Inapproximability of Nondeterministic State and Transition Complexity Assuming P $\neq$ NP

Hermann Gruber[1] and Markus Holzer[2]

[1] Institut für Informatik, Ludwig-Maximilians-Universität München,
Oettingenstraße 67, D-80538 München, Germany
email: gruberh@tcs.ifi.lmu.de
[2] Institut für Informatik, Technische Universität München,
Boltzmannstraße 3, D-85748 Garching bei München, Germany
email: holzer@informatik.tu-muenchen.de

Abstract. Inapproximability results concerning minimization of nondeterministic finite automata relative to given deterministic finite automata were obtained only recently, modulo cryptographic assumptions [4]. Here we give upper and lower bounds on the approximability of this problem utilizing only the common assumption **P** $\neq$ **NP**, in the setup where the input is a finite language specified by a truth table. To this end, we derive an improved inapproximability result for the biclique edge cover problem. The obtained lower bounds on approximability can be sharpened in the case where the input is given as a deterministic finite automaton over a binary alphabet. This settles most of the open problems stated in [4]. Note that the biclique edge cover problem was recently studied by the authors as lower bound method for the nondeterministic state complexity of finite automata [5].

1 Introduction

Finite automata are one of the oldest and most intensely investigated computational models. As such, they found widespread use in many other different areas such as circuit design, natural language processing, computational biology, parallel processing, image compression, to mention a few, see [13] and references therein. As some of these applications deal with huge masses of data, the amount of space needed by finite automata is an important research topic.

On the one hand, it is well known that while nondeterministic finite automata and deterministic finite automata are equal in expressive power, nondeterministic automata can be exponentially more succinct than deterministic ones. On the other hand, minimizing deterministic finite automata can be carried out efficiently, whereas the state minimization problem for nondeterministic finite state machines is **PSPACE**-complete, even if the regular language is specified as a deterministic finite automaton [8]. This prompted the authors of the aforementioned paper to ask whether there exist at least polynomial-time approximation algorithms with a reasonable performance guarantee. However, recent work [4]

shows that this problem cannot be approximated within $\frac{\sqrt{n}}{\text{polylog}(n)}$ for state minimization and $\frac{n}{\text{polylog}(n)}$ for transition minimization, provided some cryptographic assumption holds. As the result is based on a rather strong assumption, the authors asked for proving approximation hardness results under the weaker (and more familiar) assumption $\mathbf{P} \neq \mathbf{NP}$ as an open problem. Moreover, they asked to determine the approximation complexity when given a regular language specified by a truth table.

In this paper we solve these open problems. To summarize, we have obtained the following results on the minimization problems for nondeterministic finite automata:

- If the input is given as a *nondeterministic* finite automaton with n states (transitions, respectively), the state (transition, respectively) minimization problem is not fixed-parameter tractable (the parameter being the upper bound on the number of states/transitions to be reached) by Theorem 1, unless $\mathbf{P} = \mathbf{PSPACE}$. Earlier work established that this problem is not approximable within $o(n)$, provided $\mathbf{P} = \mathbf{PSPACE}$ [4], and this holds even for unary input alphabets, unless $\mathbf{P} = \mathbf{NP}$ [7].
- If the input is given by a *truth table* specifying a Boolean function of total size N, the state minimization problem is **NP**-complete (Theorem 4). Moreover we establish a lower bound of $N^{1/6-\varepsilon}$ on the approximability both for state and transition minimization, provided $\mathbf{P} \neq \mathbf{NP}$ (Theorems 7 and 10). These results are nicely contrasted by two simple polynomial-time algorithms achieving ratios of $O(\sqrt{N}/\log N)$ for state minimization, and $O(N/(\log N)^2)$ for the case of transition minimization, respectively (Theorem 5).
- Finally, if the input is given by a *deterministic* finite automaton, Theorem 1 asserts that the corresponding state and transition minimization problems become fixed-parameter tractable. But assuming $\mathbf{P} \neq \mathbf{NP}$, the state minimization problem is not approximable within $n^{1/3-\varepsilon}$ for alphabets of size $O(n)$ (Corollary 14), and not approximable within $n^{1/5-\varepsilon}$ for a binary alphabet, for all $\varepsilon > 0$ (Theorem 15). Under the same assumption, we show that the transition minimization problem for binary input alphabets is not approximable within $n^{1/5-\varepsilon}$, for all $\varepsilon > 0$ (Corollary 16). Before this work, the known inapproximability results for these problems were based on a much stronger assumption [4].

Some of the hardness results are based on a reduction from the biclique edge cover problem, which we prove to be neither approximable within $|V|^{1/3-\varepsilon}$ nor $|E|^{1/5-\varepsilon}$ unless $\mathbf{P} = \mathbf{NP}$ in Theorem 6.

