Intersection and Union Hierarchies of Deterministic Context-Free Languages and Pumping Lemmas

Tomoyuki Yamakami(✉)

Faculty of Engineering, University of Fukui, 3-9-1 Bunkyo, Fukui 910-8507, Japan
TomoyukiYamakami@gmail.com

Abstract. We study the computational complexity of finite intersections and unions of deterministic context-free languages. Earlier, Wotschke (1978) demonstrated that intersections of $(d+1)$ deterministic context-free languages are in general more powerful than intersections of d deterministic context-free languages for any positive integer d based on the hierarchy separation of Liu and Weiner (1973). The argument of Liu and Weiner, however, works only on bounded languages of particular forms, and therefore Wotschke's result cannot be extended to disprove any other language to be written in the form of an intersection of d deterministic context-free languages. To deal with the non-membership of a wide range of languages, we circumvent their proof argument and instead devise a new, practical technical tool: a pumping lemma for finite unions of deterministic context-free languages. Since the family of deterministic context-free languages is closed under complementation, this pumping lemma enables us to show a non-membership relation of languages made up with finite intersections of even non-bounded languages as well. We also refer to a relationship to Hibbard's limited automata.

Keywords: Deterministic pushdown automata · Intersection and union hierarchies · Pumping lemma · Limited automata

1 A Historical Account and an Overview of Contributions

1.1 Intersection and Union Hierarchies and Historical Background

In formal language theory, context-free languages constitute a fundamental family CFL, which is situated in between the family REG of regular languages and that of context-sensitive languages. It has been well known that this family CFL is closed under an operation of union but not closed under intersection. As a quick example, the language $L_{abc} = \{a^n b^n c^n \mid n \geq 0\}$ is not context-free but it can be expressed as an intersection of two context-free languages. This non-closure property can be further generalized to any intersection of d (≥ 1) context-free languages. For later notational convenience, we here write CFL(d) for the family of such languages, namely, the d intersection closure of CFL (see,

A. Leporati et al. (Eds.): LATA 2020, LNCS 12038, pp. 341–353, 2020.
https://doi.org/10.1007/978-3-030-40608-0_24

e.g., [13]). With this notation, the above language L_{abc} belongs to CFL(2)−CFL. Similarly, the language $L_d = \{a_1^{n_1}a_2^{n_2}\cdots a_d^{n_d}b_1^{n_1}b_2^{n_2}\cdots b_d^{n_d} \mid n_1, n_2, \ldots, n_d \geq 0\}$ over an alphabet $\{a_1, a_2, \ldots, a_d, b_1, b_2, \ldots, b_d\}$ falls into CFL(d) because L_d can be expressed as an intersection of d context-free languages of the form $\{a_1^{n_1}a_2^{n_2}\cdots a_d^{n_d}b_1^{m_1}b_2^{m_2}\cdots b_d^{m_d} \mid n_1, n_2, \ldots, n_d, m_1, m_2, \ldots, m_d \geq 0, n_k = m_k\}$ ($1 \leq k \leq d$). In 1973, Liu and Weiner [8] gave a contrived proof to their key statement that (*) L_d *is outside of* CFL($d-1$) *for any index* $d \geq 2$. Therefore, the collection $\{\mathrm{CFL}(d) \mid d \geq 1\}$ truly forms an infinite hierarchy.

Deterministic context-free (dcf) languages have been a focal point in CFL since a systematic study of Ginsburg and Greibach [1]. The importance of such languages can be exemplified by the facts that dcf languages are easy to parse and that every context-free language is simply the homomorphic image of a dcf language. Unlike CFL, the family DCFL of dcf languages is closed under neither union nor intersection. We use the terms of *d-intersection deterministic context-free (dcf) languages* and *d-union deterministic context-free (dcf) languages* to express intersections of d dcf languages and unions of d dcf languages, respectively. For brevity, we write DCFL(d) and DCFL[d] respectively for the family of all d-intersection dcf languages and that of all d-union dcf languages, while Wotschke [11,12] earlier referred DCFL(d) to the d-intersection closure of DCFL. In particular, we obtain DCFL(1) = DCFL[1] = DCFL. Since DCFL is closed under complementation, it follows that the complement of DCFL(d) coincides with DCFL[d]. For our convenience, we call two hierarchies $\{\mathrm{DCFL}(d) \mid d \geq 1\}$ and $\{\mathrm{DCFL}[d] \mid d \geq 1\}$ the *intersection and union hierarchies of dcf languages*, respectively. Concerning these hierarchies, we set DCFL(ω) to be the intersection closure of DCFL, which is $\bigcup_{d\geq 1}\mathrm{DCFL}(d)$. In a similar way, we write DCFL[ω] for the union closure of DCFL, that is, $\bigcup_{d\geq 1}\mathrm{DCFL}[d]$.

Wotschke [11,12] noted that the aforementioned result (*) of Liu and Weiner leads to the conclusion that $\{\mathrm{DCFL}[d] \mid d \geq 1\}$ truly forms an infinite hierarchy. To be more precise, since the language L_d belongs to DCFL(d), the statement (*) implies $\mathrm{DCFL}(d) \not\subseteq \mathrm{CFL}(d-1)$, which instantly yields $\mathrm{DCFL}(d-1) \neq \mathrm{DCFL}(d)$. Wotschke's argument, nonetheless, heavily relies on the separation result of Liu and Weiner, who employed a notion of *stratified semi-linear set* to prove the statement (*). Notice that the proof technique of Liu and Weiner was developed only for a particular form of *bounded languages*[1] and it is therefore applicable to specific languages, such as L_d. In fact, the key idea of the proof of Liu and Weiner for L_d is to focus on the number of the occurrences of each base symbol in $\{a_1, \ldots, a_d, b_1, \ldots, b_d\}$ appearing in each given string w and to translate L_d into a set $\Psi(L_d)$ of *Parikh images* $(\#_{a_1}(w), \#_{a_2}(w), \ldots, \#_{a_d}(w), \#_{b_1}, \ldots, \#_{b_d}(w))$ in order to exploit the semi-linearity of $\Psi(L_d)$, where $\#_\sigma(w)$ expresses the total number of symbols σ in a string w.

