ITHEA

# International Journal

# INFORMATION THEORIES & APPLICATIONS

2009 Volume 16 Number 1

# International Journal
# INFORMATION THEORIES & APPLICATIONS
**Volume 16 / 2009, Number 1**




**IJ ITA is official publisher of the scientific papers of the members of the ITHEA® International Scientific Society**

IJ ITA welcomes scientific papers connected with any information theory or its application.
IJ ITA rules for preparing the manuscripts are compulsory.
The **rules for the papers** for IJ ITA as well as the **subscription fees** are given on *www.ithea.org* .
**The camera-ready copy of the paper should be received by http://ij.ithea.org.**
Responsibility for papers published in IJ ITA belongs to authors.
General Sponsor of IJ ITA is the **Consortium FOI Bulgaria** (www.foibg.com).


**International Journal "INFORMATION THEORIES & APPLICATIONS" Vol.16, Number 1, 2009**

**Printed in Bulgaria**

Edited by the **Institute of Information Theories and Applications FOI ITHEA®**, Bulgaria,
in collaboration with the V.M.Glushkov Institute of Cybernetics of NAS, Ukraine,
and the Institute of Mathematics and Informatics, BAS, Bulgaria.

Publisher: **ITHEA®**
Sofia, 1000, P.O.B. 775, Bulgaria. www.ithea.org, e-mail: info@foibg.com





**ISSN 1310-0513 (printed)** **ISSN 1313-0463 (online)** **ISSN 1313-0498 (CD/DVD)**

# METHOD AND ALGORITHM FOR MINIMIZATION OF PROBABILISTIC AUTOMATA

**Olga Siedlecka**

***Abstract***: *The theory of probabilistic automata is still evolving discipline of theory of information. As in classical theory of automata, it might be a base for computations, can be exploited in design and verification of circuits and algorithms, in lexical analyzers in compilers computers in future. Minimization of any type of automata gives always saving in resources and time, and is important problem that has been analyzed for almost sixty years. Different types of automata are used for modeling systems or machines with finite number of states.*

*In this article we show few specific type of probabilistic automata, especially the reactive probabilistic finite automata with accepting states (in brief the reactive probabilistic automata), and definitions of languages accepted by it. We present definition of bisimulation relation for automata's states and define relation of indistinguishableness of automata states, on base of which we could effectuate automata minimization. Next we present detailed algorithm reactive probabilistic automata's minimization with determination of its complexity and analyse example solved with help of this algorithm.*

***Keywords***: *minimization algorithm, reactive probabilistic automata, equivalence of states of automata, bisimulation relation.*

***ACM Classification Keywords***: *F. Theory of Computation, F.1 Computation by Abstract Devices, F.1.1 Models of Computation, Automata; F.4 Mathematical logic and formal languages, F.4.3 Formal Languages*

## Introduction

The automata theory is older than any physical computer, after defining abstract machines like Turing machine, scientist searched for equally simple model that resolve problems that doesn't need to write symbols, but only read - they created automata. Like in Turing machine occurred many types of this model: deterministic, nondeterministic, finite, probabilistic, and many others. They could be used for simulation of circuits, algorithms, and every system that have states and read symbols, or react on some action. If we have states as a simulation of real resources, it is welcomed to narrow down their number.

The problem of finite automata minimization appeared in the end of fifties of last century and its main point is to find automata with the minimum number of states accepting the same language as input automata. During last fifty years many algorithms for minimization of finite deterministic automata came into existence, most of which (except Brzozowsky algorithm which is based on derivatives [Brzozowski, 1962]), is based on equivalence of states. One of the most popular minimization algorithms is Hopcroft and Ullman's algorithm with running time $O(|\Sigma|n^2)$ (where $|\Sigma|$ is the number of symbols in the alphabet, $n$ is the number of states) [Hopcroft, 2000]. Another algorithm with the same time complexity, but better memory complexity ($O(|\Sigma|n)$) is Aho-Sethi-Ullman's algorithm [Aho, 2006]. The most efficient deterministic finite automata minimization algorithm is Hopcroft's algorithm [Hopcroft, 1971] with time complexity $O(|\Sigma|n\log n)$.

In the same period of time scientists were searching for another models of computation. They developed probabilistic automata [Rabin, 1963], which are extensions of Markov chains with read symbols [Sokolova, 2004], models of finite automata over infinite words [Thomas, 1990], timed automata [Alur, 1994], hybrid automata [Henzinger, 1998] etc. We can find their ontological review in article: [Kryvyi, 2007]. In last few years new type of

automata is researched by scientist - quantum automata, the probabilistic automata is intermediate way to understand them. It became important to find minimization algorithms for new types of automata. So far minimization of reactive probabilistic automata hasn't been described and it also is a step to minimize quantum automata.

