On a maximal NFA without mergible states.

Igor Grunsky[1] Oleksiy Kurganskyy[1] and Igor Potapov[2]

[1] Institute of Applied Mathematics and Mechanics,
Ukrainian National Academy of Sciences
74 R. Luxemburg St, Donetsk, Ukraine,
grunsky@iamm.ac.donetsk.ua, kurgansk@gmx.de
[2] Department of Computer Science,
University of Liverpool, Chadwick Building,
Peach St, Liverpool L69 7ZF, U.K.,
igor@csc.liv.ac.uk

Abstract. In this paper we answer an open question about the exact bound on the maximal number of non-mergible states in nondeterministic finite automaton (NFA). It is shown that the maximal possible number of non-mergible states in a NFA that accepts a given regular language L is not greater than $2^n - 1$, where n is the number of states in the minimal deterministic finite automaton that accepts L. Next we show that the bound is reachable by constructing a NFA that have exactly $2^n - 1$ non-mergible states. As a generalization of this result we show that the number of states in a NFA that does not contain a subset of k mergible states, where $k > 1$, is bounded by $(k-1)(2^n - 1)$ and the bound is reachable.

1 Introduction

In this paper we answer an open question about the exact bound on the maximal number of non-mergible states in nondeterministic finite automaton (NFA). The notion of mergibility is closely related to the problem of finding small representations of regular languages, for which there is both practical and theoretical motivation.

It is well known that in some cases we can reduce the size of a finite automaton by merging its states without changing its accepting power. The question about reducing the number of states and transition of finite automata is well studied and has efficient solution for deterministic finite automata (DFA), but it is more complex for non-deterministic finite automata (NFA).

In many cases problems for non-deterministic automata are highly non-trivial and usually cannot be solved using classical methods for deterministic automata [6, 7]. For example the problem of identification of mergible states is quite difficult for NFA since the order of merging could lead to different non-mergible representations of the same regular language.

In this paper we work on the problem of mergible states [3] existence that can be formulated as follows:

[3] I.e. the states that we can merge without changing the accepted language of NFA

Problem 1. Given a regular language L. Is there a constant E_L such that any NFA that accepts L with a number of states greater than E_L has mergible states?

According to [2] one can distinguish two main ways of merging states: (1) a *weak* method (the weak mergibility), where two states are merged by simply collapsing one into the other and consolidating all their input and output transitions, and (2) a *strong* method (the strong mergibility), where one state is merged into another one by redirecting its input transitions toward the other state and completely deleting all its output transitions.

Very recently the upper bound on non-mergible states in case of the weak mergibility was found in [1]. Then it was shown in [2] that in the case of the strong mergibility the upper bound does not exist. The proof of the upper bound for the weak mergibility provided in [1] gives an effectively computable constant which, however, is very large and involves some imbricated Stirling numbers. In this paper we provide a solution for the Problem 1 (in a context of the week mergibility) by proving the lower and upper bounds that match exactly. We show that the maximal possible number of non-mergible states in a NFA that accepts a given language L is not greater than $2^n - 1$, where n is the number of states in the minimal deterministic finite automaton that accepts L. Next we show that the bound is reachable by construction of a NFA that have exactly $2^n - 1$ non-mergible states. As a generalization of this result we show that the number of states in a NFA that does not contain a subset of k mergible states, where $k > 1$, is bounded by $(k-1)(2^n - 1)$ and the bound is reachable.

The proof is based on our recent technique that was used in [4, 5] for characterization of the languages representable by the undirected graphs. In this paper, without loss of generality, we use edge labeled graphs instead of NFAs. In particular we represent a language L as a set of all paths in an edge-labeled graph that lead us to the final vertex instead of saying that L is accepted by a NFA.

On the one hand, this approach helps to study the properties of mergible and non-mergible states of a NFA in terms of edge-label graphs. On the other hand we believe that presented proofs, in the current form, are more readable and can be adapted for further research on graph representable languages [5]. The paper is organized as follows. In the first part of the paper we introduce definitions and prove several useful properties about equivalent and quasi-equivalent vertices. Then in the next section we prove our main result. The paper ends with some conclusions.

