A Characterisation of $\mathsf{NL}/poly$ via Nondeterministic Finite Automata.

Rob Myers and Henning Urbat
Institut für Theoretische Informatik
TU Braunschweig, Germany

April 13, 2013

Abstract

For each language $L \subseteq \mathbf{2}^*$ and non-zero monotonic function $t : \mathbb{N} \to \mathbb{N}$, we define a language $t * L \subseteq \mathbf{2}^*$ depending only on L and t. We then prove that a language L lies in $\mathsf{NL}/poly$ if and only if there exists a polynomially bounded t (as above), such that the projections $(t * L) \cap \mathbf{2}^n$ are accepted by nondeterministic finite automata of size polynomial in n. Therefore, proving super-polynomial lower bounds for *unrestricted nondeterministic branching programs* is equivalent to proving super-polynomial lower bounds for *oblivious read-once nondeterministic branching programs* i.e. nondeterministic finite automata.

1 Introduction

In this paper we characterise the non-uniform complexity class $\mathsf{NL}/poly$ in terms of nondeterministic finite automata. Recall that $\mathsf{NL}/poly$ consists of those languages accepted by non-uniform logspace-bounded nondeterministic Turing machines with polynomial advice [5]. It is known to coincide with those languages $L \subseteq \mathbf{2}^*$ whose projections $L_n = L \cap \mathbf{2}^n$ are accepted by a sequence of nondeterministic branching programs (nbps) of size polynomial in n [8, 2]. We cannot simply replace nbps by nondeterministic finite automata (nfas) in the previous statement, since the latter correspond to very restricted nondeterministic branching programs i.e. oblivious read-once nbps. For example, an nfa accepting L_n may only make n steps before terminating, and this lack of time very often forces its width to become exponential in n, since it must 'store' potentially exponentially many different computational paths.

To counter this problem, we use simple models of time i.e. non-zero monotonic functions $t : \mathbb{N} \to \mathbb{N}$. Then for each $n, d \geq 0$ we define:

$$d \cdot L_n := \{w^d : w \in L_n\} \subseteq \mathbf{2}^{nd} \qquad t * L := \bigcup_{n \in \mathbb{N}} \overline{t(n) \cdot L_n} \;\subseteq\; \mathbf{2}^*$$

The finite language $t(n) \cdot L_n$ consists of all $t(n)$-powers of words from L_n and the finite language $\overline{t(n) \cdot L_n} \subseteq \mathbf{2}^{nt(n)}$ denotes its relative complement. Regarding $t(n) \cdot L_n$, nfas accepting this language have depth $n \cdot t(n)$ which provides them with extra time in which to work. Regarding $t * L$, the reason why we define it in terms of the relative complements can be explained as follows.

1. It is well-known that the Immerman-Szelepcsényi theorem [4, 10] implies nondeterministic branching programs can be complemented with only a polynomial blow-up [8, 2]. In particular, given small nbps $\mathcal{B}_n$ for L_n we can construct small nbps $\mathcal{B}'_n$ for the relative complements $\overline{L_n} \subseteq \mathbf{2}^n$. Define $t(n) = d(\mathcal{B}'_n)$ to be the depth of these nbps, where one may assume $t : \mathbb{N} \to \mathbb{N}$ is non-zero and monotonic (else add suitable dummy paths).

2. Crucially, given small nondeterministic branching programs $\mathcal{B}'_n$ for $\overline{L_n}$, one can construct *small nfas* accepting the language $\overline{t(n) \cdot L_n}$. For each $n \in \mathbb{N}$, the construction is performed in two steps: (i) translate $\mathcal{B}'_n$ into an nfa by replacing each edge by n suitable edges, (ii) put it in parallel with a polynomially sized nfa accepting $\overline{t(n) \cdot \mathbf{2}^n}$ i.e. all those words of length $n \cdot t(n)$ that are not $t(n)$-powers.

Let $\mathsf{nfa}(\mathit{poly})$ contain those languages $L \subseteq \mathbf{2}^*$ such that there exist poly-size nfas accepting the projections L_n. Then we can use the above argument to prove half of our main result.

Theorem. *$L \in \mathsf{NL}/\mathit{poly}$ if and only if there exists some polynomially bounded non-zero monotonic $t : \mathbb{N} \to \mathbb{N}$ such that $t * L \in \mathsf{nfa}(\mathit{poly})$.*

This is equivalent to Theorem 4.3 below, which characterises languages accepted by poly-size nondeterministic branching programs, these being precisely the languages in $\mathsf{NL}/\mathit{poly}$ as remarked above. The converse direction starts with a sequence of poly-size nfas accepting the projections $(t * L)_n$. One translates each nfa into a small nbp accepting L_n by (i) viewing nfas as special nbps and constructing their complement, (ii) identifying variables in a suitable manner.

The rest of this paper is devoted to making this proof outline precise. However we first provide some motivation and suggestions for future work.

