# Boolean Language Operations on Nondeterministic Automata with a Pushdown of Constant Height

Viliam Geffert[1,⋆], Zuzana Bednárová[1,⋆],
Carlo Mereghetti[2], and Beatrice Palano[2]

[1] Dep. Computer Sci., P. J. Šafárik Univ., Jesenná 5, 04154 Košice, Slovakia
viliam.geffert@upjs.sk, ivazuzu@eriv.sk
[2] Dip. Informatica, Univ. degli Studi di Milano, v. Comelico 39, 20135 Milano, Italy
mereghetti@di.unimi.it, palano@di.unimi.it

**Abstract.** We study the size-cost of Boolean operations on *constant height nondeterministic pushdown automata*, i.e. on pushdown automata with a constant limit on the size of the pushdown. For *intersection*, we show an exponential simulation and prove that the exponential blow-up is necessary. For *union*, instead, we provide a linear trade-off while, for *complement*, we show a double-exponential simulation and prove a single-exponential lower bound.

**Keywords:** descriptional complexity, finite state automata, regular languages, nondeterministic pushdown automata.

## 1 Introduction

A primary task in the area of descriptional complexity is the analysis of how succinctly a given device is able to describe a certain class of languages. Quite often, languages that are more "complex" are obtained from "simpler" ones by the use of some "standard" language operations in the class, which requires evaluating the cost of implementing these language operations by the given device.

The largest amount of results devoted to descriptional complexity is related to *regular languages*. Among others, these languages are representable by regular grammars, expressions, and several variants of automata, starting from the original model of a *deterministic finite state automaton* (DFA) and ranging over enhanced models with additional features, like nondeterminism, alternation, probabilism, quantum or two-way versions... For a brief survey, see, e.g., [6,12].

In this paper, we study the descriptional power of a *constant height nondeterministic pushdown automaton* (*constant height* NPDA). Such machine is a traditional pushdown automaton (see, e.g., [7]) with a constant limit on the height of the pushdown, not depending on the input length. This model was introduced in [4], together with its *deterministic version* (*constant height* DPDA). It is easy to see that such devices accept regular languages. However, a representation by

⋆ Supported by the Slovak grant contracts VEGA 1/0479/12 and APVV-0035-10.

A.A. Bulatov and A.M. Shur (Eds.): CSR 2013, LNCS 7913, pp. 100–111, 2013.

constant height pushdown automata can be more succinct. In [4], an optimal exponential gap was shown between the sizes of NPDAs and of *nondeterministic finite state automata* (NFAs). The same gap was found for the deterministic case, between DPDAs and DFAs. Converting a constant height NPDA into an equivalent DPDA uses, and also requires, a double-exponential blow-up [1].

Here we concentrate on a classical problem, the *cost of Boolean operations*, on constant height NPDAs. There exists a wide literature on this issue with respect to, e.g., finite state automata [6,13], regular expressions [3,5], or grammars [8,9]. The cost of these operations for constant height DPDAs was also investigated, in [2] (see also Tab. 1 in Sect. 5).

First, we analyze the cost of *intersection*. Given two constant height NPDAs $A$ and $B$ with respective sets of states $Q_A, Q_B$, pushdown alphabets $\Gamma_A, \Gamma_B$, and pushdown heights $h_A, h_B$, we design an NPDA for $L(A) \cap L(B)$ with at most $\|Q_A\| \cdot \|\Gamma_A^{\leq h_A}\| \cdot \|Q_B\|$, states, $\|\Gamma_B\|$ pushdown symbols, and the pushdown height $h_B$. Since the roles of $A$ and $B$ can be swapped, the number of states is actually exponential in $h = \min\{h_A, h_B\}$. In the worst case, an exponential blow-up is necessary: for each fixed $c \geq 2$, we exhibit $\{L'_n\}_{n\geq 1}$ and $\{L''_n\}_{n\geq 1}$, two families of languages over a $(c+1)$-letter alphabet, such that: *(i)* both $L'_n$ and $L''_n$ are accepted by NPDAs with $O(c)$ states, $c$ pushdown symbols, and the pushdown height $n$, but *(ii)* their intersection cannot be accepted by an NPDA in which both the number of states and the pushdown height are below $c^{n/3-O(\log n)}$.

The *union* operation, instead, turns out to be easy. We propose an NPDA for $L(A) \cup L(B)$ with a linear trade-off, namely, with at most $\max\{1, |h_A - h_B|\} + \|Q_A\| + \|Q_B\|$ states, $1 + \max\{\|\Gamma_A\|, \|\Gamma_B\|\}$ pushdown symbols, and the pushdown height bounded by $\max\{h_A, h_B\}$.

Finally, for the *complement* $L(A)^{\mathrm{c}}$, we provide an NPDA with $2^{\|Q_A\| \cdot \|\Gamma_A^{\leq h_A}\|}$ many states, actually a DFA with the pushdown height equal to zero. Although we leave as open the problem of showing the optimality of such double-exponential blow-up, we prove a single-exponential lower bound for the cost, by providing $\{\tilde{L}_n\}_{n\geq 1}$, a family of languages over a $(c+1)$-letter alphabet, such that: *(i)* $\tilde{L}_n$ is accepted by an NPDA with $n + O(c)$ states, $c+1$ pushdown symbols, and the pushdown height $n+1$, but *(ii)* its complement cannot be accepted by an NPDA in which both the number of states and the pushdown height are below $c^{n/3-O(\log n)}$.

These lower bounds required some new techniques, for several reasons: *(i)* Already a deterministic machine with a polynomial pushdown height can use exponentially many different pushdown contents and hence exponential lower bounds cannot be obtained directly, by standard pigeonhole arguments. *(ii)* Moreover, our machines are *nondeterministic* and hence, after reading the first $i$ symbols of an input $a_1 \cdots a_\ell$, the state and the pushdown content do not depend only on $a_1 \cdots a_i$, but, using a guess-and-verify fashion, on the entire input.

