1. Leksička analiza

1.1 Determinističku konačni automati

Zadatak 1. **(1.1.1)**

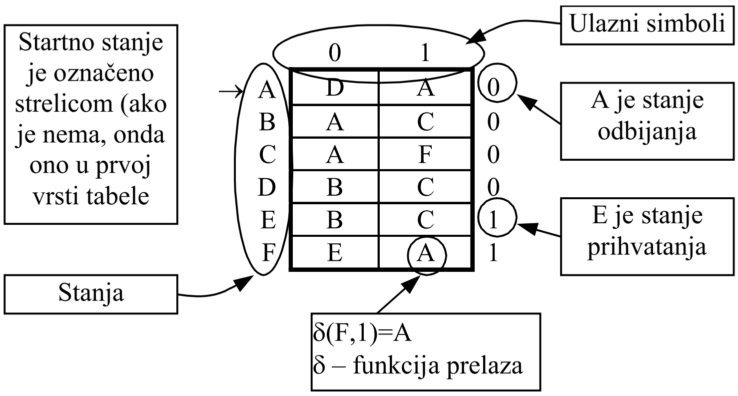
a) Predstaviti zadati konačni automat u alternativnoj formi, pomoću grafa prelaza.

b) Izložiti nekoliko sekvenci koje ovaj automat prepoznaje. Navesti nekoliko sekvenci koje ovaj automat odbija.

c) Pronaći najkraću sekvencu koju dati automat prepoznaje.

	0	1	
→ A	D	A	0
B	A	C	0
C	A	F	0
D	B	C	0
E	B	C	1
F	E	A	1

Analiza problema



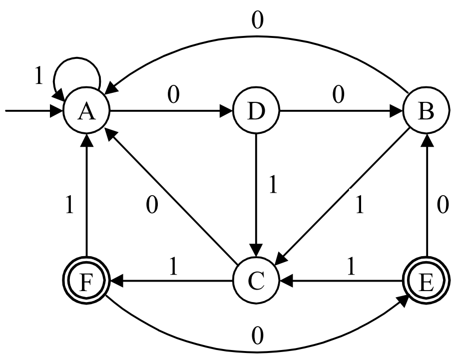
- Algoritam rada automata:

```
tekuće_stanje := S_t;          { S_t – startno stanje}
tekući_ulaz := prvi simbol ulazne sekvence;
while not (kraj ulazne sekvence)
        tekuće_stanje := δ( tekuće_stanje, tekući ulaz );
        tekući_ulaz := sledeći znak ulazne sekvence;
end while;
if ( tekuće_stanje ∈ P )      {P – skup stanja prihvatanja}
        then ulazna sekvenca se prihvata;
        else ulazna sekvenca se ne prihvata;
end if.
```

- Jezik L(K) automata K = skup svih sekvenci ulaznih simbola koje automat prihvata.

Rešenje

a)



b)

- Posmatrajmo rad automata za ulaz 0110

$$A \xrightarrow{0} D \xrightarrow{1} C \xrightarrow{1} F \xrightarrow{0} E$$

- završno stanje E je stanje prihvatanja $\Rightarrow$ 0110 pripada jeziku automata L(K).
- Probanjem $\{1011, 0011, 011011, 011011011,...\} \in L(K)$
  $\{0, 00, 00111, 1, 11, 111, 1111,...\} \notin L(K)$

c) Najkraća sekvenca koju automat prihvata, izaziva najkraći niz promena stanja počev od startnog stanja A do nekog od stanja prihvatanja E ili F.

- Možemo gledati "unazad" od stanja prihvatanja E i F.
- Dolazak u E pri radu automata isključivo preko F $\Rightarrow$ najkraća sekvenca završava u F
- Dolazak u F:

$$A \Rightarrow C \xrightarrow{1} F$$

- Dolazak u C:

$$A \Rightarrow B \xrightarrow{1} C \xrightarrow{1} F$$

$$A \Rightarrow D \xrightarrow{1} C \xrightarrow{1} F$$

$$A \Rightarrow E \xrightarrow{1} C \xrightarrow{1} F \quad \text{– otpada zbog E}$$

- Dolazak u B preko D ili E $\Rightarrow$ otpada
- Dolazak u D

$$A \xrightarrow{0} D \xrightarrow{1} C \xrightarrow{1} F$$

- najkraća sekvenca koja se prihvata je 011

Zadatak Error! No text of specified style in document.**2.** **(1.1.4)**

Projektovati automat sa konačnim brojem stanja čija je ulazna azbuka skup {0,1} i koji prepoznaje tačno sledeće sekvence simbola:

a) sve ulazne sekvence

b) nijednu ulaznu sekvencu

c) sekvencu 101

d) samo praznu sekvencu.

Rešenje

a)

	0	1	
A	A	A	1

b)

	0	1	
A	A	A	0

c)

	0	1	
A	E	B	0
B	C	E	0
C	E	D	0
D	E	E	1
E	E	E	0

d)

	0	1	
A	E	E	1
E	E	E	0

Zadatak 3. (1.2.1)

Pronaći sva suvišna stanja u automatu prikazanom na slici.

	0	1	2	
→ A	C	E	G	0
B	J	E	G	0
C	J	A	H	1
D	F	A	G	0
E	E	J	H	1
F	D	I	A	0
G	H	A	J	0
H	G	J	B	1
I	D	F	G	0
J	B	H	G	0

Analiza problema

- Suvišna (nedostižna) stanja automata su ona stanja u koja se nikad ne može stići pod dejstvom ulaznih simbola iz startnog stanja, to jest, stanja koja nemaju uticaja na rad automata (mogu se izbaciti). Najčešće se ne radi samo o jednom suvišnom stanju, već postoji skup suvišnih stanja unutar kojeg se može kružiti ali se u njega nikako ne može doći iz početnih stanja. Postupak nalaženja suvišnih stanja je indirektan: zapravo se traže sva stanja u koja se može doći iz početnog.

