LUCRAREA 3

Automate finite de stări

Cuprinsul lucrării:

- Automate finite de stări. Definiţie. Reprezentări.
- Automate finite deterministe (AFD)
- Automate finite nedeterministe (AFN)
- Transformarea unui AFN într-un AFD

Unele exemple şi figuri din acest capitol au fost preluate şi adaptate din [8] şi [9].

3.1 Breviar teoretic

În cadrul acestei secţiuni se vor trece în revistă definiţiile, tipurile şi modalităţile de reprezentare ale automatelor finite de stări.
Un automat finit de stări este definit de următorul ansamblu de elemente:

- O mulţime finită de stări
- O stare iniţială
- O submulţime de stări acceptoare (sau finale)

- O tabelă de tranziţii sau o reprezentare sub formă de graf care specifică starea următoare pe baza perechii $(stare, caracter\ de\ intrare)$
- O mulţime de simboluri de intrare posibile

Există două tipuri de automate finite de stări:

1. Automate finite deterministe (AFD)
2. Automate finite nedeterministe (AFN)

Un AFD este definit de 5 elemente $(Q, \Sigma, \delta, q_0, F)$:

- Q este o mulţime finită de stări
- Σ este alfabetul de intrare (mulţimea simbolurilor de intrare posibile)
- $\delta : Q \times \Sigma \to Q$ este funcţia de tranziţie
- q_0 este starea iniţială
- F este mulţimea de stări finale (acceptoare)

Caracteristica esenţială a AFD-urilor este că fiecare tranziţie a unui astfel de automat este determinată complet de starea curentă şi de simbolul de intrare.
în cazul unui AFN, formal, diferenţele sunt:

- Funcţia de tranziţie este diferită $\delta : Q \times \Sigma \to P(Q)$ deoarece un AFD dintr-o stare prin acelaşi simbol de intrare poate ajunge într-o mulţime de stări
- $\lambda \in \Sigma$, ceea ce înseamnă că putem avea tranziţii care să nu necesite simboluri de intrare

Cu alte cuvinte, un AFN este un automat finit care poate avea mai multe tranziţii din aceaşi stare cu acelaşi simbol precum şi λ-tranziţii care nu implică vreun simbol de intrare.

Datorită particularităţilor de mai sus, se observă că un AFD este definit mai strict, ceea ce face ca un AFN să fie de regulă mult mai redus ca număr de stări şi de tranziţii decât un AFD care recunoaşte acelaşi limbaj, deci este mai uşor de implementat un AFN decât un AFD.

3.2 Reprezentarea automatelor

În Figura 3-1 este descris un automat care recunoaşte identificatori C++ valizi prin intermediul unui graf de tranziţii. Fiecare stare a automatului este un nod al grafului. Fiecare arc din graf leagă două stări şi are unul sau mai multe simboluri de intrare.
Dacă un arc pleacă din q către q_0 cu simbolul a, acest lucru semnifică faptul că dacă automatul se află în starea q şi citeşte un simbol de intrare a, atunci el intră în starea q_0, proces ce poartă numele de tranziţie a automatului. Pentru orice automat starea iniţială este indicată printr-o săgeată care nu pleacă de nicăieri, iar stările acceptoare prin două cercuri concentrice.

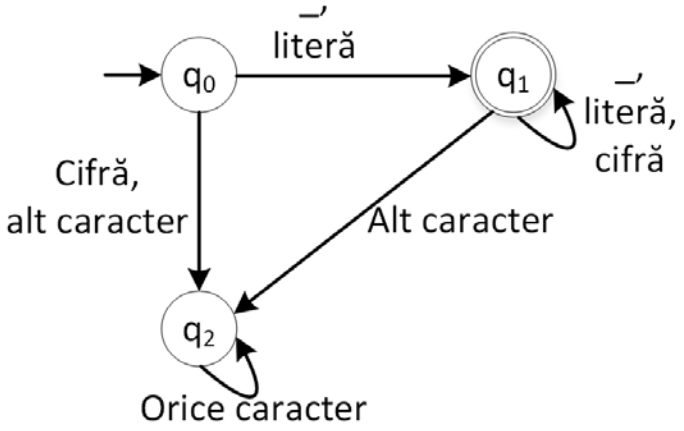

Figura 3-1: Graful de tranziţie pentru un AFD care determină dacă un şir de caractere este un identificator C++ valid.