## 2 Preliminaries

We assume the reader is familiar with the standard models of deterministic and nondeterministic *finite state automata* (DFA and NFA, for short) and *pushdown*

*automata* (DPDA and NPDA, see, e.g., [7]). Here we briefly recall these models. By $\Sigma^*$, we denote the set of words over an alphabet $\Sigma$. For a word $\varphi = a_1 \cdots a_\ell \in \Sigma^*$, let $\varphi^{\mathrm{R}} = a_\ell \cdots a_1$ denote its reversal and $|\varphi| = \ell$ its length. The set of words of length $i$ is denoted by $\Sigma^i$, with $\Sigma^{\leq h} = \bigcup_{i=0}^{h} \Sigma^i$. By $\|S\|$, we denote the cardinality of a set $S$ and by $S^{\mathrm{c}}$ its complement.

For technical reasons, the NPDAs are introduced here in the form that clearly distinguishes instructions manipulating the pushdown store from those reading the input tape [4]. An NPDA is a sextuplet $A = \langle Q, \Sigma, \Gamma, H, q_{\mathrm{I}}, F\rangle$, where $Q$ is the finite set of states, $\Sigma$ the input alphabet, $\Gamma$ the pushdown alphabet, $q_{\mathrm{I}} \in Q$ the initial state, $F \subseteq Q$ the set of accepting (final) states, and $H \subseteq Q \times (\{\varepsilon\} \cup \Sigma \cup \{-,+\}\cdot\Gamma) \times Q$ the *transition relation*, with the following meaning:

(i) $(q, \varepsilon, q') \in H$: $A$ gets from the state $q$ to the state $q'$ without using the input tape or the pushdown store.
(ii) $(q, a, q') \in H$: if the next input symbol is $a$, $A$ gets from $q$ to $q'$ by reading the symbol $a$, not using the pushdown store.
(iii) $(q, -X, q') \in H$: if the symbol on top of the pushdown is $X$, $A$ gets from $q$ to $q'$ by popping $X$, not using the input tape.
(iv) $(q, +X, q') \in H$: $A$ gets from $q$ to $q'$ by pushing the symbol $X$ onto the pushdown, not using the input tape.

An *accepting computation* begins in the state $q_{\mathrm{I}}$ with the empty pushdown store, and ends in an accepting state $q' \in F$ after reading the entire input. As usual, $L(A)$ denotes the language accepted by the NPDA $A$. A *deterministic pushdown automaton* (DPDA) is obtained by claiming that the transition relation does not allow executing more than one transition at a time. (See [2] for a more formal definition.) The following "normal form" of NPDAs will be required later.

**Lemma 1 ([4, Lem. 1]).** *For any* NPDA $A = \langle Q, \Sigma, \Gamma, H, q_{\mathrm{I}}, F\rangle$, *there exists an equivalent* NPDA $A' = \langle Q \cup \{q_{\mathrm{F}}\}, \Sigma, \Gamma, H', q_{\mathrm{I}}, \{q_{\mathrm{F}}\}\rangle$, *such that* $A'$ *accepts by entering the unique* $q_{\mathrm{F}}$ *with empty pushdown store at the end of the input.*

Given a constant $h \geq 0$, the NPDA $A$ is *of pushdown height* $h$ if, for any $\varphi \in L(A)$, there exists an accepting computation along which the pushdown store never contains more than $h$ symbols. Such a machine will be denoted by a septuplet $A = \langle Q, \Sigma, \Gamma, H, q_{\mathrm{I}}, F, h\rangle$, where $h \geq 0$ is a constant denoting the pushdown height, and all other elements are defined as above. By definition, the meaning of the transitions in the form (iv) is modified as follows:

(iv') $(q, +X, q') \in H$: *if the current pushdown store height is smaller than* $h$, then $A$ gets from the state $q$ to the state $q'$ by pushing the symbol $X$ onto the pushdown, not using the input tape; *otherwise* $A$ aborts and rejects.

A fair *descriptional complexity measure* takes into account all the components the device consists of, i.e., *(i)* the number of finite control states, *(ii)* the size of the pushdown alphabet, and *(iii)* the height of the pushdown store [4].

**Lemma 2 ([2, Lem. 2]).** *For each constant height* NPDA $A = \langle Q, \Sigma, \Gamma, H, q_{\mathrm{I}}, F, h\rangle$, *there exists an equivalent* NFA $A' = \langle Q', \Sigma, H', q'_{\mathrm{I}}, F'\rangle$ *with* $\|Q'\| \leq \|Q\| \cdot \|\Gamma^{\leq h}\|$ *states.*

We conclude this section by a combinatorial lemma required later. The lemma says that each sufficiently large subset of $A \times B$ (where $A$ and $B$ are some finite sets) must contain a trio of elements forming a "rectangular triangle".

**Lemma 3.** *Let $A$ and $B$ be arbitrary two finite sets satisfying $\|A\| \geq 2$ and $\|B\| \geq 2$, and let $C$ be a subset of $A \times B$ satisfying $\|C\| \geq \|A\| + \|B\| - 1$. Then there must exist some elements $\dot{a}, \ddot{a} \in A$ and $\dot{b}, \ddot{b} \in B$, with $\dot{a} \neq \ddot{a}$ and $\dot{b} \neq \ddot{b}$, such that $[\dot{a}, \dot{b}]$, $[\dot{a}, \ddot{b}]$, $[\ddot{a}, \dot{b}]$ are all in $C$.*

## 3 Intersection for Constant Height NPDAs

Here we consider the amount of computational resources that are sufficient and necessary for a constant height NPDA accepting the intersection $L(A) \cap L(B)$, for the given constant height NPDAs $A$ and $B$. After transforming both $A$ and $B$ into the equivalent NFAs, such machine can be built easily. However, by exploiting the power of pushdown storage, we obtain a better construction:[1]

**Theorem 1.** *Given two constant height NPDAs $A = \langle Q_A, \Sigma, \Gamma_A, H_A, q_A, F_A, h_A \rangle$ and $B = \langle Q_B, \Sigma, \Gamma_B, H_B, q_B, F_B, h_B \rangle$, there exists a constant height NPDA $C$ accepting the intersection $L(A) \cap L(B)$ with the number of states bounded by $\|Q_C\| \leq \|Q_A\| \cdot \|\Gamma_A^{\leq h_A}\| \cdot \|Q_B\|$, using $\|\Gamma_C\| = \|\Gamma_B\|$ pushdown symbols and the pushdown height $h_C = h_B$.*

The main idea is to turn the machine $A$ into an NFA $A'$ (by Lem. 2) and then construct $C$ simulating $A'$ and $B$ simultaneously, using the pushdown store for the simulation of $B$. Final states are chosen so that $C$ accepts if and only if the input is accepted by both machines. Since the roles of $A$ and $B$ can be swapped, the number of states is actually exponential in $h = \min\{h_A, h_B\}$.

