Quantifying Nondeterminism in Finite Automata

Alexandros Palioudakis, Kai Salomaa and Selim G. Akl

Abstract - Various ways of quantifying the nondeterminism in finite automata have been considered since the 1970's. Roughly speaking, a nondeterminism measure can count the number of accepting computations (ambiguity), the number of all computations (computation width) or the amount of nondeterminism on a single best (or worst) computation on a given input. This paper surveys results on the growth rate of the nondeterminism measures and the descriptional complexity of nondeterministic finite automata with limited nondeterminism.

Key words and phrases : finite automata, limited nondeterminism, state complexity

1 Introduction

As finite automata are not equipped with external memory, for a quantitative understanding of regular languages the commonly used descriptional complexity measures count the number of states or the number of transitions of a (nondeterministic) finite automaton. The descriptional complexity of finite automata has been studied for over half a century [19, 21, 22, 23], and there has been particularly much work done over the last two decades [9, 14, 34].

Besides the number of states (or transitions) a further resource to quantify is the amount of nondeterminism used by a finite automaton [13]. Various ways of quantifying the amount of nondeterminism in finite automata have been considered. The degree of ambiguity counts the number of accepting computations on a given input. The degree of ambiguity of finite automata has been considered since the 1970's and it is, perhaps, the most well studied measure [16, 29, 33]. Other nondeterminism measures are based on the amount of nondeterminism used in all, accepting as well as non-accepting, computations and further distinctions arise depending on whether the measure is a best case or a worst case measure [6, 7, 25, 28].

The *computation width* (a.k.a. 'leaf size' [1, 11], a.k.a. 'tree width' [24, 28]) of a nondeterministic finite automaton (NFA) measures the total number of computations on a given input. On the other hand, the *guessing measure* [7] counts the number of bits of information the NFA needs to encode the nondeterministic choices on the "best" accepting path, that is, the path using the least amount of nondeterminism. A closely related best case measure is *deviation number* [5]. The product of the degrees of nondeterministic choices on the best accepting computation is called *branching* [7] and, by definition, the guessing of an NFA A is the logarithm of the branching of A. A worst case variant of the branching measure is called *trace* [25, 28].

General references on descriptional complexity include the surveys by Gao et al. [3], Goldstine et al. [6], Holzer and Kutrib [9], Kutrib and Pighizzini [14] and the handbook article by Yu [34], and more comprehensive references can be found therein.

2 Preliminaries

We assume that the reader is familiar with the basics of finite automata and regular languages [32, 34].

The set of strings over a finite alphabet Σ is Σ^*, the length of $w \in \Sigma^*$ is $|w|$, the set of strings of length at most m is $\Sigma^{\leq m}$ and ε is the empty string. The set of positive integers is $\mathbb{N}$ and the cardinality of a finite set F is $|F|$.

A *nondeterministic finite automaton* (NFA) is a 5-tuple $A = (Q, \Sigma, \delta, q_0, F)$ where Q is the finite set of states, Σ is the input alphabet, $\delta : Q \times \Sigma \to 2^Q$ is the transition function, $q_0 \in Q$ is the initial state and $F \subseteq Q$ is the set of final states. The transition function δ is in the usual way extended as a function $Q \times \Sigma^* \to 2^Q$ and the language recognized by A is $L(A) = \{w \in \Sigma^* \mid \delta(q_0, w) \cap F \neq \emptyset\}$. By the size of the NFA A, size(A), we mean the number of states, that is, the cardinality of Q.

A transition of A is a triple $\mu = (q, a, p)$, $q, p \in Q$, $a \in \Sigma$ such that $p \in \delta(q, a)$. The transition μ is *nondeterministic* if $\delta(q, a)$ has at least 2 elements, and otherwise μ is deterministic. The degree of nondeterminism of the transition (q, a, p) is the cardinality of $\delta(q, a)$.

If all transitions are deterministic, A is a *deterministic finite automaton* (DFA). Note that we allow a DFA to have undefined transitions.

