ℓ. We thus can apply the induction hypothesis onto $X \xRightarrow{(\gamma)_1 \ldots (\gamma)_{n-1}}_G w_{n-1}$ and conclude that $X \xRightarrow{(\gamma)_1 \ldots (\gamma)_{n-1}}_{G^\sharp} w_{n-1}$. Next, since $w_{n-1} \xRightarrow{(\gamma)_n} w_n$ it cannot be the case that $w_{n-1} \xRightarrow{(\gamma)_n / p} w_n$ where $p \neq \ell$ and $(\gamma)_n = (Y, t)$ with $Y \in \Xi_{\overset{\frown}{\mathrm{1..d}}}$ for otherwise $X \Rightarrow^*_G w_{n-1}$ contradicts (iii)

(recall that both $(w_{n-1})_\ell$ and X belong to $\Xi_{\widehat{1..d}}$). Thus we have $(\gamma)_n \in \Delta^\sharp$, hence $w_{n-1} \xRightarrow{(\gamma)_n}_{G^\sharp} w_n$, and finally $X \xRightarrow{\gamma}_{G^\sharp} u\, X_i\, v$.

(v) If $L_1, L_2 \subseteq \widetilde{\mathbf{b}}$ and $L_1 \cdot L_2 \subseteq a_\ell^* \dots a_r^*$, for some $1 \leqslant \ell \leqslant r \leqslant d$, then there exists $\ell \leqslant q \leqslant r$ such that $L_1 \subseteq a_\ell^* \dots a_q^*$ and $L_2 \subseteq a_q^* \dots a_r^*$. Assume, by contradiction, that there is no such q. Then there exist words $w_1 = a_\ell^{i_\ell} \dots a_r^{i_r} \in L_1$ and $w_2 = a_\ell^{j_\ell} \dots a_r^{j_r} \in L_2$, two positions p_1, p_2 such that $\ell \leqslant p_2 < p_1 \leqslant r$ such that $i_{p_1} \neq 0$, $j_{p_2} \neq 0$. Because all a_i are distinct, we conclude that $w_1 \cdot w_2 \notin a_\ell^* \dots a_r^*$, hence a contradiction.

We continue with the proof of the five items of the lemma:

1. The derivation $X \xRightarrow[\mathbf{df}(k)]{\gamma} w$, where $|\gamma| = n$, has a unique corresponding r-step sequence $X^{\langle 0\rangle} = w_0^{\boldsymbol{\alpha}_0} \xRightarrow{(\gamma)_1} w_1^{\boldsymbol{\alpha}_1} \dots \xRightarrow{(\gamma)_n} w_n^{\boldsymbol{\alpha}_n} = w^{\boldsymbol{\alpha}_n}$. Now, we define a *parent* relationship in that step sequence, denoted $\lhd$, between r-annotated nonterminals: $Y^{\langle a\rangle} \lhd Z^{\langle b\rangle}$ iff there exists a step in the sequence that rewrites $Y^{\langle a\rangle}$ to $Z^{\langle b\rangle}$, that is $u^{\boldsymbol{\alpha}} \xRightarrow{(Y,t)/j} v^{\boldsymbol{\beta}}$ where $(u^{\boldsymbol{\alpha}})_j = Y^{\langle a\rangle}$, and $(v^{\boldsymbol{\beta}})_\ell = Z^{\langle b\rangle}$ for some $j \leqslant \ell \leqslant j - 1 + |t|$.
Let $(\gamma)_{\ell_p} = (X_{i_p}, a\, y\, b\, z)$ be the last occurrence, in γ, of a production with head $X_{i_p} \in \Xi_{\widehat{1..d}}$. Notice that such an occurrence always exists since $X \in \Xi_{\widehat{1..d}}$ and moreover we have that $a, b \in \mathcal{A} \cup \{\varepsilon\}$, $y, z \in \Xi_{\widetilde{1..d}} \cup \{\varepsilon\}$. In fact, since γ is a derivation, if $y \in \Xi_{\widehat{1..d}}$ or $z \in \Xi_{\widehat{1..d}}$ then $(\gamma)_{\ell_p}$ would clearly not be the last such occurrence. Let $X = X_{i_0}^{\langle r_0\rangle} \lhd X_{i_1}^{\langle r_1\rangle} \lhd \cdots \lhd X_{i_p}^{\langle r_p\rangle}$ be the sequence of ranked ancestors of X_{i_p} in the r-step sequence, and $(\gamma)_{\ell_j} = (X_{i_j}, a\, y_{m_j}\, b\, X_{i_{j+1}}) \in \Delta$ (or, symmetrically $(\gamma)_{\ell_j} = (X_{i_j}, a\, X_{i_{j+1}}\, b\, z_{m_j}) \in \Delta$), for some $a, b \in \mathcal{A} \cup \{\varepsilon\}$, $z_{m_j}, y_{m_j} \in \Xi \cup \{\varepsilon\}$, be the productions introducing these nonterminals, for all $0 \leqslant j < p$.
If $y_{m_j} \in \Xi$, let $\overline{\gamma}_j$ be the subword of γ corresponding to the derivation $y_{m_j} \xRightarrow{\overline{\gamma}_j} w_{m_j}$, for some $w_{m_j} \in \mathcal{A}^*$. Notice that no X_{i_ℓ} has y_{m_j} for ancestor, and that $y_{m_j} \xRightarrow{\overline{\gamma}_j} w_{m_j}$ must be a depth-first derivation because $X \xRightarrow{\gamma} w$ is. Otherwise, if $y_{m_j} = \varepsilon$, let $\overline{\gamma}_j = \varepsilon$. Let $\gamma^\sharp = (\gamma)_{\ell_0} \cdot \overline{\gamma}_0 \cdot (\gamma)_{\ell_1} \cdot \overline{\gamma}_1 \cdots (\gamma)_{\ell_{p-1}} \cdot \overline{\gamma}_{p-1}$. Observe that, since each $y_{m_j} \xRightarrow{\overline{\gamma}_j} w_{m_j}$ is a depth-first derivation, we have $X_{i_{j+1}}^{\langle b\rangle} y_{m_j}^{\langle b\rangle} \xRightarrow{\overline{\gamma}_j} X_{i_{j+1}}^{\langle b\rangle} w_{m_j}^{\boldsymbol{\alpha}}$ (or with $X_{i_{j+1}}$ and y_{m_j} swapped) is a depth-first step sequence because y_{m_j} and $X_{i_{j+1}}$ have the same rank b. Clearly, $\gamma^\sharp$ corresponds to a valid step sequence of G which, moreover, is depth first, since whenever $(\gamma)_{\ell_j}$ fires, X_{i_j} is the only nonterminal left (and whose rank is therefore maximal). It follows from (iv) that because $X \xRightarrow{\gamma^\sharp}_G u\, X_{i_p}\, v$ holds and $X, X_{i_p} \in \Xi_{\widehat{1..d}}$ then $X \xRightarrow{\gamma^\sharp}_{G^\sharp} u\, X_{i_p}\, v$ holds (notice the use of $G^\sharp$ instead of G). Moreover, the definition of $\gamma^\sharp$ shows that $X \xRightarrow{\gamma^\sharp}_{G^\sharp} u\, X_{i_p}\, v$ is a depth-first step sequence and $u, v \in \mathcal{A}^*$.