- Nalaženje skupa D dostižnih stanja:

1. Ubaciti startno stanje u D i označiti ga kao neobrađeno.
2. Dok postoji neobrađeno stanje S u D ponavljati korake 2.– 4.
    3. Ubaciti u D (kao neobrađena) sva stanja koja se pojavljuju u vrsti S tabele prelaza.
    4. Označiti S kao obrađeno.

Rešenje

1. D = {A}
2. Razmatramo vrstu A: D = {<u>A</u>, C, E, G}
3. Razmatramo vrstu C: D = {<u>A</u>, <u>C</u>, E, G, J, H}
4. Razmatramo vrstu E: D = {<u>A</u>, <u>C</u>, <u>E</u>, G, J, H}
5. Razmatramo vrstu G: D = {<u>A</u>, <u>C</u>, <u>E</u>, <u>G</u>, J, H}
6. Razmatramo vrstu J: D = {<u>A</u>, <u>C</u>, <u>E</u>, <u>G</u>, <u>J</u>, H, B}
7. Razmatramo vrstu H: D = {<u>A</u>, <u>C</u>, <u>E</u>, <u>G</u>, <u>J</u>, <u>H</u>, B}
8. Razmatramo vrstu B: D = {<u>A</u>, <u>C</u>, <u>E</u>, <u>G</u>, <u>J</u>, <u>H</u>, <u>B</u>}

Znači suvišna su stanja: {D, F, I} i možemo jednostavno izbaciti odgovarajuće vrste iz tabele prelaza.

Zadatak 4. **(1.2.2)**

a) Dati primer dva ekvivalentna konačna automata M i N, gde jednom stanju automata N odgovara više ekvivalentnih stanja automata M. Dokazati da primer zadovoljava uslove zadatka.

b) Dati primer dva ekvivalentna konačna automata M i N, gde automat M ima stanje koje nije ekvivalentno ni sa jednim stanjem automata N. Dokazati da primer zadovoljava uslove zadatka.

Analiza problema

- Stanje S_x automata X ekvivalentno je stanju S_y automata Y ako i samo ako automat X startujući od stanja S_x prihvata tačno isti skup sekvenci kao i automat Y startujući od stanja S_y.
- Ako stanja S_x automata X i S_y automata Y nisu ekvivalentna, onda postoji najmanje jedna sekvenca koju automat X, polazeći iz stanja S_x prihvata, a automat Y polazeći iz stanja S_y odbacuje, ili obratno, automat X, polazeći od stanja S_x, sekvencu odbacuje, a automat Y, polazeći od stanja S_y je prihvata. Takva sekvenca naziva se *sekvenca razlikovanja*.
- Operativna definicija ekvivalencije stanja: Stanja S_x i S_y su ekvivalentna ako i samo ako su zadovoljeni:

1. Uslov kompatibilnosti - stanja S_x i S_y moraju oba biti ili stanja prihvatanja ili stanja odbijanja.
2. Uslov propagacije - prelazi iz ovih stanja, za svaki ulazni simbol, moraju biti u međusobno ekvivalentna stanja.

- Dva automata su ekvivalentna ako i samo ako su im startna stanja ekvivalentna.

Rešenje

a)

Automat M:

	x	y	
→ A1	B1	C1	0
B1	B1	A1	1
C1	B1	A1	1

Automat N:

	x	y	
→ A2	C2	C2	0
C2	C2	A2	1

- Na slici je prikazan proizvoljan automat M koji ima dva ekvivalentna stanja B1 i C1 (oba predstavljaju stanja prihvatanja i vrste B1 i C1 su identične tako da su zadovoljeni uslovi kompatibilnosti i propagacije).
- Uklanjanjem stanja B1 (zamena sa C1 u tabeli), preimenovanjem A1 u A2, C1 u C2 dobijamo automat N.
- da bi važilo M $\Leftrightarrow$ N treba da bude A1 $\Leftrightarrow$ A2
- A1 je kompatibilno sa A2 (oba stanja odbijanja).
- Ispitivanje uslova propagacije $\Rightarrow$ konstrukcija tabele ekvivalencije stanja

	x	y
A1, A2	B1, C2	C1, C2

- da bi bilo A1 $\Leftrightarrow$ A2 treba da bude B1 $\Leftrightarrow$ C2, kao i C1 $\Leftrightarrow$ C2. Kompatibilnost OK, propagacija? nastavljamo popunjavanje tabele:

	x	y
A1, A2	B1, C2	C1, C2
B1, C2	B1, C2	A1, A2
C1, C2	B1, C2	A1, A2

- nema novih parova stanja, popunjavanje tabele završeno.
- Nema parova nekompatibilnih stanja $\Rightarrow$ svi parovi iz tabele predstavljaju međusobno ekvivalentna stanja.
- Dakle M $\Leftrightarrow$ N i B1 $\Leftrightarrow$ C2 i C1 $\Leftrightarrow$ C2, čime su zadovoljeni uslovi zadatka.

b)

- Automatu M iz tačke a) dodaćemo stanje D1, a automat N ostaje isti.

Automat M:

	x	y	
→ A1	B1	C1	0
B1	B1	A1	1
C1	B1	A1	1
D1	D1	D1	0

Automat N:

	x	y	
→ A2	C2	C2	0
C2	C2	A2	1

Stanje D1 nije dostižno iz stanja A1, to jest nema uticaja na rad automata M, pa su automati M i N ekvivalentni kao što je pokazano u tački a). Stanje D1 nije ekvivalentno stanju C2 jer nije zadovoljen uslov kompatibilnosti. Što se tiče para D1, A2 konstrukcijom tabele ekvivalencije:

	x	y
D1, A2	D1, C2	D1, C2
D1, C2		

vidimo da uslov propagacije nije zadovoljen jer stanja D1 i C2 nisu međusobno ekvivalentna.