We shall now show that the exponential cost cannot be avoided. To this purpose, for arbitrary fixed input alphabet $\Sigma$, we define two families of languages $\{L'_n\}_{n \geq 1}$ and $\{L''_n\}_{n \geq 1}$ accepted by constant height NPDAs with $O(\|\Sigma\|)$ states, $\|\Sigma\|$ pushdown symbols, and the pushdown height $h_n = n$, but with a lower bound $\|\Sigma\|^{\Omega(n)}$ for accepting $\{L'_n \cap L''_n\}_{n \geq 1}$. First, fix a special symbol $\$ \notin \Sigma$. Then, for each $n \geq 1$, define the following two languages:

$$\begin{aligned} L'_n &= \{u_1 \$ v_1 \$ u_2^{\mathrm{R}} \$ v_2^{\mathrm{R}} : u_1, v_1, u_2, v_2 \in \Sigma^*,\ |u_1| \leq n,\ \ u_2 \text{ is a suffix of } u_1\}, \\ L''_n &= \{u_1 \$ v_1 \$ u_2^{\mathrm{R}} \$ v_2^{\mathrm{R}} : u_1, v_1, u_2, v_2 \in \Sigma^*,\ |v_1| \leq n,\ \ v_2 \text{ is a suffix of } v_1\}. \end{aligned} \tag{1}$$

**Lemma 4.** *For any given $\Sigma$ and $n \geq 1$, the languages $L'_n$ and $L''_n$ can be accepted by DPDAs (hence, also by NPDAs) $A'_n$ and $A''_n$, respectively, with $2 \cdot \|\Sigma\| + 4 \leq O(\|\Sigma\|)$ states, $\|\Sigma\|$ pushdown symbols, and the pushdown height $h_n = n$.*

[1] Using the transition function in a form introduced in standard textbooks, only a single state would be required, since the language $L(A) \cap L(B)$ is regular, and hence context free. (See e.g. [7, Sect. 6.3.1].) This indicates that the "traditional" transition function $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times \Gamma \to 2^{Q \times \Gamma^*}$ (combining input and pushdown operations into a single step) is not realistic if the state-set size is at stake.

*Proof.* On input $u_1\$v_1\$u_2^{\mathrm{R}}\$v_2^{\mathrm{R}}$, the DPDA $A'_n$ compares the pushdown content filled while reading $u_1$ with $u_2^{\mathrm{R}}$ in order to check whether $v_2$ is a suffix of $v_1$, which leaves the first $|u_1| - |u_2^{\mathrm{R}}|$ symbols of $u_1$ in the pushdown. (The machine $A''_n$ runs in a similar way.) The tricky detail is that, to reduce the number of states from $\Omega(n{\cdot}\|\Sigma\|)$ to $O(\|\Sigma\|)$, we do allow both $A'_n$ and $A''_n$ to reject in the middle of the input by pushdown overflow. □

Denote now the intersection of $L'_n$ and $L''_n$ as

$$L_n = L'_n \cap L''_n = \{u_1\$v_1\$u_2^{\mathrm{R}}\$v_2^{\mathrm{R}} : u_1, v_1, u_2, v_2 \in \Sigma^*,\ |u_1| \le n,\ |v_1| \le n,\\ u_2 \text{ is a suffix of } u_1 \quad \text{and} \quad v_2 \text{ is a suffix of } v_1\}.$$

Clearly, if $|u_1| = |v_1| = |u_2| = |v_2| \le n$, the conditions for membership are simplified: $u_1\$v_1\$u_2^{\mathrm{R}}\$v_2^{\mathrm{R}}$ is in $L_n$ if and only if $u_2 = u_1$ and $v_2 = v_1$.

**Theorem 2.** *Let $\{A_n\}_{n\ge1}$ be constant height* NPDA*s accepting the languages $\{L_n\}_{n\ge1}$, for some non-unary alphabet $\Sigma$, and let $Q_n$ and $h_n$ be, respectively, the number of states and the pushdown height in $A_n$. Then $(\|Q_n\|{+}1)^2{\cdot}(h_n{+}1) > \|\Sigma\|^n/(4n^2+6n)$, for each $n \ge 1$. Consequently, in $\{A_n\}_{n\ge1}$, the number of states and the pushdown height cannot be both polynomial in $n$; either $\|Q_n\|+1$ or $h_n+1$ (or both values) are above $\|\Sigma\|^{n/3}/\sqrt[3]{4n^2+6n} \ge \|\Sigma\|^{n/3-O(\log n)}$.*

*Proof.* Let $A_n = \langle Q_n, \Sigma, \Gamma_n, H_n, q_{\mathrm{I},n}, F_n, h_n\rangle$ be a constant height NPDA accepting $L_n$. For contradiction, assume first that the NPDA $A_n$ is in the "normal form" of Lem. 1, that is, it accepts each input by entering the unique final state $q_{\mathrm{F},n}$ with *empty pushdown store* at the end of the input (hence, $F_n = \{q_{\mathrm{F},n}\}$), and that $p_n = \|Q_n\|^2{\cdot}(h_n{+}1) \le \|\Sigma\|^n/(4n^2+6n)$, for some $n$. From now on, for the sake of readability, we simply write $p$ instead of $p_n$, as well as $A, Q, \Gamma, H, q_{\mathrm{I}}, q_{\mathrm{F}}, h$ instead of $A_n, Q_n, \Gamma_n, H_n, q_{\mathrm{I},n}, q_{\mathrm{F},n}, h_n$. From these assumptions we get that

$$p = \|Q\|^2{\cdot}(h+1) \le \|\Sigma\|^n/(4n^2+6n). \tag{2}$$

Next, define the following set of pairs:

$$V_0 = \Sigma^n \times \Sigma^n = \{[u,v] : u, v \in \Sigma^*,\ |u| = |v| = n\}.$$