Since $X \overset{\gamma}{\Rightarrow}_G w$ is a k-index derivation, each step sequence $y_{m_j} \overset{\overline{\gamma_j}}{\Longrightarrow} w_{m_j}$ are of index at most k. Therefore the index of each step sequence $X_{i_{j+1}} y_{m_j} \overset{\overline{\gamma_j}}{\Longrightarrow} X_{i_{j+1}} w_{m_j}$ (or in reverse order) is at most $k+1$. Also, when each $(\gamma)_{\ell_j}$ fires, X_{i_j} is the only nonterminal left and so the index of the step is at most 2. Therefore we find that $X \xRightarrow[(k+1)]{\gamma^\sharp} u\, X_{i_p}\, v$, and finally that $X \xRightarrow[\mathbf{df}(k+1)]{\gamma^\sharp} u\, X_{i_p}\, v$ in $G^\sharp$.

2. Assume that $y, z \in \Xi_{\underparen{1..d}}$ (the cases $y = \varepsilon$ or $z = \varepsilon$ are similar). Since γ of length n induces a k-index depth first derivation, we have that $y\, z \xRightarrow[\mathbf{df}(k)]{(\gamma)_{\ell_p+1} \ldots (\gamma)_n} u_y\, u_z \in \mathcal{A}^*$ can be split into two derivations of G as follows: $y \xRightarrow[\mathbf{df}(k_y)]{\gamma_y} u_y$ and $z \xRightarrow[\mathbf{df}(k_z)]{\gamma_z} u_z$ such that $\max(k_z, k_y) \leqslant k$ and $\min(k_z, k_y) \leqslant k-1$ (see Lem. 7 for a proof). Assume $k_y \leqslant k-1$, the other case being symmetric. Since the only production in $(\gamma)_{\ell_p} \cdots (\gamma)_n$ whose left hand side is a nonterminal from $\Xi_{\overparen{1..d}}$ is $(\gamma)_{\ell_p} = (X_{i_p}, a\, y\, b\, z)$, which, moreover, occurs only in the first position, we have that $\gamma_y \in \Gamma_y^{\mathbf{df}(k-1)}(G_{i_p,aybz})$ and $\gamma_z \in \Gamma_z^{\mathbf{df}(k)}(G_{i_p,aybz})$, by the definition of $G_{i_p,aybz}$.
3. It suffices to notice that $\gamma^\sharp \cdot (\gamma)_{\ell_p} \cdots (\gamma)_n$ results from reordering the productions of γ and that reordering the productions of γ result into a step sequence producing the same word $w = a_1^{i_1} \ldots a_d^{i_d}$ since $L_X(G) \subseteq \widetilde{\mathbf{b}}$ where $\widetilde{\mathbf{b}}$ is a strict d-letter bounded expression. That the resulting derivation has index k and is depth-first follow easily from (1) and (2).
4. Given that $\Delta^\sharp \subseteq \Delta$ we find that $X \Rightarrow^*_{G^\sharp} u\, X_{i_p}\, v$ implies $X \Rightarrow^*_G u\, X_{i_p}\, v$, hence $Q(u,v)$ holds by (ii) and $X, X_{i_p} \in \Xi_{\overparen{1..d}}$. By the definition of $Q(u,v)$, we have:

$$\{u' \in (\Xi \cup \mathcal{A})^* \mid u \Rightarrow^* u'\} \subseteq (\{a_1\} \cup \Xi_{\underparen{1..d}})^* \text{ and}$$
$$\{v' \in (\Xi \cup \mathcal{A})^* \mid v \Rightarrow^* v'\} \subseteq (\{a_d\} \cup \Xi_{\underparen{1..d}})^*$$