Zaključujemo da stanje D1 automata M nije ekvivalentno ni sa jednim stanjem automata N čime su zadovoljeni uslovi zadatka.

Zadatak Error! No text of specified style in document.**5.** **(1.2.3 deo)**

Pronaći minimalnu ekvivalentnu tabelu prelaza za zadati automat.

	0	1	
→ S1	S1	S3	0
S2	S7	S4	1
S3	S6	S5	0
S4	S1	S4	1
S5	S1	S4	0
S6	S7	S6	1
S7	S7	S3	0

Analiza problema

- Ekvivalentni automat je onaj koji prihvata tačno isti skup sekvenci kao i polazni automat. Za određeni automat postoji mnogo različitih ekvivalentnih automata. Od interesa je među takvim naći onaj sa najmanjim brojem stanja. Takav automat nema suvišnih niti međusobno ekvivalentnih stanja.
- postupak minimizacije automata:

1. uklanjanje suvišnih stanja;
2. nalaženje skupova međusobno ekvivalentnih stanja i zamena takvih skupova stanja sa po jednim predstavnikom za svaki od skupova.

Rešenje

a)

- Najpre eliminišimo suvišna stanja.. Dostižna stanja su {S_1, S_3, S_5, S_6, S_7, S_4}, što znači da je stanje S_2 suvišno, pa ga izbacujemo iz tabele prelaza.

<u>Prva varijanta postupka minimizacije</u>:

- Proveravamo ekvivalenciju stanja po parovima na način opisan u prethodnom zadatku

1. par stanja: S_1, S_3. Oba su stanja neprihvatanja. Konstruišemo tabelu ekvivalencije.

	0	1
S_1, S_3	S_1, S_6	S_3, S_5
S_1, S_6		

Stanja S_1 i S_6 nisu ekvivalentna jer nisu kompatibilna pa ni S_1 i S_3 nisu ekvivalentna.

2. par stanja: S_1, S_4. Nisu kompatibilna, pa nisu ni ekvivalentna.
3. par stanja: S_1, S_5 – kompatibilna stanja. Tabela ekvivalencije glasi:

	0	1
S_1, S_5	S_1	S_3, S_4
S_3, S_4		

Stanja S_3 i S_4 nisu ekvivalentna jer nisu kompatibilna pa ni S_1 i S_5 nisu ekvivalentna.

4. par stanja: S_1, S_6. Nisu kompatibilna, pa nisu ni ekvivalentna.
5. par stanja: S_1, S_7 – kompatibilna stanja. Tabela ekvivalencije glasi:

	0	1
S_1, S_7	S_1, S_7	S_3

Tabela je kompletirana, što znači da su stanja S_1 i S_7 ekvivalentna.

6. par stanja: S_3, S_4 – nekompatibilna stanja, pa nisu ni ekvivalentna.
7. par stanja: S_3, S_5 – kompatibilna stanja. Tabela ekvivalencije glasi:

	0	1
S_3, S_5	S_1, S_6	S_5, S_4

Stanja S_1 i S_6 nisu kompatibilna, pa ni stanja S_3 i S_5 nisu ekvivalentna.

8. par stanja: S_3, S_6. Nisu kompatibilna, pa nisu ni ekvivalentna.
9. par stanja: S_3, S_7. Nisu ekvivalentna jer su S_1 i S_7 ekvivalentna, a S_1 i S_3 nisu kako je ranije utvrđeno.
10. par stanja: S_4, S_5. Nisu kompatibilna, pa nisu ni ekvivalentna.
11. par stanja: S_4, S_6 - kompatibilna stanja. Tabela ekvivalencije glasi:

	0	1
S_4, S_6	S_1, S_7	S_4, S_6

Kako su S_1 i S_7 ekvivalentna to su i S_4 i S_6 ekvivalentna.

12. par stanja: S_4, S_7. Nisu kompatibilna, pa nisu ni ekvivalentna.
13. par stanja: S_5 , S_6. Nisu kompatibilna, pa nisu ni ekvivalentna.
14. par stanja: S_5 , S_7. Nisu ekvivalentna jer su S_1 i S_7 ekvivalentna, a S_1 i S_5 nisu kako je ranije utvrđeno.
15. par stanja: S_6 , S_7. Nisu kompatibilna, pa nisu ni ekvivalentna.

Svi parovi su ispitani i utvrđeno je da su međusobno ekvivalentni parovi stanja S_1, S_7 i S_4, S_6, te svaki od parova možemo zameniti jednim stanjem. Obeležimo stanja na sledeći način: A = {S_1, S_7}, B = {S_4, S_6}, C = S_3, D = S_5 . Rezultujući minimalni automat:

0 1

→ A	A	C	0
B	A	B	1
C	B	D	0
D	A	B	0

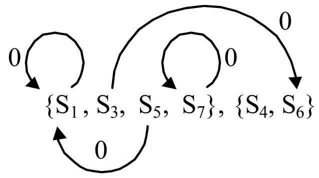
Druga varijanta postupka minimizacije (particioni metod):

- ekvivalencija stanja je relacija ekvivalencije ⇒ skup svih stanja deli na međusobno disjunktne podskupove. U svakom podskupu su stanja koja su međusobno ekvivalentna.
- Da bismo našli podelu skupa stanja prema ovoj relaciji, primenjujemo iterativan postupak u više koraka, u svakom koraku rafiniramo podelu skupa stanja dok ne dobijemo konačnu podelu.

