Deterministic Generalized Automata *

Dora Giammarresi†
Dipartimento di Matematica Pura ed Applicata, Università di L'Aquila
via Vetoio, loc. Coppito, 67100 L'Aquila, ITALY (dora@univaq.it)

Rosa Montalbano‡
Dipartimento di Matematica e Applicazioni, Università di Palermo
via Archirafi 34, 90123 Palermo, ITALY (rosalba@altair.math.unipa.it)

TR-96-016

Abstract

A generalized automaton (GA) is a finite automaton where the single transitions are defined on words rather than on single letters. Generalized automata were considered by K. Hashiguchi who proved that the problem of calculating the size of a minimal GA is decidable.

We define the model of deterministic generalized automaton (DGA) and study the problem of its minimization. A DGA has the restriction that, for each state, the sets of words corresponding to the transitions of that state are prefix sets. We solve the problem of calculating the number of states of a minimal DGA for a given language, by giving a procedure that effectively constructs a minimal DGA starting from the minimal equivalent (conventional) deterministic automaton.

*Work partially supported by the ESPRIT II Basic Research Actions Program of the EC under Project ASMICS 2 (contract No. 6317) and in part by MURST under project 40% *Algoritmi, Modelli di Calcolo, Strutture Informative.*

†Work partially done while visiting the *International Computer Science Institute (ICSI)* Berkeley, CA.

‡Partially supported by a research fellowship by *Consiglio Nazionale delle Ricerche (CNR)*, Grant n.201.01.121.

1 Introduction

Generalized automata (GA) were introduced by Eilenberg as a model of representation for regular languages that extends the notion of finite automata by allowing the single transitions to be defined on words rather than on single letters. Intuitively, a generalized automaton can be obtained from a conventional one by shrinking long paths of the graph in a unique edge with a "long" label. Therefore, generalized automata are usually more concise than conventional ones representing the same event.

In the past decades, several efforts have been devoted to compute the complexity of representation of a given language inside different models of representation (deterministic, non-deterministic, unambiguous, two-way, alternating, probabilistic, pebbles automata, regular expressions, logical formalisms and so on). The complexity of a language in a given model is generally understood as the size of the minimal representation of the language in that model. For example, a classical measure of the complexity of a finite automaton is its number of states and the complexity of a language in this model is the number of states of a minimal (with respect to the number of states) automaton recognizing it.

In this context, Hashiguchi in 1991 investigated the problem of computing the size of the minimal representation of a given regular language in the model of generalized automata (see [H91]). In particular, he proved that the problem of calculating the number of states of a minimal GA is actually decidable.

A strictly related problem consists of effectively computing a minimal representation of a given language in a model. In the case of conventional deterministic finite automata, it can be proved that the minimal automaton is unique and an algorithm to calculate it starting from any equivalent deterministic automaton can be obtained using the Myhill-Nerode's theorem (see, for example, [HU79]). For non-deterministic automata there are only partial results stating that there is no unique minimal automaton but there are no constructive procedures for computing it, excepting the one that lists all possible automata. In [JR91] the computational complexity of different problems concerning minimization is studied in a general setting for non-deterministic automata and it is proved that all these problems are computationally hard.

In this paper we introduce the model of *deterministic generalized automata* (DGA) and deal with the minimization problem for this model. In order to preserve all properties implied by the notion of determinism in the case of conventional automata, DGA have the restriction that the sets of words corresponding to the transitions of each state are prefix sets. We solve the problem of computing the number of states of a minimal DGA by giving a procedure to construct a minimal DGA for a given language starting from the minimal (conventional) deterministic one. We introduce two operations that allow one to reduce the number of states of a DGA: the first, called *$\mathcal{I}$-reduction*, contracts states that are "indistinguishable" and the second, called *$\mathcal{S}$-reduction*, suppresses states that are "superfluous". Then we give the conditions under which such operations can be performed. We show that there can be deterministic GA that are irreducible (with respect to the above operations) but not minimal and give necessary and sufficient conditions to reduce a deterministic GA to get a minimal one. Moreover, we show that, differently from the case of conventional deterministic automata, the minimal deterministic GA is not unique.

The size of the minimal representation of a language in a given model (which measures the complexity of the language) plays a primary role also in comparing different models according to their intrinsic succinctness. Much work has been devoted to studying succinctness of representation when transducers are considered (see, for example [WK94, M94]). In the case of finite automata, very recently, Harel et al. studied exponential discrepancies in the succinctness of finite automata when augmented by combinations of various additional mechanisms like alternation (i.e. both universal and existential branching), concurrency, "two-wayness" and pebbles (see [GH94]). We conclude the paper by discussing problems of discrepancy in succinctness between non-deterministic and deterministic versions of generalized automata and give some open problems.

Part of the results of this paper appeared already in [GM95].

2 Preliminaries

In this section we give first some terminology on languages and automata. Then we recall some definitions, properties and problems related to the minimization of finite automata. The notations we use are mainly borrowed from [P90].

2.1 Basic notations

We denote by Σ a finite alphabet and by Σ^* the free monoid generated by Σ. The elements of Σ are called letters, those of Σ^* are called words. A language over Σ is a subset of Σ^* (i.e. a set of words).

Given two words v and w, we say that v is a *prefix* of w if there exists a word u such that $w = vu$. Given a set of words X, we say that X is a *prefix set* if no word in X is prefix of some other word in X. Given two sets of words X and Y, the *concatenation of* X *and* Y, indicated as $X \cdot Y$, contains all words xy with $x \in X$ and $y \in Y$. The length of a word w is denoted by $|w|$.

A *finite (non deterministic) automaton* is a quintuple $\mathcal{A} = (\Sigma, Q, I, F, E)$ where Q is a finite set of *states*, $I, F \subseteq Q$ are the sets of *initial* and of *final states* respectively and $E \subseteq Q \times \Sigma \times Q$ is a set of labeled *edges*. We denote an edge of $\mathcal{A}$ by $e = (r, a, s)$, where $r, s \in Q$ and $a \in \Sigma$ is the label of e. A *path* of *length* n in $\mathcal{A}$ is a sequence of edges $e_i = (r_i, a_i, r_{i+1}) \in E$, for $i = 1, \ldots n$, that we denote by $[r_1, a_1 \ldots a_n, r_{n+1}]$. The word $w = a_1 \ldots a_n$ is the label of the path $e_1, \ldots, e_n$. If $r_1 \in I$ and $r_{n+1} \in F$ then $e_1, \ldots, e_n$ is called *accepting path* and word w is said *accepted by* (or *recognized by*) $\mathcal{A}$.

The set of all words accepted by an automaton $\mathcal{A}$ is called the *language accepted (or recognized)* by $\mathcal{A}$ and it will be referred to as $\mathcal{L}(\mathcal{A})$. A language L is *recognizable* if L is the language accepted by some finite automaton.

Two finite automata are *equivalent* if they accept the same language. An automaton $\mathcal{A} = (\Sigma, Q, I, F, E)$ is *deterministic* if $|I| = 1$, and for any state $q \in Q$ and any letter $a \in \Sigma$, there exists at most one state $p \in Q$ such that edge $(q, a, p) \in E$. Deterministic and non-deterministic automata recognize the same family of languages: that is, given any non-deterministic automaton it can be constructed an equivalent deterministic automaton (see, for example, [HU79]).