A *partial computation* of A from the initial state q_0 to state s on input $w = a_1 \cdots a_\ell$, $a_i \in \Sigma$, $i = 1, \ldots, \ell$, $\ell \geq 1$, is a sequence of transitions

$$C = (\mu_1, \ldots, \mu_k),\ \ k \leq \ell,\ \mu_i = (q_{i-1}, a_i, q_i),\ 1 \leq i \leq k,\ s = q_k$$

such that either $k = \ell$ or $k < \ell$ and $\delta(q_k, a_{k+1})$ is undefined. The sequence of transitions C is a *(full) computation* (of A) on w if $k = \ell$. An accepting computation is a (full) computation to a final state. The set of partial computations (respectively, accepting computations) of A on a string w is part-comp$_A(w)$ (respectively, acc-comp$_A(w)$).

A computation (of A on string w) is a special case of a partial computation (of A on w). Partial computations are sequences of transitions that either consume the entire string w or become “stuck” before processing the entire string. If a partial computation does not reach the end of w, it must have encountered an undefined transition, that is, a prefix of a partial computation is not a partial computation. When using the term computation, we always mean a full computation.

The minimal size of a DFA or an NFA recognizing a regular language L is called the deterministic (nondeterministic) state complexity of L and denoted, respectively, $\mathrm{sc}(L)$ and $\mathrm{nsc}(L)$. Note that we allow DFAs to be incomplete and, consequently, the deterministic state complexity of L may differ by one from a definition using complete DFAs.

A *deterministic finite automaton with multiple initial states* (MDFA) is an extension of a DFA that allows more than one initial state [10, 12]. An MDFA A accepts string w if the computation of A on w started from any of the initial states accepts. A k-MDFA is an MDFA with $k \in \mathbb{N}$ initial states.

Following the convention used by the overwhelming majority of the literature, including the central work done on various nondeterminism measures [6, 7, 8, 11], we have defined an NFA to have only one initial state. This means, that strictly speaking, a k-MDFA is not an NFA. However, a k-MDFA can always be converted to an NFA with one more state. For dealing with MDFAs, defining NFAs to have more than one initial state would probably be a more natural choice. However, allowing more than one initial state changes the precise relationships between the different nondeterminism measures. In this brief survey we do not wish to include technical recomputations of the various bounds and consequently use the standard definition.

3 Measures of nondeterminism

The ambiguity of an NFA counts the number of accepting computations while the computation width counts the total number of partial computations. Additionally we recall the definitions of further measures that are based on a best accepting computation or a worst computation. In the rest of this subsection, $A = (Q, \Sigma, \delta, q_0, F)$ is always an NFA.

The *degree of ambiguity* [16, 29] of A on $w \in \Sigma^*$, $\mathrm{amb}_A(w)$, is the number of accepting computations of A on w. The degree of ambiguity of A on strings of length m is defined as

$$\mathrm{amb}_A(m) = \max\{\mathrm{amb}_A(w) \mid w \in \Sigma^{\leq m}\}, \quad m \in \mathbb{N}.$$

We say that A has finite ambiguity if the value $\mathrm{amb}_A =^{\mathrm{def}} \sup_{m \in \mathbb{N}} \mathrm{amb}_A(m)$ is finite. Otherwise, the ambiguity of A is infinite (unbounded). If $\mathrm{amb}_A = 1$, A is an *unambiguous finite automaton* (UFA).

The *computation width* of A on $w \in \Sigma^*$, $\mathrm{cw}_A(w)$, is the number of partial computations of A on w. The notion of computation width has been called also 'leaf size' [11] or 'tree width' [24, 28], and it can be viewed as the number of leaves in the computation tree of A on w. Note that the number of partial computations, as defined in section 2, coincides with the number of leaves in the computation tree of A on w, as defined in [24]. Again, the computation width of A on strings of length $m \in \mathbb{N}$ is defined as $\mathrm{cw}_A(m) = \max\{\mathrm{cw}_A(w) \mid w \in \Sigma^{\leq m}\}$, and we say that A has finite computation width if the quantity $\mathrm{cw}_A =^{\mathrm{def}} \sup_{m\in\mathbb{N}} \mathrm{cw}_A(m)$ is finite.

Next we consider measures that, instead of counting the number of all (accepting) computations, are based on a single computation of A. Consider a partial computation of A on input $w = a_1 a_2 \cdots a_\ell$,

$$C = (\mu_1, \ldots, \mu_k), \;\; k \leq \ell, \; \mu_i = (q_i, a_i, q_{i+1}). \tag{1}$$

The *guessing* $\gamma_A(C)$ (respectively, the *branching* $\beta_A(C)$) of the partial computation C is

$$\gamma_A(C) = \sum_{i=1}^{k} \log_2 |\delta(q_i, a_i)|, \qquad \beta_A(C) = \prod_{i=1}^{k} |\delta(q_i, a_i)|.$$

Intuitively, $\gamma_A(C)$ represents the amount of guessing, in bits of information, that occurs during the computation. If A is a DFA, the amount of guessing in any computation of A is zero. By definition $\beta_A(C) = 2^{\gamma_A(C)}$ [7].