1° Inicijalna podela na stanja prihvatanja i stanja odbijanja

$P_0 = (\{S_1, S_3, S_5, S_7\}, \{S_4, S_6\})$

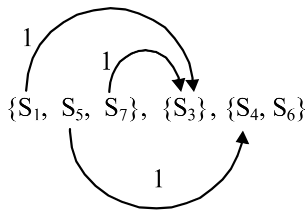
2° U narednim koracima pokušavamo da dalje razbijemo pojedine skupove stanja, utvrđujući koja stanja ne zadovoljavaju uslov propagacije. Prvo ćemo razmotriti skup stanja $\{S_1, S_3, S_5, S_7\}$ i prelaz iz ovih stanja za ulazni simbol 0:



S_3 treba izdvojiti jer njegov prelaz je u drugi skup. Tako imamo drugu podelu:

$P_1 = (\{S_1, S_5, S_7\}, \{S_3\}, \{S_4, S_6\})$

Prelazi iz S_1, S_5, S_7 pod dejstvom nule završavaju u istom skupu, pa treba razmotriti prelaze za ulazni simbol 1:



$P_2 = (\{S_1, S_7\}, \{S_5\}, \{S_3\}, \{S_4, S_6\})$

Dalje posmatramo prelaze iz $\{S_1, S_7\}$ pod dejstvom 0 i 1 i vidimo da prelaze u iste skupove. Isto važi i za prelaze iz $\{S_4, S_6\}$. Sada se vidi da je P_2 istovremeno i završna podela. Stanja koja su se našla u istim skupovima u P_2 su međusobno ekvivalentna.

Podela pomoću particija se može prikazati sledećom tabelom:

Ulaz	Particije					
	P0				P1	
	S1	S3	S5	S7	S4	S6
0	S1	S6	S1	S7	S1	S7
	P0	P1	P0	P0	P0	P0
	P00			P01	P1	
	S1	S5	S7	S3	S4	S6

1	S3	S4	S3		S4	S6
	P01	P1	P01		P1	P1
	P000		P001	P01	P1	
	S1	S7	S5	S3	S4	S6

Na početku imamo dve particije P0 i P1, gde P0 sadrži samo stanja odbijanja dok P1 sadrži prihvatanja. Particije ispitujemo pomoću ulaznih simbola.

Za ulazni simbol 0 stanja S1, S5 i S7 prelaze u particiju P0, dok stanje S3 prelazi u particiju P1. Iz toga zaključujemo da S3 ne može biti ekvivalentno sa stanjima S1, S5 i S7. Stanja iz particije P1 prelaze u istu particiju, prema tome ona su i dalje kandidati za ekvivalentna stanja.

Particiju P0 delimo na dve particije P00 i P01 od kojih jedna sadrži S1, S5 i S7 dok druga sadrži samo S3. Particija P1 ostaje netaknuta.

Za ulazni simbol 1 stanja S1 i S7 prelaze u particiju P01, dok stanje S5 prelazi u particiju P1, što znači da ne može da bude ekvivalentno sa stanjima S1 i S7. Oba stanja iz praticije P1 prelaze u particiju P1, tako da su ona i dalje kandidati za ekvivalenciju.

Particija P00 se deli na dve particije P000 i P001 od kojih jedna sadrži stanja S1 i S7, dok druga sadrži stanje S5. Particija P1 se ne deli na nove particije.

Pošto nema novih simbola analiza se zaustavlja i stanja koja su ostala u istim particijama se proglašavaju za ekvivalentna.

1.2 Nedeterministički konačni automati

Zadatak 1. **(1.3.1)**

Opisati postupak kojim se za proizvoljni nedeterministički automat i proizvoljnu sekvencu ulaznih simbola utvrđuje da li je automat prihvata. Demonstrirati postupak za sekvencu abcacbc i automat sa slike.

	a	b	c	
→ A	A, B, C	C	E	0
→ B	E	A	E	0
→ C	C, A	C, B	D	1
D	D	D	B, A	0
E	E	E	E	0

Analiza problema

	a	b	c	
→ A	A, B, C	C	E	0
→ B	E	A	E	0
→ C	C, A	C, B	D	1
D	D	D	B, A	0
E	E	E	E	0

Moguć veći broj startnih stanja

$\delta(D,c)=\{B, A\}$ prelaz može biti u bilo koje stanje iz navedenog skupa

- za određenu ulaznu sekvencu postoji više različitih scenarija promene stanja, počevši od nekog od startnih stanja

$A \xrightarrow{a} A \xrightarrow{b} C$ prihvata

$A \xrightarrow{a} B \xrightarrow{b} A$ odbija

$A \xrightarrow{a} C \xrightarrow{b} C$ prihvata

$A \xrightarrow{a} C \xrightarrow{b} B$ odbija

. . .

- Po definiciji, ako postoji bar jedan scenario prihvatanja sekvenca se prihvata

Rešenje

- Moramo voditi evidenciju, u svakom koraku rada automata, o skupu svih mogućih stanja u kojima se automat može nalaziti, prema sledećem algoritmu:

tekući_skup_stanja := S_t;

tekući_ulaz := prvi simbol ulazne sekvence;

while not (kraj ulazne sekvence)

    novi_skup_stanja := Ø;

    for (za svako s ∈ tekući_skup_stanja)

        novi_skup_stanja := novi_skup_stanja U δ(s, tekući_ulaz);

    end for;

tekući_skup_stanja := novi_skup_stanja;

tekući_ulaz := novi simbol ulazne sekvence;

end while;

if (tekući_skup_stanja ∩ P ≠ Ø)

then ulazna sekvenca se prihvata;

else ulazna sekvenca se ne prihvata;

end if;

- Primer za sekvencu abcacbc:

  Krećemo od skupa startnih stanja {A, B,C}. Za ulaz a:
  iz A se može ići u A, B, ili C,
  iz B u E
  iz C u C ili A
  dakle u zbiru iz {A, B, C} se može ići u {A, B, C, E}

$$\{A, B, C\} \xrightarrow{a} \{A, B, C, E\} \xrightarrow{b} \{A, B, C, E\} \xrightarrow{c} \{D, E\} \xrightarrow{a}$$
$$\longrightarrow \{D, E\} \xrightarrow{c} \{A, B, E\} \xrightarrow{b} \{A, C, E\} \xrightarrow{c} \{D, E\}$$

Sekvenca se ne prihvata jer su i E i D stanja odbijanja (mora biti bar jedno stanje prihvatanja).