2 Preliminaries

2.1 Basic Notations and Definitions

Let X be a finite alphabet and X^* be a free monoid on the set X. We denote the empty word by e. The length of a word $w = x_1 x_2 \dots x_k$, where $x_i \in X$, $1 \le i \le k$, is denoted by $|w|$. The concatenation of two words $u \in X^*$, $v \in X^*$ we denote as uv. We call u as a prefix of the word uv. We denote by X_e the language $X \bigcup \{e\}$.

Definition 1. *Given languages $L, L' \subseteq X^*$. The basic language operations that we use are defined as follows:*

- *the concatenation of two languages: $LL' = \{wu | w \in L, u \in L'\}$*
- *the prefix contraction of a word w from the language L: $w^{-1}L = \{u | wu \in L\}$*
- *the prefix contraction of the language L' from the language L: $(L')^{-1}L = \bigcap_{w \in L'} w^{-1}L$.*

Note that $(L')^{-1}L$ we can define as follows:

$$(L')^{-1}L = \{u | \forall w \in L' : wu \in L\}.$$

Definition 2. *Let $G = (S, X, E, s_0, F)$ be a simple (i.e. without multiple edges), directed edge-labeled graph, where S is a set of vertices, X is a finite set of labels, $E \subset S \times X_e \times S$ is a set of directed labeled edges, $F \subset S$ is a set of final vertices. By sxt or (s, x, t) we denote an edge from the vertex s to the vertex t with the label x.*

Note that an edge in a graph G can be labeled with the empty word e. From now on by a *graph* we understand an edge-labeled graph. Any path in a graph is defined as a finite sequence

$$p = s_1 x_1 s_2 x_2 \ldots s_{k-1} x_{k-1} s_k$$

such that $s_i x_i s_{i+1} \in E$ for $1 \leq i < k$. The path p begins in the vertex s_1 and ends in s_k. The word $x_1 x_2 \ldots x_{k-1} \in X^*$ is a label of the path p that we denote as $\mu(p)$.

By $\mathbf{s}$ we denote the set of labels of all paths from the initial vertex to the vertex s. The set of labels of all paths from a vertex s to final vertices we denote by $\mathbf{s^{-1}}$. We also say that $\mathbf{s^{-1}}$ is a language generated by the vertex s and $L(G)$ is a language generated by the initial vertex of the graph G, i.e. $L(G) = \mathbf{s_0^{-1}}$.

Let A be a NFA that accepts a regular language L. We can think about a structure of A as a finite directed edge-labeled graph G_A. In accordance with our definitions we can state that $L = L(G_A)$. Moreover, it is obvious that we can uphold the above language equality by adding/deleting any states or transitions in A and adding/deleting corresponding vertices or labeled edges at the same time in G_A.

Definition 3. *Two vertices $s, t \in S$ are equivalent, if $\mathbf{s^{-1}} = \mathbf{t^{-1}}$. The relation between s and t we denote as ϵ.*

Definition 4. *Given a graph G that generates a language L. Two vertices s and t from the graph G are* quasi-equivalent *if $(\mathbf{s})^{-1}L = (\mathbf{t})^{-1}L$. The relation between s and t we denote as α.*

Definition 5. *A graph G is deterministic if every vertex of G does not have any two outgoing incident edges with the same label and does not have any outgoing edges lebeled by the empty word.*

Let us consider an equivalence relation $\rho \subseteq S \times S$, i.e. a reflexive, transitive and symmetric relation. By $\rho(s)$, where $s \in S$, we denote a set $\{t|(s,t) \in \rho\}$.

Definition 6. *The graph $G/\rho = (G', X', E', s'_0, F')$ is a factor graph of a graph G if $G' = \{\rho(s)|s \in S\}$, $\rho(s)x\rho(s') \in E'$ for all edges $sxs' \in E$, $s'_0 = \rho(s_0)$ and $F' = \{\rho(s)|s \in F\}$.*

By the merging of two vertices we understand a weak method, where two vertices are merged by simply collapsing one into the other and consolidate all their incomming and outgoing edges. We say that the graph G/ρ can be constructed from G by merging vertices of each set $\rho(s)$, $s \in S$.