2 Preliminaries

First we recall some definitions from formal language and automata theory. In particular, let Σ be an alphabet and Σ^* the set of all words over the alphabet Σ, containing the empty word λ. The length of a word w is denoted by $|w|$, where

$|\lambda| = 0$. The reversal of a word w is denoted by w^R and the reversal of a language $L \subseteq \Sigma^*$ by L^R, which equals the set $\{ w^R \mid w \in L \}$. Furthermore let $\Sigma^n = \{ w \in \Sigma^* \mid |w| = n \}$.

A *nondeterministic finite automaton* (NFA) is a 5-tuple $A = (Q, \Sigma, \delta, q_0, F)$, where Q is a finite set of states, Σ is a finite set of input symbols, $\delta : Q \times \Sigma \to 2^Q$ is the transition function, $q_0 \in Q$ is the initial state, and $F \subseteq Q$ is the set of accepting states. The transition function δ is extended to a function from $\delta : Q \times \Sigma^* \to 2^Q$ in the natural way, i.e., $\delta(q, \lambda) = \{q\}$ and $\delta(q, aw) = \bigcup_{q' \in \delta(q,a)} \delta(q', w)$, for $q \in Q$, $a \in \Sigma$, and $w \in \Sigma^*$. A nondeterministic finite automaton $A = (Q, \Sigma, \delta, q_0, F)$ is a *deterministic* finite automaton (DFA), if $|\delta(q, a)| = 1$ for every $q \in Q$ and $a \in \Sigma$. The *language accepted* by a finite automaton A is $L(A) = \{ w \in \Sigma^* \mid \delta(q_0, w) \cap F \neq \emptyset \}$. Two automata are equivalent if they accept the same language.

For a regular language L, the deterministic (nondeterministic, respectively) state complexity of L, denoted by $\mathrm{sc}(L)$ ($\mathrm{nsc}(L)$, respectively) is the minimal number of states needed by a deterministic (nondeterministic, respectively) finite automaton accepting L. The transition complexity is analogously defined as the state complexity and we abbreviate the deterministic (nondeterministic, respectively) transition complexity of a regular language L by $\mathrm{tc}(L)$ ($\mathrm{ntc}(L)$, respectively).

Here we are interested in the state (transition, respectively) minimization problem for nondeterministic finite automata. This problem is defined as follows: For a given finite automaton A and an integer k, decide whether there exists a nondeterministic finite automaton B with at most k states (transitions, respectively) such that $L(A) = L(B)$. As already mentioned in the introduction, this problem is **PSPACE**-complete even if the given automaton is guaranteed to be deterministic [8]. However, other computational complexity aspects may vary if the instance to minimize is described as a nondeterministic or deterministic finite automaton. The following theorem describes such a situation—we omit the proof due to lack of space.

Theorem 1. *(i) The problem to determine for a given* deterministic *finite automaton, whether there exists a nondeterministic finite automaton B with at most k states (transitions, respectively) such that $L(A) = L(B)$ is fixed-parameter tractable w.r.t. parameter k. (ii) Provided* $\mathbf{P} \neq \mathbf{PSPACE}$*, the aforementioned problems are* not *fixed-parameter tractable, if the input is given as a* nondeterministic *finite automaton instead.* □

We also need some notions from graph theory. A *bipartite graph* is a 3-tuple $G = (U, V, E)$, where U and V are finite sets of vertices, and $E \subseteq U \times V$ is a set of edges. A bipartite graph $H = (U', V', E')$ is a *subgraph* of G if $U' \subseteq U$, $V' \subseteq V$, and $E' \subseteq E$. The subgraph H is *induced* if $E' = (U' \times V') \cap E$, the subgraph induced by U' and V' is denoted by $G[U', V']$. A set $C = \{H_1, H_2, \ldots, H_k\}$ of non-empty bipartite subgraphs of G is an *edge cover* of G if every edge in G is present in at least one subgraph. An edge cover C of the bipartite graph G is a *biclique edge cover* if every subgraph in C is a biclique, where a *biclique* is a

bipartite graph $H = (U, V, E)$ satisfying $E = U \times V$. The *bipartite dimension* of G is referred to as $d(G)$ and is defined to be the size of a smallest biclique edge cover of G. The associated decision problem is a classical one [3, GT18], and a reformulation of the biclique edge cover problem in terms of finite sets gives the set basis problem [3, SP7]. The following result was shown in [10]:

Theorem 2. *Deciding whether for a given bipartite graph G and an integer k there exists a biclique edge cover for G consisting of at most k bicliques is* **NP***-complete.*