Note that in order to define the guessing and branching measures [7, 8] above it would be sufficient to consider only full computations C. We have allowed the possibility that C is a partial computation because below we consider also a corresponding worst case measure.

The *deviation number* [5] of the computation C (as in (1)) is defined as $\mathrm{dn}_A(C) = \sum_{i=1}^{k}(|\delta(q_i, a_i)| - 1)$. Note that if all transitions of A are either deterministic or have degree of nondeterminism two, then the deviation number coincides with the guessing measure, that is, for any computation C, $\mathrm{dn}_A(C) = \gamma_A(C)$. The deviation number directly counts the number of paths that branch out from a nondeterministic computation. On the other hand, the guessing of one transition with degree of nondeterminism three is less than the guessing of a computation where we achieve three choices by first making a binary choice between states q_1 and q_2 and then another binary choice in state q_1.

The guessing of a string $w \in L(A)$ is the guessing of the best accepting computation: $\gamma_A(w) = \min\{\gamma_A(C) \mid C \in \text{acc-comp}_A(w)\}$. Similarly, the branching (respectively, the deviation number) of A on $w \in L(A)$, $\beta_A(w)$, (respectively, $\mathrm{dn}_A(w)$) is defined in terms of the best accepting computation of A on w.

As with ambiguity and computation width, the value of the measures on strings of length $m \in \mathbb{N}$ is their value on the string of length m that needs the largest amount of nondeterminism. Let α be one of the measures γ, β or dn. Then $\alpha_A(m) = \max\{\alpha_A(w) \mid w \in L(A) \cap \Sigma^{\leq m}\}$. We say that the α-value of A is finite if the quantity $\alpha_A =^{\mathrm{def}} \sup_{m\in\mathbb{N}} \alpha_A(m)$ is finite.

A variant of the guessing and branching measures that is defined in terms of the worst computation of A on a string (as opposed to the best computation used to define $\gamma_A(w)$ and $\beta_A(w)$) has been considered [25]. When defining a worst case measure, we should consider also failed (partial) computations and the values are not restricted to strings in $L(A)$.

The *maximum guessing* of A on a string $w \in \Sigma^*$ is $\gamma_A^{\max}(w) = \max\{\gamma_A(C) \mid C \in \text{part-comp}_A(w)\}$, and the *trace* of A on w is $\tau_A(w) = 2^{\gamma_A^{\max}(w)}$ [25]. Let α be either $\gamma^{\max}$ or τ and $m \in \mathbb{N}$. Then $\alpha_A(m) = \max\{\alpha_A(w) \mid w \in \Sigma^m\}$. The maximum guessing (respectively, the trace) of A is said to be finite if the value $\gamma_A^{\max} =^{\text{def}} \sup_{m\in\mathbb{N}} \gamma_A(m)$ (respectively, $\tau_A =^{\text{def}} \sup_{m\in\mathbb{N}} \tau_A(m)$) is finite.

Instead of counting the amount of guessing in bits of information, Leung [17] counts the number of nondeterministic steps on the best accepting computation and Hromkovič et al. [11] use the *advice measure* that counts the number of nondeterministic steps on the worst computation on a given input. These measures are within a multiplicative factor of $\max\{\log_2 |\delta(q,a)| \mid \delta(q,a) \neq \emptyset\}$ of the guessing and maximum guessing, respectively.

For descriptional complexity comparisons of the various nondeterminism measures the relevant quantity is the optimal size of an NFA where nondeterminism is bounded by one of the measures. Let α be one of the measures ambiguity (amb), computation width (cw), guessing (γ) , branching (β), deviation number (dn), maximum guessing ($\gamma^{\max}$), or trace (τ). For a regular language L and $k \in \mathbb{N}$, we denote

$$\text{nsc}_{\alpha\leq k}(L) = \min\{\text{size}(A) : \alpha_A \leq k,\ L(A) = L\}.$$

Above we defined the nondeterministic state complexity of L only in the case where the value of the α-measure is bounded by a constant. In the natural way, the notion can be extended for NFAs A where the α-value on strings of length m, $\alpha_A(m)$, is bounded by a value $f(m)$ for some function f. Except for the descriptional complexity comparisons between different ambiguity growth rates (discussed in section 5), very few results are known on the descriptional complexity of NFAs with non-constant nondeterminism that is bounded by input length.