Because of the aforementioned limitation of Liu and Weiner's proof technique, the scope of their proof cannot be extended to other forms of languages. Simple examples of such languages include $L_d^{(\leq)} = \{a_1^{n_1}\cdots a_d^{n_d}b_1^{m_1}\cdots b_d^{m_d} \mid \forall i \in [d](n_i \leq m_i)\}$, where $[d]$ denotes the set $\{1, 2, \ldots, d\}$. This is a bounded

[1] A bounded language satisfies $L \subseteq w_1^*w_2^*\cdots w_k^*$ for fixed strings $w_1, w_2, \ldots, w_k$.

language expanding L_d but its Parikh images do not have semi-linearity. As another example, let us take a look at a "non-palindrome" language $NPal_d^{\#} = \{w_1\#w_2\#\cdots\#w_d\#v_1\#v_2\#\cdots\#v_d \mid \forall i \in [d](w_i, v_i \in \{0,1\}^* \wedge v_i \neq w_i^R)\}$, where w_i^R expresses the *reversal* of w_i. This $NPal_d^{\#}$ is not even a bounded language. Therefore, Liu and Weiner's argument is not directly applicable to verify that neither $L_d^{(\leq)}$ nor $NPal_d^{\#}$ belongs to CFL$(d-1)$ unless we dextrously pick up its core strings that form a certain bounded language. With no such contrived argument, how can we prove $L_d^{(\leq)}$ and $NPal_d^{\#}$ to be outside of DCFL(d)? Moreover, given a language, how can we verify that it is not in DCFL(ω)? We can ask similar questions for d-union dcf languages and the union hierarchy of dcf languages. Ginsburg and Greibach [1] remarked *with no proof* that the context-free language $Pal = \{ww^R \mid w \in \Sigma^*\}$ for any non-unary alphabet Σ is not in DCFL$[\omega]$. It is natural to call for a formal proof of the remark of Ginsburg and Greibach. Using a quite different language $L_{wot} = \{wcx \mid w, x \in \{a,b\}^*, w \neq x\}$, however, Wotschke [11,12] actually proved that L_{wot} does not belong to DCFL(ω) (more strongly, the Boolean closure of DCFL) by employing the closure property of DCFL(d) under inverse gsm mappings as well as complementation and intersection with regular languages. Wotschke's proof relies on the following two facts. (i) The language L_{d+1} can be expressed as the inverse gsm map of the language $Dup_c = \{wcw \mid w \in \{a,b\}^*\}$, restricted to $a_1^+a_2^+\cdots a_{d+1}^+a_1^+a_2^+\cdots a_{d+1}^+$. (ii) Dup_c is expressed as the complement of L_{wot}, restricted to a certain regular language. Together with these facts, the final conclusion comes from the aforementioned result (*) of Liu and Weiner because $Dup_c \in$ DCFL(d) implies $L_{d+1} \in$ DCFL(d) by (i) and (ii). To our surprise, the fundamental results on DCFL(d) that we have discussed so far are merely "corollaries" of the main result (*) of Liu and Weiner!

For further study on DCFL(d) and answering more general non-membership questions to DCFL(d), we need to divert from Liu and Weiner's contrived argument targeting the statement (*) and to develop a completely different, new, more practical technical tool. The sole purpose of this exposition is, therefore, set to (i) develop a new proof technique, which can be applicable to many other languages, (ii) present an alternative proof for the fact that the intersection and union hierarchies of DCFL are infinite hierarchies, and (iii) present other languages in CFL that do not belong to DCFL(ω) (in part, verifying Ginsburg and Greibach's remark for the first time).

In relevance to the union hierarchy of dcf languages, there is another known extension of DCFL using a different machine model called *limited automata*,[2] which are originally invented by Hibbard [3] and later discussed extensively in, e.g., [9,14]. Of all such machines, a *d-limited deterministic automaton* (or a d-lda, for short) is a deterministic Turing machine that can rewrite each tape cell in between two endmarkers only during the first d visits (except that making a

[2] Hibbard [3] actually defined a rewriting system, called "scan-limited automata." Later, Pighizzini and Pisoni [9] re-formulated Hibbard's system as restricted linear automata.

turn of a tape head counts as double visits). We can raise a question of whether there is any relationship between the union hierarchy and d-lda's.

1.2 Overview of Main Contributions

In Sect. 1.1, we have noted that fundamental properties associated with DCFL(d) heavily rely on the single separation result (*) of Liu and Weiner. However, Liu and Weiner's technical tool that leads to their main result does not seem to withstand a wide variety of direct applications. It is thus desirable to develop a new, simple, and practical technical tool that can find numerous applications for a future study on DCFL(d) and DCFL[d]. Thus, our main contribution of this exposition is to present a simple but powerful, practical technical tool, called the *pumping lemma* of languages in DCFL[d] with $d \geq 1$, which also enriches our understanding of DCFL[d] as well as DCFL(d). Notice that there have been numerous forms of so-called pumping lemmas (or iteration theorems) for variants of context-free languages in the past literature, e.g., [2,4–7,10,15]. Our pumping lemma is a crucial addition to the list of such lemmas.

For a string x of length n and any number $i \in [n]$, $x[i]$ stands for the ith symbol of x and x^i for the i repetitions of x.