## Probabilistic Automata

A probabilistic automata, just like nondeterministic, has no consistently specified state, in which it will remain after reading symbol. But for probabilistic automata we have probability of reaching a state.

There exist many types of probabilistic automata which differ with properties, applications or probability distributions (continuous or discrete). Hereunder we itemize few of probabilistic automata's types with discrete probability distribution:

- reactive automata,
- generative automata,
- I\O automata,
- Vardi automata,
- alternating model of Hansson,
- Segala automata,
- bundle probabilistic automata,
- Pnueli-Zuck automata and others.

The algorithm showed in article was formulated for the reactive probabilistic automata.

A Markov chain is a transitive system, which has a probability of reaching state, but has no symbols to read, so it is the middle course to the probabilistic automata.

A **Markov chain** is a tuple $PA=(Q, \delta)$, where

- $Q$ is the finite set of states,
- $\delta$ is the transition probability function given by $\delta:Q \rightarrow D(Q)$ (where $D(Q)$ is the set of all discrete probability distribution on the set $Q$) [Sokolova, 2004] .

If $q$ is a member of $Q$ and $\delta(q) = P$ with $P(q') = p > 0$, then we say that Markov chain comes from state $q$ to state $q'$ with probability p (it may be written in many ways: $\delta(q) = P(q)$ or $\delta(q)(q') = p$.

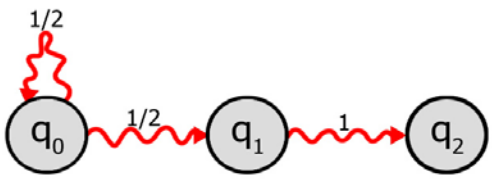


Fig.1. The Markov chain

The example of Markov chain is shown on figure 1, on which we can see probability of going out from state $q_0$.

The reactive probabilistic automata is a type of automata that react on reading symbol by going to another or the same state with given probability (sometimes we can interpret symbol as an action of simulated system ).

A **reactive probabilistic automata** is a triple $PA=(Q, \Sigma, \delta)$, where

- $Q$ is the finite set of states,
- $\Sigma$ is the finite set of input symbols (an alphabet),
- $\delta$ is the transition probability function given by $\delta: Q \times \Sigma \rightarrow D(Q)$ (where $D(Q)$ is the set of all discrete probability distribution on the set $Q$) [Sokolova, 2004] .

An **initial reactive probabilistic automata** with accepting states is a five $PA=(Q, \Sigma, \delta, q_0, F)$, in which we have additionally two elements:

- $q_0$ - a member of $Q$, is the start state,
- $F \subseteq Q$ is the set of final (accepting) states.

After reading given symbol automata is in state of **superposition** of states:

$$p_0q_0 + p_1q_1 + \ldots + p_nq_n,$$

where $p_0 + p_1 + \ldots + p_n=1$. Henceforth we will use shorter name of probabilistic automata within the meaning of initial reactive probabilistic automata with accepting states. An example of this type of automata we show on figure 2.

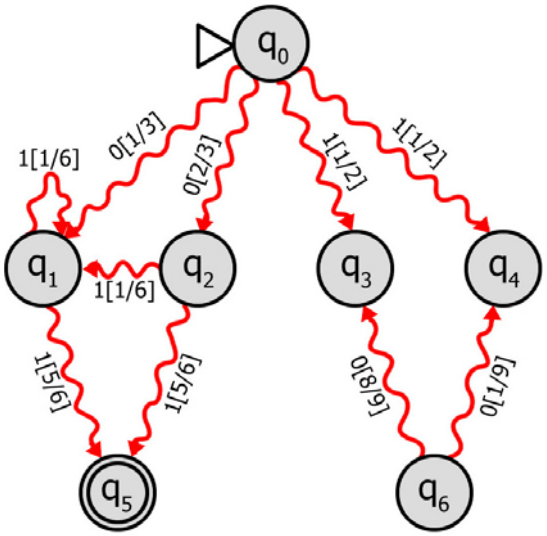


Fig.2. The initial reactive probabilistic automata with accepting states

## Language Accepted by PA

Every type of automata is strictly connected with idea of language accepted by it. In deterministic and nondeterministic finite automata we say, that language is accepted by given automata if and only if for all words from this language, automata after reading of those words is always in its final state. In probabilistic automata we must also consider the probability of acceptance.