Since G is reduced, $\{u' \in \mathcal{A}^* \mid u \Rightarrow^* u'\} \neq \emptyset$ and $\{v' \in \mathcal{A}^* \mid v \Rightarrow^* v'\} \neq \emptyset$. But because $X_{i_p} \in \Xi_{\overparen{1..d}}$, it must be the case that $\{u' \in \mathcal{A}^* \mid u \Rightarrow^* u'\} \subseteq a_1^*$ and $\{v' \in \mathcal{A}^* \mid v \Rightarrow^* v'\} \subseteq a_d^*$, otherwise we would contradict the fact that $L_X(G) \subseteq \widetilde{\mathbf{b}}$.
5. Since $X \overset{\gamma^\sharp}{\Rightarrow}_G u\, X_{i_p}\, v \xRightarrow{(X_{i_p}, a\, y\, b\, z)}_G u\, a\, y\, b\, z\, v$ and G is reduced, we have that $\{u' \in \mathcal{A}^* \mid u \Rightarrow^*_G u'\} \cdot a \cdot L_y(G) \cdot b \cdot L_z(G) \cdot \{v' \in \mathcal{A}^* \mid v \Rightarrow^*_G v'\} \subseteq L_X(G) \subseteq \widetilde{\mathbf{b}}$, and thus $L_y(G) \cdot L_z(G) \subseteq \widetilde{\mathbf{b}}$. We consider only the case $y, z \in \Xi_{\underparen{1..d}}$—the cases $y = \varepsilon$ or $z = \varepsilon$ use similar arguments, and are left as an easy exercise. Hence, our proof falls into 4 cases:
   (*a*) $L_y(G_{i,a\,y\,b\,z}) \cap (a_1 \cdot \mathcal{A}^*) = \emptyset$ and $L_z(G_{i,a\,y\,b\,z}) \cap (a_1 \cdot \mathcal{A}^*) = \emptyset$. Thus $L_y(G_{i,a\,y\,b\,z}) \cdot L_z(G_{i,a\,y\,b\,z}) \subseteq a_2^* \ldots a_d^*$. Then fact (v) for $\ell = 2$ and $r = d$ concludes this case.
   (*b*) $L_y(G_{i,a\,y\,b\,z}) \cap (\mathcal{A}^* \cdot a_d) = \emptyset$ and $L_z(G_{i,a\,y\,b\,z}) \cap (\mathcal{A}^* \cdot a_d) = \emptyset$. Thus $L_y(G_{i,a\,y\,b\,z}) \cdot L_z(G_{i,a\,y\,b\,z}) \subseteq a_1^* \ldots a_{d-1}^*$. Then fact (v) for $\ell = 1$ and $r = d-1$ concludes this case.

(c) $L_y(G_{i,a\,y\,b\,z}) \cap (\mathcal{A}^* \cdot a_d) = \varnothing$ and $L_z(G_{i,a\,y\,b\,z}) \cap (a_1 \cdot \mathcal{A}^*) = \varnothing$. Thus we have $L_y(G_{i,a\,y\,b\,z}) \subseteq a_1^* \ldots a_{d-1}^*$ and $L_z(G_{i,a\,y\,b\,z}) \subseteq a_2^* \ldots a_d^*$. By the fact (v) (with $\ell = 1$, $r = d$) there exists q, $1 \leqslant q \leqslant d$ such that $L_y(G_{i,a\,y\,b\,z}) \subseteq a_1^* \ldots a_q^*$ and $L_z(G_{i,a\,y\,b\,z}) \subseteq a_q^* \ldots a_d^*$. Next we show $1 < q < d$ holds. In fact, assume the inclusions hold for $q = 1$. Then they also hold for $q = 2$ since $L_z(G_{i,a\,y\,b\,z}) \subseteq a_2^* \ldots a_d^*$. A similar reasoning holds when $q = d$ since $L_y(G_{i,a\,y\,b\,z}) \subseteq a_1^* \ldots a_{d-1}^*$.

(d) $L_y(G_{i,a\,y\,b\,z}) \cap (a_1 \cdot \mathcal{A}^*) = \varnothing$ and $L_z(G_{i,a\,y\,b\,z}) \cap (\mathcal{A}^* \cdot a_d) = \varnothing$. We first observe that it cannot be the case that $L_y(G_{i,a\,y\,b\,z})$ contains some word where a_d occurs and $L_z(G_{i,a\,y\,b\,z})$ contains some word where a_1 occurs for otherwise concatenating those two words shows $L_y(G_{i,a\,y\,b\,z}) \cdot L_z(G_{i,a\,y\,b\,z}) \nsubseteq a_1^* \ldots a_d^*$. This leaves us with three cases: (a) If $L_y(G_{i,a\,y\,b\,z}) \cap (\mathcal{A}^* \cdot a_d) \neq \varnothing$ we find that $L_z(G_{i,a\,y\,b\,z}) \subseteq a_d^*$, hence that $L_y(G_{i,a\,y\,b\,z}) \subseteq a_2^* \ldots a_d^*$ since $L_y(G_{i,a\,y\,b\,z}) \cap (a_1 \cdot \mathcal{A}^*) = \varnothing$. (b) If $L_z(G_{i,a\,y\,b\,z}) \cap (a_1 \cdot \mathcal{A}^*) \neq \varnothing$ we find that $L_y(G_{i,a\,y\,b\,z}) \subseteq a_1^*$, hence that $L_z(G_{i,a\,y\,b\,z}) \subseteq a_1^* \ldots a_{d-1}^*$ since $L_z(G_{i,a\,y\,b\,z}) \cap (\mathcal{A}^* \cdot a_d) = \varnothing$. (c) Then $L_y(G_{i,a\,y\,b\,z}) \cap (\mathcal{A}^* \cdot a_d) = \varnothing$ and $L_z(G_{i,a\,y\,b\,z}) \cap (a_1 \cdot \mathcal{A}^*) = \varnothing$. Hence $L_y(G_{i,a\,y\,b\,z}) \cdot L_z(G_{i,a\,y\,b\,z}) \subseteq a_2^* \ldots a_{d-1}^*$ and by the fact (v) for $\ell = 2$ and $r = d-1$ there exists $1 < q < d$ such that $L_y(G_{i,a\,y\,b\,z}) \subseteq a_2^* \ldots a_q^*$ and $L_z(G_{i,a\,y\,b\,z}) \subseteq a_q^* \ldots a_{d-1}^*$. $\square$

B.4 Proof of Theorem 3

Proof (of Theorem 3). We prove the theorem by induction on $d > 0$. If $d = 1, 2$, we obtain $\Gamma_{\tilde{\mathbf{b}}}$ from Lemma 2, and time needed to compute $\Gamma_{\tilde{\mathbf{b}}}$, using Algorithm 1, is $|G|^{\mathcal{O}(k)}$. Moreover, we have $L_X^{(k)}(G) = \hat{L}_X(\Gamma_{\tilde{\mathbf{b}}} \cap \Gamma_X^{\mathtt{df}(k)}, G) \subseteq \hat{L}_X(\Gamma_{\tilde{\mathbf{b}}} \cap \Gamma_X^{\mathtt{df}(k+1)}, G)$.