For the sequel we will assume to deal with trim automata (see [E74], p.23 for the formal definition) that is automata whose states are all *accessible* and *co-accessible* (i.e. all the states are in some path from an initial to a final state). This is without loss of generality, since automata are considered because of the languages they recognize and, given any automaton, we can delete all not accesssible states without changing the recognized language. Notice that this has the consequence that the automata could be not complete (that is, that the labels of the edges leaving a given state could not cover all the letters of the alphabet Σ) and therefore that not all possible words of Σ^* are labels of paths starting from an initial state. When a word $w \in \Sigma^*$ is not a label of any path starting from an initial state, we will assume that w is not accepted by the automaton.

We conclude this section by giving some further notations on graphs that will be useful in the sequel. Given an edge $e = (p, a, q)$, we call p and q as the beginning and the end of e, respectively. An edge e is said *incident* to a state q if it begins or ends in q. An edge $e = (p, a, q)$ is a *self-loop* if $p = q$. A path $e_1, e_2, \ldots, e_n$, where $e_i = (r_i, a_i, r_{i+1})$, is a *cycle* if $r_1 = r_{n+1}$. Notice that, a self-loop is a cycle of length one. A graph is acyclic if it does not contain any cycles.

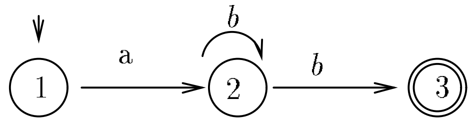
Let $G = (Q, E)$ be a directed graph where Q is the set of vertices and E is the set of edges and let $S \subseteq Q$: the *subgraph of G induced by S* is the graph $G_S = (S, E')$ such that $E' \subseteq E$ is the set of all edges whose beginnings and ends are in S. We say that S induces a *maximal acyclic* subgraph if G_S is an acyclic subgraph of G that is not a subgraph of any other acyclic subgraph of G.

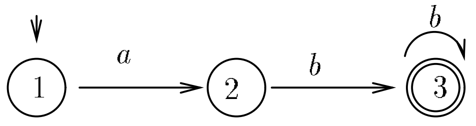
2.2 Minimal automata

We now consider the problem of minimizing a given automaton, that is the problem of finding a "minimum size" automaton equivalent to a given one. The size of an automaton is usually measured by counting the number of its states (notice that the number of edges is linearly related to the number of states). Then, formally, a *minimal automaton* for a language L is an automaton with the minimum number of states among all equivalent automata accepting L.

Remark that, in general, given language L,there is not a unique minimal non-deterministic automaton recognizing L. This is shown by the following example.

Example 2.1 Let $L = (ab)b^*$ a language over $\Sigma = \{ a, b \}$. Then L is recognized by the two different finite automata given below, that are minimal for L:





There are no known (efficient) algorithms to compute a minimal non deterministic automaton that recognizes a given language. The best we can do is to compute all possible non-deterministic automata in an incremental fashion (starting with a one-state automaton and adding states) until we find one that recognizes the given language. In [JR91] many problems regarding minimization of non-deterministic automata are investigated, and it is proved that they are all computationally hard. Moreover, in last years, there have been many attemps to define particular "normal forms" for non-deterministic automata which solve the problems of unicity and calculability of the minimal automaton (for more details see, for example, [C86, JMR]).

When we restrict the minimization problem to deterministic automata, everything becames easier to handle. A *minimal deterministic automaton* for a given language L is an automaton with minimal number of states among all equivalent deterministic automata accepting L. Notice that, in general, a minimal deterministic automaton has many more states than the corresponding non-deterministic one. Given a deterministic automaton $\mathcal{A} = (\Sigma, Q, i, F, E)$, there is a unique minimal deterministic automaton equivalent to a $\mathcal{A}$ and it can be to obtained as follows (see [HU79] or [P90] for more details). We define an equivalence relation in the set of states Q called *indistinguishability*: two states $p, q \in Q$ are *indistinguishable* if for any word $w \in \Sigma^*$, there exists a path $[p, w, f]$ with $f \in F$ if and only if there exists a path $[q, w, f']$ with $f' \in F$. The minimal deterministic automaton equivalent to $\mathcal{A}$ can be obtained by contracting the classes of indistinguishable states of $\mathcal{A}$.

3 Generalized automata

In this section we consider a generalization of the model of automata described above: we will allow the labels of the edges to be words of any finite length instead of single letters only. For the sequel we will refer to the model of automaton described in previous sections as "conventional automaton" while this more general model will be referred as "generalized automaton".

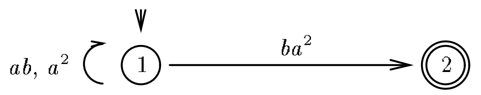
Generalized automata were introduced by Eilenberg in [E74] and they can be formally defined as follows.

Definition 3.1 *A* generalized (non-deterministic) automaton *(GA) is a quintuple* $\mathcal{A} = (\Sigma, Q, I, F, E)$ *where* Q *is a finite set of* states*,* $I, F \subseteq Q$ *are the sets of* initial *and* final *states and* $E \subseteq Q \times \Sigma^* \times Q$ *is a* finite *set of labeled* edges.

Notice that the finiteness condition for the set of edges E is now necessary to get a finite device: without this restriction we could have as many edges as the words in Σ^*.

The notion of recognizability for generalized automata is the same as for conventional automata. More precisely, a word $w \in \Sigma^*$ is recognized by a generalized automaton $\mathcal{A}$ if there exist words $w_1, w_2, \ldots, w_n \in \Sigma^*$ and edges $e_1, e_2, \ldots, e_n \in E$ such that w_i is the label of e_i, for $i = 1, \ldots, n$, the sequence $e_1, e_2, \ldots, e_n$ is an accepting path and $w_1 w_2 \ldots w_n = w$.

Observe that, in this case, the fact that a word w is accepted by a generalized automaton does not imply that all factors of w are labels of some path in the automaton. Consider, for example, the generalized automaton below recognizing the language $L = (ab + a^2)^* ba^2$ over $\Sigma = \{a, b\}$. Notice that the prefix ab^2 of the accepted word ab^2a^2 does not correspond to any path in the graph.



In general, by allowing the labels of the edges to be words of any length, a generalized automaton gives a representation of a language by means of a graph that is possibly much smaller (at least in the number of vertices) than the corresponding representation by conventional automaton. For example, if S is a finite language, then it can be described (recognized) by a GA with only two states, despite of the length of its words. Moreover, the language S^* can be recognized by a GA with one state only.

Generalized automata were considered by Hashiguchi in [H91]. He studied the problem of calculating the number of states of a minimal generalized automaton for a given language and proved that this problem is decidable.

If $\mathcal{A}$ is a GA, denote by $D(\mathcal{A})$ the maximal length of the labels of the edges in $\mathcal{A}$. The decidability is a consequence of the following theorem.

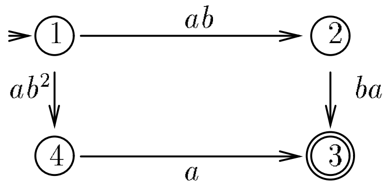
Theorem 3.1 [K. Hashiguchi 1991] *Let L be a recognizable language and m the cardinality of the syntactic monoid for L. There exists a minimal generalized automaton $\mathcal{A}$ recognizing L such that $D(\mathcal{A}) \leq 2m(m+2)(4m(m+2)+3)$.*

We observe that the number $D(\mathcal{A})$ in the statement of the theorem is actually a very huge number. This because the cardinality of the syntactic monoid of a language is of the order of n^n where n is the number of states of the minimal deterministic (conventional) automaton for L (see [P90]).