Lemma 1 (Pumping Lemma for DCFL[d]). *Let d be any positive integer and let L be any d-union dcf language over an alphabet Σ. There exist a constant $c > 0$ such that, for any $d+1$ strings $w_1, w_2, \ldots, w_{d+1} \in L$, if w_i has the form $xy^{(i)}$ for strings $x, y^{(i)} \in \Sigma^*$ with $|x| > c$ and $y^{(i)}[1] = y^{(j)}[1]$ for any pair $i, j \in [d+1]$, then there exists two distinct indices $j_1, j_2 \in [d+1]$ for which the following conditions (1)–(2) hold. Let $k \in [d+1]$.*

1. *If $k \notin \{j_1, j_2\}$, then either (a) or (b) holds.*
   (a) *There is a factorization $x = u_1u_2u_3u_4u_5$ with $|u_2u_4| \geq 1$ and $|u_2u_3u_4| \leq c$ such that $u_1u_2^iu_3u_4^iu_5y^{(k)}$ is in L for any number $i \geq 0$.*
   (b) *There are two factorizations $x = u_1u_2u_3$ and $y^{(k)} = y_1y_2y_3$ with $|u_2| \geq 1$ and $|u_2u_3| \leq c$ such that $u_1u_2^iu_3y_1y_2^iy_3$ is in L for any number $i \geq 0$.*
2. *In the case of $k \in \{j_1, j_2\}$, either (a) or (b) holds.*
   (a) *There is a factorization $x = u_1u_2u_3u_4u_5$ with $|u_2u_4| \geq 1$ and $|u_2u_3u_4| \leq c$ such that, for each $z \in \{y^{(j_1)}, y^{(j_2)}\}$, $u_1u_2^iu_3u_4^iu_5z$ is in L for any $i \geq 0$.*
   (b) *Let $x'y = xy^{(j_1)}$ and $x'\hat{y} = xy^{(j_2)}$. There are three factorizations $x' = u_1u_2u_3$, $y = y_1y_2y_3$, and $\hat{y} = z_1z_2z_3$ with $|u_2| \geq 1$ and $|u_2u_3| \leq c$ such that $u_1u_2^iu_3y_1y_2^iy_3$ and $u_1u_2^iu_3z_1z_2^iz_3$ are in L for any number $i \geq 0$.*

As a special case of $d = 1$, we obtain Yu's pumping lemma [15, Lemma 1] from Lemma 1. Since there have been few machine-based analyses to prove various pumping lemmas in the past literature, one of the important aspects of this exposition is a clear demonstration of the *first alternative proof* to Yu's pumping lemma, which is solely founded on an analysis of behaviors of 1dpda's instead of derivation trees of LR(k) grammars as in [15]. The proof of Lemma 1, in fact, exploits early results of [14] on an *ideal shape* form (Sect. 2.3) together

with a new approach of *ε-enhanced machines* by analyzing transitions of *crossing state-stack pairs* (Sect. 2.4). These notions will be explained in Sect. 2 and their basic properties will be explored therein.

Using our pumping lemma (Lemma 1), we can expand the scope of the statement (*) of Liu and Weiner [8] targeting specific bounded languages to other types of languages, including $L_d^{(\leq)}$ and $NPal_d^{\#}$ for each index $d \geq 2$.

Theorem 1. *Let $d \geq 2$ be any index.*

1. *The language $L_d^{(\leq)}$ is not in* DCFL$(d-1)$.
2. *The language $NPal_d^{\#}$ is not in* DCFL$(d-1)$.

Since Lemma 1 concerns with DCFL$[d]$, in our proof of Theorem 1, we first take the complements of the above languages, restricted to suitable regular languages, and we then apply Lemma 1 appropriately to them. The proof sketch of this theorem will be given in Sect. 3. From Theorem 1, we instantly obtain the following consequences of Wotschke [11,12].

Corollary 1. [11,12] *The intersection hierarchy of dcf languages and the union hierarchy of dcf languages are both infinite hierarchies.*

Concerning the limitation of DCFL(ω) and DCFL$[\omega]$ in recognition power, since all *unary* context-free languages are also regular languages and the family REG of regular languages is closed under intersection, all unary languages in DCFL(ω) are regular as well. It is thus easy to find languages that are not in DCFL(ω). Such languages, nevertheless, cannot serve themselves to separate CFL from DCFL$(\omega) \cup$ DCFL$[\omega]$. As noted in Sect. 1.1, Ginsburg and Greibach [1] remarked *with no proof* that the context-free language $Pal = \{ww^R \mid w \in \{0,1\}^*\}$ does not belong to DCFL(ω) (as well as DCFL$[\omega]$). As another direct application of our pumping lemma, we give a formal written proof of their remark.

Theorem 2. *The context-free language Pal is not in* DCFL$(\omega) \cup$ DCFL$[\omega]$.

As an immediate consequence of the above theorem, we obtain Wotschke's separation of DCFL(ω) from CFL. Here, we stress that, unlike the work of Wotschke [11,12], our proof does not depend on the main result (*) of Liu and Weiner.

Corollary 2. [11,12] CFL $\not\subseteq$ DCFL(ω) *and* DCFL$[\omega] \subsetneqq$ CFL.

We turn our interest to limited automata. Let us write d-LDA for the family of all languages recognized by d-limited deterministic automata, in which their tape heads are allowed to rewrite tape symbols only during the first d accesses (except that, in the case of tape heads making a turn, we treat each turn as double visits). Hibbard [3] demonstrated that d-LDA $\neq (d-1)$-LDA for any $d \geq 3$. A slightly modified language of his, which separates d-LDA from $(d-1)$-LDA, also belongs to the 2^{d-2}-th level of the union hierarchy of dcf languages but not in the $(2^{d-2}-1)$-th level. We thus obtain the following separation.

Proposition 1. *For any* $d \geq 3$, d-LDA $\cap$ DCFL$[2^{d-2}] \not\subseteq (d-1)$-LDA $\cup$ DCFL$[2^{d-2}-1]$.

The proofs of all the above assertions will be given after introducing necessary notions and notation in the subsequent section.

2 Preparations: Notions and Notation

2.1 Fundamental Notions and Notation

The set of all *natural numbers* (including 0) is denoted by $\mathbb{N}$. An *integer interval* $[m,n]_{\mathbb{Z}}$ for two integers m,n with $m \leq n$ is the set $\{m, m+1, m+2, \ldots, n\}$. In particular, for any integer $n \geq 1$, $[1,n]_{\mathbb{Z}}$ is abbreviated as $[n]$. For any string x, $|x|$ indicates the total number of symbols in x. The special symbol ε is used to denote the *empty string* of length 0. For a language L over alphabet Σ, $\overline{L}$ denotes $\Sigma^* - L$, the *complement* of L. Given a family $\mathcal{F}$ of languages, co-$\mathcal{F}$ expresses the *complement family*, which consists of languages $\overline{L}$ for any $L \in \mathcal{F}$.