AFD-ul din Figura 3-1 poate fi definit formal ca $(\{q_0, q_1, q_2\}, \Sigma, \delta, q_0, \{q_1\})$, unde Σ este mulţimea tuturor caracterelor ce pot fi introduse de la o tastatură obişnuită iar δ este definită ca fiind:

$$\delta(q_0, c) = \begin{cases} q_1, & \text{pentru } c = \text{caracterul \_ sau literă} \\ q_2, & \text{în rest} \end{cases}$$

$$\delta(q_1, c) = \begin{cases} q_1, & \text{pentru } c = \text{caracterul \_ sau literă sau cifră} \\ q_2, & \text{în rest} \end{cases}$$

$$\delta(q_0, c) = \{q_2, \text{ pentru orice } c \in \Sigma$$

O altă modalitate de a reprezenta funcţia de tranziţii este prin intermediul unui tabel de tranziţii (a se vedea Tabelul 3-1). Dintre toate cele trei metode, graful de tranziţii are avantajul de a oferi o perspectivă vizuală asupra automatului.

Tabelul 3-1: Tabelul de tranziţii pentru un AFD care determină dacă un şir de caractere este un identificator C++ valid.

δ	_	literă	cifră	alt caracter
q_0	q_1	q_1	q_2	q_2
q_1	q_1	q_1	q_1	q_2
q_2	q_2	q_2	q_2	q_2

Fiecare AFD rezolvă o anumită problemă acceptând unele şiruri de caractere de intrare şi refuzând altele, în funcţie de tranziţiile dintre stări şi starea în care se găseşte automatul la sfârşitul şirului de intrare. Dacă la sfârşitul şirului de intrare AFD-ul se află într-o stare acceptoare se poate spune că acesta recunoaşte sau acceptă şirul de intrare.

3.3 Exemple de automate finite

Exemplul 1: Automate finite pentru recunoaşterea limbajului tuturor şirurilor de 0 şi 1 care conţin subşirul 001:

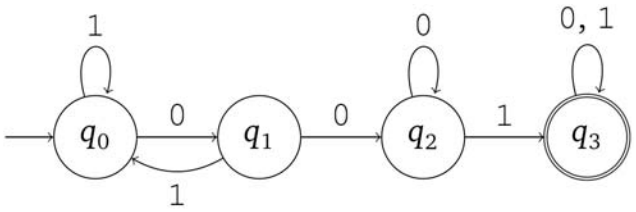

Figura 3-2: AFD care recunoaşte limbajul şirurilor de 0 şi 1 ce conţin subşirul 001.

În Figura 3-2 este prezentat AFD-ul care recunoaşte limbajul respectiv, adică pentru acele şiruri de intrare care conţin caracterele 0, 0, şi 1 consecutive în această ordine (de exemplu, şirul 1010010), automatul va ajunge într-o stare acceptoare. Automatul nu va recunoaşte însă şirurile de intrare care nu respectă aceasta proprietate, cum ar fi şirul 101101.

Iniţial, automatul se află în starea q_0 însă, pe măsură ce primeşte caractere de intrare, au loc tranziţii între stări în funcţie de fiecare caracter. Dacă şirul

de intrare începe cu 1, rămâne în starea q_0, dacă începe cu 0, va trece în starea q_1. Din starea q_1, citind caracterul 1 se întoarce în starea inițială, iar cu 0 va tranzita în q_2 unde rămâne până la întâlnirea unui 1, caz în care trece în starea q_3, stare acceptoare, deoarece a fost identificat un subșir 001. Dacă nu va urma niciun caracter 1, automatul rămâne în starea q_2, până la sfârșit, deci șirul de intrare nu va fi acceptat.

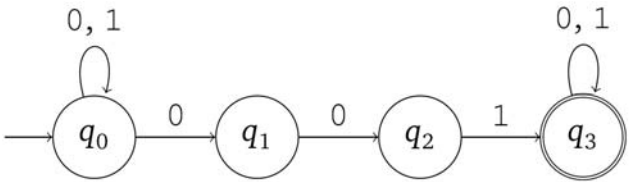

Figura 3-3: AFN care recunoaște limbajul șirurilor de 0 și 1 ce conțin subșirul 001.