Consider now the computation on the input $z = u\$v\$u^{\mathrm{R}}\$v^{\mathrm{R}}$, for each $[u,v] \in V_0$. It is clear that $z \in L_n$, and hence there must exist at least one accepting computation of $A$ on this input. From among all possible accepting computations for this input, let us fix the "leftmost" accepting computation path. (That is, each time the machine gets into a configuration from which several nondeterministic choices lead to successful acceptance, take the leftmost choice, using some lexicographical ordering on $H$, the transition relation.) Now, let us fix some significant parameters for this leftmost path (see either side of Fig. 1):

- $y_\ell \in \{0, \ldots, h\}$, the lowest height of pushdown store in the course of reading the substring $\$v\$u^{\mathrm{R}}\$$,
- $q_\ell \in Q$, the state in which the height $y_\ell$ is attained for the last time, along $\$v\$u^{\mathrm{R}}\$$,

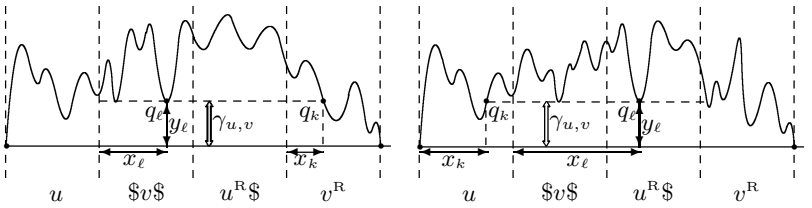


**Fig. 1.** Parameters $y_\ell, q_\ell, x_\ell$ and the pushdown content $\gamma_{u,v}$ along the computation (either side). Parameters $q_k, x_k$ depend on whether $q_\ell$ is reached in the course of reading $\$v\$$ (shown on the left), or in the course of reading $u^{\mathrm{R}}\$$ (shown on the right).

- $x_\ell \in \{1, \ldots, |\$v\$u^{\mathrm{R}}\$|\} = \{1, \ldots, 2n+3\}$, the distance from the beginning of $\$v\$u^{\mathrm{R}}\$$ to the input position in which $q_\ell$ is entered, and
- $\gamma_{u,v}$, the pushdown content at this moment.

The values for the next two parameters, namely, for $q_k \in Q$ and $x_k \in \{1, \ldots, n\}$, depend on whether $x_\ell \leq |\$v\$| = n+2$ or $x_\ell > n+2$:

If $x_\ell \leq |\$v\$| = n+2$, that is, if the computation reaches the state $q_\ell$ in the course of reading $\$v\$$ (see the left part of Fig. 1), then

- $q_k \in Q$ is the state at the moment when the machine is going to decrease, for the first time, the pushdown height from $y_\ell$ to $y_\ell - 1$, in the course of reading $v^{\mathrm{R}}$ (because our automaton always accepts with empty pushdown store — by Lem. 1, such situation must happen), and
- $x_k \in \{1, \ldots, |v^{\mathrm{R}}|\} = \{1, \ldots, n\}$ is the distance from the beginning of $v^{\mathrm{R}}$ to the input position in which $q_k$ is entered.

If $x_\ell > |\$v\$| = n+2$, that is, if the computation reaches the state $q_\ell$ in the course of reading $u^{\mathrm{R}}\$$ (see the right part of Fig. 1), then

- $q_k \in Q$ is the state at the moment when the machine has just increased, for the last time, the pushdown height from $y_\ell - 1$ to $y_\ell$, in the course of reading $u$ (because our automaton always starts with empty pushdown store — by definition, such situation must happen), and
- $x_k \in \{1, \ldots, |u|\} = \{1, \ldots, n\}$ is the distance from the beginning of $u$ to the input position in which $q_k$ is entered.

It is easy to see that, independent of whether $x_\ell \leq |\$v\$| = n+2$ or $x_\ell > n+2$, we have $y_\ell \in \{0, \ldots, h\}$, $q_\ell \in Q$, $x_\ell \in \{1, \ldots, 2n+3\}$, $q_k \in Q$, and $x_k \in \{1, \ldots, n\}$. Therefore, the number of different quintuples $[y_\ell, q_\ell, x_\ell, q_k, x_k]$ is bounded by $\|Q\|^2 \cdot (h+1) \cdot (2n+3) \cdot n = \|Q\|^2 \cdot (h+1) \cdot (2n^2+3n)$. Thus, by using also (2), the number of such quintuples can be bounded by $\|Q\|^2 \cdot (h+1) \cdot (2n^2+3n) \leq \|\Sigma\|^n / (4n^2+6n) \cdot (2n^2+3n) = \|\Sigma\|^n / 2$.

In conclusion, for each $[u, v] \in V_0$, we took the input $u\$v\$u^{\mathrm{R}}\$v^{\mathrm{R}}$, and fixed the unique leftmost accepting computation path, which gives the unique quintuple of parameters $[y_\ell, q_\ell, x_\ell, q_k, x_k]$. Thus, each pair $[u, v] \in V_0$ can be associated with exactly one quintuple $[y_\ell, q_\ell, x_\ell, q_k, x_k]$. Hence, a simple pigeonhole argument