Finally, we assume the reader to be familiar with the basic notations of approximation theory as contained in textbooks such as [12]. In particular, transferring known inapproximability results to new problems is most easily achieved if we use some kind of approximation-preserving reduction. Several such types of reduction have been proposed; our weapon of choice is the S-reduction introduced in [9]: Loosely speaking, for two minimization problems Π and Π' and associated functions $|x|_\Pi$ and $|y|_{\Pi'}$ measuring the size of respective inputs, an S-reduction from Π to Π' with amplification $(a(n), |x|_\Pi, |y|_{\Pi'})$, where $a(n)$ is monotonically increasing, consists of a polynomial-time computable function f which maps each instance x of Π to an instance y of Π' such that $|y|_{\Pi'} \leq a(|x|_\Pi)$, and a polynomial-time computable function g that maps back instance-solution pairs of Π' to instance-solution pairs of Π such that the performance ratios of the solutions are linearly related. This kind of reduction has the following nice property [9, Proposition 1]:

Lemma 3. *Let $b : \mathbb{N} \to \mathbb{R}^+$ be a positive function, and let $\Pi = (I, \mathrm{sol}, m)$, $\Pi' = (I', \mathrm{sol}', m')$ be two minimization problems. Assume Π' is approximable within $b(|y|_{\Pi'})$, for all $y \in I'$, and there is an S-reduction from Π to Π' with amplification $(a(n), |\cdot|_\Pi, |\cdot|_{\Pi'})$. Then Π is approximable within $O(b(a(|x|_\Pi)))$, for all instances x of Π.*

3 Synthesizing a Minimal Nondeterministic Finite Automaton From a Given Truth Table

In this section we investigate the approximation complexity of minimizing nondeterministic finite automata when specifying the input by a truth table, an open question in [4]. First we show that the decision version of the problem of minimizing the number of states is **NP**-complete. Then we present two simple approximation algorithms for minimizing the number of states or transitions. Moreover, we show that the best possible approximation ratio is related to the one of the biclique edge cover problem. In order to formally define the problem we are interested in, we need some more notations.

To each m-bit Boolean function $f : \{0,1\}^m \to \{0,1\}$, where $m \geq 1$ is some natural number, we can naturally associate a finite binary language as follows:

$$L_f = \{\, x_1 x_2 \ldots x_m \in \{0,1\}^m \mid f(x_1, x_2, \ldots, x_m) = 1 \,\}.$$

In [4], the following problem was proposed: Given a truth table specifying a Boolean function $f : \{0,1\}^m \to \{0,1\}$ and an integer k, is there a nondeterministic finite automata with at most k states (transitions, respectively) accepting the language L_f?

For the the above stated problem we are able to show **NP**-completeness in case of state minimization by a reduction from the biclique edge cover problem. But the used reduction does not preserve approximability. The proof of the following Theorem can be found in the full version of the paper.

Theorem 4. *Deciding whether for a given truth table $f : \{0,1\}^m \to \{0,1\}$ and a positive integer k there is a nondeterministic finite automaton with at most k states accepting language L_f is* **NP***-complete.* □

Next we consider how well the problem under consideration can be approximated. By very simple algorithms, we obtain the following situation:

Theorem 5. *(i) Given a truth table of size $N = 2^m$, specifying an m-bit Boolean function function f, then there is a polynomial-time algorithm approximating the number of states of a state minimal nondeterministic (unambiguous, respectively) finite automaton accepting L_f within a factor of $O(\sqrt{N}/\log N)$. (ii) When considering transition minimization the performance ratio changes to $O(N/(\log N)^2)$.*

Proof (Sketch). First we note that nondeterministic state and transition complexity are both at least $m = \log N$, except when L_f is empty. For state minimization we use a construction given in [6] to obtain a NFA with $O(\sqrt{N})$ states. For transition minimization recall that the minimal deterministic finite automaton accepting L_f can have at most $O(N/\log N)$ transitions [2]. Then the stated bounds easily follow. □

In the remainder of this section we derive an inapproximability result for the problem under consideration. In order to get good inapproximability ratios, the biclique edge cover problem is a natural candidate to reduce from. By combining a recent inapproximability result for the chromatic number problem [14] with the approximation preserving reduction from the minimum clique partition problem given in [11], we see that the problem is not approximable within $|V|^{1/5-\varepsilon}$. But that is not the end of the line:

Theorem 6. *For all $\varepsilon > 0$, the biclique edge cover problem cannot be approximated within $|V|^{1/3-\varepsilon}$ or $|E|^{1/5-\varepsilon}$, unless* $\mathbf{P} = \mathbf{NP}$*. This still holds in the restricted case where the input $G = (U, V, E)$ is a balanced bipartite graph, that is $|U| = |V|$, and has bipartite dimension at least $\Omega(|V|^{2/3})$.*

Proof. Let the clique partition number $\overline{\chi}(I)$ of a graph I be defined as the smallest number k such that the vertex set of I can be covered by at most k cliques. The associated decision problem is **NP**-complete [3, GT15], and as a simple reformulation of the graph coloring problem, not approximable within $|V|^{1-\varepsilon}$, for all $\varepsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$ [14]. We briefly recall the construction for reducing

the clique partition problem to the biclique edge cover problem given in [11, Theorem 5.1a].