4 Deterministic generalized automata

We now define and study the model of generalized automaton in the deterministic case. We remark that, in the case of conventional automata, the "local" condition that, given any state q, for any letter a there is at most one edge beginning in q with label a implies the "global" condition that also for any word w there is at most one path beginning in q with label w. In some sense we can say that a "local determinism" implies also a "global determinism". The same does not hold in the case of generalized automata as shown by the following example.

Example 4.1 Consider the following GA.



$$\begin{tikzcd} \arrow[r] & 1 \arrow[r, "ab"] \arrow[d, "ab^2"'] & 2 \arrow[d, "ba"] \\ & 4 \arrow[r, "a"'] & 3 \end{tikzcd}$$

This automaton is deterministic in the "classical" sense: in fact the labels of the edges beginning in any given state q are all different. Nevertheless there exist two paths with label ab^2 connecting state 1 to state 3.

In order to capture the "global" properties of the classical notion of determinism we need stronger conditions on the set of edges incident any given state. We give first a definition.

Definition 4.1 *Let $\mathcal{A} = (\Sigma, Q, I, F, E)$ be a generalized automaton and let $q \in Q$ be a state of $\mathcal{A}$. The set of* words of q *is $W(q) = \{w \in \Sigma^* \mid (q, w, r) \in E\}$.*

That is, set $W(q)$ contains the labels of all edges beginning in q.

Definition 4.2 *Let $\mathcal{A} = (\Sigma, Q, I, F, E)$ be a generalized automaton. We say that $\mathcal{A}$ is* deterministic *if $I = \{i\}$ and for any state $q \in Q$, the set $W(q)$ is a prefix set.*

Notice that the condition that $W(q)$ is a prefix set effectively guarantees that for any state q and for any word w there is at most one path beginning in q with label w. Moreover conventional deterministic automata satisfy the above definition because the $W(q)$'s are subsets of the alphabet that is a prefix set.

In this paper we focus on the problem of minimizing (as reducing the number of states) a given DGA. We will define two operations that transform a DGA into a smaller equivalent one. The first operation contracts indistinguishable states similarly to the minimization operation for conventional deterministic automata. The second operation exploits the definition of generalized automaton that allows labels of any length: the number of states can be reduced by shrinking long paths in a unique edge with a "long" label. This two operations will be called *$\mathcal{I}$-reduction* and *$\mathcal{S}$-reduction*, respectively.

4.1 $\mathcal{I}$-reductions

Given a DGA $\mathcal{A} = (\Sigma, Q, i, F, E)$, for any $q \in Q$, we denote by L_{qF} the set of words corresponding to paths from state q to a final state. We give the following definition.

Definition 4.3 *Let $\mathcal{A} = (\Sigma, Q, i, F, E)$ be a DGA. Two states $p, q \in Q$ are* indistinguishable *(write $p \sim q$) if $L_{pF} = L_{qF}$.*

Notice that the above definition of indistinguishability among states is an extension to generalized automata of the corresponding definition for conventional automata (cf. Section 2.2 or [HU79]).

The indistinguishability $\sim$ is an equivalence relation over the set of states Q. Therefore we can define an operation, that we call *$\mathcal{I}$-reduction*, that, given a DGA $\mathcal{A}$, defines a new DGA $\mathcal{A}' = \mathcal{I}(\mathcal{A})$ by contracting all the states belonging to the same equivalence class in one state. Then, the set of states of the $\mathcal{I}$-reduced DGA $\mathcal{A}'$ is the quotient of Q by $\sim$. The edges of $\mathcal{A}'$ are defined as follows. If $[q]$ denotes the equivalence class of state q, $W([q])$ is defined as the maximal prefix set of the shortest words in $\bigcup_{p \sim q} W(p)$. Observe that, for any $w \in W([q])$ there exists at least a state $p \sim q$ such that $w \in W(p)$. Then, for any (p, w, p') in $\mathcal{A}$ there is edge $([q], w, [p'])$ in $\mathcal{A}'$.

We now give a formal definition for the $\mathcal{I}$-reduction.

Definition 4.4 *Given a DGA $\mathcal{A} = (\Sigma, Q, i, F, E)$, the corresponding $\mathcal{I}$-reduced automaton $\mathcal{I}(\mathcal{A}) = \mathcal{A}' = (\Sigma, Q', i', F', E')$ is defined as follows.*

- $Q' = Q/\!\sim\, = \{q' \mid q' = [q] = \bigcup_{p \sim q}\{p\}\}$;
- $i' = [i]$;
- $F' = \{[f] \mid f \in F\}$;
- *$W([q])$ is the maximal prefix subset of $\bigcup_{p \sim q} W(p)$ such that if $w_1, w_2 \in \bigcup_{p \sim q} W(p), w_2 = w_1 v \Rightarrow w_2 \notin W([q])$;*
- *$E' = \{([p], w, [q]) \mid w \in W([p])$ and $\exists\, p' \sim p, q' \sim q : (p', w, q') \in E\}$.*

We now prove that the DGA $\mathcal{A}'$ as in the definition above, is equivalent to $\mathcal{A}$.

Lemma 4.1 *Let $\mathcal{A} = (\Sigma, Q, i, F, E)$ be a DGA and let $\mathcal{A}' = \mathcal{I}(\mathcal{A})$. Then $\mathcal{A}'$ is a DGA equivalent to $\mathcal{A}$.*

Proof: Let $\mathcal{A}' = (\Sigma, Q', i', F', E')$ as in the Definition 4.4. The fact that $\mathcal{A}'$ is a *DGA* holds by construction since it has a unique initial state and for any state $q' \in Q'$ the set $W(q')$ is a prefix set.

We now prove that $\mathcal{A}$ and $\mathcal{A}'$ recognize the same language. Let us first show that $\mathcal{L}(\mathcal{A}') \subseteq \mathcal{L}(\mathcal{A})$. Let $w \in \mathcal{L}(\mathcal{A}')$: there exists a final state f such that $[\,[i], w, [f]\,]$ is an accepting path in $\mathcal{A}'$. Then, in $\mathcal{A}$ there exist two states p, q indistinguishable from i and f, respectively, such that $[p, w, q]$ is a path in $\mathcal{A}$. Since $q \sim f$, then $q \in F$ and $w \in L_{p,F} = L_{i,F} = \mathcal{L}(\mathcal{A})$.

Similar arguments can be used to prove the reverse inclusion $\mathcal{L}(\mathcal{A}) \subseteq \mathcal{L}(\mathcal{A}')$. □

Notice that $\mathcal{I}(\mathcal{A})$ does not contain any pair of indistinguishable states. We give the following definition.

Definition 4.5 *A generalized automaton is* $\mathcal{I}$-irreducible *if it has no indistinguishable states.*

4.2 $\mathcal{S}$-reductions

We now define another transformation, that we call $\mathcal{S}$-reduction, to reduce the number of states of a generalized automaton. Let $\mathcal{A}$ be a DGA and q be a state of $\mathcal{A}$: the $\mathcal{S}$-reduced automaton $\mathcal{S}(\mathcal{A}, q)$ is obtained from $\mathcal{A}$ by suppressing the state q and redefining all the edges that were incident in q. More precisely, we suppress state q together with all its incident edges and, for any pair of edges (r, u, q) and (q, v, s) that were in $\mathcal{A}$, we define a new edge (r, uv, s).