Recently there has been renewed interest in the construction of state-minimal nfas accepting regular languages [3, 9, 11, 1, 6, 7]. Of course, every language $L \subseteq \mathbf{2}^n$ is finite and hence regular. In turns out that any particular regular language L has associated 'canonical' nondeterministic automata, see [6]. In fact there is one for each locally finite variety containing a two element algebra e.g. boolean algebras yield L's *átomaton* [1], join-semilattices yield L's *canonical residual nfa* [3, 7], and vector spaces over $\mathbb{Z}_2$ yield L's *minimal xor nfa* [11], the latter being oblivious read-once nbps with the parity acceptance condition. Various conditions are known on L such that the respective canonical nfa is *state-minimal.* For example: the result proved in [9] yields conditions under which the átomaton is state-minimal, and minimal xor nfas are *always* state-minimal amongst xor nfas. Also, it was recently proved in [7] that if the closure of L's left derivatives under unions forms a distributive lattice (i.e. its underlying inclusion-ordered poset does), then the canonical residual nfa is state-minimal.

Our main result clearly motivates the search for state-minimal nfas. Indeed, since poly-size nfas accepting $\overline{t(n) \cdot L_n}$ exist iff state-minimal nfas are poly-size, it follows that:

> $L \in \mathsf{NL}/$*poly iff there exists a non-zero monotonic* $t \in n^{O(1)}$ *such that* state-minimal *nfas for* $\overline{t(n) \cdot L_n}$ *are of size polynomial in* n.

Furthermore, the canonical nfa constructions now induce *canonical nondeterministic branching programs*, so if they are also 'small' they provide an automatic way of constructing small (non-uniform) algorithms.

We finish with a potentially interesting observation. By our main result, proving *super-polynomial lower bounds for nbps* on an explicit language $L \subseteq \mathbf{2}^*$ is *equivalent* to proving super-polynomial lower bounds on the size of nfas accepting $n^k \ast L$, for each $k \geq 0$. The latter are oblivious read-once nbps parametric in n and k, so perhaps this open problem has been reduced to an easier one.

2 Nondeterministic Branching Programs

In this section we define nondeterministic branching programs, providing comparisons and a normal form. We first fix some notational conventions.

Notation 2.1. Let $\mathbf{2} = \{0, 1\}$ be the booleans and $\mathbb{N} = \{0, 1, 2, \dots\}$ be the set of natural numbers. For any language $L \subseteq \mathbf{2}^*$ let $\widetilde{L} = \mathbf{2}^* \setminus L$ denote its complement, and for any finite language $L \subseteq \mathbf{2}^n$ write $\overline{L} = \mathbf{2}^n \setminus L$ for its relative complement. Given any word $w \in \mathbf{2}^n$ and $1 \leq i \leq n$ let $w_i \in \mathbf{2}$ be the i^{th} letter of w. For any $d \in \mathbb{N}$ let $w^d = w \dots w$ be the d-fold composition. Finally, fix a set $X_n = \{x_1, \dots, x_n\}$ of n variables for each natural $n \in \mathbb{N}$.

Definition 2.2. (a) A *nondeterministic branching program* (*nbp*) over n variables is a quadruple $\mathcal{B} = (G, s, \theta, \tau)$ consisting of:

(i) a finite directed multigraph $G = (V, E)$;

(ii) a *source node* $s \in V$;

(iii) a *node labelling* i.e. a function $\theta : V \to X_n \cup \mathbf{2}$ where every node labelled with 0 or 1 is a *sink* (has out-degree 0);

(iv) an *edge labelling* i.e. a function $\tau : E \to \mathbf{2}$.

We use the notation $(u \| l) \xrightarrow{b} (v \| m)$ to indicate that node u has label l, node v has label m, and there is an edge from u to v with label b.

(b) A *deterministic branching program* (*dbp*) is an nbp whose variable-labelled nodes have degree 2, where one outgoing edge is labelled 0, the other by 1.

(c) The *size* $s(\mathcal{B})$ of $\mathcal{B}$ is its number of nodes. For acyclic $\mathcal{B}$, its *depth* $d(\mathcal{B})$ is the number of edges of any longest directed path starting at the source.

(d) A word $w = w_1 \dots w_n \in \mathbf{2}^n$ is *accepted* by an nbp $\mathcal{B}$ if there exists some path:

$$(s \| x_{k_0}) \xrightarrow{b_0} (v_1 \| x_{k_1}) \xrightarrow{b_1} \cdots \xrightarrow{b_{m-2}} (v_{m-1} \| x_{k_{m-1}}) \xrightarrow{b_{m-1}} (v_m \| 1)$$

consistent with w, i.e. $b_i = w_{k_i}$ for every $0 \leq i < m$. The *language* $L_{\mathcal{B}} \subseteq \mathbf{2}^n$ of $\mathcal{B}$ is the set of all words accepted by $\mathcal{B}$.

Remark 2.3. Many authors make additional assumptions on the structure of (nondeterministic) branching programs, e.g. that the graph is acyclic and every node is reachable from the source. These restrictions emerge in Lemma 2.9.

Example 2.4. Here is an nbp $\mathcal{B} = (G, s, \theta, \tau)$ for $n = 4$:

$$\begin{tikzcd}
 & \boxed{s\|x_4} \arrow[dl, "0"'] \arrow[dr, "1"] & \\
\boxed{v_1\|x_2} \arrow[d, "0"'] \arrow[dr, "0"] & & \boxed{v_2\|x_3} \arrow[dl, "0"'] \arrow[d, "1"] \\
\boxed{v_3\|x_4} \arrow[d, "0"'] \arrow[dr, "1"] & \boxed{v_4\|x_2} \arrow[d, "0,1"] & \boxed{v_5\|x_1} \arrow[dl, bend left, "0"] \\
\boxed{\mathbb{0}\|0} & \boxed{\mathbb{1}\|1} &
\end{tikzcd}$$

Then $\mathcal{B}$ accepts the language:

$$L_{\mathcal{B}} = \{0000, 1000, 0010, 1010, 0001, 0101, 1001, 1101, 0011, 0111\}$$

i.e. the satisfying assignments of $(\bar{x}_4 \wedge \bar{x}_2) \vee (x_4 \wedge (\bar{x}_3 \vee (x_3 \wedge \bar{x}_1)))$. For example, 1010 is accepted via the path $(s\|x_4) \xrightarrow{0} (v_1\|x_2) \xrightarrow{0} (v_4\|x_2) \xrightarrow{0} (\mathbb{1}\|1)$.