For an undirected graph $I = (V, E)$ with $V = \{v_1, v_2, \ldots, v_n\}$, we construct its bipartite version by setting $I_B = (X_B, Y_B, E_B)$ as set of left vertices $X_B = \{x_1, x_2, \ldots, x_n\}$, as set of right vertices $Y_B = \{y_1, y_2, \ldots, y_n\}$, and $(x_i, y_j) \in E_B$ if and only if $i = j$ or $\{v_i, v_j\} \in E$. An edge (x_i, y_j) is called ascending if $i < j$, descending if $i > j$, and horizontal if $i = j$.

The biclique edge cover instance $G = (X, Y, \mathcal{E})$ consists of t^2 copies (the number t to be fixed later accordingly) of I_B, which we think of as being arranged in a $t \times t$ grid; and the bipartition of the vertex set is inherited from I_B. The ith left (right, respectively) vertex in copy (p, q) is denoted by (x_i, p, q) ((y_i, p, q), respectively). Two vertices (x_i, p, q) and (y_j, r, s) in different copies are connected by an edge if: either they are in the same row, i.e., $p = r$, and (x_i, y_j) is an ascending edge in E_B, or they are in the same column, i.e., $q = s$, and (x_i, y_j) is a descending edge in E_B. Accordingly, we say that an edge in $\mathcal{E}$ connecting vertices (x_i, p, q) and (y_j, r, s) is ascending if $i < j$, descending if $i > j$, and horizontal if $i = j$.

In [11], it is noted that if there is a system of s bicliques covering all horizontal edges in $\mathcal{E}$, then a partition of I into at most s/t^2 cliques can be constructed in polynomial time from this system, and

$$\overline{\chi}(I) \leq d(G)/t^2. \tag{1}$$

Conversely, each partition of I into r cliques corresponds to a system of rt^2 bicliques which cover all the horizontal edges in $\mathcal{E}$, and maybe some non-horizontal edges. However, note that the rt^2 bicliques do not necessarily cover all edges involving only vertices of a single copy of I_B: As an example, consider the partition of the graph I given in Figure 1 into $r = 3$ cliques.

To cover the remaining edges, we can do somewhat better than proposed in the original reduction: For $x_i \in X_B$, define $X_{i,p}$ as the set of ith left vertices in the copies of I_B which are in row p, and define $Y_{i,p}$ as the set of right vertices y in row p such that $((x_i, p, q), y)$ is an ascending edge in $\mathcal{E}$. It is not hard to see that the induced subgraph $G[X_{i,p}, Y_{i,p}]$ is a biclique which covers all ascending edges in row p incident to x_i, see Figure 1 for illustration by example.

By proceeding in this way for each row and each left vertex x_i in X_B, all ascending edges in G can be covered using no more than tn bicliques. The descending edges in G can be covered by tn bicliques in a similar manner. Thus

$$d(G) \leq t^2\overline{\chi}(I) + 2tn. \tag{2}$$

Suppose now C is a biclique edge cover for G of size s. Then we can construct a clique partition for I of size s/t^2 in polynomial time from C, see [11] for details. Now we fix $t = 4n$, and compare performance ratios using Inequality (2):

$$\frac{s/t^2}{\overline{\chi}(I)} \leq \frac{s}{d(G) - 2tn} \leq 2\frac{s}{d(G)},$$