Prin comparație, în Figura 3-3 este prezentat un AFN care recunoaște același limbaj. Diferențele față de AFD sunt evidente: în primul rând există tranziții lipsă (q_1 nu are nicio tranziție cu simbolul de intrare 1). Într-o astfel de situație, un AFD s-ar bloca, nu ar mai putea termina de citit întregul șir de caractere de intrare și, evident, nu l-ar mai putea accepta. În al doilea rând, unele stări au mai multe tranziții care consumă același simbol de intrare, cum e cazul stării q_0 care are două tranziții distincte cu simbolul 0. În acest caz, AFN-ul are de făcut o alegere cu privire la starea următoare: fie rămâne în starea q_0, fie trece în starea q_1.

În general, în situații de decizie se ia în considerare acea opțiune, dacă există, care va conduce AFN-ul în final spre o stare acceptoare. În exemplul de mai sus, dacă șirul de intrare conține o secvență 001, AFN-ul va aștepta în starea q_0 până ajunge la începutul unei astfel de secvențe (pot fi mai multe), va trece prin următoarele stări parcurgând simbolurile 0, 0, 1 ajungând în final în starea acceptoare unde va rămâne până la sfârșitul șirului de intrare.

Exemplul 2: Automate finite pentru recunoașterea limbajului tuturor șirurilor de 0 și 1 care conțin un număr par de simboluri 1 sau un număr par de simboluri 0:

În Figura 3-4 este ilustrat AFD-ul care recunoaște limbajul șirurilor de 0 și 1 care conțin un număr par de 1 sau un număr par de 0. AFN-ul pentru același limbaj este ilustrat în Figura 3-5. Se poate observa faptul că cel din urmă prezintă o construcție mai simplă decât AFD-ul și, în același

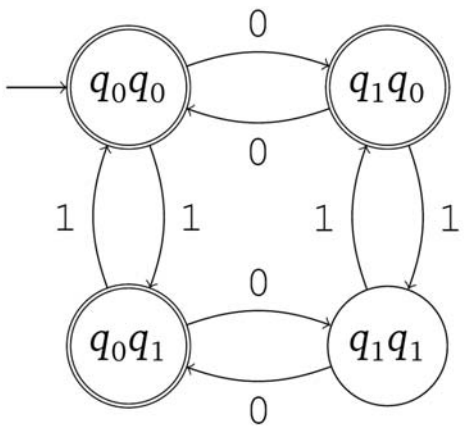

Figura 3-4: AFD care recunoaşte limbajul şirurilor de 0 şi 1 care conţin un număr par de 1 sau un număr par de 0.

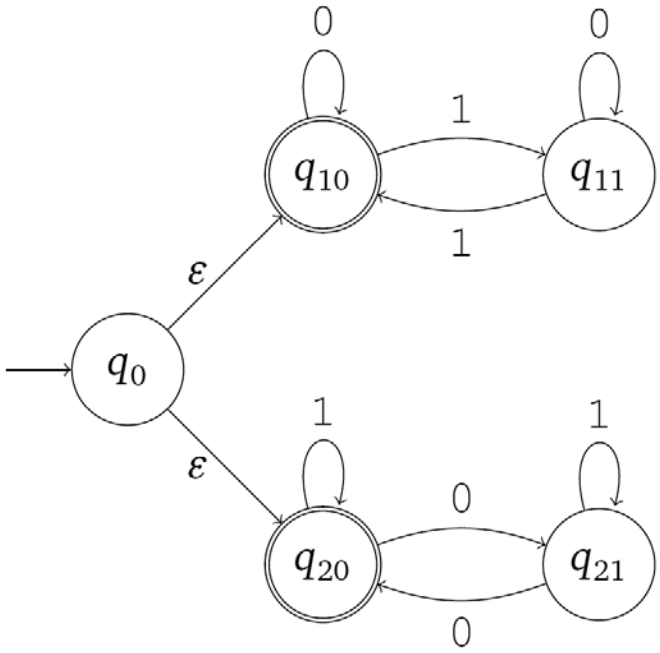

Figura 3-5: AFN care recunoaşte limbajul şirurilor de 0 şi 1 care conţin un număr par de 1 sau un număr par de 0.

timp, include o altă particularitate specifică AFN-urilor, şi anume ε–tranziţii. AFN-ul poate utiliza aceste tranziţii pentru a trece dintr-o stare într-alta fără a consuma simboluri de intrare. Automatul de mai sus are două astfel de

tranziţii, ambele provin din starea iniţială, ceea ce înseamnă că, înainte de a citi primul caracter din şirul de intrare, AFN-ul trebuie să ia decizia de a trece fie în starea q_{10}, fie q_{20}.