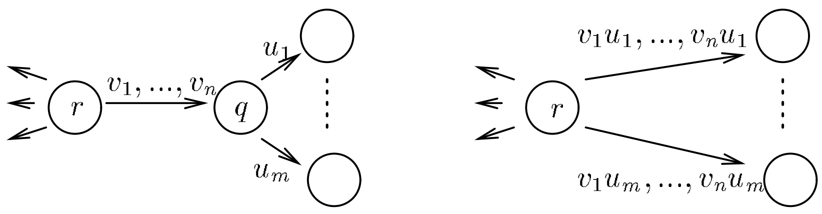
Given two states r, s of $\mathcal{A}$, we denote by L_{rs} the set of words corresponding to the labels of all paths from r to s in $\mathcal{A}$. The $\mathcal{S}$-reduction suppress states in $\mathcal{A}$ preserving sets L_{rs} for any pair of states r, s not suppressed. Observe that, in order to preserve sets L_{rs} without compromising the finiteness of automaton $\mathcal{S}(\mathcal{A}, q)$, state q must not have self-loops. Moreover, since our final goal is to minimize a DGA, we are actually interested in transformations that reduce a DGA preserving the recognized language (i.e. preserving all sets L_{if} where $f \in F$): therefore we do not apply $\mathcal{S}$-reduction both to i and to any final state. We give the following definition.

Definition 4.6 *Let $\mathcal{A} = (\Sigma, Q, i, F, E)$ be a DGA. A state $q \in Q$ is a* superfluous state *for $\mathcal{A}$ if q is neither an initial nor a final state and it has no self-loops.*

The set of all superfluous states for $\mathcal{A}$ will be denoted by *Superf*(Q). We now formally define the $\mathcal{S}$-reductions.

Definition 4.7 *Let $\mathcal{A} = (\Sigma, Q, i, F, E)$ be a DGA and $q \in Q$ be a superfluous state. Then $\mathcal{S}(\mathcal{A}, q) = (\Sigma, Q_q, i, F, E_q)$ is a (generalized) automaton where $Q_q = Q - \{q\}$ and $(r, u, s) \in E_q$ if either $(r, u, s) \in E$ or there exist $(r, u_1, q), (q, u_2, s) \in E$ such that $u_1 u_2 = u$.*

For each $r \in Q_q$, the set $W_q(r)$ of words associated to r in the transformed automaton can be calculated starting from the sets $W(r)$ and $W(q)$ as follows. We split the set $W(r)$ in two disjoint subsets $W(r) = X(r, q) \cup \overline{X}(r, q)$ such that $X(r, q)$ contains the words that are labels of edges ending in state q and $\overline{X}(r, q)$ is its complement in $W(r)$. Then, we have: $W_q(r) = X(r, q) \cdot W(q) \cup \overline{X}(r, q)$.



Lemma 4.2 *Let $\mathcal{A}$ be a DGA and let q be a superfluous state for $\mathcal{A}$. The transformed automaton $\mathcal{S}(\mathcal{A}, q)$ is a DGA equivalent to $\mathcal{A}$.*

Proof: Let $\mathcal{A}_q = \mathcal{S}(\mathcal{A}, q) = (\Sigma, Q_q, i, F, E_q)$ be as in the Definition 4.7. First, we prove that $\mathcal{S}(\mathcal{A}, q)$ is a DGA by showing that the set $W_q(r)$ is a prefix set for any $r \in Q_q$. If, in the original automaton $\mathcal{A}$, the state r has not outgoing edges entering q then the set $X(r, q) = \emptyset$. Therefore $W_q(r) = \overline{X}(r, q) = W(r)$ is a prefix set since $\mathcal{A}$ is a DGA. Otherwise, $W_q(r) = X(r, q) \cdot W(q) \cup \overline{X}(r, q)$, and, since $(X(r, q), \overline{X}(r, q))$ is a partition of a prefix set and $W(q)$ is prefix then $W_q(r)$ is prefix (see Proposition 4.1 in [BP85]).

It remains to prove that $\mathcal{A}$ and $\mathcal{A}_q$ recognize the same language. Notice that, by construction, each edge (r, u, s) in $\mathcal{A}_q$ corresponds in $\mathcal{A}$ either to the same edge or to the path $\{(r, u_1, q), (q, u_2, s)\}$ where $u_1u_2 = u$. Then, it is easy to verify that, for any word $v \in \Sigma^*$, v is the label of an accepting path in $\mathcal{A}_q$ if and only if v is label of an accepting path in $\mathcal{A}$. $\Box$

We give the following definition.

Definition 4.8 *A generalized automaton is $\mathcal{S}$-*irreducible *if it has not superfluous states.*

5 Irreducible DGA

In the previous section we have defined two ways of reducing the number of states of a given generalized automaton to get an equivalent smaller one: contracting indistinguishable states ($\mathcal{I}$-reductions) or suppressing superfluous states ($\mathcal{S}$-reductions).

We give the following definition.

Definition 5.1 *A DGA is* irreducible *if it is both $\mathcal{I}$-irreducible and $\mathcal{S}$-irreducible.*

We now consider the problem of calculating irreducible DGA that are equivalent to a given DGA. First observe that, if we apply $\mathcal{S}$-reductions to an $\mathcal{I}$-irreducible DGA, this remains $\mathcal{I}$-irreducible: this is because $\mathcal{S}$-reductions preserve, in particular, all sets $L_{p,f}$ where f is a final state. Then, a procedure that makes a given DGA first $\mathcal{I}$-irreducible and then $\mathcal{S}$-irreducible leads surely to an irreducible automaton. The converse holds too, that is $\mathcal{I}$-reductions preserve $\mathcal{S}$-irreducibility of a DGA, since they transform initial (final) state into initial (final) state and states with no self-loops into states with no self-loops again. Therefore, in the rest of the section, we concentrate our attention to find conditions to apply $\mathcal{S}$-reduction to a given DGA in order to "suppress" as many as possible superfluous states to make it $\mathcal{S}$-irreducible.

Notice that, if p and q are both superfluous states of a given DGA $\mathcal{A}$, then p is not necessarily still a superfluous state for the transformed automaton $\mathcal{S}(\mathcal{A}, q)$. In general, the set of superfluous states of a DGA changes when it is reduced by transformation $\mathcal{S}$. We now establish conditions under which two superfluous states p and q can be both suppressed.

Lemma 5.1 *Let $\mathcal{A} = (\Sigma, Q, i, F, E)$ be a DGA and p, q be two superfluous states for $\mathcal{A}$ such that there is no cycle of length two between p and q. Then p and q are superfluous states for $\mathcal{S}(\mathcal{A}, q)$ and $\mathcal{S}(\mathcal{A}, p)$ respectively and $\mathcal{S}(\mathcal{S}(\mathcal{A}, q), p) = \mathcal{S}(\mathcal{S}(\mathcal{A}, p), q)$.*

Proof: Let $\mathcal{A}_q = \mathcal{S}(\mathcal{A}, q) = (Q_q, i, F, E_q)$, $\mathcal{A}_p = \mathcal{S}(\mathcal{A}, p) = (Q_p, i, F, E_p)$. We first prove that p is a superfluous state for $\mathcal{A}_q$. Obviously p is not either an initial or a final state of $\mathcal{A}_q$; we have to show that p has not self-loops in $\mathcal{A}_q$. By contradiction: if edge $(p, u, p) \in E_q$ then, by construction, there exist edges $(p, u_1, q), (q, u_2, p) \in E$ with $u = u_1 u_2$. But these edges constitute a cycle of length two that contradict the hypothesis. The same argument proves that q is a superfluous state for $\mathcal{A}_p$. Therefore $\mathcal{S}(\mathcal{A}_q, p)$ and $\mathcal{S}(\mathcal{A}_p, q)$ are defined. We denote them with $\mathcal{A}_{qp} = (Q_{qp}, i, F, E_{qp})$, $\mathcal{A}_{pq} = (Q_{pq}, i, F, E_{pq})$ respectively.