4 Growth rate of nondeterminism

For NFAs with unbounded nondeterminism, the growth rate of the nondeterminism measures is counted as a function of input length. In this section we discuss results on the growth rate of nondeterminism, bounds for finite nondeterminism in terms of the number of states and comparisons between the different nondeterminism measures. The results deal mainly with the nondeterminism of a specific NFA because, in general, not much is known about descriptional complexity comparisons between the various nondeterminism measures in the case where the amount of nondeterminism is unbounded.

It is known since the 1970's that the problem of determining whether the degree of ambiguity of a given NFA is finite, polynomial or exponential is decidable [20, 30]. The problem of computing the degree of ambiguity of an NFA with finite ambiguity was shown by Chan and Ibarra [2] to be PSPACE-hard. On the other hand, based on a characterization of NFAs with, respectively, polynomial and exponential ambiguity due to Ravikumar and Ibarra [29], Weber and Seidl [33] have given a polynomial time algorithm for testing whether the degree of ambiguity of an NFA is finite or whether it grows polynomially or exponentially.

Hromkovič et al. [11] have characterized the possible growth rates of computation width (for which they use the name 'leaf size').

Theorem 4.1 (Hromkovič et al. [11]) *For any NFA A, the function* $\mathrm{cw}_A(m)$ *is either bounded by a constant, or between linear and polynomial in m, or otherwise in* $2^{\Theta(m)}$.

The above characterization can be effectively decided. An NFA A has unbounded computation width if and only if some cycle of A contains a nondeterministic transition and this observation yields a simple polynomial time algorithm to test whether cw_A is finite.

Hromkovič et al. [11] have shown that if A is an n-state NFA with bounded computation width, then $\mathrm{cw}_A \le n^n$. Palioudakis et al. [24] have improved the bound to 2^{n-2} and, furthermore, shown that all values up to 2^{n-2} are possible. We say that an NFA A has *optimal computation width* if $L(A)$ cannot be recognized by an NFA B where $\mathrm{size}(B) \le \mathrm{size}(A)$, $\mathrm{cw}_B \le \mathrm{cw}_A$ and one of the inequalities is strict.

Theorem 4.2 (Palioudakis et al. [24]) *The bounded computation width of an n-state NFA is at most 2^{n-2}. For every $n \ge 2$ and $1 \le k \le 2^{n-2}$ there exists an n-state NFA A over a binary alphabet with optimal computation width k.*

The following relationships between the growth rates of maximum guessing, computation width and ambiguity are known. Hromkovič et al. [11] use "advice" instead of maximum guessing and a different name for computation width. The maximum guessing of an NFA A is always within a constant factor of the advice measure of A.

Theorem 4.3 (Hromkovič et al. [11]) *If A is a minimal NFA, then*

$$(\forall m \in \mathbb{N})\ \ \gamma_A^{\max}(m), \mathrm{amb}_A(m) \le \mathrm{cw}_A(m) = O(\mathrm{amb}_A(m) \cdot \gamma_A^{\max}(m)).$$

Goldstine et al. [8] have constructed NFAs having unbounded, but sublinear guessing, and have considered the relation between ambiguity and guessing.

Theorem 4.4 (Goldstine et al. [8]) *For every $k \in \mathbb{N}$, there is an NFA A such that $\gamma_A(m) = \Theta(m^{\frac{1}{k}})$.*

If $\gamma_A(m) = \Theta(m^{\frac{1}{k}})$ with $k \geq 2$, then the ambiguity of A must be unbounded. If $\gamma_A(m)$ is $\Theta(m)$ or $\Theta(1)$, then the ambiguity of A may be either bounded or unbounded.

Directly from the definitions it follows that if A has finite computation width, then the guessing and branching of A is finite but the converse implication does not hold, in general. The computation width of A is finite if and only if the trace of A is finite.