2.2 Deterministic Pushdown Automata

A *one-way deterministic pushdown automaton* (or a 1dpda, for short) M is a tuple $(Q, \Sigma, \{\mathfrak{c}, \$\}, \Gamma, \delta, q_0, Z_0, Q_{acc}, Q_{rej})$, where Q is a finite set of inner states, Σ is an input alphabet with $\check{\Sigma} = \Sigma \cup \{\varepsilon, \mathfrak{c}, \$\}$, Γ is a stack alphabet, δ is a deterministic transition function from $Q \times \check{\Sigma} \times \Gamma$ to $Q \times \Gamma^*$, q_0 is the initial state in Q, Z_0 is the bottom marker in Γ, and Q_{acc} and Q_{rej} are subsets of Q. The symbols $\mathfrak{c}$ and \$ respectively express the left-endmarker and the right-endmarker. Let $\Gamma^{(-)} = \Gamma - \{Z_0\}$. We assume that, if $\delta(p, \varepsilon, a)$ is defined, then $\delta(p, \sigma, a)$ is undefined for all symbols $\sigma \in \check{\Sigma} - \{\varepsilon\}$. Moreover, we require $\delta(q, \sigma, Z_0) \neq (p, \varepsilon)$ for any $p, q \in Q$ and $\sigma \in \check{\Sigma}$. Each content of a stack is expressed as $a_1 a_2 \cdots a_k$ in which a_1 is the topmost stack symbol, a_k is the bottom marker Z_0, and all others are placed in order from the top to the bottom of the stack.

Given $d \in \mathbb{N}^+$, a *d-intersection deterministic context-free (dcf) language* is an intersection of d deterministic context-free (dcf) languages. Let DCFL(d) denote the family of all d-intersection dcf languages. Similarly, we define *d-union dcf languages* and DCFL$[d]$ by substituting "union" for "intersection" in the above definitions. Note that DCFL$[d]$ = co-(DCFL(d)) because DCFL = co-DCFL.

For two language families $\mathcal{F}_1$ and $\mathcal{F}_2$, the notation $\mathcal{F}_1 \wedge \mathcal{F}_2$ (resp., $\mathcal{F}_1 \vee \mathcal{F}_2$) denotes the family of all languages L for which there are two languages $L_1 \in \mathcal{F}_1$ and $L_2 \in \mathcal{F}_2$ over the same alphabet satisfying $L = L_1 \cap L_2$ (resp., $L = L_1 \cup L_2$).

Lemma 2. [11,12] DCFL(d) *is closed under union, intersection with* REG. *In other words,* DCFL$(d) \wedge$ REG $\subseteq$ DCFL(d) *and* DCFL$(d) \vee$ REG $\subseteq$ DCFL(d). *A similar statement holds for* DCFL$[d]$.

Lemma 3. *Let $d \geq 1$ be any natural number.*

1. DCFL(d) = DCFL$(d+1)$ *iff* DCFL$[d]$ = DCFL$[d+1]$.
2. *If* $L \in$ DCFL(d), *then it follows that* $A \cap \overline{L} \in$ DCFL$[d]$ *for any* $A \in$ REG.

From Lemma 3(1) follows Corollary 1, provided that Theorem 1 is true. Theorem 1 itself will be proven in Sect. 3.

2.3 Ideal Shape

Let us recall from [14] a special "pop-controlled form" (called an *ideal shape*), in which the pop operations always take place by first reading an input symbol and then making a series (one or more) of the pop operations without reading any further input symbol. This notion was originally introduced for *one-way probabilistic pushdown automata* (or 1ppda's); however, in this exposition, we apply this notion only to 1dpda's. To be more formal, a 1dpda *in an ideal shape* is a 1dpda restricted to take only the following transitions. (1) Scanning $\sigma \in \Sigma$, preserve the topmost stack symbol (called a *stationary operation*). (2) Scanning $\sigma \in \Sigma$, push a new symbol u ($\in \Gamma^{(-)}$) without changing any other symbol in the stack. (3) Scanning $\sigma \in \Sigma$, pop the topmost stack symbol. (4) Without scanning an input symbol (i.e., *ε-move*), pop the topmost stack symbol. (5) The stack operations (4) comes only after either (3) or (4).

It was shown in [14] that any 1ppda can be converted into its "error-equivalent" 1ppda in an ideal shape. In Lemma 4, we restate this result for 1dpda's. We say that two 1dpda's are *(computationally) equivalent* if, for any input x, their acceptance/rejection coincide. The *push size* of a 1ppda is the maximum length of any string pushed into a stack by any single move.

Lemma 4 (Ideal Shape Lemma for 1dpda's). (cf. [14]) *Let $n \in \mathbb{N}^+$. Any n-state 1dpda M with stack alphabet size m and push size e can be converted into another (computationally) equivalent 1dpda N in an ideal shape with $O(en^2m^2(2m)^{2enm})$ states and stack alphabet size $O(enm(2m)^{2enm})$.*

2.4 Boundaries and Crossing State-Stack Pairs

We want to define two basic notions of boundaries and crossing state-stacks. For this purpose, we visualize a *single move* of a 1dpda M as three consecutive actions: (i) firstly replacing the topmost stack symbol, (ii) updating an inner state, and (iii) thirdly either moving a tape head or staying still.

A *boundary* is a borderline between two consecutive tape cells. We index all such boundaries from 0 to $|¢x\$|$ as follows. The boundary 0 is located at the left of cell 0 and boundary $i+1$ is in between cell i and $i+1$ for every index $i \geq 0$. When a string xy is written in $|xy|$ consecutive cells, the *(x, y)-boundary* indicates the boundary $|x| + 1$, which separates between x and y. A *boundary block* between boundaries t_1 and t_2 with $t_1 \leq t_2$ is a consecutive series of boundaries between t_1 and t_2 (including t_1 and t_2). These t_1 and t_2 are called *ends* of this boundary block. For brevity, we write $[t_1, t_2]$ to denote a boundary block between t_1 and t_2. For two boundaries t_1, t_2 with $t_1 < t_2$, the *(t_1, t_2)-region* refers to the consecutive cells located in the boundary block $[t_1, t_2]$. When an input string x is written in the (t_1, t_2)-region, we conveniently call this region the *x-region* unless the region is unclear from the context.