Remark 2.5. Nbps are closely related to *switching-and-rectifier networks* (*srns*) [8]. An srn $\mathcal{S} = (G, s, t, \mu)$ is a finite directed multigraph $G = (V, E)$ equipped with two vertices $s, t \in V$ and a partial edge-labelling $\mu : E \rightharpoonup X_n \times \mathbf{2}$. A word $w \in \mathbf{2}^n$ is accepted iff there exists a directed path from s to t such that for each label (x_i, b) we have $w_i = b$. Define the *size* $s(\mathcal{S})$ of an srn $\mathcal{S}$ to be the number of nodes, although it is more standard to consider the number of *labelled edges*.

Every nbp $\mathcal{B}$ has an equivalent srn $\mathcal{S}$ with at most one more node. Every srn $\mathcal{S}$ has an equivalent nbp $\mathcal{B}$ such that $s(\mathcal{B}) \leq 1 + n \cdot s(\mathcal{S})$. By 'equivalent' we mean they accept the same language. The constructions resemble the translation between Moore and Mealy machines, where we may assume $n > 0$.

(a) Given an nbp $\mathcal{B} = (G, s, \theta, \tau)$ one can assume it has exactly one 1-labelled node $\mathbb{1}$ (else introduce a new 1-labelled node $\mathbb{1}$ and merge 1-labelled nodes). Then an equivalent srn $\mathcal{S}$ is obtained by labelling each edge (u, v) in G by $(\theta(u), \tau(u, v))$ and requiring $s/\mathbb{1}$ to be the source/target node, respectively. Therefore $s(\mathcal{S}) \leq 1 + s(\mathcal{B})$.

(b) Given any srn $\mathcal{S} = (G, s, t, \mu)$ we may assume that:

1. t is a sink, else add a new target t' and unlabelled edge $t \to t'$.

2. Any two labelled edges with the same source are labelled by the same variable x_i i.e. they have labels (x_i, b_j) for $j = 1, 2$. One can force this by adding unlabelled edges to 'dummy' nodes, used as the source of conflicting edges. At most $(n-1) \cdot s(\mathcal{S})$ new nodes are required.
3. All edges are labelled, else replace every unlabelled edge (u, v) by two parallel edges (u, v), one labelled by $(x_1, 0)$ and one labelled by $(x_1, 1)$.

Then $\mathcal{S}$ and $\mathcal{S}'$ accept the same language and $s(\mathcal{S}') \leq 1 + n \cdot s(\mathcal{S})$. We obtain $\mathcal{B}$ from $\mathcal{S}'$ as follows. Replace each (x_i, b)-labelled edge (u, v) by the b-labelled edge (u, v) and set $\theta(u) = x_i$ (well-defined by (2)). Finally label $\theta(t) = 1$ and $\theta(v) = 0$ for each sink $v \neq t$, and let s be the source. Then $\mathcal{B}$ accepts the same language as $\mathcal{S}$, and $s(\mathcal{B}) = s(\mathcal{S}') \leq 1 + n \cdot s(\mathcal{S})$.

We now define a 'canonical form' for nondeterministic branching programs.

Definition 2.6. An nbp $\mathcal{B} = (G, s, \theta, \tau)$ is called *stratified* if

(1) for any pair $e \neq e'$ of parallel edges, one has $\tau(e) \neq \tau(e')$;

(2) G is acyclic;

(3) every node is reachable from s;

(4) all sinks are labelled by 0 or 1;

(5) every path from s to a sink has length $d(\mathcal{B})$.

Remark 2.7. Assuming that (3) holds, the conditions (2) and (5) are equivalent to the existence of a (necessarily unique) partition $V = V_0 \cup V_1 \cup \ldots \cup V_{d(\mathcal{B})}$ such that $V_0 = \{s\}$, all sinks are contained in $V_{d(\mathcal{B})}$, and every edge of $\mathcal{B}$ goes from V_i to V_{i+1} for some $0 \leq i < d(\mathcal{B})$. In fact, choose V_i to be the nodes reachable from s via a path of length i.

Example 2.8. The nbp in Example 2.4 is stratified.

Lemma 2.9. *Every nbp $\mathcal{B}$ has an equivalent stratified nbp of size $O(s(\mathcal{B})^4)$.*

Proof. For $1 \leq k \leq 5$, we show that every nbp $\mathcal{B} = (G = (V, E), s, \theta, \tau)$ satisfying the first $k - 1$ conditions of Definition 2.6 can be turned into an equivalent nbp satisfying the first k conditions.

$\underline{k = 1:}$ Whenever an nbp $\mathcal{B}$ has parallel edges with the same label, delete all but one of them. This yields an equivalent nbp satisfying (1).