Zadatak 2. **(1.3.2)**

Odrediti deterministički konačni automat ekvivalentan datom nedeterminističkom

	x	y	z	
→ A	A, D	C	A	0
B	A, B			0
→ C		C, D		1
D	D	C	A	1

Analiza problema

- Svako stanje traženog determinističkog automata je skup onih stanja nedeterminističkog automata u kojima se isti može naći u datom trenutku procesiranja ulazne sekvence.
- Tabela prelaza determinističkog automata se određuje na sledeći način:
  1. Startno stanje determinističkog automata se određuje kao skup startnih stanja nedeterminističkog automata. Obeležiti prvu vrstu tabele ovim stanjem.
  2. Izabrati nepopunjenu vrstu tabele označenu stanjem S.
  3. S je stanje prihvatanja ako sadrži bar jedno stanje prihvatanja nedeterminističkog automata, u suprotnom se radi o stanju odbijanja.
  4. Svaki ulaz u izabranoj vrsti tabele popunjava se na sledeći način: za svaki ulazni simbol u odrediti skup S’ = $\bigcup \delta(s, u)$, $s \in S$, gde je δ funkcija prelaza nedeterminističkog automata. Ulaz tabele u vrsti S i koloni u popuniti sa S’. Ako nijedna vrsta tabele prelaza nije označena sa S’, označiti novu vrstu tabele prelaska sa S’.

  Ako postoji nepopunjena vrsta tabele, preći na korak 2. U suprotnom, konstrukcija automata je završena.

Rešenje

	x	y	z	
→ {A, C}	{A, D}	{C, D}	{A}	1
{A, D}	{A, D}	{C}	{A}	1
{C, D}	{D}	{C, D}	{A}	1
{A}	{A, D}	{C}	{A}	0
{C}	{}	{C, D}	{}	1
{D}	{D}	{C}	{A}	1
{}	{}	{}	{}	0

{} – stanje greške.

Zadatak 3. (1.3.6)

Konstruisati minimalni deterministički konačni automat koji prihvata sve i samo one sekvence oblika s_1s_2 gde je s_1 sekvenca koju prihvata automat sa slike (a), a s_2 sekvenca koju prihvata automat sa slike (b).

	a	b	
→ 0	1		1
1	2	1	0
2		0	1

(a)

	b	c	
→ A	B	C	0
B	C		1
C		A	0

(b)

Analiza problema

Opštija formulacija problema glasi: dokazati da je skup regularnih jezika (odnosno jezika koji mogu biti opisani konačnim automatima) zatvoren u odnosu na operaciju konkatenacije. Drugim rečima, potrebno je dokazati da je jezik L_D koji se dobija operacijom konkatenacije dva proizvoljna regularna jezika L_1 i L_2, takođe regularan. Operacija konkatenacije nad jezicima, kao skupovima sekvenci, definiše se kao Dekartov proizvod tih skupova, kao što je u postavci zadatka navedeno.

Prvi korak u rešavanju problema je formiranje nedeterminističkog automata koji prihvata traženi skup sekvenci, po sledećem algoritmu:

- Skup ulaznih simbola novog automata je unija skupova ulaznih simbola polaznih automata.
- Skup stanja novog automata je unija skupova stanja polaznih automata.
- Funkcija prelaza novog automata sadrži sve prelaze koji postoje među stanjima polaznih automata. Dodatno, od stanja u novom automatu koja odgovaraju stanjima prihvatanja automata sa slike (a) postoje prazni prelazi, odnosno prelazi za ulaznu sekvencu ε dužine nula simbola, do stanja novog automata koje odgovara startnom stanju automata sa slike (b). Zbog postojanja praznih prelaza novi automat je nedeterministički.
- Startno stanje novog automata odgovara startnom stanju automata sa slike (a).
- Stanja prihvatanja novog automata odgovaraju stanjima prihvatanja automata sa slike (b).

Drugi korak u rešavanju problema je prevođenje prethodno dobijenog nedeterminističkog automata u ekvivalentan deterministički. U zadatku 2. je dat algoritam za prevođenje NKA u DKA, ali se on u ovom zadatku mora modifikovati zbog postojanja ε-prelaza u NKA:

1. Startno stanje determinističkog automata se određuje kao ε-zatvaranje skupa startnih stanja nedeterminističkog automata. Obeležiti prvu vrstu tabele ovim stanjem. Operacija ε-zatvaranja je definisana kasnije u tekstu.
2. Izabrati nepopunjenu vrstu tabele označenu stanjem S.

3. S je stanje prihvatanja ako sadrži bar jedno stanje prihvatanja nedeterminističkog automata, u suprotnom se radi o stanju odbijanja.

4. Svaki ulaz u izabranoj vrsti tabele popunjava se na sledeći način: za svaki ulazni simbol u odrediti skup S' = $\bigcup \delta(s, u)$, $s \in S$, gde je δ funkcija prelaza nedeterminističkog automata. Ulaz tabele u vrsti S i koloni u popuniti sa skupom koji se dobija ε-zatvaranjem skupa S'. Ako nijedna vrsta tabele prelaza nije označena sa S', označiti novu vrstu tabele prelaska sa S'.