$$\delta(q_1, w\sigma) = \sum_{q \in Q} \delta(q_1, w)(q) \cdot \delta(q, \sigma).$$

The probability of going from state $q_1$ to state $q_2$ after reading symbol σ we denote as $\delta(q_1,\sigma)(q_2)=p$. An extended transition probability function, for given word $v$ and prefix $w$, *so* $v = w\,\sigma$, denoted by the same notation, is given by [Cao, 2006]:

The **language accepted** by the probabilistic automata is defined as function:

$$L_{PA}:\Sigma^{*}\rightarrow[0,1],$$

such that [Cao, 2006]:

$$\forall w \in \Sigma^{*}, L_{PA}(w) = \sum_{q\in F} \delta(q_0, w)(q).$$

We say that language $L$ is recognized with bounded error by a probabilistic automata $PA$ with interval $(p_1,p_1)$, if $p_1<p_2$ and

$$p_1 = \sup\{P_w | w \notin L\},\ p_1 = \inf\{P_w | w \in L\}$$ [Golovkins, 2002].

We say that language $L$ is recognized with probability $p$, if the language is recognized with interval $(1\text{-}p,p)$ [Golovkins, 2002].

We say that language $L$ is recognized with probability $1 - \varepsilon$, if for every $\varepsilon > 0$ there exist an automata which recognizes the language with interval $(\varepsilon_1, 1-\varepsilon_2)$, where $\varepsilon_1, \varepsilon_2 \leq \varepsilon$ [Golovkins, 2002].

## Bisimulation and Indistinguishableness

When two automata accept the same language? When they possess equivalent states? Maybe one of them has smaller number of states and accepts the same language? These questions are very important for automata minimization problem. So, if we can find equivalent states, we can minimize some types of automata, but relevant relation is needed. One of the manners is to find first bisimulation relation and on the base of it define indistinguishableness of states.

Firstly we say that two deterministic finite automata are equivalent if they accept the same language, and two states are equivalent, if for every given word, reading this word after going out from these states always will finish for both states in accepting state or finish for both states in nonaccepting state. Automata is called minimal if all its states are nonequivalent.

For two deterministic finite automata: $DFA_1=(S, \Sigma, \delta)$ and $DFA_2=(T, \Sigma, \delta)$ *exists a* strong **bisimulation relation** $R \subseteq S \times T$ if for all $(s,t) \in R$ and for all $\sigma \in \Sigma$:

- if $\delta(s,\sigma)=s'$ then there exists $t' \in T$ such that $\delta(t,\sigma)=t'$ and $(s',t') \in R$ *and*
- if $\delta(t,\sigma)=t'$ then there exists $s' \in S$ such that $\delta(s,\sigma)=s'$ and $(s',t') \in R$ [Kozen, 1997].

The relation of strong bisimulation $R$ has such properties as:

- a diagonal $\Delta_S \subseteq S \times S$ is bisimulation on $(S, \delta)$,
- an inverse relation $R^{-1}$ is bisimulation,
- a sum of bisimulation relations is also bisimulation.

The equivalence relation $R$ is a **congruence** on set of automata states for $(q_1,q_2) \in Q$ and symbols $\sigma \in \Sigma$ if $q_1 R q_2$ and $\delta(q_1,\sigma) R \delta(q_2, \sigma)$ [Gecseg, 1986].

The relation of strong bisimulation $R$ is a congruence [Milner, 1989].

For two initial deterministic finite automata with accepting states DF$A_1$=$(S, \Sigma, \delta, q_0, F_S)$ and DF$A_2$=$(T, \Sigma, \delta, q_0, F_T)$ exists an **indistinguishableness relation** $N \subseteq S \times T$, if for all $(s,t) \in N$ and for all $\sigma \in \Sigma$:

- $(s,t) \in N^0$ if and only if $((s \in F_S \wedge t \in F_T) \vee (s \notin F_S \wedge t \notin F_T))$,
- $(s,t) \in N^k$ if and only if $(s,t) \in N^{k-1}$ and
- if $\delta(s,\sigma)=s'$ then there exists $t' \in T$ such that $\delta(t,\sigma)=t'$ and $(s',t') \in N^{k-1}$and
- if $\delta(t,\sigma)=t'$ then there exists $s' \in S$ such that $\delta(s,\sigma)=s'$ and $(s',t') \in N^{k-1}$.

The relation of indistinguishableness $N$ is a congruence [Milner, 1989].