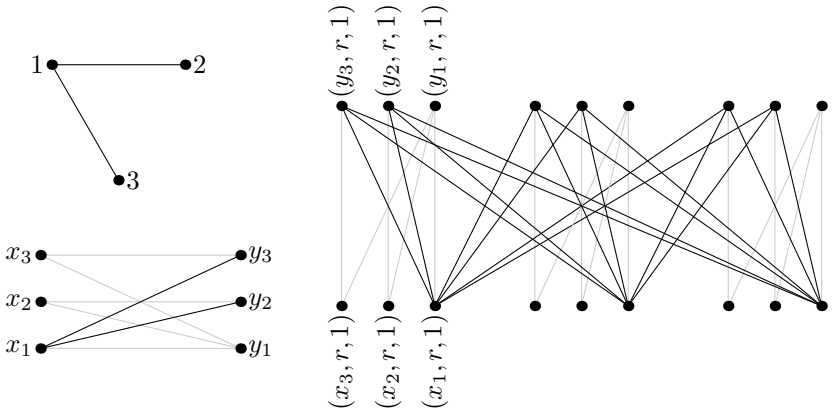

Fig. 1. The original graph I (top left), the bipartite graph I_B (lower left), and the subgraph of G induced by the vertices in row r (right), consisting of $t = 3$ copies of I_B. The induced subgraph $G[X_{1,r}, Y_{1,r}]$ forms a biclique.

where the last statement above holds since $2tn = \frac{1}{2}t^2 \leq \frac{1}{2}d(G)$ by Inequality (1). We have established a S-reduction with expansion $(O(n^3), |V|, |X|)$. Then the desired hardness result regarding the measure $|X|$ follows by Lemma 3. Estimating the number of edges in $\mathcal{E}$ gives a total number of at most $t^2|E_B| + 2t \cdot \binom{t}{2}|E_B| = O(|V|^5)$, so this is equally a S-reduction with expansion $(O(n^5), |V|, |\mathcal{E}|)$. Again by Lemma 3, the claimed inapproximability result follows. Finally, we note that Inequality (1) implies that $d(G) \geq t^2 = \Omega(|X|^{2/3})$, since $|X| = \Theta(n^3)$ and $t = 4n$. □

Theorem 7. *Given a truth table of size N specifying a Boolean function function f, no polynomial-time algorithm can approximate the number of states of a state minimal nondeterministic finite automaton accepting L_f with performance ratio $N^{1/6-\varepsilon}$, for all $\varepsilon > 0$, unless* $\mathbf{P} = \mathbf{NP}$.

Proof. We use the finite language to encode the edges in the graph $G = (X, Y, \mathcal{E})$ from the proof of Theorem 6, and the notations defined therein. Recall X consists of nodes of the form (x_i, p, q) with $x_i \in X_B$, p denotes a row index and q a column index, and similar for y_j, that is (x_i, p, q) and (y_j, p, q) belong to the same copy of I_B. Without loss of generality we assume $V = \{0, 1\}^m$ for some m. The t addresses for rows and columns can be respectively encoded in binary using a fixed length of $\log t = m + 2$. Throughout the rest of the proof, $c_1, c_2, \ldots, c_t$ denote the words encoding the t column addresses, and in a similar manner, $r_1, r_2, \ldots, r_t$ the row addresses. We then encode the edges $((x, p, q), (y, a, b))$ in $\mathcal{E}$ as $xr_pc_q(r_ac_b)^Ry$, and define L_G as the set of all codewords corresponding to an edge in $\mathcal{E}$. In the following, we will use the term "edge" to denote a word encoding an edge in $\mathcal{E}$ if there is no risk of confusion.

Claim 8. The nondeterministic state complexity of L_G is bounded below by the bipartite dimension of G.

Proof. We apply the biclique edge cover technique [5, Theorem 4] to give a lower bound for $\mathrm{nsc}(L_G)$. Let $\Gamma = (A, B, E_{L_G})$ be the bipartite graph given by $A = B = \{0,1\}^{m+2(m+2)}$, and $E_{L_G} = \{\, (u,v) \in A \times B \mid uv \in L_G \,\}$. By an obvious bijection holds $d(G) = d(\Gamma)$, and the latter gives a lower bound for the nondeterministic state complexity of L_G. □

Claim 9. $\mathrm{nsc}(L_G) = O(d(G)) + O(|X|^{2/3} \log |X|)$.

Proof. We establish the claim by constructing a sufficiently small NFA accepting the language L_G from a minimum biclique edge cover for G. For the horizontal edges in $\mathcal{E}$, we give a construction inspired by the proof of Theorem 6. Let $\{\, (X_j, Y_j) \mid 1 \leq j \leq \overline{\chi}(I) \,\}$ be a minimum set of bicliques covering all horizontal edges in $I_B = (X_B, Y_B, E_B)$. For the ith biclique, we define an auxiliary language H_j as $H_j = X_j \cdot M \cdot Y_j$, where $M = \{\, rc(rc)^R \mid r, c \in \{0,1\}^{m+2} \,\}$ is the language ensuring that the row and column address of $x \in X_j$ is the same as the row and column address of $y \in Y_j$. As there are no horizontal edges between different copies of I_B, language M provides that the union of languages $\bigcup_j H_j$ contains all codewords corresponding to horizontal edges in $\mathcal{E}$, and is a subset of L_G. Each H_j can be described by a nondeterministic finite automaton having $O(t^2)$ states: As all words in the sets X_j and Y_j have length m, each of them can be accepted by a NFA with $O(2^{m/2}) = o(t)$ states. The language M can be accepted by a NFA with $O(2^{2(m+2)}) = O(t^2)$ states. A schematic drawing of such an automaton is given in Figure 2. And a standard construction for nondeterministic finite automata yields an automaton with $O(t^2)$ states for the concatenation of these languages. Finally, the union of these languages can be accepted by a NFA having $O(t^2 \cdot \overline{\chi}(I)) = O(d(G))$ many states.

