# Generarea și utilizarea imaginilor la studierea automatelor finite

CIUBOTARU Constantin

## 1. Introducere

Teoria și aplicarea practică a automatelor, în particular a automatelor finite ($AF$), reprezintă unul dintre cele mai vechi și activ studiate domenii ale informaticii. De rând cu aplicațiile tradiționale ce țin de proiectarea compilatoarelor, inteligența artificială și procesarea textelor, în prezent sunt elaborate aplicații moderne pentru procesarea limbajului natural, recunoașterea vorbirii, modelarea și testarea produselor software, probabilități (lanțuri Markov), jocuri video, procesarea imaginilor, teoria codificării etc. La elaborarea acestor aplicații inevitabil apare necesitatea unor transformări echivalente asupra AF [1], cum ar fi:

- eliminarea stărilor inaccesibile și neproductive,
- eliminarea $\varepsilon$-tranzițiilor,
- convertirea automatului nedeterminist în automat determinist echivalent
- minimizarea $AF$ și altele.

Pentru majoritatea acestor transformări este foarte utilă reprezentarea $AF$ sub formă de graf. Dacă reprezentările analitică și tabelară nu implică dificultăți la vizualizare, reprezentarea și vizualizarea $AF$ sub formă de graf este mai dificilă.

Unul și același graf poate fi desenat în mai multe moduri. Unele pot fi mai simple, mai comprehensibile, având un aspect estetic atrăgător, altele - mai greu de sesizat, cu o structurare nereușită. Pentru automatul finit care recunoaște limbajul $L = \{0,1\}^*\{00,11\}\{0,1\}^*$ putem desena mai multe reprezentări grafice. De exemplu, graful din Figura 1($a$) conține mai multe intersecții ale muchiilor, acestea incomodează vizualizarea și urmărirea procesului de recunoaștere. Astfel de reprezentări datorită intersecțiilor de muchii se vor numi reprezentări "spaghetti".

Graful din Figura 1($b$) reprezintă acelasi automat, dar, spre deosebire de primul, este mai lesne de vizualizat și ilustrează evident procesul de