We now prove that $\mathcal{A}_{qp} = \mathcal{A}_{pq}$. By definition $Q_{qp} = Q - \{q, p\} = Q_{pq}$, then we only have to show that $E_{qp} = E_{pq}$. From the hypothesis we can assume without loss of generality that there are no edges from p to q. We show that $E_{qp} \subseteq E_{pq}$. If edge $(r, u, s) \in E_{qp}$ then there exists a path $[r, u, s]$ in $\mathcal{A}$. There are four cases:

i) $(r, u, s) \in E$;
ii) $(r, u_1, p), (p, u_2, s) \in E$ and $u_1 u_2 = u$;
iii) $(r, v_1, q), (q, v_2, s) \in E$ and $v_1 v_2 = u$;
iv) $(r, w_1, q), (q, w_2, p), (p, w_3, s) \in E$ and $w_1 w_2 w_3 = u$.

If one of the first three cases occurs then it is easy to see that $(r, u, s) \in E_{pq}$. If *iv)* holds then $(r, w_1, q), (q, w_2 w_3, s) \in E_p$ and this implies, by definition, that $(r, w_1 w_2 w_3, s) = (r, u, s) \in E_{pq}$. In a similar way we prove the converse inclusion $E_{pq} \subseteq E_{qp}$. $\square$

Lemma 5.1 allows us to adopt the notation

$$\mathcal{S}(\mathcal{S}(\mathcal{A}, q), p) = \mathcal{S}(\mathcal{S}(\mathcal{A}, p), q) = \mathcal{S}(\mathcal{A}, \{p, q\}).$$

We now want to investigate the conditions under which this notation can be extended to any set $S = \{s_1, s_2, \ldots, s_h\} \subseteq Q$.

We indicate by $\mathcal{A}_i$ the DGA obtained from $\mathcal{A}$ by suppressing in order states $s_1, s_2, \ldots, s_i$ for $i = 1, \ldots, h$. Notice that the transformation

$$\mathcal{S}((\ldots \mathcal{S}(\mathcal{S}(\mathcal{A}, s_1), s_2) \ldots), s_h) \tag{1}$$

can be realized only if, for any $i = 1, \ldots, h-1$, state s_{i+1} is an superfluous state for $\mathcal{A}_i$. We use the notation $\mathcal{S}(\mathcal{A}, \{s_1, \ldots, s_h\}) = \mathcal{S}(\mathcal{A}, S)$ to refer to expression (1).

We recall that *Superf*(Q) denotes the set of all superfluous states of $\mathcal{A}$. The following lemma characterizes those sets $S \subseteq Q$ for which $\mathcal{S}(\mathcal{A}, S)$ can be calculated.

Lemma 5.2 *Let $\mathcal{A} = (\Sigma, Q, i, F, E)$ be a DGA and $S \subseteq Q$. Then $\mathcal{S}(\mathcal{A}, S)$ can be calculated if and only if $S \subseteq$ Superf(Q) and it induces an acyclic subgraph in $\mathcal{A}$.*

Proof: First, observe that, given automata $\mathcal{A}$ and $\mathcal{S}(\mathcal{A}, q)$, there is a cycle containing two states $r, s \neq q$ in $\mathcal{A}$ if and only if there is a cycle containing r and s in $\mathcal{S}(\mathcal{A}, q)$. Then we prove by induction on the cardinality of S, that if $S \subseteq$ *Superf*(Q) and it induces an acyclic subgraph in $\mathcal{A}$ then $\mathcal{S}(\mathcal{A}, S)$ is defined. As base of the induction, we take the case when $|S| = 2$ that is true by Lemma 5.1. Assume that the statement is true for $|S| \leq h - 1$: we show that this implies the case $|S| = h$. Let $S = \{s_1, s_2, \ldots, s_h\}$: since $S \subseteq$ *Superf*(Q) then $\mathcal{S}(\mathcal{A}, s_1)$ is defined. As consequence of the observation at the beginning of the proof, the set $\{s_2, \ldots, s_h\}$ still induces an acyclic subgraph in $\mathcal{S}(\mathcal{A}, s_1)$. Then by inductive hypothesis we have that $\mathcal{S}(\mathcal{S}(\mathcal{A}, s_1), \{s_2, \ldots, s_h\})$ is defined.

Conversely, we suppose that $\mathcal{S}(\mathcal{A}, S)$ is defined and prove that S induces an acyclic subgraph in $\mathcal{A}$. Consider the case $|S| = 2$, say $S = \{p, q\}$. Suppose, by contradiction, that there exists a cycle between p and q in $\mathcal{A}$: then there is a self-loop in q (resp. in p) in $\mathcal{S}(\mathcal{A}, p)$ (resp. $\mathcal{S}(\mathcal{A}, q)$). Therefore the automaton $\mathcal{S}(\mathcal{A}, S)$ cannot be defined. Using this case and applying techniques similar to the ones in the first part, the proof can be completed by induction. □

Lemma 5.1 guaranties that the computation of DGA $\mathcal{S}(\mathcal{A}, S)$ is independent of the order in which the states s_i's are suppressed from $\mathcal{A}$ and justifies the notation $\mathcal{S}(\mathcal{A}, S)$ to refer to expression (1).

Remark 5.1 It is easy to verify that the fact that set S induces an acyclic subgraph in $\mathcal{A}$ has the consequence that the length of labels in $\mathcal{S}(\mathcal{A}, S)$ can increase at most of $|S|$.

We recall that a DGA $\mathcal{A} = (\Sigma, Q, i, F, E)$ is $\mathcal{S}$-irreducible if the set of its superfluous states *Superf*$(Q) = \emptyset$. We refer to the subgraph induced by *Superf*(Q) as $\mathcal{A}_{Superf(Q)}$.

As immediate consequence of Lemma 5.2 we get the following theorem that gives necessary and sufficient conditions on the set S in order $\mathcal{S}(\mathcal{A}, S)$ to be irreducible.

Theorem 5.1 *Let $\mathcal{A} = (\Sigma, Q, i, F, E)$ be a DGA and $S \subseteq$ Superf(Q). The DGA $\mathcal{S}(\mathcal{A}, S)$ is $\mathcal{S}$-irreducible if and only if S induces a maximal acyclic subgraph in $\mathcal{A}_{Superf(Q)}$.*

Remark 5.2 Given a graph G with set of vertices V, we can find different subsets of V that induce a maximal acyclic subgraph in G. In particular, we can find some of such different subsets that have also different size. Then, if we want to find the "minimal" $\mathcal{S}$-reduced automaton, when applying Theorem 5.1, we have to choose set S as a maximum size set among all possible sets that induce an acyclic subgraph in $\mathcal{A}_{Superf(Q)}$.