Definition 7. *A subset R of vertices S in a graph G is mergible iff the language generated by the graph after merging of these vertices does not change, i.e. $L(G) = L(G/\rho)$, where $(s,t) \in \rho$ iff $s = t$ or $s,t \in R$.*

Now we define a simple graph that we call *Line*. Let $w = x_1x_2 \dots x_{k-1}$ be a word of a length $k-1$. By $Line(w)$ we denote a directed graph that is isomorphic to the graph with k vertices $s_1, ..., s_k$, and with the set of edges $\{s_ix_is_{i+1}|1 \le i < k\}$. We also call the vertex s_1 as initial and the vertex s_k as final. An example of a graph $Line(w)$, where $w = abcb$ is shown on the Figure 1.

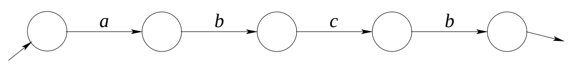

Fig. 1. An example of $Line(w)$, where $w = abcb$.

2.2 Intermediate Results

In this paper we consider languages in a given finite alphabet X, and graphs with the same alphabet X for labels. Without loss of generality we assume that all vertices in graphs are reachable from initial vertices and from any vertex at least one finite vertex is reachable in G.

Lemma 1. *Let G be a directed edge-labeled graph and s,t are two vertices in G, that $x^{-1}\mathbf{t^{-1}} \supseteq \mathbf{s^{-1}}$, and the edge (t,x,s) does not belong to the graph G. Then adding the new edge (t,x,s) into the graph G does not the language $L(G)$.*

Proof. Let us add one edge (t,x,s) to the graph G, that is obviously can only extend a language $L(G)$. Thus, it is enough to show that for any path p which goes via the edge (t,x,s) from the initial vertex to a final one we can find a path p' from the initial vertex to a final one without passing the edge (t,x,s) with the same label, i.e. $\mu(p) = \mu(p')$.

Let us consider a path $p = s_1x_1s_2x_2 \dots s_{k-1}x_{k-1}s_k$ from the initial vertex to a final one such that passes the edge (t,x,s) in the extended graph G. Let i be the maximum number, such that $(t, x, s) = (s_i, x_i, s_{i+1})$. According to the initial condition $x^{-1}\mathbf{t^{-1}} \supseteq \mathbf{s^{-1}}$ there is a path $s_i y_1 t_1 y_2 t_2 \dots t_{l-1} y_l t_l$ from s_i to a final vertex t_l with the label $y_1 y_2 \dots y_l$ such that $y_1 y_2 \dots y_l = x_i x_{i+1} x_{i+2} \dots x_{k-1}$, where $x_i = x$, but which does not pass the edge (t, x, s). Thus the number of (t, x, s)-edges in the path $p' = s_1 x_1 s_2 \dots s_i y_1 t_1 y_2 t_2 \dots t_{l-1} y_l t_l$ is less than in p by one. From it follows that, by repeating the above reduction a finite number of times, for any path p in G from the initial vertex to a final one we can construct a path p' in G from the initial vertex to a final one, that does not contain the edge (t, x, s), and $\mu(p) = \mu(p')$.

Lemma 2. *For any (finite or infinite) graph G the identity $L(G) = L(G/\epsilon)$ holds.*

Proof. Let G be an initial edge-labeled graph and let s and t be equivalent vertices in G. Let us assume that for some vertex s' the graph G contains the edge (s, x, s') and there is no edge (t, x, s') in the graph G. In this case we meet initial condition of Lemma 1, so the language represented by the graph G would not be changed by extending G with the edge (t, x, s'). Moreover, a language generated by any vertex in the extended graph would not be changed as well.

Let us extend the graph G in the following way: for any equivalent vertices s and t and some vertex s' we add the edges (t, x, s') iff the edge (s, x, s') belongs to the graph.