Lemma 4.1 (Palioudakis et al. [25]) *For any NFA A with finite computation width, $\mathrm{cw}_A \leq \tau_A \leq 2^{\mathrm{cw}_A - 1}$.*

Furthermore, it is known that the bounds cannot be improved [25]. Contrasting the result of Theorem 4.1, the growth rate of trace cannot be polynomial.

Theorem 4.5 (Palioudakis et al. [25]) *If A is an n-state NFA then either the trace of A is finite or, for all $m \in \mathbb{N}$, $\tau_A(m) \geq 2^{\lfloor \frac{m}{n} \rfloor}$.*

Determining the possible growth rates of the corresponding "best case" measure branching is considerably more difficult. By Theorem 4.4 we know that there exist NFAs A such that $\beta_A(m) = 2^{\Theta(m^{\frac{1}{k}})}$, for all $k \geq 1$, however, it is not known whether the branching of an NFA can be polynomial but unbounded.

Open problem 4.1 Is the growth rate of $\beta_A(m)$ superpolynomial for all NFAs that have infinite branching?

If A is an NFA over a unary alphabet and A has unbounded branching, then $\beta_A(m) = 2^{\Omega(m)}$ [27].

5 Limited nondeterminism and state complexity

It is well known that an incomplete DFA equivalent to an n state NFA needs in the worst case $2^n - 1$ states. Schmidt [31] first developed methods to prove lower bounds for the size of UFAs and gave a family of n state NFAs A_n with finite ambiguity such that any UFA for $L(A_n)$ needs $2^{\Omega(\sqrt{n})}$ states. Schmidt's lower bound criterion can be viewed as a special case of the communication complexity techniques [11]. The worst case trade-off for converting an NFA with finite ambiguity to a UFA was given by Leung [18].

Theorem 5.1 (Leiss [15], Leung [18]) *The worst case size of a UFA equivalent to an n state NFA with finite ambiguity, and the worst case size of a DFA equivalent to an n state UFA, is $2^n - 1$.*[1]

Descriptional complexity comparisons between NFAs with different unbounded ambiguity were first considered by Ravikumar and Ibarra [29]. Leung [16] has established the following important separation between polynomial and exponential ambiguity.

Theorem 5.2 (Leung [16]) *For every $n \in \mathbb{N}$ there exists an NFA A_n with n states such that any polynomially ambiguous NFA for $L(A_n)$ needs $2^n - 1$ states.*

Hromkovič et al. [11] have given a substantially simpler proof for an exponential size gap between NFAs of polynomial ambiguity and general NFAs, however, their result does not yield the precise $2^n - 1$ lower bound of Theorem 5.2. The main open problem for NFAs with unbounded ambiguity is the size trade-off between NFAs with finite ambiguity and polynomial ambiguity.

Open problem 5.1 Is there an exponential gap between (optimal) sizes of NFAs with finite ambiguity and polynomial ambiguity?

Goldstine et al. [7] have shown that there exist NFAs A with n states such that any NFA with finite branching equivalent to A needs to have almost worst case size blow-up.

Theorem 5.3 (Goldstine et al. [7]) *For $n \in \mathbb{N}$ there exists a regular language L with $\mathrm{nsc}(L) = n$ such that for any $k \in \mathbb{N}$, $\mathrm{nsc}_{\beta \le k}(L) = 2^{n-1}$.*

Conversely, it is known that there exist NFAs that require almost the worst case size blow-up of determinization and, furthermore, different finite amounts of branching allow incremental savings in the number of states. The theorem below is the "spectrum result" of [7] stated in a slightly simplified form and translated into our notations.

Theorem 5.4 (Goldstine et al. [7]) *For $n \geq 2$ there exists a regular language L_n such that $\mathrm{nsc}(L_n) = n + 1$, $\mathrm{sc}(L_n) = 2^n$ and*

$$\mathrm{nsc}_{\beta \le k}(L_n) \textit{ is between } \begin{cases} 2^{\frac{n}{k}} \textit{ and } 2k \cdot 2^{\frac{n}{k}} - 1 \textit{ for } 2 \le k < \frac{n}{\log_2 n}, \\ n+1 \textit{ and } 2k \cdot 2^{\frac{n}{k}} - 1 \textit{ for } \frac{n}{\log_2 n} \le k < n, \\ n+1 \textit{ and } 4n - 1 \textit{ for } k \geq n. \end{cases}$$

[1]The latter is given as 2^n in [15, 18] because the paper requires the DFAs to be complete.