For Markov chain the bisimulation relation was defined in article [Sokolova, 2004], and its construction is helpful for defining the same relation for reactive probabilistic automata.

Let $R$ be an equivalence relation on the set $S$, and let $P_1, P_2 \in D(S)$ be discrete probability distributions. Then

$$P_1 \equiv_R P_2 \Leftrightarrow \forall C \in S/R: P_1[C] = P_2[C],$$

where $C$ is an equivalence class [Sokolova, 2004].

Let $R$ be an equivalence relation on the set $S$, let $A$ be a set, and $P_1, P_2 \in D(S)$ be discrete probability distributions. Then:

$$P_1 \equiv_{R,A} P_2 \Leftrightarrow \forall C \in S/R, \forall a \in A: P_1[a,C] = P_2[a,C]$$

[Sokolova, 2004].

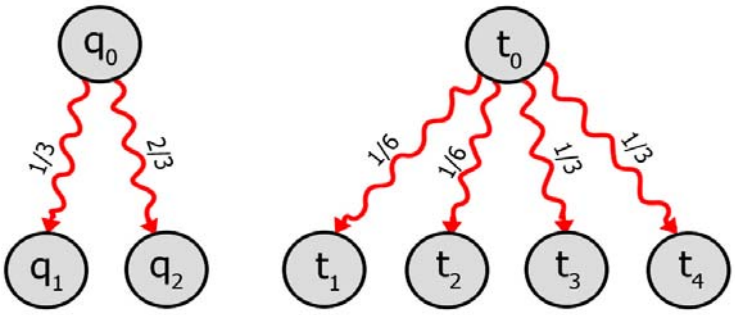


Fig.3. The bisimulation relation on MC

The equivalence relation on the set $Q$ of states of Markov chain $(Q, \delta)$ will be strong **bisimulation relation** $R \subseteq S \times T$ then and only then when for all $(q,t) \in R$:

if $\delta(q)=P_1$ then there exists a distribution $P_2$ with $t \in T$ such that $\delta(t)=P_2$ and $P_1 \equiv_R P_2$ [Sokolova, 2004].

On figure 3 we have five pairs in bisimulation relation: $\{(q_0,t_0), (q_1,t_1), (q_1,t_2), (q_2,t_3), (q_2,t_4)\}$.

On base of bisimulation relation on Markov chain states we can define the same type of bisimulation for reactive probabilistic automata.

Let $PA_1=(S, \Sigma, \delta)$ and $PA_2=(T, \Sigma, \delta)$ be two reactive probabilistic automata. A **bisimulation relation** $R \subseteq S \times T$ exists if for all $(s,t) \in R$ and for all $\sigma \in \Sigma$:

if $\delta(s,\sigma)=P_1$ then there exists a distribution $P_2$ with $t \in T$ such that $\delta(t,\sigma)=P_2$ and $P_1 \equiv_{R,\Sigma} P_2$ [Sokolova, 2004].

States $(s,t) \in R$ we call bisimilar, what is denoted by $s \approx t$.

On figure 4 we have six pairs in bisimulation relation: {$(s_0,t_0)$, $(s_1,t_1)$, $(s_2,t_1)$, $(s_3,t_2)$, $(s_4,t_2)$, $(s_5,t_3)$}.

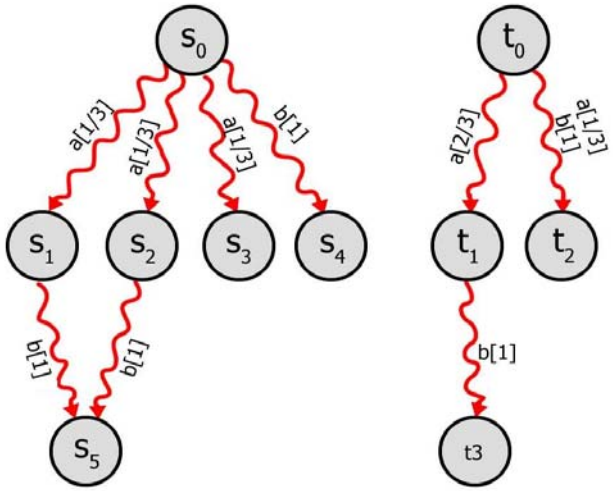


Fig.4. The bisimulation relation on PA

Let $PA_1=(S, \Sigma, \delta, q_0, F_S)$ and $PA_2=(T, \Sigma, \delta, q_0, F_T)$ be two initial reactive probabilistic automata with accepting states. We can define **indistinguishableness relation** $N \subseteq S \times T$, if for all $(s,t) \in N$ and for all $\sigma \in \Sigma$:

$(s,t) \in N^0$ if and only if $((s \in F_S \wedge t \in F_T) \vee (s \notin F_S \wedge t \notin F_T))$,

$(s,t) \in N^k$ if and only if $(s,t) \in N^{k-1}$ and

if $\delta(s,\sigma)=P_1$ then exists the probability distribution $P_2$ with $t \in T$ such that $\delta(t,\sigma)=P_2$ and $P_1 \equiv_{R,\Sigma} P_2$.