For the induction step, assume $d \geqslant 3$. W.l.o.g. we assume that G is reduced for X, and that $a_1^* \ldots a_d^*$ is the minimal bounded expression such that $L_X(G) \subseteq a_1^* \ldots a_d^*$. Consider the partition $\Xi_{\widehat{1..d}} \cup \Xi_{\widetilde{1..d}} = \Xi$ and $\Xi_{\widehat{1..d}} \cap \Xi_{\widetilde{1..d}} = \varnothing$, defined in the previous. Since G is reduced for X, then $X \in \Xi_{\widehat{1..d}}$. Define

$$\Delta_{pivot} = \{(X_i, a\,y\,b\,z) \in \Delta \mid X_i \in \Xi_{\widehat{1..d}} \text{ and } a, b \in \mathcal{A} \cup \{\varepsilon\},\ y, z \in \Xi_{\widetilde{1..d}} \cup \{\varepsilon\}\} \ .$$

By Lemma 2, for each $X_i \in \Xi$, such that $L_{X,X_i}(G) \subseteq a_1^* a_d^*$, there exists a bounded expression $\Gamma_{1,d}^{X,X_i}$ such that $L_{X,X_i}^{(k+1)} = \hat{L}_{X,X_i}(\Gamma_{1,d}^{X,X_i} \cap \Gamma_{X,X_i}^{\mathtt{df}(k+1)}, G)$. Moreover, by the induction hypothesis, for each ℓ, m, r such that $1 \leqslant \ell \leqslant m \leqslant r \leqslant d$, $m - \ell < d - 1$ and $r - m < d - 1$, and for each $Y, Z \in \Xi$ such that $L_Y(G) \subseteq a_\ell^* \ldots a_m^*$ and $L_Z(G) \subseteq a_m^* \ldots a_r^*$, there exist two sets $\mathcal{S}_{\ell\ldots m}^Y, \mathcal{S}_{m\ldots r}^Z$ of bounded expressions over $\Delta_{i,aybz}$ such that $L_Y^{(k)}(G) \subseteq \hat{L}_Y(\bigcup \mathcal{S}_{\ell\ldots m}^Y \cap \Gamma_Y^{\mathtt{df}(k+1)}, G)$ and $L_Z^{(k)}(G) \subseteq \hat{L}_Z(\bigcup \mathcal{S}_{m\ldots r}^Z \cap \Gamma_Z^{\mathtt{df}(k+1)}, G)$. We extend this notation to ε, and assume that $\mathcal{S}_{i\ldots j}^\varepsilon = \{\varepsilon\}$. We define:

$$\mathrm{IH} = \{(\ell, m, r) \mid 1 \leqslant \ell \leqslant m \leqslant r \leqslant d,\ m - \ell < d - 1 \wedge r - m < d - 1\}$$

$$\begin{aligned}\mathcal{S}_{\tilde{\mathbf{b}}} = \{\Gamma_{1,d}^{X,X_i} \cdot (X_i, a\,y\,b\,z)^* \cdot \Gamma' \cdot \Gamma'' \mid (X_i, a\,y\,b\,z) \in \Delta_{pivot} \wedge \\ L_{X,X_i}(G) \subseteq a_1^* a_d^* \wedge \Gamma' \in \mathcal{S}_{\ell\ldots m}^y \wedge \Gamma'' \in \mathcal{S}_{m\ldots r}^z \wedge (\ell, m, r) \in IH\}\end{aligned}$$

First, let us prove that $L_X^{(k)}(G) \subseteq \hat{L}_X(\bigcup \mathcal{S}_{\tilde{\mathbf{b}}} \cap \Gamma_X^{\mathbf{df}(k+1)}, G)$. Let $w \in L_X^{(k)}(G)$ be a word, and $X \xRightarrow[\mathbf{df}(k)]{\gamma} w$ be a k-index depth first derivation of w in G. Since $w \in L_X^{(k)}(G)$, such a derivation is guaranteed to exist. By Lemma 3, there exists $(X_i, a\,y\,b\,z) \in \Delta_{pivot}$, and $\gamma^\sharp \in (\Delta^\sharp)^*$, $\gamma_y, \gamma_z \in (\Delta_{i,aybz})^*$, such that $\gamma^\sharp \cdot (X_i, a\,y\,b\,z) \cdot \gamma_y \cdot \gamma_z$ is a permutation of γ, and:

- $X \xRightarrow[\mathbf{df}(k+1)]{\gamma^\sharp} u\,X_i\,v$ is a step sequence of $G^\sharp$ with $u, v \in \mathcal{A}^*$;
- $y \xRightarrow[\mathbf{df}(k_y)]{\gamma_y} u_y$ and $z \xRightarrow[\mathbf{df}(k_z)]{\gamma_z} u_z$ are derivations of $G_{i,aybz}$ (hence $u_y\,u_z \in \mathcal{A}^*$), $\max(k_y, k_z) \leqslant k$ and $\min(k_y, k_z) \leqslant k-1$;
- $X \xRightarrow[\mathbf{df}(k+1)]{\gamma^\sharp \cdot (X_i, aybz) \cdot \gamma_y \cdot \gamma_z} w$ is a derivation of $G^\sharp$ if $y \xRightarrow[\mathbf{df}(k-1)]{\gamma_y} u_y$ is a derivation of $G_{i,aybz}$;
- $X \xRightarrow[\mathbf{df}(k+1)]{\gamma^\sharp \cdot (X_i, aybz) \cdot \gamma_z \cdot \gamma_y} w$ is a derivation of $G^\sharp$ if $z \xRightarrow[\mathbf{df}(k-1)]{\gamma_z} u_z$ is a derivation of $G_{i,aybz}$;
- $L_{X,X_i}(G^\sharp) \subseteq a_1^* a_d^*$;
- $L_y(G_{i,aybz}) \subseteq a_\ell^* \ldots a_m^*$ if $y \in \Xi_{\widetilde{1..d}}$; $L_z(G_{i,aybz}) \subseteq a_m^* \ldots a_r^*$ if $z \in \Xi_{\widetilde{1..d}}$, with $1 \leqslant \ell \leqslant m \leqslant r \leqslant d$, such that $m - \ell < d - 1$ and $r - m < d - 1$.