The *stack height* of M at boundary t is the length of the stack content while passing the boundary t. E.g., a stack content $a_1a_2\cdots a_k$ has stack height k.

A boundary block $[t_1, t_2]$ is called *convex* if there is a boundary s between t_1 and t_2 (namely, $s \in [t_1, t_2]$) such that there is no pop operation in the (t_1, s)-region and there is no push operation in the (s, t_2)-region. A boundary block $[t_1, t_2]$ is *flat* if the stack height does not change in the (t_1, t_2)-region. A boundary block $[t_1, t_2]$ with $t_1 < t_2$ is *pseudo-convex* if the stack height at every boundary $s \in [t_1, t_2]$ does not go below $h_2 - \frac{h_1-h_2}{t_2-t_1}(s - t_1)$, where h_i is the stack height at boundary t_i for any $i \in \{1, 2\}$. By their definitions, either convex or flat boundary blocks are also pseudo-convex.

A *peak* is a boundary t such that the stack heights at the boundaries $t-1$ and $t+1$ are smaller than the stack height at the boundary t. A *plateau* is a boundary block $[t, t']$ such that any stack height at a boundary $i \in [t, t']$ is the same. A *hill* is a boundary block $[t, t']$ such that (i) the stack height at the boundary t and the stack height at the boundary t' coincide, (ii) there is at least one peak at a certain boundary $i \in [t, t']$, and (iii) both $[t, i]$ and $[i, t']$ are convex. The *height* of a hill is the difference between the topmost stack height and the lowest stack height.

Given strings over alphabet Σ, *ε-enhanced strings* are strings over the extended alphabet $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$, where ε is treated as a special input symbol expressing the absence of symbols in Σ. An *ε-enhanced 1dpda* (or an ε-1dpa, for short) is a 1dpda that takes ε-enhanced strings and works as a standard 1dpda except that a tape head always moves to the right without stopping. This tape head movement is sometimes called "real time".

Lemma 5. *For any 1dpda M, there exists an ε-1dpda N such that, for any input string x, there is an appropriate ε-enhanced string $\hat{x}$ for which M accepts (resp., rejects) x iff N accepts (resp., rejects) $\hat{x}$. Moreover, $\hat{x}$ is identical to x except for the ε symbol and is uniquely determined from x and M.*

Let M be either a 1dpda or an ε-1dpda, and assume that M is in an ideal shape. A *crossing state-stack pair* at boundary i is a pair (q, γ) of inner state q and stack content γ. In a computation of M on input x, a crossing state-stack pair (q, γ) at boundary i refers to the machine's current status where (1) M is reading an input symbol, say, σ at cell $i - 1$ in a certain state, say, p with the stack content $a\gamma'$ and then M changes its inner state to q, changing a by either pushing another symbol b satisfying $\gamma = ba\gamma'$ or popping a with $\gamma = \gamma'$. Any computation of M on x can be expressed as a series of crossing state-stack pairs at every boundary in the ¢x\$-region.

Two boundaries t_1 and t_2 with $t_1 < t_2$ are *mutually correlated* if there are two crossing state-stack pairs (q, γ) and (p, γ) at the boundaries t_1 and t_2, respectively, for which the boundary block $[t_1, t_2]$ is pseudo-convex. Moreover, assume that $t_1 < t_2 < t_3 < t_4$. Two boundary blocks $[t_1, t_2]$ and $[t_3, t_4]$ are *mutually correlated* if (i) $[t_1, t_2]$, $[t_2, t_3]$, and $[t_3, t_4]$ are all pseudo-convex, (ii) (q, γ) and $(p, \alpha\gamma)$ are crossing state-stack pairs at the boundaries t_1 and t_2, respectively, and (iii) $(s, \alpha\gamma)$ and (r, γ) are also crossing state-stack pairs at the boundaries t_3 and t_4, respectively, for certain $p, q, r, s \in Q$, $\gamma \in (\Gamma^{(-)})^* Z_0$, and $\alpha \in (\Gamma^{(-)})^*$.

If an ε-1dpda is in an ideal shape, then it pops exactly one stack symbol whenever it reads a single symbol of a given ε-enhanced input string.

Lemma 6. *Let w be any string.*

1. *Let $t_1, t_2 \in \mathbb{N}$ with $1 \leq t_1 < t_2 \leq |w| + 1$. Let $w = x_1x_2x_3$ be a factorization such that t_1 is the (x_1, x_2)-boundary and t_2 is the (x_2, x_3)-boundary. If the boundaries t_1 and t_2 are mutually correlated and inner states at the boundaries t_1 and t_2 coincide, then it follows that $w \in L$ iff $x_1x_2^ix_3 \in L$ for any $i \in \mathbb{N}$.*
2. *Let $t_1, t_2, t_3, t_4 \in \mathbb{N}$ with $1 \leq t_1 < t_2 < t_3 < t_4 \leq |w| + 1$. Let $w = x_1x_2x_3x_4x_5$ such that each t_i is (x_i, x_{i+1})-boundary for each $i \in [4]$. If two boundary blocks $[t_1, t_2]$ and $[t_3, t_4]$ are mutually correlated, inner states at the boundaries t_1 and t_2 coincide, and inner states at the boundaries t_3 and t_4 coincide, then it follows that $w \in L$ iff $x_1x_2^ix_3x_4^ix_5 \in L$ for any number $i \in \mathbb{N}$.*

3 Proof Sketches of Three Separation Claims

We intend to present the proof sketches of three separation claims (Theorems 1 and 2 and Proposition 1) before verifying the pumping lemma. To understand our proofs better, we demonstrate a simple and easy example of how to apply Lemma 1 to obtain a separation between DCFL[d] and DCFL[$d-1$].