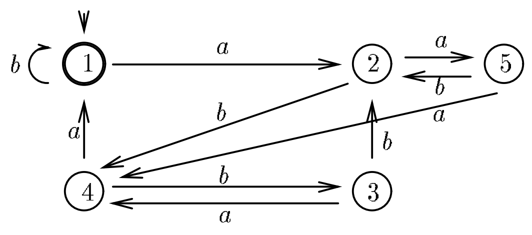
We conclude this section by remarking that, given an automaton $\mathcal{A}$, the problem of finding a maximum set of states S as required by Theorem 5.1 is NP-complete. In fact, it is strictly related to the following NP-complete problem (see [LY80]): "Given a direct graph, find the minimum number of states to be deleted so that resulting subgraph is acyclic".

6 Minimal DGA

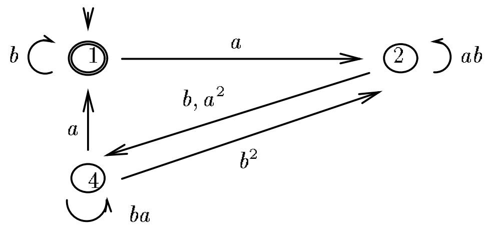
In this section we consider the minimization problem: given a DGA $\mathcal{A}$, find a minimal DGA equivalent to $\mathcal{A}$. Since a minimal DGA must be irreducible, we surely should apply to $\mathcal{A}$ both $\mathcal{I}$-reductions and $\mathcal{S}$-reductions.

We first remark that irreducibility does not necessarely imply minimality. And this is true even if, when applying $\mathcal{S}$-reductions, we choose a set with the maximum number of states among all sets of superfluous states that induce a maximal acyclic subgraph in $\mathcal{A}$ (see Remark 5.2). The reason for this derives from the fact that the procedure consisting of taking the DGA and appling an $\mathcal{S}$-reduction followed by an $\mathcal{I}$-reduction is not equivalent to the procedure that inverts these two operations. This is evident from the following example.

Example 6.1 Consider the DGA $\mathcal{A}$ over the alphabet $\Sigma = \{a, b\}$ given below.

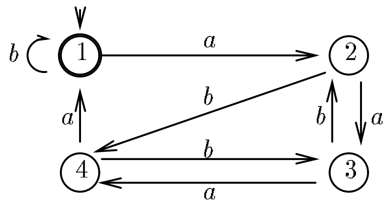


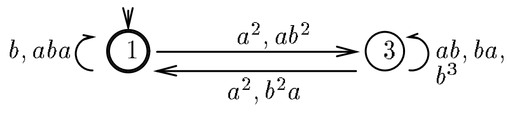
We make $\mathcal{A}$ first $\mathcal{S}$-irreducible and then $\mathcal{I}$-irreducible. Observe that the subset $S = \{3, 5\}$ induces in $\mathcal{A}$ a maximal acyclic subgraph so that $\mathcal{A}_1 = \mathcal{S}(\mathcal{A}, S)$ is an $\mathcal{S}$-irreducible DGA. $\mathcal{A}_1$ is represented below.



Since in $\mathcal{A}_1$ there are no indistinguishable states we conclude that $\mathcal{I}(\mathcal{A}_1) = \mathcal{A}_1$ is both an $\mathcal{S}$-irreducible and an $\mathcal{I}$-irreducible DGA equivalent to $\mathcal{A}$.

Now we invert the procedure and we make $\mathcal{A}$ first $\mathcal{I}$-irreducible and then $\mathcal{S}$-irreducible. Observe, that, in $\mathcal{A}$, states 3 and 5 are indistinguishable: thus we can contract them in a unique state, that we call again 3. We obtain the $\mathcal{I}$-irreducible automaton $\mathcal{A}_2 = \mathcal{I}(\mathcal{A})$ given below on the left. Then the set of states $S' = \{2, 4\}$ induces a maximal acyclic graph in $\mathcal{A}_2$ so that $\mathcal{A}'_2 = \mathcal{S}(\mathcal{A}_2, S')$ is an $\mathcal{S}$-irreducible DGA. $\mathcal{A}'_2$ is represented by the graph given below on the right.





Notice that the two resulting DGA, $\mathcal{A}_1$ and $\mathcal{A}'_2$, are both $\mathcal{S}$- and $\mathcal{I}$-irreducible but they have a different number of states.

We now state the following theorem.

Theorem 6.1 *Let L be a regular language and let $\mathcal{N}$ be a minimal DGA recognizing L. If $\mathcal{M}$ is the equivalent minimal conventional deterministic automaton then there exists a set S of states of $\mathcal{M}$ such that $\mathcal{N} = \mathcal{S}(\mathcal{M}, S)$.*

Proof: Let $\mathcal{N} = (Q_\mathcal{N}, i_\mathcal{N}, F_\mathcal{N}, E_\mathcal{N})$ and $\mathcal{M} = (Q_\mathcal{M}, i_\mathcal{M}, F_\mathcal{M}, E_\mathcal{M})$. The proof consists in defining a mapping $\varphi : Q_\mathcal{N} \to Q_\mathcal{M}$ such that all the states in $Q_\mathcal{M}$ that have no counterimage by φ in $Q_\mathcal{N}$ are superfluous states for $\mathcal{M}$. More precisely, the proof is given in three steps:

1. Define mapping $\varphi : Q_\mathcal{N} \to Q_\mathcal{M}$ and show that it is a well defined function.

2. Define set $S \subset Q_M$ by means of mapping φ and show that it satisfy conditions of Lemma 5.2.

3. Prove that the two automata $\mathcal{N}$ and $\mathcal{S}(\mathcal{M}, S)$ coincide.

To accomplish step 1. we define mapping φ as follows: $\varphi(i_{\mathcal{N}}) = i_{\mathcal{M}}$ and $\varphi(q_{\mathcal{N}}) = q_{\mathcal{M}}$ if and only if there exists a word $w \in L_{i_{\mathcal{N}}q_{\mathcal{N}}} \cap L_{i_{\mathcal{M}}q_{\mathcal{M}}}$ where $q_{\mathcal{N}} \in Q_{\mathcal{N}}$ and $q_{\mathcal{M}} \in Q_{\mathcal{M}}$.

We now show that φ is a well-defined function over $Q_{\mathcal{N}}$. Since automata $\mathcal{N}$ and $\mathcal{M}$ are equivalent then, given $q_{\mathcal{N}} \in Q_{\mathcal{N}}$ there exists $q_{\mathcal{M}} \in Q_{\mathcal{M}}$ such that $\varphi(q_{\mathcal{N}}) = q_{\mathcal{M}}$. Such state $q_{\mathcal{M}}$ is unique. In fact, suppose that there exists also $p_{\mathcal{M}} \in Q_{\mathcal{M}}$ such that $\varphi(q_{\mathcal{N}}) = p_{\mathcal{M}}$: then, by the definition of φ, there exist two words u, v such that paths $[i_{\mathcal{N}}, u, q_{\mathcal{N}}]$ and $[i_{\mathcal{N}}, v, q_{\mathcal{N}}]$ are in $\mathcal{N}$ and paths $[i_{\mathcal{M}}, u, q_{\mathcal{M}}]$ and $[i_{\mathcal{M}}, v, p_{\mathcal{M}}]$ are in $\mathcal{M}$. But the equivalence of $\mathcal{N}$ and $\mathcal{M}$ implies that $q_{\mathcal{M}}$ and $p_{\mathcal{M}}$ are indistinguishable and this contradicts the hypothesis that $\mathcal{M}$ is minimal.