Let us consider the case where $y, z \in \Xi$ (the other cases of $y = \varepsilon$ or $z = \varepsilon$ being similar, are left to the reader). We also assume $k_y \leqslant k - 1$ the other case being symmetric.

Therefore, by the induction hypothesis there exist bounded expressions $\Gamma' \in \mathcal{S}^y_{\ell\ldots m}$ and $\Gamma'' \in \mathcal{S}^z_{m\ldots r}$ such that $y \xRightarrow[\mathbf{df}(k_y+1)]{\gamma'} u_y$ and $z \xRightarrow[\mathbf{df}(k_z+1)]{\gamma''} u_z$, for some control words $\gamma' \in \Gamma'$ and $\gamma'' \in \Gamma''$. If $L_{X,X_i}(G^\sharp) \subseteq a_1^* a_d^*$, by Lemma 2, there exists a control word $\gamma^\sharp \in \Gamma_{1,d}^{X,X_i}$ such that $X \xRightarrow[\mathbf{df}(k+1)]{\gamma^\sharp} u\,X_i\,v$ is a $(k+1)$-index depth first step sequence in $G^\sharp$. It follows that:

$$X \xRightarrow[\mathbf{df}(k+1)]{\gamma^\sharp} u\,X_i\,v \xRightarrow{(X_i, aybz)} u\,a\,y\,b\,z\,v \xRightarrow[\mathbf{df}(k_y+2)]{\gamma'} u\,a\,u_y\,b\,z\,v \xRightarrow[\mathbf{df}(k_z+1)]{\gamma''} u\,a\,u_y\,b\,u_z\,v = w \ .$$

Observe that $u\,a\,y\,b\,z\,v \xRightarrow[\mathbf{df}(k_y+2)]{\gamma'} u\,a\,u_y\,b\,z\,v$ because $a, b, u, v \in \mathcal{A}^*$, $z \in \Xi$ and $y \xRightarrow[\mathbf{df}(k_y+1)]{\gamma'} u_y$. Since $k_y \leqslant k-1$ and $k_z \leqslant k$, we find that $k_y + 2 \leqslant k+1$ and $k_z + 1 \leqslant k+1$, respectively. Hence the overall index of the foregoing derivation with control word $(\gamma^\sharp\,(X_i, aybz)\,\gamma'\,\gamma'')$ is at most $k+1$. Since it is also a depth-first derivation, we finally find that $w \in \hat{L}_X(\bigcup \mathcal{S}_{\tilde{\mathbf{b}}} \cap \Gamma_X^{\mathbf{df}(k+1)}, G)$, i.e. $L_X^{(k)}(G) \subseteq \hat{L}_X(\bigcup \mathcal{S}_{\tilde{\mathbf{b}}} \cap \Gamma_X^{\mathbf{df}(k+1)}, G)$.

In the following, we address the time complexity of the construction of $\mathcal{S}_{\tilde{\mathbf{b}}}$, and of each bounded expression $\Gamma \in \mathcal{S}_{\tilde{\mathbf{b}}}$. We refer to Algorithm 2 in the following. Notice first that both the minimizeExpression and partitionNonterminals functions take time $\mathcal{O}(|G|)$, because emptiness of the intersection between a context-free grammar and a finite automaton of constant size is linear in the size of the grammar [5, Section 5]. Moreover, the inclusion check on (line 12) is possible also in time $\mathcal{O}(|G|)$ (see Remark2). By Lemma 2, a call to ConstantBoundedControlSet$(G, \mathbf{b}, k)$ will take time $|G|^{\mathcal{O}(k)}$. Lemma 3 shows that the sizes of the bounded expression considered at lines 16 and 19, in a recursive call, sum up to the size of the bounded expression for the current call. Thus the total number of recursive calls is at most d. We thus let $T(d)$ denote the time needed for the top-level call of the function LetterBoundedControlSet$(G, X, a_1^* \dots a_d^*, k)$ to complete. Since the loop on (lines 11–21) will be taken at most $\|\Delta\| \leqslant |G|$ times, we obtain:

$$T(d) = |G|^{\mathcal{O}(k)} + |G|(\mathcal{O}(|G|) + 2\,T(d-1))$$

where $2\,T(d-1)$ is the time needed for the two recursive calls at lines 16 and 19 to complete. Because $T(0) = \mathcal{O}(|G|) + |G|^{\mathcal{O}(k)}$, we find that $T(d) = |G|^{\mathcal{O}(k)+d}$.