We use a similar matching language as M to construct a NFA accepting a subset of the codewords of $\mathcal{E}$ which contains all ascending edges. This time, the language has to ensure that the the left and the right vertex share the same row address, that is $M' = \{\, rc_1c_2r^R \mid r, c_1, c_2 \in \{0,1\}^{m+2} \,\}$, and this language can be accepted by a NFA with only $O(t \log t)$ states, see Figure 2 for illustration.

Following the idea in the proof of Theorem 6, the graph G has for every row p and every vertex $x_i \in X_B$ a biclique $G[X_{i,p}, Y_{i,p}]$ containing only ascending edges. As we have an ascending biclique for each $x_i \in \{0,1\}^m$, it is more economic to share the states needed for addressing. Thus, a part of the automaton is a binary tree, whose root is the start state and whose leaves address the nodes in X_B. That is, after reading a word x of length m, the automaton is in a unique leaf of the binary tree. In a symmetric manner, we construct an inverted binary tree whose leaves address the nodes in Y_B, and whose transitions are directed towards the root, which is the final state of the automaton. It remains to wire the copies of the automaton accepting M' into these two binary trees appropriately, using no more than $|X_B|$ copies of it: Each leaf x_i of the binary tree, addressing some node in X_B, is identified with the start state of a fresh copy of the NFA. The transitions entering the final state of this copy are replaced with transitions

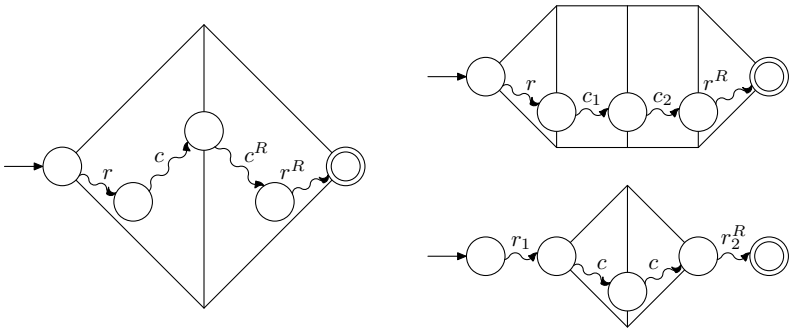

Fig. 2. Schematic drawings of the nondeterministic finite automata accepting M (left), M' (top right), and M'' (bottom right).

entering the inverted binary tree at the appropriate address. This completes the description of the construction for a NFA having $O(|X_B| + |Y_B| + |X_B| t \log t) = O(|X|^{2/3} \log |X|)$ states accepting a subset of $\mathcal{E}$ including all ascending edges, since $|X| = \Theta(t^3)$ and $|X_B| = |Y_B| = \Theta(t)$.

For the descending edges, we carry out a similar construction, this time using the language $M'' = \{\, r_1 c c^r r_2 \mid r_1, c, r_2 \in \{0,1\}^{m+2} \,\}$ ensuring that the column addresses match, see Figure 2. Then a similar construction gives a compact NFA describing the codewords of a set of edges including all descending edges in G. Finally, the union of all these languages can be described by a NFA with the desired upper bound on the number of states. □

Assume now there exists a polynomial-time algorithm approximating $\mathrm{nsc}(L_G)$ within $|X|^{1/3-\varepsilon}$, that is it finds a NFA A of size at most $|X|^{1/3-\varepsilon} \cdot \mathrm{nsc}(L_G)$. By Claim 8, this can be seen equivalently as an algorithm finding a biclique edge cover for G of size at most $|X|^{1/3-\varepsilon} \cdot \mathrm{nsc}(L_G)$. We estimate the performance ratio of the latter algorithm:

$$\frac{|X|^{1/3-\varepsilon} \cdot \mathrm{nsc}(L_G)}{d(G)} = |X|^{1/3-\varepsilon} \cdot \frac{O(d(G)) + O(|X|^{2/3} \log |X|)}{d(G)}$$

by using Claim 9. The latter term is $O(|X|^{1/3-\varepsilon} \log |X|)$ because Theorem 6 asserts that $d(G) \geq |X|^{2/3}$. If we choose a small positive real δ such that $\varepsilon - \delta > 0$, then for $|X|$ large enough, $|X|^{1/3-\varepsilon} \log |X| < |X|^{1/3-(\varepsilon-\delta)}$. Together with the final argument given in Theorem 6, this implies $\mathbf{P} = \mathbf{NP}$.