Decizia se ia în funcţie de existenţa unei alegeri care va conduce în final AFN-ul la acceptarea şirului de intrare. În acest exemplu, dacă numărul de simboluri 1 din şirul de intrare este par iar numărul de simboluri 0 nu, AFN-ul va trece în starea q_{10}. În situaţia inversă, va trece în starea q_{20}. Dacă atât numărul de simboluri 1 cât şi numărul de simboluri 0 este par, AFN-ul va realiza o tranziţie în oricare din cele două stări, iar dacă niciunul din aceste numere nu este par, AFN-ul nu va putea accepta şirul de intrare.

3.4 Conversia unui AFN în AFD

Orice AFN poate fi transformat, prin aplicarea unui algoritm simplu, într-un AFD echivalent, adică un AFD care recunoaşte acelaşi limbaj ca AFN-ul iniţial.

Exemplul 1:

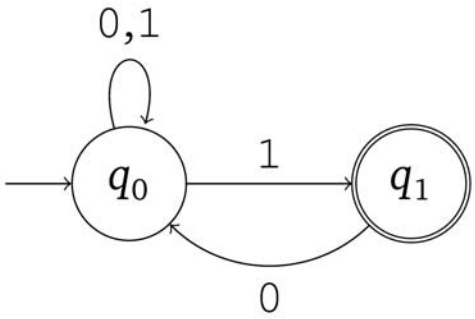

Figura 3-6: Un AFN care acceptă limbajul şirurilor ce se termină cu simbolul 1.

În Figura 3-6 de mai sus este prezentat un AFN care acceptă limbajul şirurilor de 0 şi 1 care se termină în 1. Obiectivul este să se obţină AFD-ul echivalent. Pentru acest lucru, primul pas îl reprezintă reprezentarea AFN-ului sub forma tabelului de tranziţii.

Fiind vorba despre un AFN, se observă faptul că există în tabel mulţimi de stări în care se ajunge cu acelaşi simbol de intrare din aceeaşi stare. Pasul următor al algoritmului implică adăugarea la acest tabel de noi linii care

Tabelul 3-2: . Funcţia de tranziţie al AFN-ului din Figura 3-6.

δ	0	1
q_0	q_0	$\{q_0, q_1\}$
q_1	q_0	-

Tabelul 3-3: Funcţia de tranziţie al AFD-ului echivalent AFN-ului din Figura 3-6.

δ'	0	1
q_0	q_0	$\{q_0, q_1\}$
q_1	q_0	-
$\{q_0, q_1\}$	q_0	$\{q_0, q_1\}$

reprezintă tranziţiile pentru aceste mulţimi de stări.

În acest exemplu, se va adăuga o nouă stare la tabel, $\{q_0, q_1\}$, din care se ajunge cu 0 în q_0 iar cu 1 în aceaşi stare $\{q_0, q_1\}$. Algoritmul continuă până când au fost adăugate toate mulţimile de stări apărute pe parcurs. Noul tabel astfel obţinut este tabelul de tranziţii al AFD-ului rezultat.

Pe baza noului tabel de tranziţii poate fi reprezentat AFD-ul şi sub forma de graf, cum este afişat în Figura 3-7. în noul AFD echivalent, starea iniţială rămâne cea din AFN, iar stările finale sunt atât cele din AFN, cât şi orice stare nou apărută (o mulţime de stări din AFN) care are în componenţă măcar o stare finală din AFN.

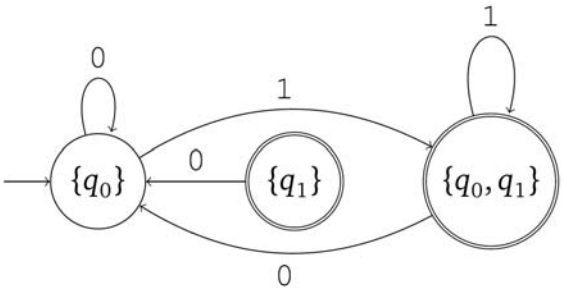

Figura 3-7: AFD-ul echivalent AFN-ului din Figura 3-6.

La o analiză a graf-ului AFD-ului echivalent rezultat, se observă faptul că starea q_1, deşi stare finală este inaccesibilă din moment ce nu există nicio tranziţie dintr-o altă stare către q_1. Astfel, AFD-ul poate fi minimizat prin reducerea stărilor inaccesibile. În acest caz se obţine automatul din Figura 3-8.