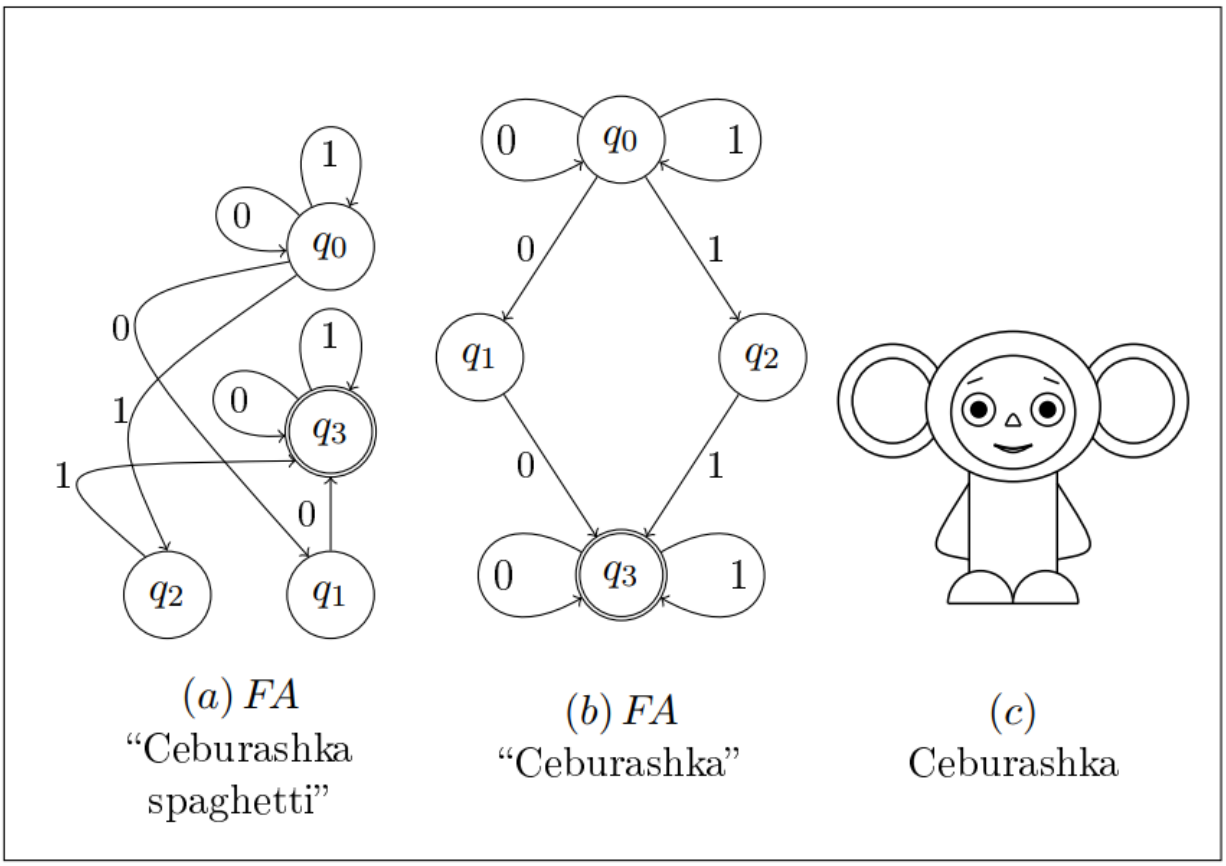



**Figure 1.** Reprezentarea grafică a AF "Cheburashka".

recunoaștere a șirurilor limbajului $L$. Am numit acest automat "Cheburashka" prin asociere cu eroul cunoscutului film cu desene animate [1], Figura 1$(c)$. Această comparație ne ajută și la memorizarea acestui automat.

Având mai multe reprezentări ale aceluiași graf este destul de ușor să alegem varianta cea mai potrivită, mai ales având la dispoziție un șir de criterii de apreciere, de obicei de natură estetică. De exemplu, graful trebuie să se integreze într-un spațiu determinat, limitat, să conțină cât mai puține intersecții ale muchiilor, să evite curburile ascuțite, să respecte proporțiile referitor la lungimea muchiilor și valorile unghiurilor de incidență, să favorizeze elementele de simetrie, de concentrare a nodurilor, să utilizeze forme adecvate pentru noduri, să respecte orientarea fluxului informativ (de sus în jos sau de la stânga la dreapta). Pentru automatele finite fluxul general al informației întotdeauna va fi orientat din starea inițială spre stările finale.

[1] https://en.wikipedia.org/wiki/Cheburashka

Este destul de dificil de transmis aceste criterii calculatorului. Aici intervine nu numai problema formalizării criteriilor, dar și faptul că unele sunt contradictorii. Inevitabil apar situații de compromis.

## 2. Soluții propuse

Pornind de la definiția $AF$ (reprezentarea analitică) în mod automat se generează reprezentarea $AF$ sub formă de graf $\Gamma = (Q, E, F)$, unde $Q$ - mulțimea nodurilor (stările automatului), $E$ - mulțimea de arce $(q_i, a, q_j)$, iar $F$ - mulțimea nodurilor-stări finale. Pentru a obține reprezentarea grafică s-a elaborat un compilator care generează pentru acest graf un program LaTeX/ Ti$k$Z [2, 3, 4, 5].