Finally, the time needed to build each bounded expression $\Gamma \in \mathcal{S}_{\widetilde{\mathbf{b}}}$ can be evaluated by observing that each such expression is uniquely determined by a sequence $\sigma \in \Delta^*$ of productions of G that are successively chosen at line 11. Let us consider now a slightly modified version of Algorithm 2 that is guided by a sequence $\sigma \in \Delta^*$ received in input — the function LetterBoundedControlSet$(G, X, a_s^* \dots a_t^*, k, \sigma)$ receives an extra parameter and returns also the suffix of σ that remains after processing the first production on σ, i.e. the recursive calls at lines 16 and 19 have returned. Since the sum of sizes of the bounded expressions for these recursive calls is at most $t - s$, by Lemma 3, we obtain that, in total, Algorithm 2 initiates at most d calls to LetterBoundedControlSet. We recall also that the prefix of each call (before making recursive calls) takes time $\mathcal{O}(|G|) + |G|^{\mathcal{O}(k)}$. Since $L_X(G) \subseteq \widetilde{\mathbf{b}}$, assuming that $\widetilde{\mathbf{b}}$ is minimal, we have $|\widetilde{\mathbf{b}}| \leqslant |G|$. Hence, the time needed to compute a bounded expression $\Gamma \in \mathcal{S}_{\widetilde{\mathbf{b}}}$ is bounded by:

$$d \cdot (\mathcal{O}(|G|) + |G|^{\mathcal{O}(k)}) \leqslant |G| \cdot (\mathcal{O}(|G|) + |G|^{\mathcal{O}(k)}) = |G|^{\mathcal{O}(k)} \ .$$

□

B.5 Proof of Lemma 4

Proof (of Lemma 4). Given $k > 0$, consider the following grammar:

$$G = \langle \{X_i \mid 0 \leqslant i \leqslant k\}, \{a\}, \{X_i \to X_{i-1}\, X_{i-1} \mid 1 \leqslant i \leqslant k\} \cup \{X_0 \to a\} \rangle \ .$$

Notice that $L_{X_k}(G) = \{a^{2^k}\} \subseteq a^*$ and $|G| = \mathcal{O}(k)$. Moreover, every depth-first derivation of G has index $k + 1$.

For each $i \in \{1, \dots, n\}$, let p_i be the production $X_i \to X_{i-1}\, X_{i-1}$ of G_n, and let p_0 be $X_0 \to a$. It is easy to see that, because the derivation is depth-first, the control word γ generating a^{2^k} from X_k is unique. Now suppose that there exists $\Gamma = w_1^* \dots w_d^*$ such that $\gamma = w_1^{i_1} \dots w_d^{i_d}$, for some $i_1, \dots, i_d \geqslant 0$. Next we show that, for all $j = 1, \dots, d$ we must have $i_j \leqslant 2$.

We first make this crucial observation, since the derivation tree is binary and its traversal is depth-first, we have that for every p_i, every three consecutive occurrences $\ell_1 < \ell_2 < \ell_3$ of p_i—$(\gamma)_{\ell_1} = (\gamma)_{\ell_2} = (\gamma)_{\ell_3} = p_i$—implies that there exists a position ℓ between ℓ_1 and ℓ_3 such that $(\gamma)_\ell = p_{i+1}$. Otherwise that would imply that the derivation tree has a node X_{i+1} with three X_i children; or that the tree was not traversed in depth-first.

Take an arbitrary w_j in Γ and let g be the greatest index of a production occurring in w_j. The number i_j of repetitions of w_j cannot be greater than two for otherwise p_g contradicts the previous fact. So this concludes that no i_j can be larger than 2.

Now, since the only string of $L_{X^k}(G)$ has length 2^k and that no rule produces more than one terminal then necessarily $|\gamma| \geqslant 2^k$. So we show that $|\Gamma|$ has to be at least 2^{k-1}. By contradiction, suppose $|\Gamma| \leqslant (2^{k-1}-1)$, then since in order to capture γ no word of Γ can occur more than twice, the longest control word that Γ can capture is $2 \cdot (2^{k-1}-1) = 2^k - 2$ which is shorter than $2^k = |\gamma|$, hence a contradiction. □

B.6 Proof of Theorem 2

Proof (of Theorem 2). The NP-hard lower bound is by reduction from the Positive Integer Linear Programming (PILP) problem, which is known to be NP-complete [26, Corollary 18.1a]. Consider the following instance of PILP, with variables $k_1, \ldots, k_m$ ranging over positive integers:

$$\begin{cases} a_{11} \cdot k_1 + \ldots + a_{m1} \cdot k_m + c_1 & \leqslant 0 \\ & \cdots \\ a_{1n} \cdot k_1 + \ldots + a_{mn} \cdot k_m + c_n & \leqslant 0 \end{cases}$$

and denote $\mathbf{a}_i = \langle a_{i1}, \ldots, a_{in} \rangle \in \mathbb{Z}^n$, for all $i = 1, \ldots, m$, and $\mathbf{c} = \langle c_1, \ldots, c_n \rangle \in \mathbb{Z}^n$. Let $\mathbf{x} = \{x_1, \ldots, x_n\}$ be a set of integer variables. Consider the program $\mathcal{P}_{\mathrm{PILP}} = \langle G, X_0, [.] \rangle$, where $G = \langle \Xi, \Sigma, \Delta \rangle$:

- $\Xi = \{X_0, \ldots, X_{m+1}\}$,
- $\Sigma = \{\tau_i \mid i = 0, \ldots, m+1\} \cup \{\lambda_i \mid i = 0, \ldots, m\}$,
- $\Delta = \{X_i \to \tau_i\, X_{i+1} \mid i = 0, \ldots, m\} \cup \{X_i \to \lambda_i\, X_i \mid i = 1, \ldots, m\} \cup \{X_{m+1} \to \tau_{m+1}\}$,
- the semantics of the words $w \in L_{X_0}(G)$ is defined by the following relations:

$$\begin{array}{lll} \rho_{\tau_0} & \equiv \mathbf{x}' = 0 & \\ \rho_{\tau_i} & \equiv \mathbf{x}' = \mathbf{x} & \text{for all } i = 1, \ldots, m-1 \\ \rho_{\lambda_i} & \equiv \mathbf{x}' = \mathbf{x} + \mathbf{a}_i & \text{for all } i = 1, \ldots, m \\ \rho_{\tau_m} & \equiv \mathbf{x}' = \mathbf{x} + \mathbf{c} & \\ \rho_{\tau_{m+1}} & \equiv \mathbf{x} \leqslant \mathbf{0} & \end{array}$$

Let $\widetilde{\mathbf{b}}_{\mathrm{PILP}} = \tau_0^* \lambda_1^* \tau_1^* \ldots \lambda_m^* \tau_m^* \tau_{m+1}^*$ be a bounded expression. It is immediate to check that the PILP problem has a solution if and only if $\mathrm{REACH}_{fo}(\mathcal{P}_{\mathrm{PILP}}, \widetilde{\mathbf{b}}_{\mathrm{PILP}})$ holds. This settles the NP-hard lower bound for the class of fo-reachability problems.