As the size of the graph and the size of the truth table are related by $N = \Theta(|X|^2)$, the problem is not approximable within $N^{1/6-2\varepsilon}$ for every positive real ε, and the theorem is established. □

For transition minimization we encounter the following situation.

Theorem 10. *Given a truth table of size N specifying a Boolean function f, no polynomial-time algorithm can approximate the number of transitions of a transition minimal nondeterministic finite automaton accepting L_f with performance ratio $N^{1/6-\varepsilon}$, for all $\varepsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$.*

Proof. The language L_G defined in the proof of Theorem 7 can be accepted by a polynomial-time constructible DFA A having at most $O(m \cdot |L_G|)$ states and transitions. We can mimic the proof of Theorem 7 if we are able to verify inequalities relating nondeterministic transition complexity of L_G to the bipartite dimension of G in a way similar to Claim 8 and Claim 9.

Claim 11. The nondeterministic transition complexity of L_G is bounded below by the bipartite dimension of G minus 1.

For an upper bound on $\mathrm{ntc}(L_G)$, we take a closer look at the NFA constructed in the proof of Claim 9.

Claim 12. $\mathrm{ntc}(L_G) = O(d(G)) + O(|X|^{2/3} \log |X|)$.

The rest of the proof follows along the lines of the proof of Theorem 7. □

4 Synthesizing a Minimal Nondeterministic Finite Automaton From a Given Deterministic One

This section contains results on the inapproximability of the minimization problem for nondeterministic finite automata, when the input is specified by a (deterministic) finite state automaton. This problem was investigated in [4, 8]: Given a finite automaton A and an integer k, is there a nondeterministic finite automaton with at most k states (transitions, respectively) accepting language $L(A)$?

Note that the minimization problems w.r.t. states (transitions, respectively) for nondeterministic finite automata are trivially approximable within $O(n/\log n)$, if the input is given by a deterministic finite automaton. Observe that the minimal deterministic finite automaton equivalent to a given deterministic one is also a feasible solution for the respective problem. The performance ratio of this solution can be bounded using the fact that the blow-up in the number of states or transitions inferred by determinization is at most exponential. While this is only a poor performance guarantee, a strong inapproximability result is obtained in [4], but under a much stronger (cryptographic) assumption than $\mathbf{P} \neq \mathbf{NP}$. We just note their result in passing:

Theorem 13. *(i) Given an n-state deterministic finite automaton A, no polynomial-time algorithm can approximate the number of states of a state minimal nondeterministic finite automaton accepting $L(A)$ with performance ratio better than $\frac{\sqrt{n}}{\mathrm{polylog}(n)}$, if nonuniform logspace contains strong pseudo-random functions. (ii) In case of transition minimization the problem remains inapproximable with the same assumption as above and performance ratio better than $\frac{n}{\mathrm{polylog}(n)}$, where t is the number of transitions of the given deterministic finite state automaton.*

In order to obtain our first inapproximability result on the problem where a DFA is given, we use Theorem 6 and a reduction from the biclique edge cover problem to the nondeterministic finite state automaton minimization problem,

where the input is a deterministic finite state automaton, given in [1, Lemma 1]. The noted reduction is an S-reduction with expansion $(O(n), |V|, |Q|)$.

Corollary 14. *Given a n-state deterministic finite automaton A accepting a finite language over an alphabet of size $O(n)$, no efficient algorithm can approximate the number of states of a state minimal nondeterministic finite automaton accepting $L(A)$ with performance ratio $n^{1/3-\varepsilon}$, for all $\varepsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$.* □

For fixed alphabet size, we obtain a corresponding result from Theorem 7, as from every truth table, an equivalent DFA of smaller size can be constructed in polynomial time. Exploiting the special structure of the language used in the proof of Theorem 7, we can get an even higher bound.