For $n=|Q|$, we have

$$N \subseteq N^{n-2} \subseteq N^{n-3} \subseteq \ldots \subseteq N^1 \subseteq N^0.$$

States $s,t$ we call indistinguishable, what is denoted by $s \equiv t$, if there exists indistinguishableness relation $N$, such that $(s,t) \in N$.

On figure 5 we have six pairs in indistinguishableness relation: $\{(s_0,t_0), (s_1,t_1), (s_2,t_1), (s_3,t_2), (s_4,t_2), (s_5,t_3)\}$.

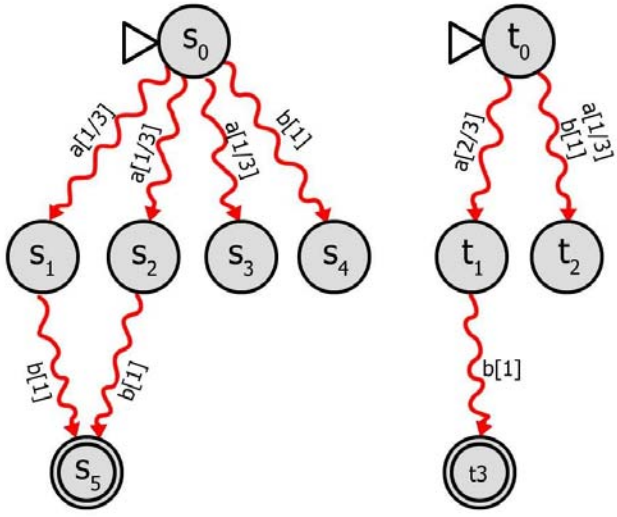


Fig.5. The indistinguishableness relation on PA

The indistinguishableness relation is a congruence in set of states of automata $PA=(Q, \Sigma, \delta, q_0, F)$ - for adequate transition function $\delta$, in analogical way as bisimulation relation, if two results of transition function belong to relation, also states from which we go out belong to relation: for $(q_1,q_2) \in Q$ and symbols $\sigma \in \Sigma$ if $q_1 R q_2$ and $\delta(q_1,\sigma) R \delta(q_2, \sigma)$.

## Minimization of Reactive Probabilistic Automata

The indistinguishableness relation defined in prior section give us possibilty to create minimization methods and algorithms for reactive probabilistic automata.

A probabilistic automata $PA=(Q, \Sigma, \delta, q_0, F)$ recognizing language $L$ with probability $p$ we call minimal, if there doesn't exist automata with smaller number of states recognizing language $L$ with not smaller probability.

In minimal automata there are no two states that could be equivalent in terms of indistinguishableness relation.

A minimization of probabilistic automata parts on two steps:

- elimination of unreachable states (probability to reach those states is 0),
- joining of indistinguishable states (using indistinguishableness relation).

First we show on below code elimination of unreachable states:

Alg.1. Algorithm of elimination of unreachable states:

**INPUT:** $PA=(Q,\Sigma,\delta,q_0,F)$**- reactive probabilistic automata.**
**OUTPUT:** $PA'=(Q',\Sigma,\delta',q_0,F')$ **- reactive probabilistic automata without unreachable states, recognizing the same language as** $PA$**.**

```
1.    FOR ALL {q∈Q} DO
2.      markedStates[q]←0;
3.    END FOR
4.    S.push(q0);
5.    markedStates[q]←1;
6.    pr←0;
7.    WHILE {S≠∅} DO
8.      p←S.first();
9.      S.pop();
10.     FOR ALL {σ∈Σ} DO
11.       FOR ALL {q∈Q} DO
12.         pr← δ(p,σ)(q);
13.         IF {pr≠0 ∧ markedStates[q0]=0} THEN
14.           S.push(q);
15.           markedStates[q]←1;
16.         END IF
17.       END FOR
18.     END FOR
19.   END WHILE
20.   FOR ALL {q∈Q} DO
```

```
21.     IF {markedStates[q]=1} THEN
22.       Q'.push(q);
23.     END IF
24.   END FOR
25.   F'←F∩Q;
26.   FOR ALL {q∈Q} DO
27.     IF {markedStates[q]=1} THEN
28.       FOR ALL {p∈Q} DO
29.         IF {markedStates[p]=1} THEN
30.           FOR ALL {σ∈Σ} DO
31.             δ'(q,σ)(p) ← δ(q,σ)(p);
32.           END FOR
33.         END IF
34.       END FOR
35.     END IF
36.   END FOR
```