Proposition 2. *Let $d \geq 2$ and let $L_{(d)} = \{a^nb^{kn} \mid k \in [d], n \geq 0\}$. It then follows that $L_{(d)} \in \mathrm{DCFL}[d] - \mathrm{DCFL}[d-1]$.*

Proof. Let $d \geq 2$. Clearly, $L_{(d)}$ belongs to DCFL[d]. Assuming $L_{(d)} \in \mathrm{DCFL}[d-1]$, we apply the pumping lemma (Lemma 1) to $L_{(d)}$. There is a constant $c > 0$ that satisfies the lemma. Let $n = c + 1$ and consider $w_i = a^nb^{in}$ for each index $i \in [d]$. Since each w_i belongs to $L_{(d)}$, we can take an index pair $j, k \in [d]$ with $j < k$ such that w_j and w_k satisfy the conditions of the lemma.

Since Condition (1) of the lemma is immediate, we hereafter consider Condition (2). Let $x' = a^nb^{jn-1}$, $y = b$, and $\hat{y} = b^{(k-j)n+1}$. Firstly, we consider Case (a) with a factorization $x' = x_1x_2x_3x_4x_5$ with $|x_2x_4| \geq 1$ and $|x_2x_3x_4| \leq c$. Since $x_1x_2^ix_3x_4^ix_5y \in L_{(d)}$ for any number $i \in \mathbb{N}$, we conclude that $x_2 \in \{a\}^*$ and $x_4 \in \{b\}^*$. Let $x_2 = a^m$ and $x_4 = b^r$ for certain numbers $m, r \in [c]$. Note that $x_1x_2^ix_3x_4^ix_5y$ equals $a^{n+(i-1)m}b^{jn+(i-1)r}$. Hence, $n + (i-1)m = g(jn + (i-1)r)$ for a certain $g \in [d]$. This implies that $(jg-1)n = (m-gr)(i-1)$. We then obtain $jg - 1 = m - gr = 0$, which further implies that $j = g = 1$ and $m = r$. Similarly, from $x_1x_2^ix_3x_4^ix_5\hat{y} \in L_{(d)}$, it follows that $n + (i-1)m = g'(kn + (i-1)r)$. Thus, $(kg' - 1)n = (m - g'r)(i-1)$. This implies $k = g' = 1$ and $m = r$. Since $j \neq k$, we obtain a contradiction.

Next, we consider Case (b) with appropriate factorizations $x' = x_1x_2x_3$, $y = y_1y_2y_3$, and $\hat{y} = z_1z_2z_3$ with $|x_2| \geq 1$ and $|x_2x_3| \leq c$ such that $x_1x_2^ix_3y_1y_2^iy_3 \in L_{(d)}$ and $x_1x_2^ix_3z_1z_2^iz_3 \in L_{(d)}$ for any number $i \in \mathbb{N}$. Since $|x_2x_3| \leq c$, we obtain $x_2 \in \{b\}^*$. Assume that $x_2 = b^m$ for a certain number $m \in [c]$. This is impossible because $x_1x_2^ix_3y_1y_2^iy_3$ has the form $a^nb^{jn+(i-1)m}$ and the exponent of b is not of the form rn for any number $r \in [d]$.

Proof Sketch of Theorem 1(1). Let $d \geq 2$ be any integer and consider $L_d^{(\leq)}$ over $\Sigma_d = \{a_1, a_2, \ldots, a_{d,1}, b_2, \ldots, b_d\}$. It is not difficult to check that $L_d^{(\leq)} \in \mathrm{DCFL}(d)$. Our goal is, therefore, to show that $L_d^{(\leq)}$ is not in $\mathrm{DCFL}(d-1)$. To lead to a contradiction, we assume that $L_d^{(\leq)} \in \mathrm{DCFL}(d-1)$.

Take $A = a_1^* a_2^* \cdots a_d^* b_1^* b_2^* \cdots b_d^*$ in REG and consider $L' = A \cap (\Sigma_d^* - L_d^{(\leq)})$, that is, $L' = \{a_1^{n_1} \cdots a_d^{n_d} b_1^{m_1} \cdots b_d^{m_d} \mid \exists i \in [d](n_i > m_i)\}$. Note by Lemma 3(2) that, since $L_d^{(\leq)} \in \mathrm{DCFL}(d-1)$, we obtain $L' \in \mathrm{DCFL}[d-1]$. Take a pumping-lemma constant $c > 0$ that satisfies Lemma 1. We set $n = c+1$ and consider the set $\{xy^{(k)} \mid k \in [d]\}$, where $x = a_1^n a_2^{2n} \cdots a_d^{dn}$ and $y^{(k)} = b_1^n b_2^{2n} \cdots b_{k-1}^{(k-1)n} b_k^{kn-1} b_{k+1}^{(k+1)n} \cdots b_d^{dn}$ for each index $k \in [d]$. Lemma 1 guarantees the existence of a specific distinct pair $\{j_1, j_2\}$ with $1 \leq j_1 < j_2 \leq d$.

By Lemma 1, since $|x'| > c$, there are two conditions to consider separately. Condition (1) is not difficult. Next, we consider Condition (2). Let $x' = a_1^n \cdots a_d^{dn} b_1^n \cdots b_{j_1-1}^{(j_1-1)n} b_{j_1}^{j_1 n-1}$, $y = b_{j_1} b_{j_1+1}^{(j_1+1)n} \cdots b_d^{dn}$, and $\hat{y} = b_{j_1+1}^{(j_1+1)n} \cdots b_{j_2-1}^{(j_2-1)n} b_{j_2}^{j_2 n-1} b_{j_2+1}^{(j_2+1)n} \cdots b_d^{dn}$. Note that $x'y = xy^{(j_1)}$ and $x'\hat{y} = xy^{(j_2)}$. There are three factorizations $x' = u_1 u_2 u_3$ with $|u_2| \geq 1$ and $|u_2 u_3| \leq c$, $y = y_1 y_2 y_3$, and $\hat{y} = z_1 z_2 z_3$ satisfying both $u_1 u_2^i u_3 y_1 y_2^i y_3 \in L'$ and $u_1 u_2^i u_3 z_1 z_2^i z_3 \in L'$ for any number $i \in \mathbb{N}$. From $|u_2 u_3| \leq c$ follows $u_2 \in \{b_{j_1}\}^+$. Let $u_2 = b_{j_1}^e$ for a certain $e \geq 1$. In particular, take $i = 2$. Note that $u_1 u_2^2 u_3 y_1 y_2^2 y_3$ has factors $a_{j_1}^{j_1 n}$ and $b_{j_1}^{j_1 n - 1 + 2e}$. Thus, we obtain $j_1 n = j_1 n + 2e - 1$, a clear contradiction. □

We omit from this exposition the proofs of Theorems 1(2), 2, and Proposition 1. These proofs will be included in its complete version.