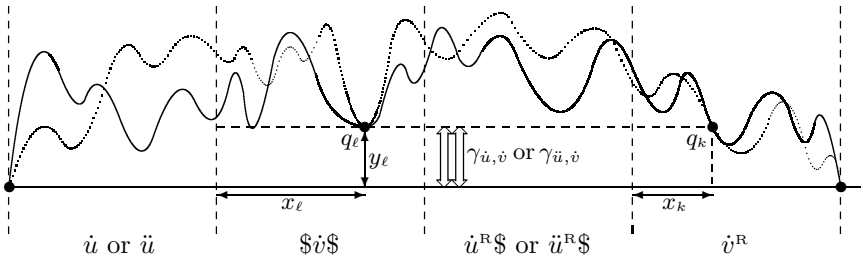


**Fig. 2.** Computation paths for the inputs $\dot{z} = \dot{u}\$\dot{v}\$\dot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}}$, $\ddot{z}_u = \ddot{u}\$\dot{v}\$\ddot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}}$, and $\delta_u = \dot{u}\$\dot{v}\$\ddot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}} \notin L_n$, for the case of $x_\ell \le |\$\dot{v}\$| = n{+}2$

proves the existence of a set $V_1 \subseteq V_0$, such that all $[u, v] \in V_1$ share the same $[y_\ell, q_\ell, x_\ell, q_k, x_k]$ and, moreover, the cardinality of such set is

$$\|V_1\| \ge \frac{\|V_0\|}{\|Q\|^2 \cdot (h+1) \cdot (2n^2+3n)} \ge \frac{\|\Sigma\|^{2n}}{\|\Sigma\|^n/2} = 2 \cdot \|\Sigma\|^n > 2 \cdot \|\Sigma\|^n - 1\,. \qquad (3)$$

Realize that $V_1 \subseteq \Sigma^n \times \Sigma^n$ and $\|\Sigma^n\| = \|\Sigma\|^n \ge 2$, for each $\|\Sigma\| \ge 2$ and $n \ge 1$. Hence, taking into account (3), the sets $A = \Sigma^n$, $B = \Sigma^n$, and $C = V_1$ satisfy the assumptions of Lem. 3. Therefore, there must exist some strings $\dot{u}, \ddot{u}, \dot{v}, \ddot{v} \in \Sigma^n$, with $\dot{u} \ne \ddot{u}$, $\dot{v} \ne \ddot{v}$, such that $[\dot{u}, \dot{v}]$, $[\dot{u}, \ddot{v}]$, and $[\ddot{u}, \dot{v}]$ are all in $V_1$. Consequently, they all share the same parameters $[y_\ell, q_\ell, x_\ell, q_k, x_k]$ on the corresponding accepting paths. Now we have to distinguish between the two cases, depending on the value $x_\ell$.

Case i: $x_\ell \le n{+}2 = |\$\dot{v}\$|$. This means that, for $[u, v] \in \{[\dot{u}, \dot{v}], [\dot{u}, \ddot{v}], [\ddot{u}, \dot{v}]\}$, all fixed leftmost computations for the inputs $u\$v\$u^{\mathrm{R}}\$v^{\mathrm{R}}$ visit the same state $q_\ell \in Q$, with the same pushdown height $y_\ell$, and at the same position $x_\ell$, in the course of reading $\$v\$$. Thus, for all these inputs, the parameter $q_k \in Q$ is taken as the state at the moment when the height is going to be decreased below $y_\ell$ for the first time, along $v^{\mathrm{R}}$, at a position $x_k$. Also the values $q_k$ and $x_k$ are the same for all these inputs. This situation is depicted in Fig. 2. Consider now $\dot{z} = \dot{u}\$\dot{v}\$\dot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}}$ and $\ddot{z}_u = \ddot{u}\$\dot{v}\$\ddot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}}$, together with their crossbreed $\delta_u = \dot{u}\$\dot{v}\$\ddot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}} \notin L_n$.

First, on the inputs $\dot{z}$ and $\ddot{z}_u$, at the moment when the machine $A$ reaches the state $q_\ell$ at the position $x_\ell$, the pushdown store contains, respectively, the string $\gamma_{\dot{u},\dot{v}}$ or $\gamma_{\ddot{u},\dot{v}}$, consisting of $y_\ell$ symbols loaded in the course of reading $\dot{u}$ (or $\ddot{u}$, respectively). On both $\dot{z}$ and $\ddot{z}_u$, these deepest $y_\ell$ symbols will stay unchanged in the pushdown until the moment when $A$ reaches the same state $q_k$ at the same position $x_k$, along $\dot{v}^{\mathrm{R}}$.

Now, for the input $\delta_u$, one of the possible computations can start by following the trajectory for $\dot{z}$, reading $\dot{u}$ and the first $x_\ell$ symbols of $\$\dot{v}\$$, until it reaches the state $q_\ell$. At this moment, the pushdown store contains the string $\gamma_{\dot{u},\dot{v}}$. Now the machine switches to the computation path for $\ddot{z}_u$, until it gets into the state $q_k$. Along this path, the computation does not visit the deepest $y_\ell$ symbols in the pushdown store, reading the remaining $|\$\dot{v}\$| - x_\ell$ symbols of $\$\dot{v}\$$, the entire block $\ddot{u}^{\mathrm{R}}\$$, and the first $x_k$ symbols of $\dot{v}^{\mathrm{R}}$. From this point forward, the

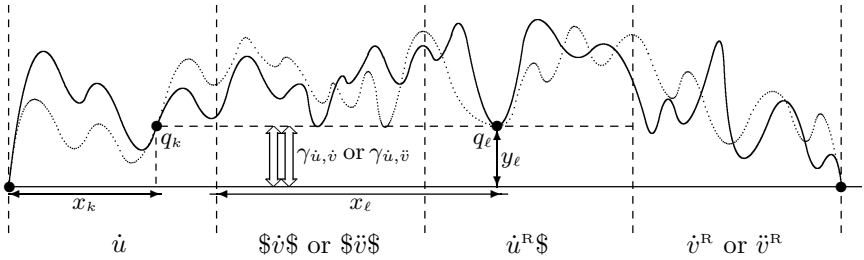


**Fig. 3.** Computation paths for the inputs $\dot{z} = \dot{u}\$\dot{v}\$\dot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}}$, $\ddot{z}_v = \dot{u}\$\ddot{v}\$\dot{u}^{\mathrm{R}}\$\ddot{v}^{\mathrm{R}}$, and $\delta_v = \dot{u}\$\ddot{v}\$\dot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}} \notin L_n$, for the case of $x_\ell > |\$\dot{v}\$| = n+2$

computation on $\delta_u$ can switch back to the trajectory for $\dot{z}$, working with the same content in the pushdown store and reading the remaining $|\dot{v}^{\mathrm{R}}| - x_k$ symbols of $\dot{v}^{\mathrm{R}}$. Clearly, such computation path stops with the empty pushdown in the accepting state $q_{\mathrm{F}}$. Thus, $A$ accepts $\delta_u = \dot{u}\$\dot{v}\$\ddot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}} \notin L_h$, which is a contradiction.