Theorem 15. *Given a n-state deterministic finite automaton A accepting a finite language over an alphabet of size two, no efficient algorithm can approximate the number of states of a state minimal nondeterministic finite automaton accepting $L(A)$ within a factor of $n^{1/5-\varepsilon}$, for all $\varepsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$.*

Proof (Sketch). To obtain the inapproximability result, we again use the language L_G defined in the proof of Theorem 7. The crucial observation is that this language contains $|\mathcal{E}| = O(2^{5m})$ words of length $O(m)$. Thus, a binary tree-like deterministic finite automaton of size $O(m \cdot |\mathcal{E}|)$ accepting all these words can be constructed in polynomial time—note that this size is much smaller than the truth table specifying L_G. Then one can show, similarly as in the proof of Theorem 7, that the stated inapproximability bound holds. □

By combining the observations in Theorems 10 and 15, we obtain for the corresponding problem of minimizing the number of transitions:

Corollary 16. *Given a deterministic finite automaton A with t transitions accepting a finite language over a binary alphabet, no efficient algorithm can approximate the number of transitions of a transition minimal nondeterministic finite automaton accepting $L(A)$ within a factor of $t^{1/5-\varepsilon}$, for all $\varepsilon > 0$, unless $\mathbf{P} = \mathbf{NP}$.* □

5 Conclusions

We compared nondeterministic finite automata minimization problems for regular languages, where the language is specified by different means—in decreasing order of succinctness: By a nondeterministic finite automaton, a deterministic automaton, or by a truth table. When given an NFA, the approximability of these problems is already settled [4, 7]. When given a DFA as input, approximation hardness was known only modulo cryptographic assumptions [4]. The main contribution of this paper is that we do not need such strong assumptions, that is, the problems are hard to approximate unless $\mathbf{P} = \mathbf{NP}$. This essentially also holds if the input is specified as a truth table, but for the latter case, we were able to provide simple approximation algorithms with nontrivial performance guarantees. This settles most of the research problems suggested in [4].

Acknowledgments. We thank Gregor Gramlich for carefully reading an earlier draft of this work, and to the anonymous referees for many valuable suggestions and corrections.

References

1. J. Amilhastre, Ph. Janssen, and M.-C. Vilarem. FA minimisation heuristics for a class of finite languages. In O. Boldt and H. Jürgensen, editors, *Proceedings of the 4th International Workshop on Implementing Automata*, number 2214 in LNCS, pages 1–12, Potsdam, Germany, July 2001. Springer.
2. J.-M. Champarnaud and J.-E. Pin. A maxmin problem on finite automata. *Discrete Applied Mathematics*, 23:91–96, 1989.
3. M. R. Garey and D. S. Johnson. *Computers and Intractability, A Guide to the Theory of NP-Completeness*. Freeman, 1979.
4. G. Gramlich and G. Schnitger. Minimizing NFA's and regular expressions. In V. Diekert and B. Durand, editors, *Proceedings of the 22nd Annual Symposium on Theoretical Aspects of Computer Science*, number 3404 in LNCS, pages 399–411, Stuttgart, Germany, February 2005. Springer.
5. H. Gruber and M. Holzer. Finding lower bounds for nondeterministic state complexity is hard (extended abstract). In O. H. Ibarra and Z. Dang, editors, *Proceedings of the 10th International Conference on Developments in Language Theory*, number 4036 in LNCS, pages 363–374, Santa Barbara, California, USA, June 2006. Springer.
6. H. Gruber and M. Holzer. On the average state and transition complexity of finite languages. *Theoretical Computer Science*, Special Issue: *Selected papers of "Descriptional Complexity of Formal Systems 2006"*. Accepted for publication.
7. H. Gruber and M. Holzer. Computational complexity of NFA minimization for finite and unary languages. In *Proceedings of the 1st International Conference on Language and Automata Theory and Applications*, LNCS, Tarragona, Spain, March 2007. Springer. Accepted for publication.
8. T. Jiang and B. Ravikumar. Minimal NFA problems are hard. *SIAM Journal on Computing*, 22(6):1117–1141, December 1993.
9. V. Kann. Polynomially bounded minimization problems that are hard to approximate. *Nordic Journal of Computing*, 1(3):317–331, Fall 1994.
10. J. Orlin. Contentment in graph theory: Covering graphs with cliques. *Indagationes Mathematicae*, 80:406–424, 1977.
11. H. U. Simon. On approximate solutions for combinatorial optimization problems. *SIAM Journal on Discrete Mathematics*, 3(2):294–310, 1990.
12. V. V. Vazirani. *Approximation Algorithms*. Springer, 2001.
13. S. Yu. Regular languages. In G. Rozenberg and A. Salomaa, editors, *Handbook of Formal Languages*, volume 1, pages 41–110. Springer, 1997.
14. D. Zuckerman. Linear degree extractors and the inapproximability of max clique and chromatic number. Report TR05-100, Electronic Colloquium on Computational Complexity (ECCC), September 2005.