We show next that the class of fo-reachability problems $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ is included in NEXPTIME. Let $\mathcal{P} = \langle G, I, [.] \rangle$ be a given program, where $G =$

$\langle \Xi, \Sigma, \Delta\rangle$ is its underlying grammar, and $\mathbf{b} = w_1^* \ldots w_d^*$ a bounded expression. By Lemma 5, there exists a grammar $G^\cap = \langle \Xi^\cap, \Sigma, \Delta^\cap\rangle$ such that:

$$\bigcup_{1 \leqslant s \leqslant x \leqslant d} L_{[\mathrm{Q}_1^{(s)} I \mathrm{Q}_1^{(x)}]}(G^\cap) = L_I(G) \cap \mathbf{b} \ .$$

Moreover, we have that $|G^\cap| = \mathcal{O}(|\mathbf{b}|^3 \cdot |G|)$. Let $\mathcal{P}_{s,x} = \langle G^\cap, [\mathrm{Q}_1^{(s)} I \mathrm{Q}_1^{(x)}], [.]\rangle$ be a program, for each $1 \leqslant s \leqslant x \leqslant d$. Since the alphabets of G and $G^\cap$ coincide, the mapping of symbols to octagonal relations is the same for G and $G^\cap$, hence:

$$\bigcup_{1 \leqslant s \leqslant x \leqslant d} [\mathcal{P}_{s,x}] = [\mathcal{P}]_{\mathbf{b}} \ .$$

Then $[P]_{\mathbf{b}} \neq \varnothing$ if and only if $[\mathcal{P}_{s,x}] \neq \varnothing$, for some $1 \leqslant s \leqslant x \leqslant d$. We have reduced the original problem $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ to $\mathcal{O}(|\mathbf{b}|^2)$ reachability problems, of size $\mathcal{O}(|\mathbf{b}|^3 \cdot |G|)$ each. In the following we fix $1 \leqslant s \leqslant x \leqslant d$, focus w.l.o.g on the problem $\mathrm{REACH}_{fo}(\mathcal{P}_{s,x}, \mathbf{b})$ and we denote by $X = [\mathrm{Q}_1^{(s)} I \mathrm{Q}_1^{(x)}]$ in the rest of this proof.

Let $\mathcal{A} = \{a_1, \ldots, a_d\}$ be an alphabet disjoint from Σ and $\widetilde{\mathbf{b}} = a_1^* \ldots a_d^*$ be a strict letter-bounded expression, such that $\mathbf{b} = h(\widetilde{\mathbf{b}})$, where $h : \mathcal{A} \to \Sigma^*$ is the homomorphism $h(a_i) = w_i$, for all $i = 1, \ldots, d$. By Lemma 6 there exists a grammar $G^{\bowtie} = \langle \Xi^\cap, \mathcal{A}, \Delta^{\bowtie}\rangle$ such that, for every $k > 0$:

1. $L_X^{(k)}(G^{\bowtie}) = h^{-1}(L_X^{(k)}(G^\cap)) \cap \widetilde{\mathbf{b}}$,
2. for each $\Gamma \subseteq (\Delta^{\bowtie})^*$, such that $L_X^{(k)}(G^{\bowtie}) \subseteq \hat{L}_X(\Gamma, G^{\bowtie})$, we have $L_X^{(k)}(G^\cap) \subseteq \hat{L}_X(\iota^{-1}(\Gamma), G^\cap)$.

Moreover, we have $|G^{\bowtie}| = \mathcal{O}(|\mathbf{b}|^3 \cdot |G|)$. Since $L_X^{(k)}(G^{\bowtie}) \subseteq \widetilde{\mathbf{b}}$, by Theorem 3, there exists a set $\mathcal{S}_{\widetilde{\mathbf{b}}}$ of bounded expressions over $\Delta^{\bowtie}$ such that:

$$L_X^{(k)}(G^{\bowtie}) \subseteq \hat{L}_X\left(\bigcup \mathcal{S}_{\widetilde{\mathbf{b}}} \cap \Gamma_X^{\mathbf{df}(k+1)}(G^{\bowtie}), G^{\bowtie}\right) \ .$$

Hence, by Lemma 6, we obtain:

$$L_X^{(k)}(G^\cap) \subseteq \hat{L}_X\left(\iota^{-1}\left(\bigcup \mathcal{S}_{\widetilde{\mathbf{b}}}\right) \cap \Gamma_X^{\mathbf{df}(k+1)}(G^\cap), G^\cap\right) \ .$$