5. Ako postoji nepopunjena vrsta tabele, preći na korak 2. U suprotnom, konstrukcija automata je završena.

Operacija ε-zatvaranja glasi:

```
Ulaz: skup stanja T, Izlaz: skup stanja ε-closure(T);
for (za svako stanje t ∈ T)
        staviti t na stek;
end for;
ε-closure(T) := T;
while (stek nije prazan)
        skinuti vršni element t sa steka;
        for (za svako stanje u za koje postoji ε-prelaz od t ka u)
                ε-closure(T) := ε-closure(T) U {u};
                staviti u na stek;
        end for;
end while;
```

Treći i poslednji korak u rešavanju problema je minimizacija prethodno dobijenog determinističkog automata na neki od načina ranije opisanih na vežbama.

Rešenje

Tabela prelaza nedeterminističkog automata prikazana je na slici. Startno stanje novog automata je startno stanje prvog automata, a stanje prihvatanja je stanje prihvatanja drugog automata.

	a	b	c	ε	
→ 0	1			A	0
1	2	1			0
2		0		A	0
A		B	C		0
B		C			1
C			A		0

Deterministički ekvivalent dobijenog automata je:

	a	b	c	
→ {0, A}	{1}	{B}	{C}	0
{1}	{2, A}	{1}		0
{B}		{C}		1
{C}			{A}	0
{2, A}		{0, A, B}	{C}	0
{A}		{B}	{C}	0
{0, A, B}	{1}	{B, C}	{C}	1
{B, C}		{C}	{A}	1

1.3 Regularni izrazi

Zadatak 1. (1.4.2)

a) Napisati regularni izraz koji opisuje označene decimalne konstante. Primeri dozvoljenih konstanti su: 1257, +0**.**0392, -12345**.**36, 2**.**0 a nedozvoljenih: **.**123, -23**.**, +, **.** Napomena: U regularnom izrazu koristiti simbole +, -, **.** i d (simbol d predstavlja proizvoljnu cifru).

b) Konstruisati nedeterministički konačni automat koji odgovara regularnom izrazu dobijenom u tački a).

c) Konstruisati minimalni deterministički konačni automat ekvivalentan automatu dobijenom u tački b).

Analiza problema:

Regularni izrazi opisuju na formalan način skupove ulaznih sekvenci. Jednako "jak" mehanizam kao konačni automati. Generatori leksičkih analizatora (npr. yacc) kao ulaz uzimaju regularne izraze, a kao izlaz generišu konačne automate.

Regularni izraz *s* nad ulaznom azbukom Σ opisuje regularni skup nizova znakova L(*s*).

Regularni izrazi definišu se na sledeći način:

- ∅ je regularan izraz koji opisuje prazan skup L(∅)=∅.
- ε je regularan izraz koji opisuje skup {ε} L(ε)={ε}.
- niz znakova *n* ∈ Σ je regularni izraz koji opisuje skup {*n*} L(*n*)={*n*}.
- složeniji regularni izrazi dobijaju se od regularnih izraza A i B primenom operacija:
  - unije, oznaka A|B, L(A|B) = L(A) ∪ L(B) = {x | x ∈ A ili x ∈ B}
  - konkatenacije (nadovezivanja) AB ili A·B
    L(AB)=L(A) x L(B)={xy | x ∈ A, y ∈ B} (Dekartov proizvod skupova)
    Konvencija: $A^n \equiv \underbrace{A \cdot A \cdot ... \cdot A}_{n}$, $A^0 \equiv \{\varepsilon\}$
  - zvezdastog zatvaranja A^*
    $A^*=A^0|A^1|A^2|A^3|\ldots$ (beskonačan broj članova)
  - pozitivnog zatvaranja A^+
    $A^+ = A^1|A^2|A^3|\ldots$

Najviši prioritet imaju operacije $^+$ i *, zatim · pa |.

Primeri regularnih izraza:

L(abc)={abc}   L(a | b | c)={a, b, c}

L[(a | b)(c | d)]={ac, ad, bc, bd} male zagrade su obavezne zbog prioriteta

L(a^*)={ε, a, aa, aaa, …}

L[$(a | b)^+$]={a, b} ∪ {a, b} x {a, b} ∪ {a, b} x {a, b} x {a, b} ∪ …

={a,b} ∪ {aa, ab, ba, bb} ∪ {aaa, aab, aba, abb, baa, bab, bba, bbb} ∪...

={a, b, aa, ab, ba, bb, …} = svi mogući nizove slova a i b dužine 1 ili više

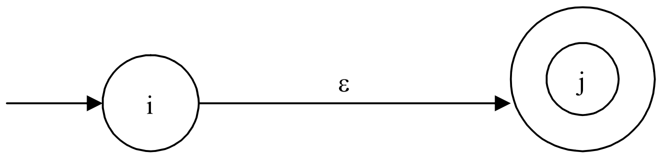
Regularni izraz slovo =A | B |…| Z | a | b |…| z

Cifra = 0 | 1 |…|9

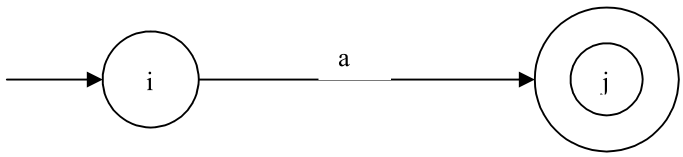
Identifikator = Slovo (Slovo | Cifra $)^*$

Pravila za konstrukciju nedeterminističkog konačnog automata (NKA) za dati regularni izraz definiše Tompsonov algoritam:

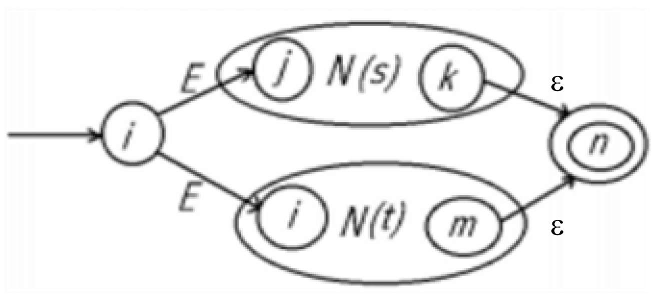
- regularnom izrazu ε odgovara sledeći automat:



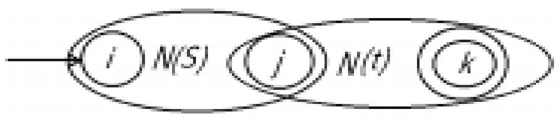
- regularnom izrazu a, gde je a simbol ulazne azbuke odgovara sledeći automat:



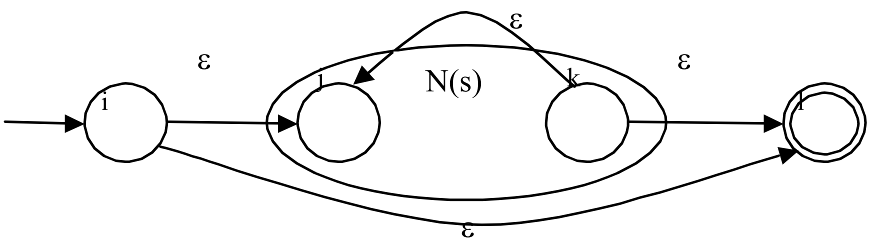
- ako je *N(s)* NKA za regularni izraz s, *N(t)* NKA za regularni izraz t, onda se NKA za uniju s|t formira na sledeći način:



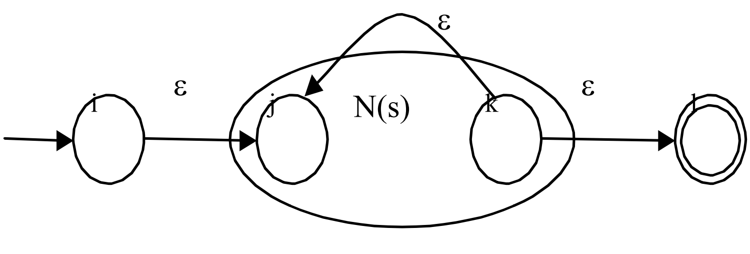
- ako je *N(s)* NKA za regularni izraz s, *N(t)* NKA za regularni izraz t, onda se NKA za konkatenaciju st formira na sledeći način:



- ako je *N(s)* NKA za regularni izraz s, onda se NKA za zvezdasto zatvaranje s^* formira na sledeći način:



- ako je *N(s)* NKA za regularni izraz s, onda se NKA za pozitivno zatvaranje s^+ formira na sledeći način:



Rešenje:

a)

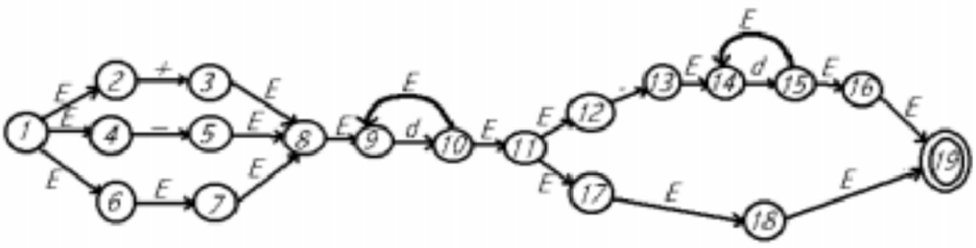
Usvojićemo da je d = [0-9] tj. 0 | 1 |…|9. Regularni izraz za decimalnu konstantu je:

$$(+ \mid - \mid \varepsilon)\ d^{+}\ (\,.\,d^{+} \mid \varepsilon)$$

Predznak je opcioni, decimale su takođe opcione, ali ako je decimalna tačka prisutna onda sa obe strane mora postojati bar po jedna cifra.

b)

Regularnom izrazu iz tačke a) odgovara nedeterministički automat prikazan na slici:



c)

Prethodno prikazanom nedeterminističkom automatu odgovara sledeći deterministički automat:

	+	-	*	d	
→ A={1, 2, 4, 6, 7, 8, 9}	B	C	D	E	0
B={3, 8, 9}	D	D	D	E	0
C={5, 8, 9}	D	D	D	E	0
D={}	D	D	D	D	0
E={9, 10, 11, 12, 17, 18, 19}	D	D	F	E	1
F={13, 14}	D	D	D	G	0
G={14, 15, 16, 19}	D	D	D	G	1

Stanja prihvatanja su E i G jer sadrže završno stanje 19 NKA. D je stanje greške i može se izostaviti.