Case ii: $x_\ell > n+2 = |\$\dot{v}\$|$. Again, the three leftmost computations on the inputs $u\$v\$u^{\mathrm{R}}\$v^{\mathrm{R}}$, for $[u,v] \in \{[\dot{u},\dot{v}],[\dot{u},\ddot{v}],[\ddot{u},\dot{v}]\}$, visit the same state $q_\ell \in Q$, with the same pushdown height $y_\ell$, and at the same position $x_\ell$, this time in the course of reading $u^{\mathrm{R}}\$$. For all of them, the parameter $q_k \in Q$ is now taken as the state reached at the moment when the height has been increased to $y_\ell$ for the last time, along $u$, at a position $x_k$, but also here the values $q_k$ and $x_k$ are the same for all these inputs. This situation is depicted in Fig. 3. This time we consider the inputs $\dot{z} = \dot{u}\$\dot{v}\$\dot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}}$ and $\ddot{z}_v = \dot{u}\$\ddot{v}\$\dot{u}^{\mathrm{R}}\$\ddot{v}^{\mathrm{R}}$, together with their crossbreed $\delta_v = \dot{u}\$\ddot{v}\$\dot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}} \notin L_n$.

First, on the inputs $\dot{z}$ and $\ddot{z}_v$, at the moment when $A$ gets to the state $q_k$ at the position $x_k$, the pushdown store contains, respectively, the string $\gamma_{\dot{u},\dot{v}}$ or $\gamma_{\dot{u},\ddot{v}}$, consisting of $y_\ell$ pushdown symbols loaded in the course of reading the first $x_k$ symbols of $\dot{u}$. On both $\dot{z}$ and $\ddot{z}_v$, these deepest $y_\ell$ symbols will stay unchanged in the pushdown until the moment when $A$ reaches the same state $q_\ell$ at the same position $x_\ell$, along $\dot{u}^{\mathrm{R}}\$$.

Now, for the input $\delta_v$, one of the possible computations can start by following the trajectory for $\dot{z}$, reading first $x_k$ symbols of $\dot{u}$, until it reaches the state $q_k$. At this moment, the pushdown store contains the string $\gamma_{\dot{u},\dot{v}}$. Here $A$ switches to the computation path for $\ddot{z}_v$, until it gets into the state $q_\ell$. Along this path, the computation does not visit the deepest $y_\ell$ symbols in the pushdown store, reading the remaining $|\dot{u}| - x_k$ symbols of $\dot{u}$ and the first $x_\ell$ symbols of $\$\ddot{v}\$\dot{u}^{\mathrm{R}}\$$ (which includes, among others, traversing across the entire block $\$\ddot{v}\$$). From this point forward, the computation on $\delta_v$ can switch back to the trajectory for $\dot{z}$, working with the same content in the pushdown store and reading the remaining $|\$\ddot{v}\$\dot{u}^{\mathrm{R}}\$| - x_\ell$ symbols of $\dot{u}^{\mathrm{R}}\$$ and the entire block $\dot{v}^{\mathrm{R}}$. Again, such path stops in $q_{\mathrm{F}}$. Thus, $A$ accepts $\delta_v = \dot{u}\$\ddot{v}\$\dot{u}^{\mathrm{R}}\$\dot{v}^{\mathrm{R}} \notin L_n$, which is a contradiction.

In conclusion, if $A = A_n$ accepts $L_n$, the inequality (2) must be reversed. Thus, the value $p = p_n$ must satisfy $p_n = \|Q_n\|^2 \cdot (h_n+1) > \|\Sigma\|^n/(4n^2+6n)$. However, recall that we have derived this lower bound for the NPDA $A_n$ in the

"normal form" of Lem. 1. For unrestricted NPDAs (not assuming this form), the lower bound changes to $(\|Q_n\|+1)^2 \cdot (h_n+1) > \|\Sigma\|^n/(4n^2+6n)$, since converting a general NPDA into the normal form does not cost more than one state, keeping the same pushdown height. Consequently, either $\|Q_n\|+1 > \|\Sigma\|^{n/3}/\sqrt[3]{4n^2+6n}$, or else $h_n+1 > \|\Sigma\|^{n/3}/\sqrt[3]{4n^2+6n} \geq \|\Sigma\|^{n/3-O(\log n)}$. □

By combining Lem. 4 with Thm. 2, we get the following blow-up:

**Theorem 3.** *For each fixed constant $c \geq 2$, there exist $\{L'_n\}_{n\geq 1}$ and $\{L''_n\}_{n\geq 1}$, some families of regular languages built over a $(c+1)$-letter alphabet, such that:*

- *(i) there exist, respectively, $\{A'_n\}_{n\geq 1}$ and $\{A''_n\}_{n\geq 1}$, sequences of constant height* NPDA*s accepting these languages with $O(c)$ states, $c$ pushdown symbols, and the pushdown height $h_n = n$, but*
- *(ii) for any constant height* NPDA*s $\{A_n\}_{n\geq 1}$ accepting the family of their intersections $\{L_n\}_{n\geq 1} = \{L'_n \cap L''_n\}_{n\geq 1}$, either the number of states in $A_n$ or else the pushdown height must be above $c^{n/3-O(\log n)}$, independently of the size of the used pushdown alphabet.*

For comparison, by the use of Thm. 1 for NPDAs from Lem. 4, we get that $\{L'_n \cap L''_n\}_{n\geq 1}$ can be accepted[2] by NPDAs with $\|Q_A\| \cdot \|\Gamma_A^{\leq h_A}\| \cdot \|Q_B\| \leq O(c^n)$ states, $\|\Gamma_B\| = c$ pushdown symbols, and the pushdown height $h_B = n$.