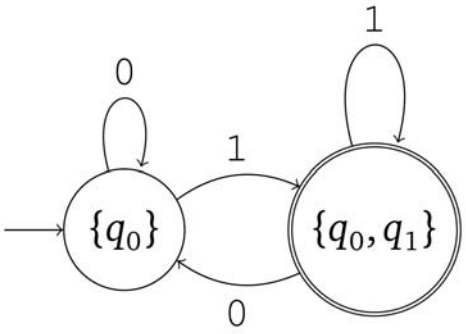

Figura 3-8: AFD-ul minimizat.

Acesta a fost un exemplu minimalist. În practică pot exista şi situaţii în care un AFN ce se doreşte convertit în AFD prezintă şi ε–tranziţii. În acel caz, pentru stabilirea tabelului de tranziţie trebuie ţinut cont şi de acest fapt, întrucât automatul poate ajunge într-o stare şi fără a consuma un caracter de intrare. Astfel, se introduce mulţimea de stări $E(R)$, care se numeşte extensia mulţimii de stări R şi conţine toate stările în care se poate ajunge din stările lui R prin oricâte ε-tranziţii (inclusiv prin niciuna).

Exemplul 2:

Fie AFN-ul din Figura 3-9. Pentru acest automat, extensiile stărilor sunt prezentate mai jos:

$$E(\{q_0\}) = \{q_0\}$$
$$E(\{q_1\}) = \{q_0, q_1\}$$
$$E(\{q_2\}) = \{q_0, q_1, q_2\}$$
$$E(\{q_2, q_3\}) = \{q_0, q_1, q_2, q_3\}$$

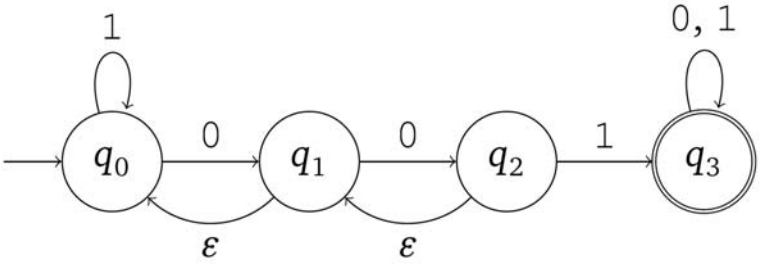

Figura 3-9: AFN ce recunoaşte limbajul şirurilor care conţin subşirul 001.

Pentru automatul de mai sus, funcţia de tranziţie a stărilor este prezentată în Tabelul 3-4. Această funcţie reflectă atât tranziţiile care se realizează pe baza simbolurilor de intrare, cât şi ε–tranziţiile, luându-se în calculul şi extensiile mulţimilor de stări prezentate mai sus. Spre exemplu, din starea q_0 se poate ajunge în starea q_1 printr-o tranziţie cu simbolul 0, dar se poate ajunge şi înapoi în starea q_0 din starea q_1 printr-o ε–tranziţie.

Tabelul 3-4: Funcţia de tranziţie pentru AFN-ul din Figura 3-9

δ	0	1
q_0	$\{q_0, q_1\}$	q_0
q_1	$\{q_0, q_1, q_2\}$	-
q_2	-	q_3
q_3	q_3	q_3

Pe baza tabelului de tranziţii se continuă algoritmul de conversie adăugându-se noile stări reprezentate de mulţimile din tabel. Rezultatul este funcţia de tranziţie a AFD-ului echivalent, prezentată în Tabelul 3-5.

Tabelul 3-5: Funcţia de tranziţie pentru AFD-ul echivalent AFN-ului din Figura 3-9

δ'	0	1
q_0	$\{q_0, q_1\}$	q_0
q_1	$\{q_0, q_1, q_2\}$	-
q_2	-	q_3
q_3	q_3	q_3
$\{q_0, q_1\}$	$\{q_0, q_1, q_2\}$	q_0
$\{q_0, q_1, q_2\}$	$\{q_0, q_1, q_2\}$	$\{q_0, q_3\}$
$\{q_0, q_3\}$	$\{q_0, q_1, q_3\}$	$\{q_0, q_3\}$
$\{q_0, q_1, q_3\}$	$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_3\}$
$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_1, q_2, q_3\}$	$\{q_0, q_3\}$

Analizând funcţia de tranziţie a AFD-ului echivalent, se observă faptul că stările q_1, q_2 şi q_3 sunt inaccesibile de către restul stărilor automatului, astfel încât pot fi reduse. AFD-ul echivalent minimizat este prezentat în Figura 3-10.