By Theorems 5.1 and 5.3 we know that a finite degree of ambiguity or finite branching does not improve the worst case exponential cost of determinization. On the other hand, an NFA with finite computation width has an equivalent DFA of polynomial size.

Theorem 5.5 (Palioudakis et al. [24]) *If A is an NFA with n states and $\mathrm{cw}_A \leq k$ for some $k \leq n-1$, then $\mathrm{sc}(L(A)) \leq 1+\sum_{j=1}^{k}\binom{n-1}{j}$. Furthermore, for every $1 \leq k \leq n-1$, there exists an n state NFA $A_{n,k}$ with computation width k over a binary alphabet such that $\mathrm{sc}(L(A_{n,k})) = 1+\sum_{j=1}^{k}\binom{n-1}{j}$.*

The determinization of MDFAs involves a similar size trade-off and the lower bound construction used for Theorem 5.5 is modified from Holzer et al. [10]. Palioudakis et al. [24] gives also an upper bound $1+\sum_{i=1}^{k-\ell+1}\binom{n-1}{i}$ for converting an n state NFA with computation width k to an NFA with computation width $2 \leq \ell < k$. However, a corresponding lower bound is missing and for computation width we do not have a spectrum result analogous to Theorem 5.4.

By noting that finite computation width implies finite ambiguity, Theorem 5.1 establishes that the worst case cost of converting an NFA with finite computation width to a UFA is the same as determinizing an arbitrary NFA. For the converse transformation Palioudakis et al. [24] has observed, by modifying constructions of Goldstine et al. [7] and Leung [18], that for any $n \geq 4$ there exists an UFA A with n states such that any NFA with finite computation width equivalent to A needs 2^{n-1} states.

The deviation number [5] counts the computations "branching out" from a best accepting computation. For any NFA A with finite computation width, $\mathrm{dn}_A \leq \mathrm{cw}_A - 1$. If the NFA A is constructed in a way that forces it to make the nondeterministic choices sequentially, and the "wrong" choice always leads to failure without further nondeterminism, then the deviation number of A is exactly one less than the computation width of A. It is known that for some regular languages the minimal NFA must have this property. Note that for any DFA A, $\mathrm{cw_A} = 1$ and $\mathrm{dn}_A = 0$. In the result below the condition $\mathrm{nsc}_{\mathrm{dn}\leq k}(L_k) < \mathrm{nsc}_{\mathrm{dn}\leq k-1}(L_k)$ guarantees that the minimal NFA for L_k needs to have deviation number at least k – otherwise the equality would be satisfied by any regular language L such that the minimal DFA for L is also minimal as an NFA.

Proposition 5.1 (Goc and Salomaa [5]) *For every $k \in \mathbb{N}$ there exists a regular language L_k (over a three letter alphabet) such that*

$$\mathrm{nsc}_{\mathrm{cw}\leq k+1}(L_k) = \mathrm{nsc}_{\mathrm{dn}\leq k}(L_k) < \mathrm{nsc}_{\mathrm{dn}\leq k-1}(L_k).$$

Contrasting the languages of Proposition 5.1, there exist languages for which an NFA of given computation width needs to be super-polynomially larger than an NFA with the same deviation number.

Theorem 5.6 (Goc and Salomaa [5]) *For $k, s \in \mathbb{N}$ there exists a language $L_{k,s}$ over an alphabet of size s recognized by an NFA having size $(k+1) \cdot s^k$ and deviation number $k\lceil \log s \rceil + 1$ such that for any constant $c \geq 1$,*

$$\mathrm{nsc}_{\mathrm{cw} \leq c \cdot k \cdot \log s}(L_{k,s}) \in \Omega(c \cdot k \cdot (\log s) \cdot 2^{\frac{s}{c \cdot \log s}}).$$

The inequalities of Lemma 4.1 imply the following state complexity comparison between finite computation width and finite trace:

Theorem 5.7 (Palioudakis et al. [25]) *For any regular L and $k \in \mathbb{N}$,*

$$\mathrm{nsc}_{\mathrm{cw} \leq k}(L) \leq \mathrm{nsc}_{\tau \leq k}(L) \leq \mathrm{nsc}_{\mathrm{cw} \leq \log(k-1)}(L).$$

It is known that the first inequality cannot be improved because for any unary regular language L and all $k \in \mathbb{N}$, $\mathrm{nsc}_{\mathrm{cw} \leq k}(L) = \mathrm{nsc}_{\tau \leq k}(L)$ [27]. It remains an open problem whether there exist regular languages for which the second inequality of Theorem 5.7 is an equality.