4 Proof Sketch of the Pumping Lemma for DCFL[*d*]

We are now ready to provide the proof of the pumping lemma for DCFL[d] (Lemma 1). Our proof has two different parts depending on the value of d. The first part of the proof targets the basis case of $d = 1$. This special case directly corresponds to Yu's pumping lemma [15, Lemma 1]. To prove his lemma, Yu utilized a so-called *left-part theorem* of his for LR(k) grammars. We intend to re-prove Yu's lemma using only 1dpda's with no reference to LR(k) grammars. Our proof argument is easily extendable to *one-way nondeterministic pushdown automata* (or 1npda's) and thus to the pumping lemma for CFL. The second part of the proof deals with the general case of $d \geq 2$. Hereafter, we give the sketches of these two parts.

Basis Case of $d = 1$: Let Σ be any alphabet and take any infinite dcf language L over Σ. Let us consider an appropriate ε-1dpda $M = (Q, \Sigma, \{\mathfrak{k}, \$\}, \Gamma, \delta, q_0, Z_0, Q_{acc}, Q_{rej})$ in an ideal shape that recognizes L by Lemmas 4–5. For the desired constant c, we set $c = 2^{|Q|}$. Firstly, we take two arbitrary strings xy and $x\hat{y}$ over Σ with $y[1] = \hat{y}[1] = a$ and $|x| > c$.

Our goal is to show that Condition (2) in the basis case of $d = 1$ holds. There are four distinct cases to deal with. Hereafter, we intend to discuss them

separately. Note that, since M is one-way, every crossing state-stack pair at any boundary in the x-region does not depend on the choice of y and $\hat{y}$.

Case 1: Consider the case where there are two boundaries t_1, t_2 with $1 \leq t_1 < t_2 \leq |xa|$ and $|t_2 - t_1| \leq c$ such that (i) the boundaries t_1 and t_2 are mutually correlated and (ii) inner states at the boundaries t_1 and t_2 coincide. In this case, we factorize x into $x_1x_2x_3$ so that $t_1 = |x_1|$ and $t_2 = |x_1x_2|$. By Lemma 6(1), it then follows that, for any number $i \in \mathbb{N}$, $x_1x_2^ix_3y \in L$ and $x_1x_2^ix_3\hat{y} \in L$.

Case 2: Consider the case where there are four boundaries t_1, t_2, t_3, t_4 with $1 \leq t_1 < t_2 < t_3 < t_4 \leq |xa|$ and $|t_4 - t_1| \leq c$ and there are $p, q \in Q$, $\gamma \in (\Gamma^{(-)})^*Z_0$, and $\alpha \in (\Gamma^{(-)})^*$ for which (i) (q, γ) and $(q, \alpha\gamma)$ are the crossing state-stack pairs respectively at the boundaries t_1 and t_2, (ii) $(p, \alpha\gamma)$ and (p, γ) are the crossing state-stack pairs respectively at the boundaries t_3 and t_4, and (iii) the boundary block $[t_i, t_{i+1}]$ for each index $i \in [3]$ is pseudo-convex. We then take a factorization $x = x_1x_2x_3x_4x_5$ such that $t_i = |x_1x_2 \cdots x_i|$ for each $i \in [4]$. Note that $|x_2x_4| \geq 2$ because of $t_1 < t_2$ and $t_3 < t_4$. By an application of Lemma 6(2), we conclude that, for any $z \in \{y, \hat{y}\}$, $x_1x_2^ix_3x_4^ix_5z \in L$ for all $i \in \mathbb{N}$.

Case 3: Assume that Cases 1–2 fail. For brevity, we set $R = (|xa| - c, |xa|)$. Consider the case where there is no pop operation in the R-region. Since R-region contains more than $|Q|^3$ boundaries, the R-region includes a certain series of boundaries $s_1, s_2, \ldots, s_m$ such that, for certain $q \in Q$, $\gamma \in (\Gamma^{(-)})^*Z_0$, and $\alpha'_1, \ldots, \alpha'_{m-1} \in (\Gamma^{(-)})^*$, there are crossing state-stack pairs of the form $(q, \gamma), (q, \alpha'_1\gamma), \ldots, (q, \alpha'_{m-1} \cdots \alpha_1\gamma)$ at the boundaries $s_1, s_2 \ldots, s_m$, respectively. Note that the boundary blocks $[s_1, s_2], [s_2, s_3], \ldots, [s_{m-1}, s_m]$ are all convex. Clearly, $m > |Q|^2$. We choose $\{t_i\}_{i\in[m]}$ and $\{r_i\}_{i\in[m]}$ so that (i) for each index $i \in [m]$, t_i and r_i are boundaries in the y-region and in the $\hat{y}$-region, respectively, satisfying that $t_1 < t_2 < \cdots < t_m$ and $r_1 < r_2 < \cdots < r_m$, and (ii) for each index $i \in [m-1]$, $[s_i, s_{i+1}]$ is mutually correlated to $[t_i, t_{i+1}]$ in the y-region and also to $[r_i, r_{i+1}]$ in the $\hat{y}$-region. Note that the boundary blocks $[t_1, t_2], \ldots, [t_{m-1}, t_m]$, $[r_1, r_2], \ldots, [r_{m-1}, r_m]$ are all pseudo-convex. Since $m > |Q|^2$, it follows that there is a pair $j_1, j_2 \in [m]$ with $j_1 < j_2$ such that inner states at the boundaries r_{j_1} and r_{j_2} coincide. Using Lemma 6(2), we can obtain the desired conclusion.