From it follows that the sequence $\epsilon(s_1)x_1\epsilon(s_2)x_2 \dots x_{k-1}\epsilon(s_k)$ is a path from the initial vertex to a final one in the factor graph G/ϵ iff there exist a path $r_1 x_1 r_2 x_2 \dots x_{k-1} r_k$ in the graph G, where $r_i \in \epsilon(s_i)$, $1 \leq i \leq k$. Above facts show that for any graph G the equality $L(G) = L(G/\epsilon)$ holds.

Lemma 3. *Given a graph G and a language L, then $L(G) \subseteq L$ iff $\mathbf{s^{-1}} \subseteq (\mathbf{s})^{-1}L$ for any vertex s in the graph G.*

Proof. The above lemma follows from the fact that a set of all paths via a vertex s to a final vertex can be represented by $\mathbf{ss^{-1}}$. Since $\mathbf{ss^{-1}} \subseteq L$ holds it is equivalent to $\mathbf{s^{-1}} \subseteq (\mathbf{s})^{-1}L$. The proof of the reverse part is trivial since from the statement $\mathbf{s^{-1}} \subseteq (\mathbf{s})^{-1}L$, where s is the initial vertex, follows that $L(G) \subseteq L$.

The next lemma gives us the common property for equivalent and quasi-equivalent vertices.

Lemma 4. *For any graph G the identity $L(G) = L(G/\alpha)$ holds.*

Proof. Let G be an initial edge-labeled graph. Note that the operation of merging any two vertices in a edge-labeled graph can only extend the language that it represents. Let us now assume that $W = (\mathbf{s})^{-1}L(G) = (\mathbf{t})^{-1}L(G)$, i.e. s and t are quasi-equivalent. From Lemma 3 follows that $\mathbf{s^{-1}} \subseteq W$ and $\mathbf{t^{-1}} \subseteq W$.

Let us construct an extension of a graph G by adding new vertices and edges in such a way that $\mathbf{s^{-1}} = \mathbf{t^{-1}} = W$ holds in the extended graph. In particular we can do it as follows. Let us add by direct sum all graphs $Line(w)$, $w \in W$, to the graph G and merge the vertex s with the initial vertices of $Line(w)$, $w \in W$.

Next we can repeat the same construction with the vertex t. According to the Lemma 1 the language represented by a graph G after its extension has not been changed.

Since the vertices s and t are now equivalent in the extended graph G we can merge s and t without changing the language represented by this graph. Now we can delete all introduced edges and nodes to get a graph that can be constructed by just merging quasi-equivalent vertices s and t.

Corollary 1. *Quasi-equivalent vertices are mergible.*

Definition 8. *A graph G is saturated if for any vertex s the equality $(\mathbf{s})^{-1}L = \mathbf{s^{-1}}$ holds, i.e. any quasi-equivalent vertices in the graph are equivalent.*

Lemma 5. *Two vertices in a saturated graph are mergible iff they are equivalent.*

Proof. It is obvious that equivalent vertices are mergible.

Assume s and t are not equivalent in a saturated graph G. Then there are two words u and w, such that either $u \in \mathbf{s}$, $u \notin \mathbf{t}$, $w \notin \mathbf{s^{-1}}$, $w \in \mathbf{t^{-1}}$ or $u \notin \mathbf{s}$, $u \in \mathbf{t}$, $w \in \mathbf{s^{-1}}$, $w \notin \mathbf{t^{-1}}$ holds. In this case the word uw does not belong to the language $L(G)$, but could belong to the language $L(G)$ if we merge the vertices s and t, i.e. s and t are not mergible.

3 Main Result

Theorem 1. *Given a regular language L and a graph G such that it does not contain any mergible vertices and $L(G) = L$. The number of vertices in the graph G is bounded by $2^n - 1$, where $n = |\{w^{-1}L | w \in L\}|$ and the bound is reachable.*

Proof. Let us consider a graph G satisfying to the above statement. Since quasi-equivalent vertices are mergible then the maximum number of non-mergible vertices is bounded by a maximal number of vertices that are not pairwise quasi-equivalent. Since a set $(W)^{-1}L$, where $W \subseteq L$, is defined as $\cap_{w \in W} w^{-1}L$ the number of different quasi-equivalent classes in G cannot exceed the number of different non-empty subsets of the set $\{w^{-1}L | w \in L\}$, which is $2^n - 1$.