Directly from the definitions it follows that, for any NFA A, $\beta_A \leq \tau_A$ and hence Theorem 5.7 implies that for any regular language L and $k \in \mathbb{N}$, $\mathrm{nsc}_{\beta \leq 2^{k-1}}(L) \leq \mathrm{nsc}_{\mathrm{cw} \leq k}(L)$.

For the converse state complexity comparison between branching and computation width, we recall that Kappes [12] has given a nice construction based on modular arithmetic that simulates an NFA with finite branching by an MDFA.

Theorem 5.8 (Kappes [12]) *An NFA with n states having branching k can be simulated by a k-MDFA with $k \cdot n$ states.*

Strictly speaking, Kappes [12] constructs a k-MDFA with $k \cdot n + 1$ states including a dead state (which can be omitted with our conventions). Now observing that a k-MDFA can be converted to an NFA with computation width k by adding one more state, we have:

Corollary 5.1 *For any regular language L, $\mathrm{nsc}_{\mathrm{cw} \leq k}(L) \leq k \cdot \mathrm{nsc}_{\beta \leq k}(L) + 1$.*

It is not known whether the bound of Corollary 5.1 is the best possible. On the other hand, we know that the transformation of Theorem 5.8 converting an NFA with finite branching to an MDFA is almost optimal.

Theorem 5.9 (Palioudakis et al. [26]) *For infinitely many values $n, k \in \mathbb{N}$, an MDFA equivalent to an NFA with n states and branching k needs, in the worst case, size $\frac{k}{1+\log k} \cdot n$.*

Most of the above state complexity results deal only with finite nondeterminism measures. With the exception of the exponential trade-off between NFAs with polynomial and exponential ambiguity, respectively (Theorem 5.2 [16]), very little is known about descriptional complexity comparisons between the various nondeterminism measures in the case where the amount of nondeterminism is unbounded. For example, for a given regular language, the comparison between optimal sizes of NFAs having polynomial and exponential growth rate of computation width remains open.

References

[1] H. Björklund and W. Martens, The tractability frontier for NFA minimization. *J. Comput. System Sci.* 78 (2012) 198–210.

[2] T. Chan and O.H. Ibarra, On the finite valuedness problem for sequential machines. *Theoret. Comput. Sci.* 23 (1983) 95–101.

[3] Y. Gao, N. Moreira, R. Reis and S. Yu, A review on state complexity of individual operations. Faculdade de Ciencias, Universidade do Porto, Technical Report DCC-2011-8 `www.dcc.fc.up.pt/dcc/Pubs/TReports/TR11/dcc-2011-08.pdf` To appear in *Computer Science Review.*

[4] Y. Gao and S. Yu, State complexity and approximation. *Int. J. Found. Comput. Sci.* 23(5) (2012) 1085–1098.

[5] D. Goc and K. Salomaa, Computation width and deviation number. Proceedings of DCFS 2014, Lect. Notes Comput. Sci. 8614, Springer (2014) 150–161.

[6] J. Goldstine, M. Kappes, C.M.R. Kintala, H. Leung, A. Malcher and D. Wotschke, Descriptional complexity of machines with limited resources. *J. Univ. Comput. Sci.* 8 (2002) 193–234.

[7] J. Goldstine, C.M.R. Kintala and D. Wotschke, On measuring nondeterminism in regular languages. *Inform. Comput.* 86 (1990) 179–194.

[8] J. Goldstine, H. Leung and D. Wotschke, On the relation between ambiguity and nondeterminism in finite automata. *Inform. Comput.* 100 (1992) 261–270.

[9] M. Holzer and M. Kutrib, Descriptional and computational complexity of finite automata — A survey. *Inform. Comput.* 209 (2011) 456–470.

[10] M. Holzer, K. Salomaa and S. Yu, On the state complexity of k-entry deterministic finite automata. *J. Automata, Languages and Combinatorics* 6 (2001) 453–466.

[11] J. Hromkovič, S. Seibert, J. Karhumäki, H. Klauck and G. Schnitger, Communication complexity method for measuring nondeterminism in finite automata. *Inform. Comput.* 172 (2002) 202–217.