Case 4: Assume that Cases 1–3 fail. In this case, we define a notion of "true gain" in the R-region and estimate its value. Choose s_1 and s_2 so that $|xa| - c \leq s_1$, $s_2 \leq |xa|$, and the boundary block $[s_1, s_2]$ is pseudo-convex. Let $G(s_1, s_2)$ denote the set of boundary blocks $[t_1, t'_1], [t_2, t'_2], \ldots, [t_m, t'_m]$ with $s_1 \leq t_1$, $t'_m \leq s_2$, $t_i < t'_i$ for every $i \in [m]$, and $t'_j < t_{j+1}$ for every $j \in [m-1]$ such that (i) $[t_i, t'_i]$ is pseudo-convex but cannot be flat, (ii) $[t'_j, t_{j+1}]$ is pseudo-convex (and could be flat), (iii) there are crossing state-stack pairs $(q_i, \gamma), (q'_i, \gamma)$ at the boundaries t_i, t'_i for every $i \in [m]$, (iv) the stack height at the boundary t'_i is higher than the stack height at the boundary t_i, (v) the boundary t_i is a pit (i.e., the lowest point within its small vicinity). Define the *true gain* $tg(s_1, s_2)$ to be $\sum_{i=1}^{m} |t'_i - t_i|$. It is possible to prove that $tg(s_1, s_2) > |Q|^3$. Using this inequality, we can employ an argument similar to Case 3 to obtain the lemma.

General Case of $d \geq 2$: We begin with proving this case by considering d 1dpda's $M_1, M_2, \ldots, M_d$. The language recognized by each machine M_i is denoted by $L(M_i)$. Let us assume that $L = \bigcup_{i=1}^{d} L(M_i)$. Take $d+1$ strings $w_1, w_2, \ldots, w_{d+1}$ in L and assume that each w_k has the form $xy^{(k)}$ with $|x| > c$. Since all w_k's are in L, define a function f as follows. Let $f(k)$ denote the minimal index i_k satisfying that $w_k \in L(M_{i_k})$ but $w_k \notin L(M_j)$ for all $j \neq i_k$. Since there are at most d different languages, there are two distinct indices $j_1, j_2 \in [d+1]$ such that $f(j_1) = f(j_2)$. In what follows, we fix such a pair (j_1, j_2).

Consider the case of $w = xy^{(j_1)}$ and $w' = xy^{(j_2)}$. Take arbitrary factorizations $w = x'y$ and $w' = x'\hat{y}$. We apply the basis case of $d = 1$ again and obtain one of the following (a)–(b). (a) There is a factorization $x = x_1x_2x_3x_4x_5$ with $|x_2x_4| \geq 1$ and $|x_2x_3x_4| \leq c$ such that $x_1x_2^ix_3x_4^ix_5y \in L$ and $x_1x_2^ix_3x_4^ix_5y \in L$ for any number $i \in \mathbb{N}$. (b) There are factorizations $x' = x_1x_2x_3$, $y = y_1y_2y_3$, and $\hat{y} = z_1z_2z_3$ such that $|x_2| \geq 1$, $|x_2x_3| \leq c$, $x_1x_2^ix_3y_1y_2^iy_3 \in L$, and $x_1x_2^ix_3z_1z_2^iz_3 \in L$ for any number $i \in \mathbb{N}$.

References

1. Ginsburg, S., Greibach, S.: Deterministic context free languages. Inf. Control **9**, 620–648 (1966)
2. Harrison, M.A.: Iteration theorems for deterministic families of languages. Fundamenta Informaticae **9**, 481–508 (1986)
3. Hibbard, T.N.: A generalization of context-free determinism. Inf. Control **11**, 196–238 (1967)
4. Igarashi, Y.: A pumping lemma for real-time deterministic context-free languages. Theor. Comput. Sci. **36**, 89–97 (1985)
5. King, K.N.: Iteration theorems for families of strict deterministic languages. Theor. Comput. Sci. **10**, 317–333 (1980)
6. Kutrib, M., Malcher, A., Wotschke, D.: The Boolean closure of linear context-free languages. Acta Inform. **45**, 177–191 (2008)
7. Li, M., Vitányi, P.: An Introduction to Kolmogorov Complexity and Its Applications. Springer, New York (1994)
8. Liu, L.Y., Weiner, P.: An infinite hierarchy of intersections of context-free languages. Math. Syst. Theory **7**, 185–192 (1973)
9. Pighizzini, G., Pisoni, A.: Limited automata and regular languages. Int. J. Found. Comput. Sci. **25**, 897–916 (2014)
10. Wise, D.S.: A strong pumping lemma for context-free languages. Theor. Comput. Sci. **3**, 359–369 (1976)
11. Wotschke, D.: The Boolean closures of the deterministic and nondeterministic context-free languages. In: Brauer, W. (ed.) GI Gesellschaft für Informatik e. V. LNCS, pp. 113–121. Springer, Heidelberg (1973). https://doi.org/10.1007/978-3-662-41148-3_11
12. Wotschke, D.: Nondeterminism and Boolean operations in pda's. J. Comput. Syst. Sci. **16**, 456–461 (1978)
13. Yamakami, T.: Oracle pushdown automata, nondeterministic reducibilities, and the hierarchy over the family of context-free languages. In: Geffert, V., Preneel, B., Rovan, B., Štuller, J., Tjoa, A.M. (eds.) SOFSEM 2014. LNCS, vol. 8327, pp. 514–525. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-04298-5_45. A complete version is found at arXiv:1303.1717 under a slightly different title

14. Yamakami, T.: Behavioral Strengths and weaknesses of various models of limited automata. In: Catania, B., Královič, R., Nawrocki, J., Pighizzini, G. (eds.) SOFSEM 2019. LNCS, vol. 11376, pp. 519–530. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-10801-4_40
15. Yu, S.: A pumping lemma for deterministic context-free languages. Inf. Process. Lett. **31**, 47–51 (1989)