## 4 Union and Complement for Constant Height NPDAs

Now we shall deal with another two basic Boolean operations, union and complement. First, given two constant height NPDAs $A$ and $B$, we construct a constant height NPDA $C$ accepting the union $L(A) \cup L(B)$. The size of $C$ is linear in all "reasonable" complexity measures. This allows us to derive an exponential lower bound for the complement $L(A)^{\mathrm{c}}$.

An NPDA $C$ accepting $L(A) \cup L(B)$ is simple: $C$ nondeterministically chooses which of the given two machines it will simulate. However, the pushdown heights $h_A, h_B$ may be different. If the chosen machine uses a lower pushdown height than the other one, the difference in the pushdown limit must be repaired, by filling $|h_A - h_B|$ copies of some extra symbol at the bottom of the pushdown.

**Theorem 4.** *Given two constant height* NPDA*s $A = \langle Q_A, \Sigma, \Gamma_A, H_A, q_A, F_A, h_A\rangle$ and $B = \langle Q_B, \Sigma, \Gamma_B, H_B, q_B, F_B, h_B\rangle$, there exists a constant height* NPDA *$C$ accepting the union $L(A) \cup L(B)$ with the number of states bounded by $\|Q_C\| \leq \max\{1, |h_A - h_B|\} + \|Q_A\| + \|Q_B\|$, using $\|\Gamma_C\| \leq 1+\max\{\|\Gamma_A\|, \|\Gamma_B\|\}$ pushdown symbols and the pushdown height $h_C = \max\{h_A, h_B\}$.*

[2] Alternatively, one can get a different NPDA for $\{L'_n \cap L''_n\}_{n\geq 1}$, using only $O(c^{n/2})$ states, with a pushdown of height $3/2 \cdot n$. The idea is to load the first half of $u_1$ and the entire $v_1$ in the pushdown, but to store the second half of $u_1$ in the finite state control. After checking the second half of $u_1$ against the first half of $u_2^{\mathrm{R}}$, we store the second half of $u_2^{\mathrm{R}}$ in the finite state control, to be checked later, after comparing $v_2^{\mathrm{R}}$ with $v_1$. This indicates that — without using a different witness language — the gap from Thm. 3 cannot be raised higher than to $\Omega(c^{n/2})$.

The last operation is complement. A trivial double-exponential upper bound is obtained by the use of Lem. 2, coding the pushdown content of the given NPDA $A$ in the finite state control, which gives a classical NFA with at most $\|Q_A\| \cdot \|\Gamma_A^{\leq h_A}\|$ states. Then we make this machine deterministic, by the standard power set construction [7,11], with $2^{\|Q_A\| \cdot \|\Gamma_A^{\leq h_A}\|}$ states. Finally, we obtain a DFA $B$ for $L(A)^{\mathrm{c}}$ by swapping the roles of accepting and rejecting states.

**Theorem 5.** *Given a constant height* NPDA $A = \langle Q_A, \Sigma, \Gamma_A, H_A, q_A, F_A, h_A \rangle$, *there exists a* DFA $B$ *(hence, also a constant height* NPDA*) accepting the complement* $L(A)^{\mathrm{c}}$ *with the number of states bounded by* $\|Q_B\| \leq 2^{\|Q_A\| \cdot \|\Gamma_A^{\leq h_A}\|}$ *(hence, using no pushdown symbols and the pushdown height equal to zero).*

At this point, one can easily combine the exponential lower bound obtained for intersection with the linear upper bound for union and conclude that the lower bound for complementing is at least exponential, by application of De Morgan's laws. Nevertheless, to see some growth rate for the gap, we shall consider a specific witness language. For this purposes, recall the languages $L'_n$ and $L''_n$, introduced by (1). For the fixed alphabet $\Sigma$ and each $n \geq 1$, let

$$\tilde{L}_n = {L'_n}^{\mathrm{c}} \cup {L''_n}^{\mathrm{c}}. \tag{4}$$

**Lemma 5.** *For any given* $\Sigma$ *and* $n \geq 1$, *the language* $\tilde{L}_n$ *can be accepted by an* NPDA $\tilde{A}_n$ *with* $\|Q_n\| = n + 4 \cdot \|\Sigma\| + 11 \leq n + O(\|\Sigma\|)$ *states,* $\|\Sigma\| + 1$ *pushdown symbols, and the pushdown height* $h_n = n + 1$.

*Proof.* Recall that, by Lem. 4, both $L'_n$ and $L''_n$ are accepted by the respective DPDAs $A'_n$ and $A''_n$ using $2 \cdot \|\Sigma\| + 4$ states, $\|\Sigma\|$ pushdown symbols, and the pushdown height $n$. Since both $A'_n$ and $A''_n$ are *deterministic*, an NPDA $\tilde{A}_n$ for $\tilde{L}_n$ can nondeterministically choose which of these two machines it will simulate, to verify that its unique computation *rejects*. The tricky detail is that, to reduce the number of states from $\Omega(n \cdot \|\Sigma\|)$ to $n + O(\|\Sigma\|)$, we do not keep track of the current pushdown height during the simulation, and hence we do not detect pushdown *overflows*. However, $A'_n$ and $A''_n$ from Lem. 4 reject by pushdown overflows *only if* the length of some block ($u_1$ or $v_1$, respectively) exceeds $n$, that is, only if the input contains a substring of length $n$+1 not containing any \$-symbols. Therefore, $\tilde{A}_n$ proceeds as follows. First, $\tilde{A}_n$ stores some new initial symbol $X_{\mathrm{I}}$ at the bottom of the pushdown (to detect pushdown *underflows* during the simulation), and then nondeterministically chooses from among *(i)* testing whether $A'_n$ rejects by a computation not blocked by a pushdown overflow, *(ii)* testing whether $A''_n$ rejects by a computation not blocked by a pushdown overflow, and *(iii)* testing whether the input contains a substring $\varphi$ of length $n$+1 without any \$-symbol, the starting position of $\varphi$ is established nondeterministically. □