We now turn to step 2. and define set $S = Q_{\mathcal{M}} - \varphi(Q_{\mathcal{N}})$. Notice that S contains all states of $\mathcal{M}$ that do not correspond to any state of $\mathcal{N}$. We now prove that $\mathcal{M}' = \mathcal{S}(\mathcal{M}, S)$ is defined, that is S satisfies the conditions of Lemma 5.2, and that $\mathcal{N} = \mathcal{M}'$. Let us first observe that $F_{\mathcal{M}} \subseteq \varphi(Q_{\mathcal{N}})$: therefore the set S does not contains both the initial state $i_{\mathcal{M}}$ and the set of final states of $\mathcal{M}$. We now show that S induces an acyclic subgraph in $\mathcal{M}$.

Suppose that in $\mathcal{M}$ there is a cycle $[s, u, s]$ whose states are all in S. Let v, w be two words such that the paths $[i_{\mathcal{M}}, v, s], [s, w, f_{\mathcal{M}}]$ are in $\mathcal{M}$ where $f_{\mathcal{M}} \in F_{\mathcal{M}}$. The words $vu^n w \in L$ for all integers $n \geq 0$: therefore in $\mathcal{N}$ for any n there exists a path $[i_{\mathcal{N}}, vu^n w, f_{\mathcal{N}}]$, where $f_{\mathcal{N}} \in F_{\mathcal{N}}$. Since $Q_{\mathcal{N}}$ is a finite set, there exist infinite values of n for which path $[i_{\mathcal{N}}, uv^n w, f_{\mathcal{N}}]$ in $\mathcal{N}$ contains a cycle and it can be split as paths $[i_{\mathcal{N}}, x, r], [r, y, r], [r, y, r], \ldots, [r, y, r], [r, z, f_{\mathcal{N}}]$ that is $vu^n w = xy^k z$ for a suitable value of k.

Therefore we can choose k, h in a way that $k \geq h$, $|xy^h| \geq |v|$ and$|y^{k-h}z| \geq |w|$ while $xy^k z = vu^n w$. We observe that $|xy^h| \leq |vu^n|$ otherwise $|xy^k z| = |xy^h| + |y^{k-h}z| > |vu^n w|$ contradicting the hypothesis. Since the word xy^h is a prefix of $vu^n w$ (that belongs to L) then there exists a state s' in $\mathcal{M}$ such that the path $[i_{\mathcal{M}}, xy^h, s']$ is in $\mathcal{M}$. ¿From the definition of φ we have: $s' = \varphi(r)$ that is $s' \in \varphi(Q_{\mathcal{N}})$ (and therefore s' does not belongs to S). Moreover, notice that since $|v| \leq |xy^h| \leq |vu^n|$, then the state s' is a state in the cycle $[s, u, s]$ in $\mathcal{M}$. But this implies that $s' \in S$ contradicting what we stated before.

It remains to prove step 3. that is to show that $\mathcal{N} = \mathcal{M}'$. We already know that $\mathcal{N}$ and $\mathcal{M}'$ are equivalent, $i_{\mathcal{M}'} = i_{\mathcal{M}} = \varphi(i_{\mathcal{N}})$ and that map φ is defined onto the set of states $Q_{\mathcal{M}'}$ of $\mathcal{M}'$ that is $Q_{\mathcal{M}'} = Q_{\mathcal{M}} - S = \varphi(Q_{\mathcal{N}})$. Mapping φ is actually a bijection from $Q_{\mathcal{N}}$ in $Q_{\mathcal{M}'}$. In fact if there exist two states $p, q \in Q_{\mathcal{N}}$ such that $\varphi(p) = \varphi(q)$ then $|Q_{\mathcal{M}'}| < |Q_{\mathcal{N}}|$ and this contradicts the hypothesis of $\mathcal{N}$ minimal DGA. $\square$

Using Theorem 6.1 together with Remark 5.2, we get a procedure to compute the size n of a minimal DGA $\mathcal{N}$ recognizing a given language L. This is described by the following algorithm.

Algorithm:

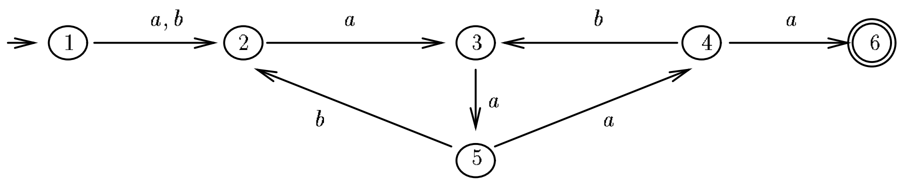
1. Calculate the minimal conventional deterministic automaton $\mathcal{M}$ for L.
2. Calculate a maximal set of states S that induces a maximal acyclic subgraph in $\mathcal{M}$.
3. Then, $n = |Q_{\mathcal{M}}| - |S|$.

This algorithm solves, in the deterministic setting, the corresponding problem studied by Hashiguchi in [H91]. In fact, let m be the number of states of the minimal automaton $\mathcal{M}$ and let $\mathcal{N}$ be the minimal DGA calculated by the above algorithm. Then, the maximal length of the labels in the edges of $\mathcal{N}$ (called $D(\mathcal{N})$ in [H91]) is at most equal to the number of states suppressed in $\mathcal{M}$ plus 1 (see Remark 5.1), that is $D(\mathcal{N}) \leq m$.

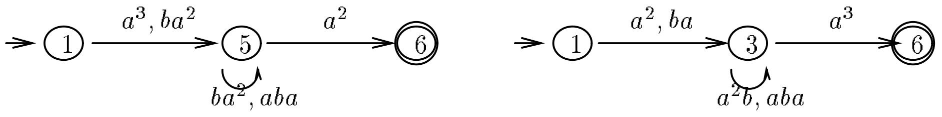
Differently from the case of conventional deterministic automata, the following theorem holds.

Theorem 6.2 *Given a language L, there is not a unique minimal DGA that recognizes L.*

Proof: The proof is given by the following example. Consider the minimal deterministic automaton $\mathcal{A}$ represented below.



In $\mathcal{A}$ there are two maximal sets of superfluous states, $S_1 = \{2, 3, 4\}$ and $S_2 = \{2, 4, 5\}$. By suppressing S_1 in $\mathcal{A}$ we obtain the minimal equivalent DGA given below on the left. In the same way, by suppressing S_2 in $\mathcal{A}$ we obtain another minimal equivalent DGA that is given below on the right.



□

As immediate consequence of Theorems 5.1 and 6.1, we obtain a procedure to find all minimal DGA equivalent to a given deterministic automaton $\mathcal{A}$. We take the minimal (conventional) deterministic automaton $\mathcal{M}$ equivalent to $\mathcal{A}$ and compute all maximal sets among all superfluous sets that induce maximal acyclic subgraphs in

$\mathcal{M}_{Superf(Q_{\mathcal{M}})}$. All minimal DGA equivalent to $\mathcal{A}$ can be computed by applying an $\mathcal{S}$-reduction to $\mathcal{M}$ with respect to such sets.