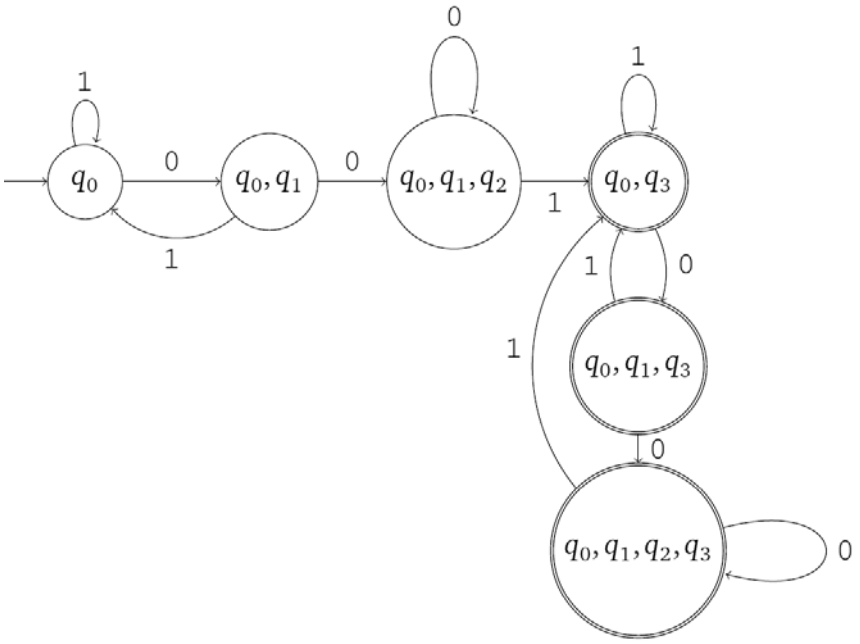

Figura 3-10: AFD-ul echivalent minimizat.

3.5 Probleme propuse

1. Propuneţi un exemplu de AFD care acceptă toate şirurile de caractere care se termină cu secvenţa aa. Alfabetul limbajului este $\{a, b\}$.

2. Propuneţi un exemplu de AFD care acceptă toate şirurile de caractere care se termină cu b şi nu conţin secvenţa aa. Alfabetul limbajului este $\{a, b\}$.

3. Propuneţi un exemplu de AFN care acceptă toate şirurile de caractere de lungime minim 2 a căror ultime două simboluri sunt la fel. Alfabetul este 0,1. AFN-ul să aibă maxim 4 stări şi 6 tranziţii.

4. Propuneţi un exemplu de AFN care acceptă toate şirurile de caractere de lungime minim 2 în care pe penultima poziţie se află întotdeauna 1. Alfabetul este {0,1}. AFN-ul să aibă maxim 4 stări şi 6 tranziţii.

5. Transformaţi AFN-ul din Figura 3-11 într-un AFD. Dacă este cazul, minimizaţi stările noului automat.

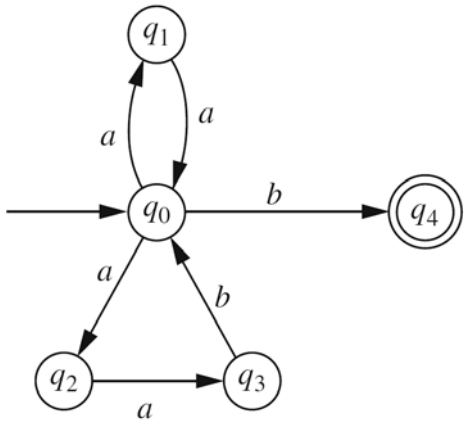

Figura 3-11: AFN-ul corespunzător problemei 5.

6. Transformaţi AFN-ul din Figura 3-12 într-un AFD. Dacă este cazul, minimizaţi stările noului automat.

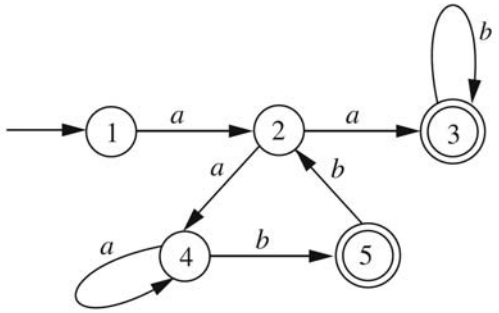

Figura 3-12: AFN-ul corespunzător problemei 6.