$\underline{k = 2:}$ If $\mathcal{B}$ satisfies (1), construct the nbp $\mathcal{B}' = ((V', E'), s', \theta', \tau')$ where:

$$V' = V \times \{0, \ldots, s(\mathcal{B})\} \qquad E' = \{((u, i), (v, i+1)) : 0 \leq i < s(\mathcal{B}),\ (u, v) \in E\}$$
$$s' = (s, 0) \qquad \theta'(v, i) = \theta(v) \qquad \tau'((u, i), (v, i+1)) = \tau(u, v)$$

Clearly $\mathcal{B}'$ satisfies (1) and (2). Furthermore $\mathcal{B}'$ is equivalent to $\mathcal{B}$: if $w \in L_{\mathcal{B}}$ then there exists a w-consistent path $(s \| x_{k_0}) \xrightarrow{b_0} (v_1 \| x_{k_1}) \xrightarrow{b_1} \cdots \xrightarrow{b_{m-1}} (v_m \| 1)$ in $\mathcal{B}$ of length $m \le s(\mathcal{B})$, which immediately yields the w-consistent path:

$$((s,0) \| x_{k_0}) \xrightarrow{b_0} ((v_1,1) \| x_{k_1}) \xrightarrow{b_1} \cdots \xrightarrow{b_{m-1}} ((v_m,m) \| 1)$$

in $\mathcal{B}'$. This shows $L_{\mathcal{B}} \subseteq L_{\mathcal{B}'}$ and the reverse inclusion is proved analogously.

<u>$k = 3$:</u> Given an nbp satisfying (1) and (2), restrict to those nodes reachable from the source via directed paths.

<u>$k = 4$:</u> Now assume that $\mathcal{B}$ satisfies (1)-(3). Then relabelling all variable-labelled sinks with 0 yields an equivalent nbp satisfying (1)-(4).

<u>$k = 5$:</u> We may assume that every sink is reachable from s via a path of length $d(\mathcal{B})$, else merge sinks with the same label, so that the resulting nbp has at most two sinks, and add dummy $0,1$-labelled paths of length $d(\mathcal{B})$ from s to each sink. In view of Remark 2.7, define the partition:

$$V_i = \{v \in V : i \text{ is the length of any longest directed path } s \xrightarrow{*} v\}$$

for each $0 \le i \le d(\mathcal{B})$. Clearly $V_0 = \{s\}$ and $V_{d(\mathcal{B})}$ contains all sinks. Furthermore, every edge goes from V_i to V_j for some $i < j$. By replacing any such edge by a $0,1$-labelled path of length $j - i$, one makes sure that every edge goes from V_i' to V_{i+1}' for some $i < d(\mathcal{B})$. By Remark 2.7, the resulting nbp satisfies (1)-(5).

Observe that in steps 2 and 5, the size of the constructed nbp is at most quadratic in the size of the given one, while in the other steps the size does not increase. Therefore, starting from any nbp $\mathcal{B}$ we have shown how to construct an equivalent stratified nbp of size $O(s(\mathcal{B})^4)$ □

3 From Stratified Nbps to Nfas

In this section we associate to each stratified nbp $\mathcal{B}$ a nondeterministic finite automaton $N_{\mathcal{B}}$ of size polynomial in $s(\mathcal{B})$. Although $N_{\mathcal{B}}$ does not accept the same language as $\mathcal{B}$, they are closely related. We start by recalling the classical notion of a nondeterministic finite automaton.

Definition 3.1. A *nondeterministic finite automaton* (*nfa*) is a tuple $N = (Z, R_b, F, I)$ where Z is a finite set of states, $R_b \subseteq Z \times Z$ is a relation representing b-transitions (one for each $b \in \mathbf{2}$), $F \subseteq Z$ is the set of final states, and $I \subseteq Z$ is the set of initial states. The *size* $s(N)$ of N is the number of states, and the *depth* $d(N)$ of N is the length of any longest path starting in an initial state (defined for acyclic nfas). N *accepts* the language $L(N) \subseteq \mathbf{2}^*$ in the usual manner: $w \in L(N)$ iff there exists a w-path from some initial state to some final state.

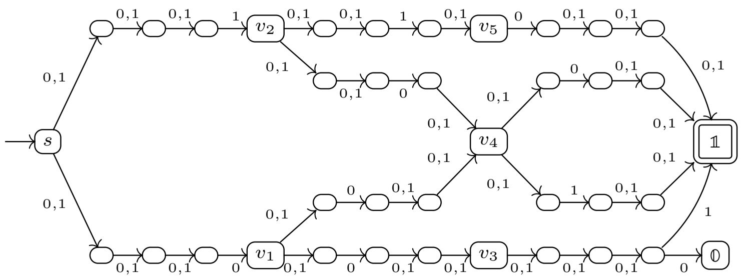
Remark 3.2. In analogy to Definition 2.6, we call an nfa N *stratified* if (i) N is acyclic and reachable, (ii) N has exactly one initial state, (iii) a state is final iff it is a sink, and (iv) all paths from the initial state to a final state have the same length $d(N)$. It is easy to see that every nfa accepting a finite language $L \subseteq \mathbf{2}^n$ can be turned into an equivalent stratified nfa with no more states. Moreover, a stratified nfa can be viewed as a stratified nbp: label final states with 1, label any nonfinal state with x_{i+1} if it is reachable via any word of length i, and choose the initial state as the source node. The resulting nbp is an instance of an *oblivious read-once* nbp: all paths from the source to a sink read each variable exactly once and in the same order.