We used the fact that $\iota^{-1}(\Gamma_X^{\mathbf{df}(k+1)}(G^{\bowtie})) = \Gamma_X^{\mathbf{df}(k+1)}(G^\cap)$. Because $L_X(G^\cap) \subseteq \mathbf{b}$, there exists $K = \mathcal{O}(|G^\cap|)$ such that $L_X(G^\cap) = L_X^{(K)}(G^\cap)$ as Theorem 1 shows. Hence $K = \mathcal{O}(|\mathbf{b}|^3 \cdot |G|)$ as well. We obtain the following:

$$L_X(G^\cap) \subseteq L_X^{(K)}(G^\cap) \subseteq \hat{L}_X\left(\iota^{-1}\left(\bigcup \mathcal{S}_{\widetilde{\mathbf{b}}}\right) \cap \Gamma_X^{\mathbf{df}(K+1)}(G^\cap), G^\cap\right) \subseteq L_X(G^\cap)$$

thus, $L_X(G^\cap) = \hat{L}_X\left(\iota^{-1}\left(\bigcup \mathcal{S}_{\widetilde{\mathbf{b}}}\right) \cap \Gamma_X^{\mathbf{df}(K+1)}(G^\cap), G^\cap\right)$. Assume that $\mathcal{S}_{\widetilde{\mathbf{b}}} = \{\Gamma_1, \ldots, \Gamma_m\}$, for some $m > 0$, and denote $\iota^{-1}(\Gamma_i)$ by $\widetilde{\Gamma}_i$. We have that, for each derivation $X \xRightarrow[\mathbf{df}(k+1)]{\gamma} w$ of $G^\cap$, $[w] = \varnothing$ iff $[\gamma] = \varnothing$ [11, Lemma 2]. As a result, $[\mathcal{P}_{s,x}] \neq \varnothing$

iff there exists $i = 1, \ldots, m$ and $\gamma \in \widetilde{\Gamma}_i \cap \Gamma_X^{\mathbf{df}(k+1)}(G^\cap)$, such that $[\gamma] \neq \emptyset$. By Theorem 3, each set Γ_i can be constructed in time:

$$|G^{\bowtie}|^{\mathcal{O}(K)} = (|\mathbf{b}|^3 \cdot |G|)^{\mathcal{O}(K)} = (|\mathbf{b}|^3 \cdot |G|)^{\mathcal{O}(|\mathbf{b}|^3 \cdot |G|)} = 2^{\mathcal{O}(|\mathbf{b}|^3 \cdot |G| \cdot (\log|\mathbf{b}| + \log|G|))} \ .$$

We have used the facts $|G^{\bowtie}| = \mathcal{O}(|\mathbf{b}|^3 \cdot |G|)$ and $K = \mathcal{O}(|\mathbf{b}|^3 \cdot |G|)$.

By Lemma 1, there exists a finite automaton $A_{G^\cap}^{\mathbf{df}(K+1)}$ that recognizes the language $\Gamma_X^{\mathbf{df}(K+1)}(G^\cap)$. Equivalently, we consider a grammar $\mathcal{G}^{\mathbf{df}(K+1)}$, such that $L_{X^{\langle 0 \rangle}}(\mathcal{G}^{\mathbf{df}(K+1)}) = \Gamma_X^{\mathbf{df}(K+1)}(G^\cap)$, where $X^{\langle 0 \rangle}$ is the ranked nonterminal corresponding to the initial state of $A_{G^\cap}^{\mathbf{df}(K+1)}$ in Lemma 1. Let $\mathcal{Q} = \langle \mathcal{G}^{\mathbf{df}(K+1)}, X^{\langle 0 \rangle}, [.] \rangle$ be the program associated with $\mathcal{G}^{\mathbf{df}(K+1)}$. If $\mathcal{P}$ was assumed to be an octagonal program, then so is $\mathcal{Q}$.

The problem $\mathrm{REACH}_{fo}(\mathcal{P}_{s,x}, \mathbf{b})$ is thus equivalent to the finite set of problems $\mathrm{REACH}_{fo}(\mathcal{Q}, \widetilde{\Gamma}_i)$, for $i = 1, \ldots, m$. The size of $\mathcal{G}^{\mathbf{df}(K+1)}$ is

$$|\mathcal{G}^{\mathbf{df}(K+1)}| = |G^\cap|^{\mathcal{O}(K)} = (|\mathbf{b}|^3 \cdot |G|)^{\mathcal{O}(K)} = 2^{\mathcal{O}(|\mathbf{b}|^3 \cdot |G| \cdot (\log|\mathbf{b}| + \log|G|))} \ .$$

Hence the size of the input to each problem $\mathrm{REACH}_{fo}(\mathcal{Q}, \widetilde{\Gamma}_i)$ is $2^{\mathcal{O}(|\mathbf{b}|^3 \cdot |G| \cdot (\log|\mathbf{b}| + \log|G|))}$. Since $\mathcal{Q}$ is a procedure-less octagonal program, and each such problem can be solved in NPTIME [7, Theorem 10], this provides a NEXPTIME decision procedure for the problem $\mathrm{REACH}_{fo}(\mathcal{P}_{s,x}, \mathbf{b})$.

We are left with proving that the $\mathrm{REACH}_{fo}(\mathcal{P}, \mathbf{b})$ problem is in NP, when $[P] = [P]^{(k)}$, for a constant $k > 0$. To this end, we define a grammar $G_k = \langle \Xi \times \{0, \bar{0}, \ldots, k, \bar{k}\}, \Sigma, \Delta_k \rangle$ such that $L_X(G)^{(k)} = L_{(X,k)}(G_k)$ [22, Definition 3.1]. Using the fact that, for each production $(Z, w) \in \Delta$, there are at most two nonterminals in w, we establish that $|G_k| \leqslant 3k|G| + k(k+1)$, hence $|G_k| = \mathcal{O}(k^2 \cdot |G|)$.

The corresponding program is $\mathcal{P}_k = \langle G_k, (I, k), [.] \rangle$. By applying the reduction above, we obtain a set of problems $\mathrm{REACH}_{fo}(\mathcal{Q}_k, \widetilde{\Gamma}_i)$, each of which of size $(|\mathbf{b}|^3 \cdot |G_k|)^{\mathcal{O}(k)} = (|\mathbf{b}|^3 \cdot (k^2 \cdot |G|))^{\mathcal{O}(k)}$. Since k is constant, we can solve this problem in NPTIME, using an NP procedure [7, Theorem 10]. Since the NP-hard lower bound was proved above, the problem is NP-complete. $\square$