We finish the section by remarking that the inverse of the $\mathcal{S}$-reduction (that is breaking edges with "long" labels and create a sequence of edges with "shorter" labels) is easy to define. Given the edge $(p, w_1 w_2 \ldots w_n, q)$ we can insert states $r_1, r_2, \ldots, r_{n-1}$ and edges $(p, w_1, r_1), (r_1, w_2, r_2), \ldots, (r_{n-1}, w_n, q)$. Therefore, to minimize a given DGA $\mathcal{A}$, we apply this inverse operation to $\mathcal{A}$ until we obtain a conventional deterministic automaton $\mathcal{A}'$; then we minimize $\mathcal{A}'$. Finally we apply Theorem 6.1.

7 Final discussions and open problems

In this paper we have defined the model of deterministic generalized automaton and studied the problem of its minimization (with respect to the number of states). In particular we have given a procedure that effectively constructs a minimal DGA starting from the minimal equivalent (conventional) deterministic automaton. This gives a solution, in the deterministic setting, for the corresponding problem studied by Hashiguchi in [H91].

The size of a minimal representation of a language in a given model is related to the comparisons of different models according to their intrinsic succinctness. The primary terms of comparisons are always the deterministic and the non-deterministic versions. In the case of conventional automata, it is well known that there is an exponential gap in the complexity of representation between the non-deterministic and deterministic versions. In fact, consider the languages $L_n = (a+b)^* a (a+b)^{n-1}$, for any integer n: the minimal deterministic automaton for L_n has exactly 2^n states while the corresponding non-deterministic one has $n+1$ states. We notice that such discrepancy in succinctness between non-deterministic and deterministic versions still holds inside the model of GA. In fact, the minimal (conventional) deterministic automaton for L_n has exactly 2^{n-1} final states (therefore not superfluous) that will be necessarily also in any minimal DGA. On the other hand the minimal (non-deterministic) GA has only two states for any n. This example suggests that, if the minimal conventional deterministic automaton has "too many" final states, then the corresponding GA cannot be reduced too much.

Consider now a slight modification of the language L_n above in order to get a "similar" language with one only final state. Let $L'_n = (a+b)^* a (a+b)^{n-1} c$: the minimal (conventional) non-deterministic automaton for L'_n has $n+2$ states while the corresponding deterministic one has $2^n + 1$ states among which there is only one final state. Then, this time, when we define the minimal DGA we can suppress many more states and in fact, the minimal DGA for L'_n has $n+2$ states!

A further direction for this work is then to investigate about the succinctness in the case of automata with only one final state. This is related with the decomposition of a regular language in unitary components ([E74]).

As final observation, notice that the $\mathcal{S}$-reductions can be defined as well for non-deterministic GA. They still give equivalent GA but in general we do not know whether there exists a procedure that compute a minimal non-deterministic (generalized) automaton.

We conclude the paper by mentioning another measure of "descriptional complexity" of a minimal DGA with respect to the equivalent minimal DFA: the sum of the lenght of all labels of the edges that we call *label-size* of the automaton.

In fact, while the label-size of a conventional automaton is linearly related to the number of states, in the case of generalized automata, it is probabily the most effective measure of the size.

Let $\mathcal{A} = \{\Sigma, Q, I, F, E\}$ be a minimal conventional automaton and let us denote by $\ell(\mathcal{A})$ the label-size of $\mathcal{A}$, $s = \Sigma$, $n = |Q|$, $m = |E|$. Suppose that, in order to get an equivalent minimal DGA $\mathcal{A}_N$, we have to suppress N states. By construction, $\mathcal{A}_N$ is a deterministic generalized automaton with $n - N$ states on an alphabet with s symbols, having labels of lenght at most $N + 1$ (see Remark 5.1); then for any state in $\mathcal{A}_N$ there is at most one edge for any word in Σ^{N+1}. If M is the number of edges in $\mathcal{A}_N$, it holds:

$$M \leq (n - N)s^{N+1}$$

and then

$$\ell(\mathcal{A}_N) \leq (N + 1)(n - N)s^{N+1}.$$

It is not difficult to find automata for which such bounds are reached. For example consider the minimal automaton recognizing the language of all words over Σ whose length is a multiple of three. If Σ has two letters, then the label-size of such automaton is equal to 6 while the label-size of the corresponding minimal DGA is equal to 24.

Let us denote *total-size* of an automaton the sum of number of states, edges and label length. It would be interesting to characterize languages (automata) for which minimal DGA are more concise with respect to the total-size than the equivalent minimal conventional automata or to find a procedure to minimize a DGA with respect to this total-size.

Acknoledgements

We are deeply grateful to Antonio Restivo for several discussions that inspired this paper and for his helpful comments and hearty encouragements. We also thank Pino Italiano and Sergio Salemi for some useful suggestions.

References

[AHU] A. Aho, J. E. Hopcroft and J. D. Ullman, *The Design and the Analysis of Computer Algorithm* (Addison-Wesley, Reading, MA 1974).

[BP85] J. Berstel and D. Perrin, *Theory of Codes* (Academic Press, 1985).

[C86] M.Chrobak, Finite automata and unary languages, *Theoret. Comp. Science* **47** (1986) 149-158.

[E74] S. Eilenberg, *Automata, Languages and Machines, Vol. A* (Academic Press, 1974).

[GH94] N. Globerman and D. Harel, Complexity results for multi-pebble automata and their logics, in:*Proc. ICALP'94)*, Lecture Notes in Computer Science, Vol. 820 (Springer-Verlag, Berlin, 1994) 73–82.

[H91] K. Hashigushi, Algorithms for determining the smallest number of nonterminals (states) sufficient for generating (accepting) a regular language, in: *Proc. (ICALP'91)*, Lecture Notes in Computer Science, Vol. 510 (Springer-Verlag, Berlin, 1991) 641–648.

[HU79] J. E. Hopcroft and J. D. Ullman, *Introduction to Automata Theory, Languages and Computation* (Addison-Wesley, Reading, MA 1979).

[GM95] D. Giammarresi and R. Montalbano, Deterministic Generalized Automata, in: *Proc. XII Symposium on Theoretical Aspects of Computer Science (STACS '95).* Lecture Notes in Computer Science, Vol. 900 (Springer-Verlag, Berlin, 1995) 325–336.

[JMR] Tao Jang, E. McDowell and B. Ravikumar, The structure and complexity of minimal NFA's over a unary alphabet, Tech.Report, University of Rhode Island (TR-200-90).

[JR91] Tao Jiang and B. Ravikumar, Minimal NFA problems are hard, *SIAM Journal on Computing* **22** (1995) 1117–1141.

[LY80] J. M. Lewis and M. Yannakakis, The node-deletion problem for hereditary properties is NP-complete, *Journal of Comp. and System Science* **20** (1980) 219–230.

[M94] M. Mohri, Minimization of sequential trasducers, in: *Proc. Combinatorial Pattern Matching (CPM'94)*, Lecture Notes in Computer Science, Vol. 807 (Springer-Verlag, Berlin, 1994) 151–163.

[P90] D. Perrin, Finite Automata. in: J. Van Leeuwen, ed., *Handbook of Theoretical Computer Science, Vol.B* (Elsevier, Amsterdam, 1990) 1–57.

[WK94] A. Weber and R. Klemm, Economy of description for single-valued transducers, *Information and Computation* **118**, 2 (1995) 327 - 340.