Definition 3.3. Given a stratified nbp $\mathcal{B} = (G, s, \theta, \tau)$ over n variables, the nfa $N_{\mathcal{B}}$ is constructed as follows:
(1) Replace every edge $(u \| x_i) \xrightarrow{b} (v \| l)$ of $\mathcal{B}$ by a path of length n from u to v, where the i-th transition has label b and all others have labels 0 and 1.

$$\textcircled{u} \xrightarrow{0,1} \bigcirc \xrightarrow{0,1} \cdots \xrightarrow{0,1} \bigcirc \xrightarrow{b} \bigcirc \xrightarrow{0,1} \cdots \xrightarrow{0,1} \bigcirc \xrightarrow{0,1} \textcircled{v}$$

(2) The source s is the only initial state of $N_{\mathcal{B}}$, and a state is final if and only if it is labelled with 1.

Example 3.4. For the nbp $\mathcal{B}$ of Example 2.4 we obtain the following nfa $N_{\mathcal{B}}$:



Lemma 3.7 below describes various relevant properties of $N_{\mathcal{B}}$. But first we need a simple (yet crucial) definition.

Definition 3.5. For each $n, d \in \mathbb{N}$ and finite language $L \subseteq \mathbf{2}^n$ define:

$$d \cdot L := \{w^d : w \in L\} \subseteq \mathbf{2}^{nd}.$$

i.e. we take collection of all d-powers of words from L.

Remark 3.6. It follows that $\overline{d \cdot L} = d \cdot \overline{L} \cup \overline{d \cdot \mathbf{2}^n}$ for any $n, d \geq 0$ and $L \subseteq \mathbf{2}^n$. That is, this relative complement consists of (a) those d-powered words w^d where $w \notin L$, and (b) those words in $\mathbf{2}^{nd}$ which are not d-powers.

Lemma 3.7. *For any stratified nbp $\mathcal{B}$ over n variables, we have:*

(a) $s(N_{\mathcal{B}}) = O(n \cdot s(\mathcal{B})^2)$*;*

(b) $d(N_{\mathcal{B}}) = n \cdot d(\mathcal{B})$*;*

(c) $L(N_{\mathcal{B}}) \subseteq \mathbf{2}^{n \cdot d(\mathcal{B})}$*;*

(d) $d(\mathcal{B}) \cdot L_{\mathcal{B}} = L(N_{\mathcal{B}}) \cap (d(\mathcal{B}) \cdot \mathbf{2}^n)$*. That is, the* $d(\mathcal{B})$*-powers of words accepted by* $\mathcal{B}$ *are precisely those* $d(\mathcal{B})$*-powered words that* $N_{\mathcal{B}}$ *accepts.*

Proof. (a)-(c) follow directly from the construction of $N_{\mathcal{B}}$.
(d) Let $d = d(\mathcal{B})$. To prove '$\supseteq$', suppose $w \in L_{\mathcal{B}}$ so there exists some path:

$$s = (v_0 \| x_{k_0}) \xrightarrow{b_0} (v_1 \| x_{k_1}) \xrightarrow{b_1} \dots \xrightarrow{b_{d-1}} (\mathbb{1} \| 1) \qquad (*)$$

in $\mathcal{B}$ with $b_i = w_{k_i}$ for all i. This yields accepting paths of the form

$$(s = v_0 \xrightarrow{c_{0,1}} \dots \xrightarrow{c_{0,n}})(v_1 \xrightarrow{c_{1,1}} \dots \xrightarrow{c_{1,n}}) \dots (v_{d-1} \xrightarrow{c_{d-1,0}} \dots \xrightarrow{c_{d-1,n}} \mathbb{1}) \qquad (**)$$

in $N_{\mathcal{B}}$ where $c_{i,j} = b_i$ for $j = k_i$, and $c_{i,j} \in \mathbf{2}$ is arbitrary otherwise. In particular, choosing $c_{i,j} = w_j$ for all i yields an accepting path for the word w^d. Hence $w^d \in L(N_{\mathcal{B}}) \cap d \cdot \mathbf{2}^n$.

Conversely, any accepting path in $N_{\mathcal{B}}$ is induced by some path $(*)$ in $\mathcal{B}$ and has the form $(**)$. If a word w^d $(w \in \mathbf{2}^n)$ is accepted in $N_{\mathcal{B}}$ via $(**)$, we have $b_i = c_{i,k_i} = w_{k_i}$ for all i, so the path $(*)$ in $\mathcal{B}$ is consistent with w. It follows that $w \in L_{\mathcal{B}}$, which proves '$\subseteq$'. ☐

4 Characterisation of NL/*poly*

We are now ready to prove the announced characterisation of NL/*poly* via nondeterministic finite automata. We first introduce the relevant complexity classes.

Notation 4.1. For any language $L \subseteq \mathbf{2}^*$ and $n \geq 0$, let $L_n := L \cap \mathbf{2}^n$.

Definition 4.2. The complexity class nbp(*poly*) contains those $L \subseteq \mathbf{2}^*$ such that each L_n is accepted by some nbp $\mathcal{B}_n$, where $s(\mathcal{B}_n) \in n^{O(1)}$ i.e. their size is bounded polynomially in n. The complexity classes dbp(*poly*) and nfa(*poly*) are defined analogously: replace 'nbp' by 'dbp' or 'nfa', respectively.