Now we prove that $2^n - 1$ is a reachable bound. The Figure 2 shows a deterministic graph H with the initial vertex n and all final vertices. The vertex k, $1 \leq k \leq n$, generates the language that contains only all possible initial prefixes of the words from the language defined by the regular expression $(a_1 a_2 \dots a_n)^*$, where $a_i \in \{0, 1\}$ for $i \in \{1, 2, \dots, n\} - \{k\}$ and $a_k = 0$. Hence the graph H does not have any equivalent vertices.

Let us construct a saturated graph G with $2^n - 1$ vertices such that $L(H) = L(G)$. First we define a set of vertices $S = \{s | s \subseteq \{1, 2, \dots, n\}, s \neq \emptyset\}$. The vertex $\{n\}$ is initial, and all vertices are final. For any $s \in S$, $t \in S$ the edge $(s, 0, t)$ belongs to G iff for all $x \in s - \{1\}$ we have that $x - 1 \in t$ and if $1 \in s$ then $n \in t$. For any $s \in S$, $t \in S$ the edge $(s, 1, t)$ belongs to G iff $1 \notin s$ and for all $x \in s$ we have that $x - 1 \in t$. It follows from the construction that for each vertex $s \in S$ the equality $\mathbf{s^{-1}} = \bigcap_{i \in s} \mathbf{i^{-1}}$ holds, where $\mathbf{i^{-1}}$ is a language generated by a vertex

i from the graph H. Hence the graph G is saturated and for any vertex $s \in S$ the language $\mathbf{s^{-1}}$ consists of all possible prefixes of the language defined by the regular expression $(a_1 a_2 \ldots a_n)^*$, where $a_i = (0 \vee 1)$ for $i \in \{1, 2, \ldots, n\} - s$ and $a_i = 0$ for $i \in s$. From it follows that for different vertices s we have different languages $\mathbf{s^{-1}}$ and according to the Lemma 5 there are no mergible vertices in G. This ends our proof.

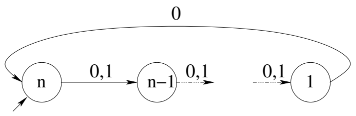

Fig. 2. Deterministic graph with the initial vertex n and all final vertices.

Corollary 2. *Let $k > 1$ be an integer, L be a regular language, and G be a finite graph that does not contain a subset of k mergible vertices and $L(G) = L$. The number of vertices in the graph G is bounded by $(k-1)(2^n - 1)$, where $n = |\{w^{-1}L | w \in L\}|$ and the bound is reachable.*

Proof. Let L be a regular language, G be a finite graph with a number of vertices greater than $(k-1)(2^n-1)$ and such that $L(G) = L$, where $n = |\{w^{-1}L | w \in L\}|$. The number of different quasi-equivalent classes in G is not greater than $2^n - 1$. According to the pigeonhole principle there is a subset of k quasi-equivalent and therefore mergible vertices in the graph G.

Consider the graph G and the language L from the proof of the Theorem 1. We have that $n = |\{w^{-1}L | w \in L\}|$ and G consists of $(2^n - 1)$ non-mergible vertices. Let G_1,G_2,. . .,G_{k-1} be $k - 1$ copies of G, and s_{0i} the inital vertex of G_i, where $1 < i < k$. Let H be a graph constructed by direct sum of the graphs G_i, $1 < i < k$, and such that H has the initial vertex s_{01} and for all vertices s, t in H, such that $x^{-1}\mathbf{t^{-1}} \supseteq \mathbf{s^{-1}}$ and the edge (t, x, s) does not belong to the graph H we construct the edge (t, x, s). does not change the langauge $L(H)$. By applying Lemma 1 we state that the above construction does not change the langauge $L(H)$. So it is obvious that $L(G) = L(H)$, all final vertices in H are reachable and it does not contain a subset of k quasi-equivalent vertices.