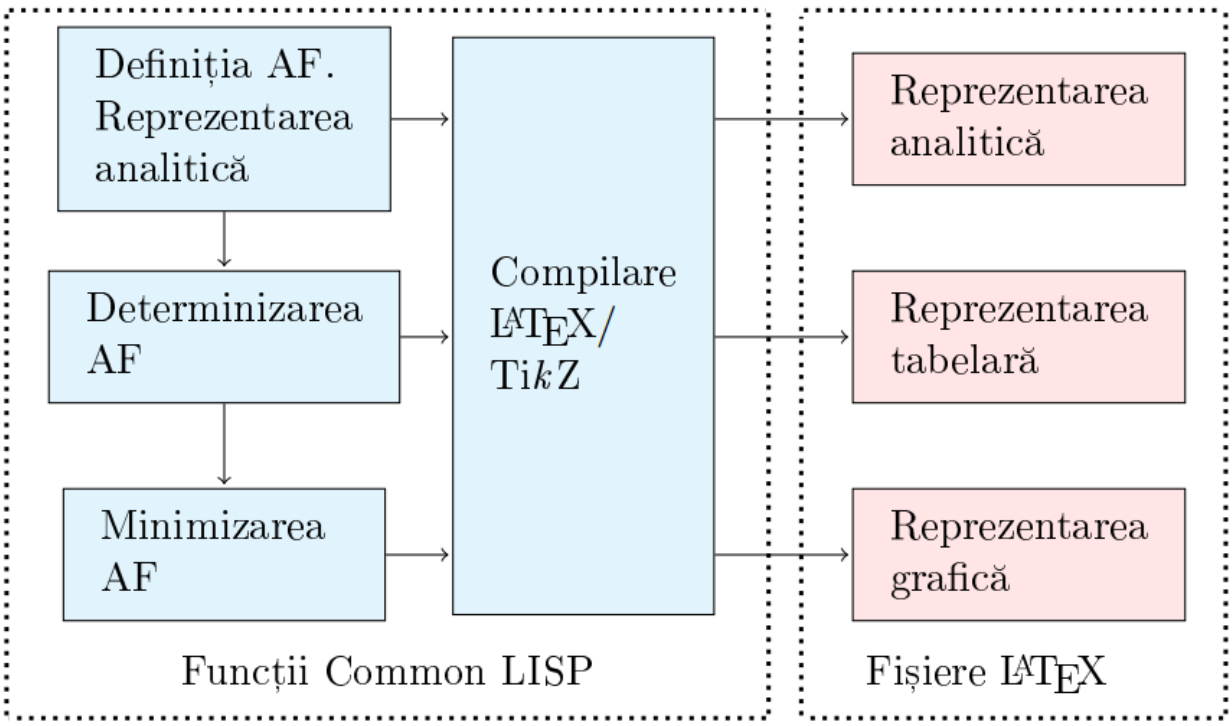



**Figure 2.** Schema aplicațiilor propuse.

Compilatorul generează în mod aleator și distribuie uniform coordonatele nodurilor. Se are în vedere respectarea unei distanțe minime între oricare două noduri, specificată în prealabil în dependență de numărul de noduri.

În Figura 2 este prezentată schema aplicațiilor elaborate și interacțiunea lor.

Generarea fișierelor LaTeX pentru reprezentările analitice și tabelare se bazează pe utilizarea pachetelor `\tabto` și `\tabular` și este relativ simplă.

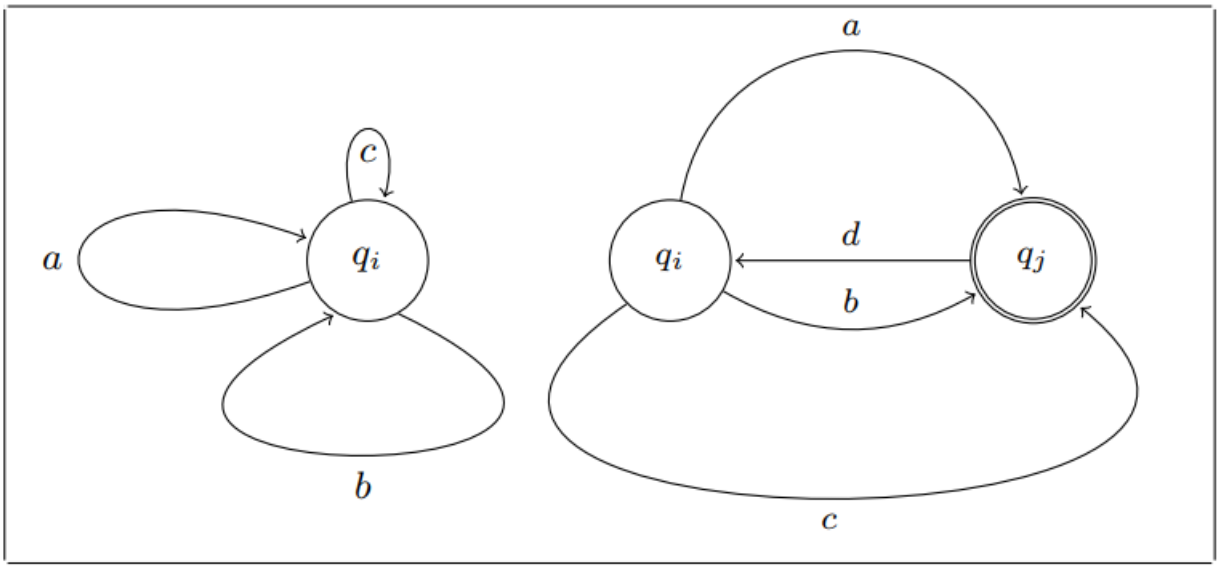



**Figure 3.** Vizualizarea fragmentelor generate.

```
\node[state](q_i)at (3.5,3.5){\scriptsize{$q_{i}$}};
\node[state,accepting](q_j)at(6.5,3.5){\scriptsize{$q_{j}$}};
```