The following relationships are well-known:

$$\mathsf{dbp}(\mathit{poly}) = \mathsf{L}/\mathit{poly} \qquad \mathsf{nbp}(\mathit{poly}) = \mathsf{NL}/\mathit{poly}$$

where L/*poly* (resp. NL/*poly*) consists of those languages accepted by some single log-space bounded deterministic (resp. nondeterministic) Turing machine with 'polynomial advice' [5]. These results are mentioned in [8], where our dbps correspond to 'BPs' and their notion of size agrees up to a linear factor. On the other hand, although our nbps are not quite the same as the switching-and-rectifier networks used in [8] (their size is the number of labelled edges), the above correspondence nevertheless holds, see [2, Theorem 1].

For any language $L \subseteq \mathbf{2}^*$ and non-zero monotonic function $t : \mathbb{N} \to \mathbb{N}$, define:

$$t * L := \bigcup_{n \geq 0} \overline{t(n) \cdot L_n} \ \subseteq \ \mathbf{2}^*$$

Recall that a function $t : \mathbb{N} \to \mathbb{N}$ is *monotonic* if $m \leq n$ implies $t(m) \leq t(n)$. We consider non-zero monotonic functions $t : \mathbb{N} \to \mathbb{N}$ to ensure that $n \mapsto n \cdot t(n)$ is injective, so that $t * L$ is actually a *disjoint* union.

Theorem 4.3. *$L \in \mathsf{nbp}(poly)$ if and only if there exists a poly-bounded non-zero monotonic function $t : \mathbb{N} \to \mathbb{N}$ such that $t * L \in \mathsf{nfa}(poly)$.*

The proof uses the following two results. The first is a corollary of the Immerman-Szelepcsényi theorem, as mentioned in [8].

Theorem 4.4 ([4, 10])**.** *The class $\mathsf{nbp}(poly)$ is closed under complement:*

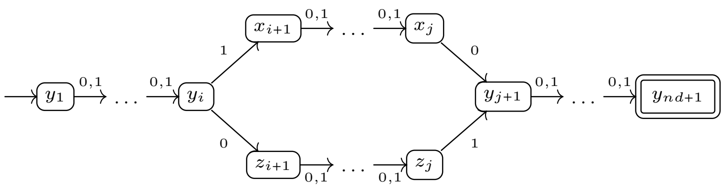
$$L \in \mathsf{nbp}(poly) \quad \textit{iff} \quad \widetilde{L} \in \mathsf{nbp}(poly)$$

for any language $L \subseteq \mathbf{2}^*$.

The second result provides poly-size nfas for certain finite languages.

Lemma 4.5. *For all $n, d \geq 0$, there exists an nfa with $O(n^2 d^3)$ states accepting the language $\overline{d \cdot \mathbf{2}^n} \subseteq \mathbf{2}^{nd}$.*

Proof. $\overline{d \cdot \mathbf{2}^n}$ consists of all words $w \in \mathbf{2}^{nd}$ such that there exists $1 \leq i < j \leq n \cdot d$ where (i) $i = j$ mod n, and (ii) $w_i \neq w_j$. The following nfa with $O(nd)$ states accepts all such words for a fixed pair (i, j):



Taking the disjoint union of these nfas yields an nfa accepting $\overline{d \cdot \mathbf{2}^n}$. Since there are $n\binom{d}{2} = O(nd^2)$ pairs (i, j) satisfying (i), this nfa has $O(n^2 d^3)$ states. □

Remark 4.6. On the other hand, poly-size nfas do *not* exist for the relative complements $d \cdot \mathbf{2}^n \subseteq \mathbf{2}^{nd}$. In fact, a state-minimal nfa for $d \cdot \mathbf{2}^n$ is obtained from its state-minimal dfa by deleting the state accepting the empty language. The latter is exponential in n for any fixed $d > 1$. To see this, one can use the fact that $d \cdot \mathbf{2}^n$ defines a *linear code* i.e. a linear subspace of $\mathbb{Z}_2^{nd}$.

Proof of Theorem 4.3. Let $L \in \mathsf{nbp}(poly)$. Then also $\widetilde{L} \in \mathsf{nbp}(poly)$ by Theorem 4.4, so there exists a family of nbps $\mathcal{B}_n$ $(n \geq 0)$ such that $\mathcal{B}_n$ accepts $(\widetilde{L})_n = \overline{L_n}$ and $s_n := s(\mathcal{B}_n)$ is polynomially bounded in n. By Lemma 2.9, we may assume that the nbps $\mathcal{B}_n$ are stratified. Moreover, we assume that the depths $d_n :=$

$d(\mathcal{B}_n)$ are non-zero and grow monotonically (otherwise add dummy paths). Let $N_n := N_{\mathcal{B}_n}$ be the nfa associated to B_n (see Definition 3.3). Then:

$$\begin{aligned}
\overline{d_n \cdot L_n} &= d_n \cdot \overline{L_n} \cup \overline{d_n \cdot \mathbf{2}^n} && \text{see Remark 3.6}\\
&= d_n \cdot L_{\mathcal{B}_n} \cup \overline{d_n \cdot \mathbf{2}^n} && \text{since } L_{\mathcal{B}_n} = \overline{L_n}\\
&= (L(N_n) \cap d_n \cdot \mathbf{2}^n) \cup \overline{d_n \cdot \mathbf{2}^n} && \text{by Lemma 3.7}(d)\\
&= (L(N_n) \cup \overline{d_n \cdot \mathbf{2}^n}) \cap (d_n \cdot \mathbf{2}^n \cup \overline{d_n \cdot \mathbf{2}^n})\\
&= L(N_n) \cup \overline{d_n \cdot \mathbf{2}^n}
\end{aligned}$$