Corollary 3. *Let $k > 1$ be an integer, L be a regular language, and A be a NFA that does not contain a subset of k mergible states and accept L. The maximal number of states in A is bounded by $(k-1)(2^n - 1)$, where n is the number of states in the minimal deterministic finite automaton that accepts L.*

Proof. The proof is straightforward and follows from the Corollary 2. By a NFA A we understand the graph G from Corollary 2. In addition to it we use the fact that a number of states in the minimal deterministic automaton that accepts L is equal to the cardinality of a set $\{w^{-1}L | w \in L\}$.

4 Conclusion

In this paper we answered a question about the exact bound on non-mergible states in a nondeterministic finite automaton. It was shown that the tight bound on the maximal possible number of non-mergible states in NFA that accepts a given language L is $2^n - 1$, where n is the number of states in the minimal deterministic finite automaton that accepts L. It is easy to see that the same bound holds for a case of ε-free NFA since the upper bound proofs cover a case of ε-free NFA and the low bound constructions do not contain any ε-transitions.

Another interesting aspect of this paper is the use of quasi-equivalence relation that have a number of important properties not only for finite but also for infinite graphs or automata.

Corollary 4. *Let L be a regular language and G be an infinite graph such that $L = L(G)$. The factor graph G/α is finite.*

Let us show using simple example that the above statement does not hold if we substitute the relation α by ϵ. The infinite graph G on Figure 3 have one initial vertex that is the vertex without incoming edges and one final vertex that is the vertex without outgoing edges.

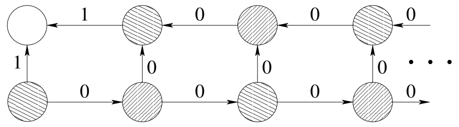

Fig. 3. A infinite graph G which is isomorphic to G/ϵ.

It is easy to see that the language generated by G corresponds to the expression $(00)^*1$. The above graph does not have any equivalent vertices since each vertex of G generates a different language. On the other hand, the vertices colored with the same pattern are quasi-equivalent and mergible, because of the fact that $(0^{2i})^{-1}L(G) = L(G)$ and $(00^{2i})^{-1}L(G) = 0^{-1}L(G)$, for any $i \geq 0$.

The factor graph G/α is shown on Figure 4.

The question about applicability of quasi-equivalence relation for the minimization of NFA that accepts a given regular language could be promising next step along the research initiated in [8].

References

1. C. Campeanu, N. Santean, S. Yu. Mergible states in large NFA. Theoretical Computer Science, 330, 23-34, (2005)
2. C. Campeanu, N. Santean, S. Yu. Large NFA without mergible states. Proceedings of 7th International Workshop on Descriptional Complexity of Formal Systems, DCFS 2005, 75-84 (2005)

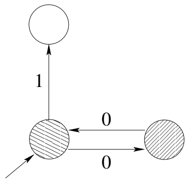

Fig. 4. The graph G/α.

3. S. Eilenberg. Automata, Languages, and Machines. Academic Press, Inc. Orlando, FL, USA (1974)
4. I.S. Grunsky, O. Kurganskyy. Properties of languages that generated by undirected vertex-labeled graphs. In proceeding of 8-th International Seminar on Discrete mathematics and its applications, Moscow State University, 267-269 (2004) [in Russian].
5. I.S. Grunsky, O. Kurganskyy, I. Potapov. Languages Representable by Vertex-labeled Graphs. MFCS, LNCS 3618, 435-446 (2005)
6. J. Hromkovic, S. Seibert, T. Wilke. Translating Regular Expressions into Small ε-Free Nondeterministic Finite Automata. J. Comput. Syst. Sci. 62(4): 565-588 (2001)
7. J. Hromkovic, G. Schnitger. NFAs With and Without ε-Transitions. ICALP 2005, 385-396 (2005)
8. L. Ilie, R. Solis-Oba, S. Yu. Reducing the Size of NFAs by Using Equivalences and Preorders. CPM 2005: 310-321 (2005)