[12] M. Kappes, Descriptional complexity of deterministic finite automata with multiple initial states. *J. Automata, Languages, and Combinatorics* 5 (2000) 269–278.

[13] C.M.R. Kintala and D. Wotschke, Amounts of nondeterminism in finite automata. *Acta Inf.* 13 (1980) 199–204.

[14] M. Kutrib and G. Pighizzini, Recent trends in descriptional complexity of formal languages. *Bulletin of the EATCS* 111 (2013) 70–86.

[15] E. Leiss, Succinct representation of regular languages by Boolean automata. *Theoret. Comput. Sci.* 13 (1981) 323–330.

[16] H. Leung, Separating exponentially ambiguous finite automata from polynomially ambiguous finite automata. *SIAM J. Comput.* 27(4) (1998) 1073–1082.

[17] H. Leung, On finite automata with limited nondeterminism. *Acta Inf.* 35 (1998) 595–624

[18] H. Leung, Descriptional complexity of NFA of different ambiguity. *Internat. J. Foundations Comput. Sci.* 16 (2005) 975–984.

[19] O.B. Lupanov, A comparison of two types of finite sources. *Problemy Kibernetiki* 9 (1963) 328–335.

[20] A. Mandel and I. Simon, On finite semi-groups of matrices, *Theoret. Comput. Sci.* 5 (1977) 183–204.

[21] A.N. Maslov, Estimates on the number of states of finite automata. *Soviet Math. Dokl.* 11 (1970) 1373–1375.

[22] A.R. Meyer and M.J. Fischer, Economy of description by automata, grammars and formal systems. Proc. SWAT (FOCS), IEEE Computer Society (1971) 188–191.

[23] F.R. Moore, On the bounds for state-set size in the proofs of equivalence between deterministic, nondeterministic, and two-way finite automata. *IEEE Transactions on Computers* C-20 (1971) 1211–1214.

[24] A. Palioudakis, K. Salomaa, and S.G. Akl, State complexity of finite tree width NFAs. *J. Automata, Languages and Combinatorics* 17 No. 2–4 (2012) 245–264.

[25] A. Palioudakis, K. Salomaa, and S.G. Akl, Comparisons between measures of nondeterminism for finite automata. Proceedings of DCFS 2013, Lect. Notes Comput. Sci. 8031, Springer, (2013) 217–228.

[26] A. Palioudakis, K. Salomaa, and S.G. Akl, Lower bound for converting an NFA with finite nondeterminism into an MDFA. *J. Automata, Languages and Combinatorics* 19 (2014) 251–264.

[27] A. Palioudakis, K. Salomaa, and S.G. Akl, Unary NFAs with limited nondeterminism. Proceedings of SOFSEM'14. Lect. Notes Comput. Sci. 8327, Springer (2014) 443–454.

[28] A. Palioudakis, *State complexity of nondeterministic finite automata with limited nondeterminism.* PhD thesis, Queen's University, Kingston, Canada, 2014.

[29] B. Ravikumar and O.H. Ibarra, Relating the degree of ambiguity of finite automata to the succinctness of their representation. *SIAM J. Comput.* 18 (1989) 1263–1282.

[30] C. Reutenauer, *Propiétés arithmétiques et topologiques des séries rationnelles en variable non commutative.* Tèse de troisuème cycle, Université Paris VI, 1977.

[31] E.M. Schmidt, *Succinctness of descriptions of context-free, regular and finite languages.* PhD thesis, Cornell University, Ithaca, NY, 1978.

[32] J. Shallit, *A Second Course in Formal Languages and Automata Theory,* Cambridge University Press, 2009.

[33] A. Weber and H. Seidl, On the degree of ambiguity of finite automata. *Theoret. Comput. Sci.* 88 (1991) 325–349.

[34] S. Yu, Regular languages, in: *Handbook of Formal Languages,* Vol. I, (G. Rozenberg, A. Salomaa, Eds.), Springer, 1997, pp. 41–110.

Alexandros Palioudakis
Institution: Department of Computer Science, Yonsei University
Post address: 50, Yonsei-Ro, Seodaemun-Gu, Seoul 120-749, Korea
Email: `alex@cs.yonsei.ac.kr`

Kai Salomaa and Selim G. Akl
Institution: School of Computing, Queen's University
Post address: Kingston, Ontario K7L 2N8, Canada
E-mail: {`ksalomaa, akl`}`@cs.queensu.ca`