By Lemma 3.7, N_n has $O(ns_n^2)$ states. Moreover, by Lemma 4.5 there exists an nfa N'_n accepting $\overline{d_n \cdot \mathbf{2}^n}$ with $O(n^2 d_n^3)$ states, this being polynomial in n because $d_n \leq s_n$. Taking the disjoint union of the nfas N_n and N'_n yields a polynomial-sized nfa N''_n for $L(N_n) \cup \overline{d_n \cdot \mathbf{2}^n} = \overline{d_n \cdot L_n}$.

Define the non-zero monotonic function $t : \mathbb{N} \to \mathbb{N}$ as $t(n) = d_n$. Then we obtain a polynomial-sized family of nfas M_k accepting $(t * L)_k$ as follows. If $k = nd_n$ for some n (unique because t is non-zero and monotonic), we have $(t * L)_k = \overline{d_n \cdot L_n}$, so take $M_k = N''_n$. The size of N''_n is polynomial in n, hence also in $k = nd_n$. If k is not of the form nd_n for some n then $(t * L)_k = \varnothing$, so let M_k be a one-state nfa accepting $\varnothing$. This proves $t * L \in \mathsf{nfa}(\mathit{poly})$.

For the converse, suppose we have $t * L \in \mathsf{nfa}(\mathit{poly})$ for some non-zero monotonic function $t : \mathbb{N} \to \mathbb{N}$. Then there exists a family of polynomial-sized nfas N_k ($k \geq 0$) accepting $(t * L)_k$. By Remark 3.2, we can turn N_k into an equivalent stratified (oblivious read-once) nbp $\mathcal{B}_k$ of the same size. Then by Theorem 4.4, there also exists a family of polynomial-sized nbps $\mathcal{B}'_k$ accepting $(\widetilde{t * L})_k$. If $k = n \cdot t(n)$ for some n (necessarily unique), then $(\widetilde{t * L})_k = t(n) \cdot L_n$ and the size of $\mathcal{B}'_k$ is polynomial in n, since it is polynomial in $k = n \cdot t(n)$ and $t(n)$ is polynomial in n. The nbp $\mathcal{B}'_k$ has $n \cdot t(n)$ variables $x_1, \ldots, x_{n \cdot t(n)}$, and replacing all node labels $x_{m \cdot n + i}$ (where $0 \leq m < t(n)$ and $1 \leq i \leq n$) by x_i yields an nbp $\mathcal{B}''_n$ accepting L_n whose size is polynomial in n. It follows that $L \in \mathsf{nbp}(\mathit{poly})$. □

Note 4.7. Consequently, for any language $L \in \mathsf{dbp}(\mathit{poly})$ there exists a non-zero monotonic $t \in n^{O(1)}$ such that $t * L \in \mathsf{nfa}(\mathit{poly})$. However one can also prove this without using the Immerman-Szelepcsényi theorem i.e. without Theorem 4.4. For suppose $L \in \mathsf{dbp}(\mathit{poly})$, so we have poly-sized dbp's $\mathcal{B}'_n$ accepting L_n with depth $d_n = d(\mathcal{B}'_n)$. Then by Lemma 3.7(d) we have $L(N_{\mathcal{B}'_n}) \cap (d_n \cdot \mathbf{2}^n) = d_n \cdot L_{\mathcal{B}'_n}$, so taking the relative complement we obtain:

$$\overline{d_n \cdot L_{\mathcal{B}'_n}} = \overline{L(N_{\mathcal{B}'_n})} \cup \overline{d_n \cdot \mathbf{2}^n}$$

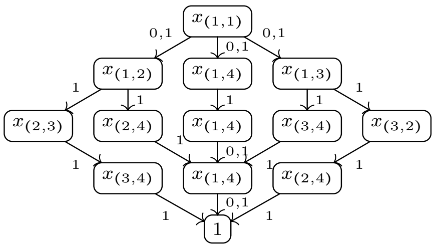
By Lemma 4.5 a small nfa accepting $\overline{d_n \cdot \mathbf{2}^n}$ exists, but there is *also* a small nfa accepting $\overline{L(N_{\mathcal{B}'_n})}$: (i) we have a small nfa accepting $L(N_{\mathcal{B}'_n})$ via Definition 3.3, and (ii) the latter nfa is a *partial deterministic* automaton since we started with a dbp. To accept the complement one makes 1-labelled states non-final and 0-labelled states final. Since $L_{\mathcal{B}'_n} = L_n$ the rest of the proof proceeds as before.