a) Șabloane pentru noduri.

```
(q_i)edge[to path={(\tikztostart.-60).. controls
    (4.5,1.5)and (-2.6,1.5) .. node[below]
    {\footnotesize $b$} (\tikztotarget.-120)}](q_i)
(q_i)edge[out=200,in=160,looseness=20]node[left]
    {\footnotesize $a$} (q_i)
(q_i)edge[loop above]node[below]{\footnotesize $c$} (q_i);
```

b) Șabloane pentru bucle.

```
(q_i) edge [out=80,in=100,looseness=1.5] node [above]
 {\scriptsize $a$} (q_j)(q_i) edge [bend right=30] node
 [above]{\scriptsize $b$} (q_j)
(q_i) edge [to path={(\tikztostart.-135).. controls (0.0,1.0)
  and (9.5,1.0) .. node [below]{\scriptsize $c$}
 (\tikztotarget.-45)}](q_j)
(q_j) edge node [above]{\scriptsize $d$} (q_i);
```

c) Șabloane pentru arce.

**Figure 4.** Șabloane generate.

Anumite dificultăți apar la generarea reprezentărilor grafice. Acestea se referă la modul de amplasare a nodurilor, desenarea buclelor, arcelor, tendința de a obține un desen estetic, compehensibil. Automatizarea totală a acestui proces practic este imposibilă, deoarece criteriile expuse sunt greu de formalizat, iar uneori sunt chiar contradictorii [6]. Soluțiile propuse în lucrare vin să ajute utilizatorul la construirea unei structuri grafice acceptabile.

Compilatorul elaborat exploatează, de rând cu posibilitățile sistemului LaTeX, pachetul `\tikz` și librăriile lui, în mod special librăriile `automata`, `positioning, arrows`. De menționat că există și posibilitatea de a evidenția stările finale și starea inițială. Avem la dispoziție și un bogat arsenal pentru desenarea buclelor/arcelor.

În Figura 3 sunt inserate câteva variante posibile. Pentru fiecare arc (buclă) se generează șabloane funcționale (Figura 4), unul fiind activ, celelalte fiind comentate și oferite utilizatorului în caz că dorește să intervină manual.

## 3. Exemplu

Drept exemplu a fost selectat un $AF$ pentru care în mod automat s-au gnerat modelele echivalente determinist și minimizat. Pentru toate aceste construcții s-au generat fișiere LaTeX pentru reprezentările tabelare și grafice. Toate acestea sunt inserate în Figurile 5,6.

## 4. Concluzii

Aplicațiile propuse în lucrare sunt utilizate în procesul de instruire la studierea automatelor finite. Desigur, se mai cere intervenția manuală, dar versiunea funcțională LaTeX, obținută în rezultatul compilării, este foarte utilă. În perspectivă se prevede implicarea schemei cadru Sugiyama [7, 8], care în linii mari minimizează numărul de intersecții ale arcelor.

**EXEMPLU**

**Definiția analitică**

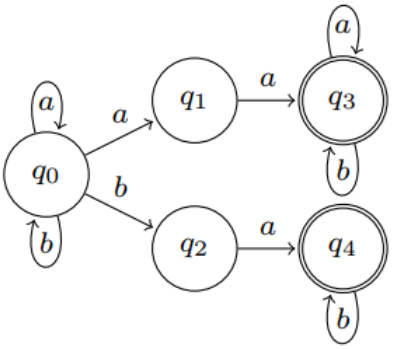
$AF=(Q,\Sigma,\delta,q_0,F)$.
$Q=\{q_0,q_1,q_2,q_3,q_4\}$,
$\Sigma=\{a,b\}$,
$F=\{q_3,q_4\}$
$\delta(q_0,a)=\{q_0,q_1\}$,
$\delta(q_0,b)=\{q_0,q_2\}$,
$\delta(q_1,a)=\{q_3\}$,
$\delta(q_2,a)=\{q_4\}$,
$\delta(q_3,a)=\{q_3\}$,
$\delta(q_3,b)=\{q_3\}$,
$\delta(q_4,b)=\{q_4\}$