In this algorithm *S* is auxiliary stack, on which we put states, which we can reach with non-zero probability going out from the start state $q_0$. The transition probability function $\sigma(p,\sigma)(q)$ gives probability *pr* of reaching state *q*, going out from state *p*, reading symbol $\sigma$. The running time of the algorithm time is bounded by:

$$T(n,|\Sigma|) \le a(7+9n+2|\Sigma|n+2n^2+6|\Sigma|n^2)+c(4+8n+2|\Sigma|n+3n^2+5|\Sigma|n^2),$$

where *a* is time of an assignment and *c* is time of comparison, clearly $O(|\Sigma|n^2)$ is the time complexity of this algorithm.

In the algorithm of joining indistinguishable states we use already defined indistinguishableness relation. In one word, states to be indistinguishable, have to be in the same equivalence class, and must have the same probability distribution for symbols and equivalence classes, which can be reach from this states. Inspired by Hopcroft-Ullman's algorithm [Hopcroft, 2000], first we assume that all pairs of states are indistinguishable, above that, that first element of pair is member of final states' set and second isn't. Next analyzing all pair of states and all symbols we find distinguishable states, until the moment that any change is made. Algorithm analyses probability distributions of reaching state from state.

Alg.2. Algorithm of joining indistinguishable states:

**INPUT:** $PA=(Q,\Sigma,\delta,q_0,F)$ **- reactive probabilistic automata.**
**OUTPUT:** $PA'=(Q',\Sigma,\delta',q_0',F')$ **- minimal reactive probabilistic automata recognizing language** $L_{PA}$.

```
1.    FOR {i←0; i<|Q|; i←i+1} DO
2.      FOR {j←0; j≤i; j←j+1} DO
3.        IF {(qi∈F ∧ qj∉F) ∨ (qi∉F ∧ qj∈F)} THEN
4.          Dqi,qj←1;
5.        ELSE
6.          Dqi,qj←0;
7.        END IF
```

```
8.      END FOR
9.    END FOR
10.   FOR {i←1; i<|Q|; i←i+1} DO
11.     FOR {j←0; j<i; j←j+1} DO
12.       IF {D_qi,qj=0} THEN
13.         FOR ALL {σ∈Σ} DO
14.           E1←0;
15.           E2←0;
16.           N1←0;
17.           N2←0;
18.           FOR ALL {p∈Q} DO
19.             IF {D_qi,p=0} THEN
20.               E1←E1+δ(q_i,σ)(p);
21.             ELSE
22.               N1←N1+δ(q_i,σ)(p);
23.             END IF
24.             IF {D_qj,p=0} THEN
25.               E2←E2+δ(q_j,σ)(p);
26.             ELSE
27.               N2←N2+δ(q_j,σ)(p);
28.             END IF
29.           END FOR
30.           IF {E1≠E2 ∨ N1≠N2} THEN
31.             D_qi,qj←1;
32.             break;
33.           END IF
34.         END FOR
35.       END IF
36.     END FOR
37.   END FOR
38.   Q′←Q;
39.   F′←F;
40.   q0′←q0;
41.   FOR {i←1; i<|Q|; i←i+1} DO
42.     FOR {j←0; j<i; j←j+1} DO
43.       IF {D_qi,qj=0} THEN
44.         Q′←Q′\{q_i,q_j};
45.         Q′←Q′∪{q_ij};
46.         IF{q_i∈F} THEN
47.           F′←F′\{q_i,q_j};
48.           F′←F′∪{q_ij};
49.         END IF
```

```
50.          IF {j=0} THEN
51.            q0←qj;
52.          END IF
53.        END IF
54.      END FOR
55.    END FOR
56.    FOR ALL {q1 xq2 xσ ∈ Q'xQ'xΣ } DO
57.      IF {q1 = q2 ∧ q1∉Q ∧ q1=p1p2 : p1p2∈Q} THEN
58.        δ'(q1,σ)(q2)← δ(p1,σ)(q2) + δ(q2,σ)(p2);
59.      IF {q1∉Q ∧ q1=p1p2 : p1p2∈Q} THEN
60.        δ'(q1,σ)(q2)← δ(p1,σ)(q2);
61.      ELSIF {q2∉Q ∧ q2=p1p2 : p1p2∈Q } THEN
62.        δ'(q1,σ)(q2)← δ(q1,σ)(p1)+ δ(q2,σ)(p2);
63.      ELSE
64.        δ'(q1,σ)(q2)← δ(q1,σ)(q2);
65.      END IF
66.    END FOR
```