5 Encoding Unreachability

To illustrate our constructions we start by considering *reachability*: given a finite directed graph (V, E) and vertices $s, t \in V$, decide whether there exists a directed path from s to t. Since reachability is in NL (in fact NL-complete), it necessarily lies in $\mathsf{NL}/poly$ and therefore has poly-size nondeterministic branching programs. Given $V_k = \{v_1, \ldots, v_k\}$ take the lexicographic ordering on $[k] \times [k]$ where $[k] = \{1, \ldots, k\}$. Next, consider the finite language:

$$W(k) = \{w \in \mathbf{2}^{k^2} : \text{there is a directed path } v_1 \to^* v_k \text{ in } (V_k, E_w)\}$$

where if $w = e_{(1,1)} \ldots e_{(k,k)}$ then $E_w = \{(v_i, v_j) \in (V_k)^2 : e_{(i,j)} = 1\}$, so that w is the adjacency matrix of the graph (V_k, E_w). Then $W(k)$ is accepted by a stratified nbp $\mathcal{B}_k$ of size polynomial in k, shown below in the case where $k = 4$:



$\mathcal{B}_4$ has depth $d(\mathcal{B}_4) = 4$ and is defined over $n = k^2 = 16$ variables. It induces the nfa $N = N_{\mathcal{B}_4}$ where: (i) edges labelled by $0, 1$ are replaced by 16 such edges, (ii) edges labelled by 1 are replaced by 16 edges $(\xrightarrow{0,1})^{r_1} \xrightarrow{1} (\xrightarrow{0,1})^{r_2}$ in a suitable manner. Next, recall the language $\overline{4 \cdot \mathbf{2}^n} \subseteq \mathbf{2}^{n \cdot 4} = \mathbf{2}^{64}$. It is accepted by the pointed nfa N' obtained by taking the disjoint union of $16 \cdot \binom{4}{2} = 96$ nfas i.e. for each $1 \leq i < j \leq 64$ with $i = j \bmod n$, one needs an nfa accepting all $w \in \mathbf{2}^{64}$ such that $w_i \neq w_j$. Let N'' be the disjoint union of N and N'. Then:

$$L(N'') \;=\; L(N) \cup \overline{4 \cdot \mathbf{2}^n} \;=\; \overline{\overline{L(N)} \cap 4 \cdot \mathbf{2}^n} \;=\; \overline{\{w^4 \in \mathbf{2}^{64} : w \in \overline{W(4)}\}}$$

because w^4 is *not* accepted by N iff $w \notin W(4)$. Then we have encoded *unreachability* in the nfa N'', which may be understood in two ways:

1. By Remark 3.2 we can view N'' as an oblivious read-once nbp and apply Theorem 4.4 to obtain an nbp accepting $\overline{L(N'')}$ with only a poly-sized blow up. Then we can restrict from 64 to 16 variables as in the proof of Theorem 4.3, yielding an nbp accepting $\overline{W(4)}$.

2. That unreachability has been 'encoded' can also be seen directly. Make final states non-final in N'', and add to each initial state q the structure:

$$\begin{tikzcd} \boxed{q} \arrow[r, "0,1"] & \boxed{q_1} \arrow[r, "0,1"] & \cdots \arrow[r, "0,1"] & \boxed{\boxed{q_{64}}} \arrow[r, "0,1"] & \boxed{q_{65}} \arrow[loop above, "0,1"] \end{tikzcd}$$

where the q_i are new states. The resulting nfa accepts $\overline{L(N'')}$ via *universal* acceptance. Thus $w \in \mathbf{2}^{16}$ represents a directed graph with no directed path from v_1 to v_4 iff w^4 is accepted by this universal machine.

References

[1] Janusz Brzozowski and Hellis Tamm. Theory of Átomata. In *Proc. 15th International Conference on Developments in Language Theory (DLT'11)*, volume 6795 of *Lecture Notes Comput. Sci.*, pages 105–116. Springer, 2011.

[2] Carsten Damm and Markus Holzer. Inductive counting below logspace. In Igor Prvara, Branislav Rovan, and Peter Ruzika, editors, *Mathematical Foundations of Computer Science 1994*, volume 841 of *Lecture Notes in Computer Science*, pages 276–285. Springer Berlin Heidelberg, 1994.

[3] François Denis, Aurélien Lemay, and Alain Terlutte. Residual finite state automata. *Fund. Inform.*, XX:1–30, 2002.

[4] Neil Immerman. Nondeterministic space is closed under complementation. *SIAM J. Comput.*, 17(5):935–938, October 1988.

[5] Richard M. Karp and Richard J. Lipton. Some connections between nonuniform and uniform complexity classes. In *Proceedings of the twelfth annual ACM symposium on Theory of computing*, STOC '80, pages 302–309, New York, NY, USA, 1980. ACM.

[6] Robert S.R. Myers, Jiří Adámek, Henning Urbat, and Stefan Milius. Canonical nondeterministic automata. To be published. Available at `www.stefan-milius.eu`.

[7] Robert S.R. Myers, Jiří Adámek, Henning Urbat, and Stefan Milius. Nondeterministic closure automata and state-minimal nondeterministic acceptors. Submitted. Available on request.

[8] Alexander A. Razborov. Lower bounds for deterministic and nondeterministic branching programs. In *Proceedings of the 8th International Symposium on Fundamentals of Computation Theory*, FCT '91, pages 47–60, London, UK, UK, 1991. Springer-Verlag.

[9] Arto Salomaa, Derick Wood, and Sheng Yu. On the state complexity of reversals of regular languages. *Theoretical Computer Science*, 320(23):315 – 329, 2004.

[10] Róbert Szelepcsényi. The method of forced enumeration for nondeterministic automata. *Acta Informatica*, 26(3):279–284, 1988.

[11] Jean Vuillemin and Nicolas Gama. Efficient Equivalence and Minimization for Non Deterministic Xor Automata. Research report, LIENS, May 2010.