**Reprezentarea tabelară**

$AF=(Q,\Sigma,\delta,q_0,F)$.
$Q=\{q_0,q_1,q_2,q_3,q_4\}$,
$\Sigma=\{a,b\}$,
$F=\{q_3,q_4\}$

| $q$ | $\delta(q,a)$ | $\delta(q,b)$ |
|---|---|---|
| $q_0$ | $\{q_0,q_1\}$ | $\{q_0,q_2\}$ |
| $q_1$ | $\{q_3\}$ | $\{\}$ |
| $q_2$ | $\{q_4\}$ | $\{\}$ |
| $q_3$ | $\{q_3\}$ | $\{q_3\}$ |
| $q_4$ | $\{\}$ | $\{q_4\}$ |

**Reprezentarea grafică**




**AF determinist, stări generalizate**

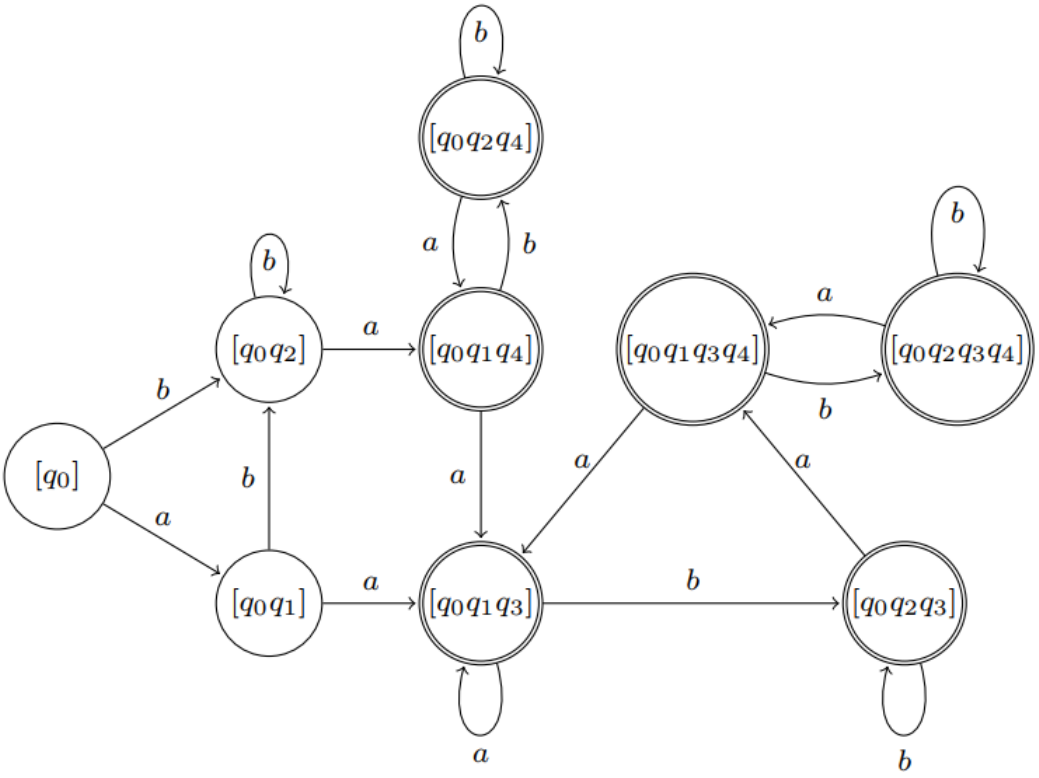



**Figure 5.** Exemplu (pagina 1).

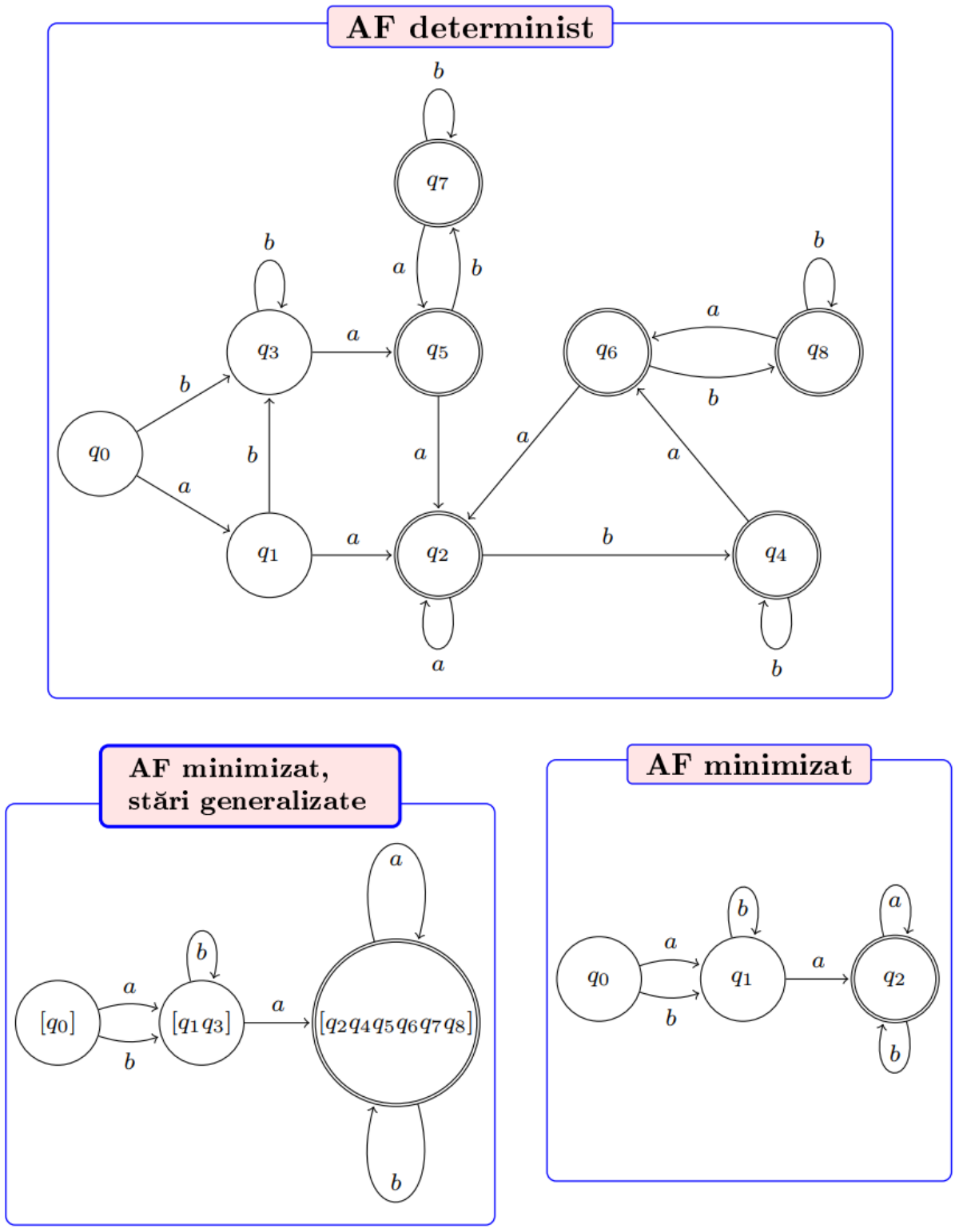



**Figure 6.** Exemplu (pagina 2)

# References

[1] J.E. Hopcroft, R. Motwani, and J.D. Ullman. *Introduction to Automata Theory, Languages, and Computation*. Addison Wesley, 1979, 427 p.

[2] Michel Goossens, Frank Mittelbach, and Alexander Samarin. *The LaTeX Companion*. Addison-Wesley, Reading, Massachusetts, 1993.

[3] Stefan Kottwitz. *LaTeX Cookbook*. Packt Publishing, 2015, 387p.

[4] *The TikZ and PGF Packages. Manual for version 3.1.6*. Institut für Theoretische Informatik Universität zu Lübeck, 2020, 1320p.

[5] Till Tantau. *Graph Drawing in TikZ*. Journal of Graph Algorithms and Applications, 17 (4), 2013, pp.495–513.

[6] G. Di Battista, P. Eades, R. Tamassia, and I.G. Tollis. *Graph Drawing. Algoriithms for the visualization of graphs*. Prentice Hall, 1999, 397 p.

[7] K. Sugiyama, S. Tagawa, and M. Toda. *Methods for visual understanding of hierarchical system structures*. IEEE Trans. Syst. Man. Cybern., 11(2), 1981, pp.109–125.

[8] N.S. Nikolov. *Sugiyama Algorithm*. Encyclopedia of Algorithms - 2016 Edition, Springer 2016, pp. 2162–2166.

Această lucrare a fost executată în cadrul proiectului de cercetare 20.80009.5007.22. „Sisteme informatice inteligente pentru soluționarea problemelor slab structurate, procesarea cunoștințelor și volumelor mari de date”.

(CIUBOTARU Constantin) Vladimir Andrunachievici Institute of Mathematics and Computer Science
*E-mail address*: chebotar@gmail.com