Analyzing algorithm in details: on input we have reactive probabilistic automata; on output we get minimal automata that accept the same language as input automata. In lines 1 to 9 we tentatively fill structure *D*, which is lower triangular matrix of all combination of automata's states. In place where one of the states is final and second isn't, we set value 1, because states are distinguishable. In other case we set 0, providing that all other pairs of states are indistinguishable. In lines 10 to 33 is the main part of algorithm, which decides if states are equal or not, comparing probability distributions. First (line 12) we verify if pair of states is indistinguishable $D_{qi,qj}=0$ (otherwise it makes no sense in analyzing them). For every symbol from alphabet Σ we reset value of auxiliary variables *E1, E2, N1, N2*, in which we will sum probabilities of reaching distinguishable states *N* or indistinguishable states *E*. States will be generally recognized as indistinguishable if values of *E1, E2* and *N1, N2* will be respectively equal. If for two analyzed states, for any symbol of alphabet, we get different values of those variable, loop is interrupted (line 32), because states are distinguishable and we go to next iteration. In the last part of algorithm (from line 38) we create output automata, so we replace indistinguishable states by single states, and calculate values for transition probability function (from line 54). Depending, if we analyze reaching state or going out from new state, values of probability will be summed or copied. The running time of the algorithm is bounded by:

$$T(n,|\Sigma|) \le a(5 + 4.5n - 3.5|\Sigma|n + 7.5n^2 + 2|\Sigma|n^2 + 3n^3 + 1.5|\Sigma|n^3) +$$
$$c(2 + 7n - 2.5|\Sigma| n + 7n^2 + |\Sigma| n^2 + 7n^3 + 1.5|\Sigma|n^3),$$

so complexity will be $O(|\Sigma|n^3)$.

Lets analyze steps of both algorithms on example from figure 2. First we reset table *markedStates[qi]*, which size is 7 (automata has 7 states). We push on stack start state. Next we mark with *1* field for this state in table *markedStates[q0]*. We pop from the stack start state and push those, which we can reach from start state reading symbol *0*, with nonzero probability (those will be $q_1$, $q_2$) and for symbol *1*, respectively $q_3$, $q_4$, in every case marking them with *1* in table *markedStates[qi]*. In next iteration we search for states we can reach from states put on the stack. Finally, the only state, which wasn't marked, is $q_6$. In next steps we exclude it from the set of states of automata (figure 6).

The algorithm of joining indistinguishable states in first part fill structure $D_{qi,qj}$ with *1* in those places where one of states is final, and second isn't – for all combinations of other states with state $q_5$. Next we check successively all combinations of states and sum probabilities of going out from this states in variables *E1, E2, N1, N2*, for example for states $q_1$, $q_0$, values for this variables are *E1=0, E2=1, N1=0, N2=0*, so this pair of states is distinguishable and $D_{qi,qj}=1$. Finally structure $D_{qi,qj}$ has value *1* only for pairs: $q_1$, $q_2$ and $q_3$, $q_4$, which will be replaced by new single states $q_{12}$, $q_{34}$. Probabilities for reaching those states will be summed, and for going out from them will be copied (figure 7).