Conversely, by De Morgan's laws, the *complement* of the language introduced by (4) is $\tilde{L}_n^{\mathrm{c}} = ({L'_n}^{\mathrm{c}} \cup {L''_n}^{\mathrm{c}})^{\mathrm{c}} = L'_n \cap L''_n = L_n$. Recall that the lower bound derived for $L_n$ in Thm. 2 is exponential. Combined with Lem. 5, this gives:

**Theorem 6.** *For each fixed constant $c \geq 2$, there exists $\{\tilde{L}_n\}_{n\geq 1}$, a family of regular languages built over a $(c+1)$-letter alphabet, such that:*

*(i) there exists $\{\tilde{A}_n\}_{n\geq 1}$, a sequence of constant height* NPDA*s accepting these languages with $Q_n \leq n + O(c)$ states, $c+1$ pushdown symbols, and the pushdown height $h_n = n+1$, but*

*(ii) for any constant height* NPDA*s $\{\tilde{A}^{\rm c}_n\}_{n\geq 1}$ accepting the family of their complements $\{\tilde{L}^{\rm c}_n\}_{n\geq 1}$, either the number of states in $\tilde{A}^{\rm c}_n$ or else the pushdown height must be above $c^{n/3-O(\log n)}$, independently of the size of the used pushdown alphabet.*

The above lower bound is far below the known conversion for complementing, presented in Thm. 5, using $2^{\|Q_A\|\cdot\|\Gamma_A^{\leq h_A}\|}$ states. It should also be pointed out that, in the case of our witness languages $\{\tilde{L}_n\}_{n\geq 1}$, their complements $\{\tilde{L}^{\rm c}_n\}_{n\geq 1} = \{L_n\}_{n\geq 1}$ *can* be accepted by NPDAs with only a *single*-exponential blow-up, namely, with $O(c^n)$ states, $c$ pushdown symbols, and the pushdown height $n$, which is obtained by combining Lem. 4 with Thm. 1.

## 5 Concluding Remarks

We have analyzed the size cost of basic Boolean operations for *nondeterministic automata with a pushdown of constant height*. For intersection, a single-exponential cost is sufficient and, in the worst case, also necessary. On the other hand, the cost of union is only linear. Combining these results, we have shown that the lower bound for complement is single-exponential, but we have derived only a double-exponential upper bound, which leaves a large gap.

It was conjectured that the lower bound for the intersection from Thm. 3 can be improved to an exponential lower bound on the number of states, independently of the pushdown height. This would give two witness regular languages the intersection of which would be "expensive" with respect to the number of states even for an NPDA with unrestricted pushdown. However, using the "traditional" transition function, we can reduce the number of states in such a machine to one, with a large pushdown alphabet. (See also Ft. 1.)

The corresponding costs for constant height DPDAs, *deterministic versions* of NPDAs, are [2]: single-exponential for intersection and union, but polynomial for complement. These results are compared in Tab. 1. It turns out that the cost of studied Boolean operations, both for DPDAs and NPDAs with a constant height pushdown, reflects the closure properties for the corresponding machines with an *unrestricted pushdown* (hence, for deterministic and general context-free languages): the unrestricted version of the pushdown machine is closed under the given operation (see e.g. [7]) if and only if the cost of the same operation for the constant height version is at most polynomial. Therefore, it could be interesting to investigate the complexity of other language operations for constant height pushdown automata and compare them with unrestricted versions.

**Table 1.** Size-cost of Boolean operations on constant height DPDAs [2] and constant height NPDAs, studied in this paper.

| Operation | constant height DPDAs | constant height NPDAs |
|---|---|---|
| Intersection | exponential | exponential |
| Union | exponential | linear |
| Complement | polynomial | exponential ... double-exponential |

Similarly, we would like to emphasize the interest in *two-way versions* of these machines, and in some *more restricted versions*. For instance, one could study the cost for *unary* languages: the same investigation on unary NFAs [10] shows interesting differences from the general case.

## References


1. Bednárová, Z., Geffert, V., Mereghetti, C., Palano, B.: Removing nondeterminism in constant height pushdown automata. In: Kutrib, M., Moreira, N., Reis, R. (eds.) DCFS 2012. LNCS, vol. 7386, pp. 76–88. Springer, Heidelberg (2012)
2. Bednárová, Z., Geffert, V., Mereghetti, C., Palano, B.: The size-cost of Boolean operations on constant height deterministic pushdown automata. Theoret. Comput. Sci. 449, 23–36 (2012)
3. Ehrenfeucht, A., Zieger, P.: Complexity measures for regular expressions. J. Comput. System Sci. 12, 134–146 (1976)
4. Geffert, V., Mereghetti, C., Palano, B.: More concise representation of regular languages by automata and regular expressions. Inf. & Comp. 208, 385–394 (2010)
5. Gruber, H., Holzer, M.: Language operations with regular expressions of polynomial size. Theoret. Comput. Sci. 410, 3281–3289 (2009)
6. Holzer, M., Kutrib, M.: Descriptional complexity — an introductory survey. In: Martín-Vide, C. (ed.) Scientific Applications of Language Methods, pp. 1–58. Imperial College Press (2010)
7. Hopcroft, J., Motwani, R., Ullman, J.: Introduction to Automata Theory, Languages, and Computation. Addison-Wesley (2001)
8. Kutrib, M., Malcher, A., Wotschke, D.: The Boolean closure of linear context-free languages. Acta Inform. 45, 177–191 (2008)
9. Meyer, A., Fischer, M.: Economy of description by automata, grammars, and formal systems. In: Proc. IEEE Symp. Switching & Automata Th., pp. 188–191 (1971)
10. Pighizzini, G., Shallit, J.: Unary language operations, state complexity and Jacobsthal's function. Internat. J. Found. Comput. Sci. 13, 145–159 (2002)
11. Rabin, M., Scott, D.: Finite automata and their decision problems. IBM J. Res. Develop. 3, 114–125 (1959)
12. Yu, S.: Regular languages. In: Rozenberg, G., Salomaa, A. (eds.) Handbook of Formal Languages, vol. I, pp. 41–110. Springer (1997)
13. Yu, S., Zhuang, Q., Salomaa, K.: The state complexities of some basic operations on regular languages. Theoret. Comput. Sci. 125, 315–328 (1994)