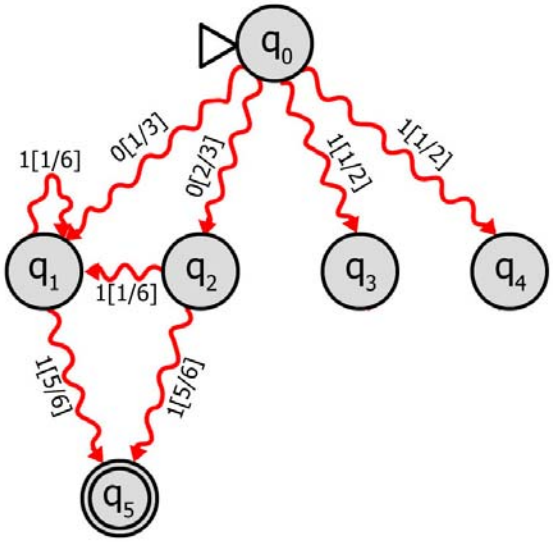


Fig.6. Elimination of unreachable states

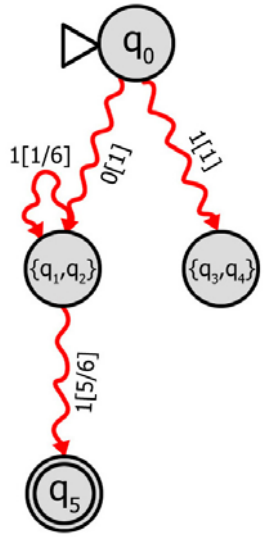


Fig.7. Joining of indistinguishable states

## Conclusion

In this article we defined indistinguishableness relation for reactive probabilistic automata, what give us opportunity to build minimization algorithm, with complexity $O(|\Sigma|n^3)$. Algorithms will terminate, because number of states or symbols in alphabet is always limitation for iterations (and we work on finite sets). Probabilities for accepting words don't change because they are respectively summed or copied.

Minimization of any types of automata is important problem that has its practical application – less number of states – less amount of resources. So, this definition of indistinguishableness relation and minimization algorithm is the base for further work on adequate algorithm for quantum automata.

## Bibliography

[Aho, 2006] A.V. Aho, M.S. Lam, R. Sethi, J.D. Ullman, Compilers: Principles, Techniques, and Tools (2nd Edition). Addison Wesley, 2006.

[Alur, 1994] R. Alur, D.L. Dill, A theory of timed automata. Theoretical Computer Science 126, 2 (1994), pp. 183 - 235.

[Brzozowski, 1962] J. A. Brzozowski, Canonical regular expressions and minimal state graphs for definite events. In Proceedings of the Symposium on Mathematical Theory of Automata (1962), vol. 12 of MRI Symposia Series, Polytechnic Press of the Polytechnic Institute of Brooklyn, pp. 529 - 561.

[Cao, 2006] Y. Cao, L. Xia, M. Ying, Probabilistic automata for computing with words. ArXiv Computer Science e-prints (2006).

[Gecseg, 1986] F. Gecseg, Products of automata. Monographs on Theoretical Computer Science, Springer-Verlag (1986).

[Golovkins, 2002] M. Golovkins, M. Kravtsev, Probabilistic reversible automata and quantum automata. Lecture Notes In Computer Science 2387 (2002), p. 574.

[Henzinger, 1998] T.A. Henzinger, P.W. Kopke, A. Puri P.Varaiya, What s decidable about hybrid automata? Journal of Computer and System Sciences 57 (1998), pp. 94 - 124.

[Hopcroft, 1971] J.E. Hopcroft, An n log n algorithm for minimizing the states in a finite automaton. In The Theory of Machines and Computations, Z. Kohavi, Ed. Academic Press, 1971, pp. 189 - 196.

[Hopcroft, 2000] J.E. Hopcroft, R. Motwani, J.D. Ullman, Introduction to Automata Theory, Languages, and Computation (2nd Edition). Addison Wesley, 2000.

[Kozen, 1997] D.C. Kozen, Automata and Computability. Springer-Verlag New York, Inc.,

Secaucus, NJ, USA, (1997).

[Kryvyi, 2007] S. Kryvyi, L. Matveeva, E. Lukianova, O. Siedlecka, The Ontology-based view on automata theory. In Proceedings of 13-th International Conference KDS-2007 (Knowledge-Dialog-Solution) (Sofia, 2007), ITHEA, Ed., vol. 2, pp. 427 – 436.

[Milner, 1989] C. Milner, Communication and Concurrency. Prentice-Hall, Inc., upper Saddle River, NJ, USA (1989).

[Rabin, 1963] M.O. Rabin, Probabilistic automata. Information and Controle 6 (1963), pp. 230 - 245.

[Sokolova, 2004] A. Sokolova, E. de Vink, Probabilistic automata: System types, parallel composition and comparison. In Validation of Stochastic Systems: A Guide to Current Research (2004), LNCS 2925, pp. 1 – 43.

[Thomas, 1990] W. Thomas, Automata on infinite objects. Handbook of theoretical computer science: formal models and semantics B (1990), pp. 133 – 191.

## Authors' Information

*Siedlecka Olga – Institute of Computer and Information Sciences, Czestochowa University of Technology, ul. Dabrowskiego 73, 42-200 Czestochowa, Poland; e-mail: olga.siedlecka@icis.pcz.pl*

*Major Fields of Scientific Research: Theory of automata, quantum computing, quantum automata*