Eliminacija suvušnih (nedostižnih) stanja:

Dostižna stanja = {A, B, C, E, F, G}

Nema suvišnih stanja. Sledi minimizacija prema particionom algoritmu:

Početna particija ima sledeći izgled: P_0 = ({A, B, C, F}, {E, G})

Razmatranje prelaza za +: P_1 = ({A}, {B, C, F}, {E, G})

Razmatranje prelaza za -: P_2 = ({A}, {B, C, F}, {E, G})

Razmatranje prelaza za *: P_3 = ({A}, {B, C, F}, {E}, {G})

Razmatranje prelaza za d: P_4 = ({A}, {B, C}, {F}, {E}, {G})

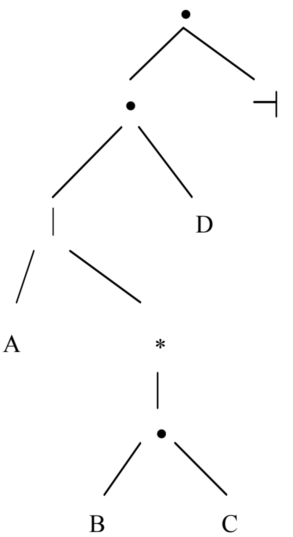
P_4 je konačna particija. Stanja B i C su ekvivalentna i zamenjujemo ih jednim stanjem X. Minimalan automat je:

\+ - * d

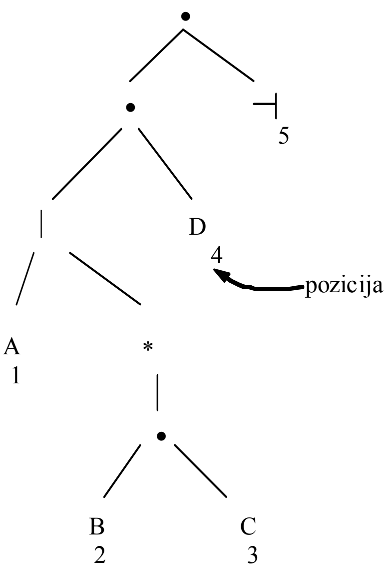
A	X	X		E	0
X				E	0
E			F	E	1
F				G	0
G				G	1

Zadatak 2. Metodom pozicija odrediti DKA za regularni izraz (A | (BC)*)D⊣ .

Sintaksno stablo za dati izraz:



- Pozicijama se označavaju listovi sintaksnog stabla (osim ε) redom pri obilasku stabla sa leva u desno.



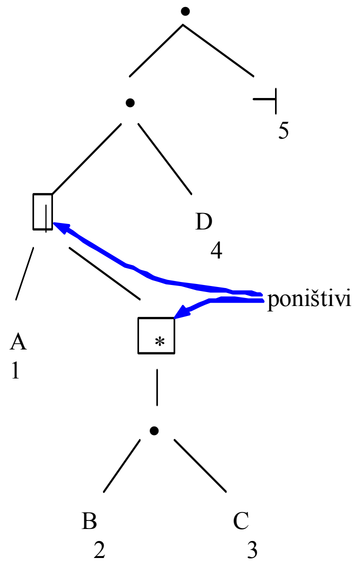
- Čvor sintaksnog stabla je poništiv ako podizraz sa korenom u datom čvoru može generisati praznu sekvencu. Računanje poništivosti:

1. Listovi nisu poništivi, osim ako su označeni sa ε.

2. Čvor konkatenacije • je poništiv ako su oba njegova naslednika poništiva.

3. Čvor unije | je poništiv ako je bar jedan od njegovih naslednika poništiv.

4. Čvor zvezdastog zatvaranja * je uvek poništiv.



- Prva pozicija za svaki čvor stabla predstavlja skup pozicija listova koji se mogu pojaviti na prvom mestu sekvence koju generiše podizraz sa korenom u datom čvoru. Računanje prvih pozicija:

1. $prva\_pozicija(\text{a}\bullet\text{b}) = \begin{cases} prva\_pozicija(\text{a}), & \text{a nije poništiv} \\ prva\_pozicija(\text{a}) \cup prva\_pozicija(\text{b}), & \text{a je poništiv} \end{cases}$

2. $prva\_pozicija(\text{a}|\text{b}) = prva\_pozicija(\text{a}) \cup prva\_pozicija(\text{b})$

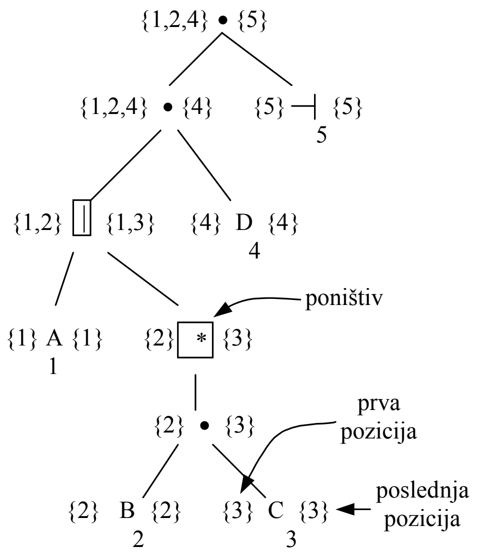
3. $prva\_pozicija(\text{a}^*)$ = prva_pozicija(a).

- Poslednja pozicija za svaki čvor stabla predstavlja skup pozicija listova koji se mogu pojaviti na poslednjem mestu sekvence koju generiše podizraz sa korenom u datom čvoru. Računanje poslednjih pozicija:

1. $poslednja\_pozicija(\text{a}\bullet\text{b}) = \begin{cases} poslednja\_poz.(\text{b}), & \text{b nije poništiv} \\ poslednja\_poz.(\text{b}) \cup poslednja\_poz.(\text{a}), & \text{b je poništiv} \end{cases}$

2. $poslednja\_pozicija(\text{a}|\text{b}) = poslednja\_pozicija(\text{a}) \cup poslednja\_pozicija(\text{b})$

3. $poslednja\_pozicija(\text{a}^*)$ = poslednja_pozicija(a).



- <u>Sledeća pozicija</u>(p) definiše se za svaku poziciju lista p kao skup pozicija koje mogu slediti posmatranu poziciju p u proizvoljnoj sekvenci znakova izvedenoj iz datog regularnog izraza. Pri računanju funkcije sledeća pozicija u sintaksnom stablu razmatraju se svi čvorovi tipa • i * na sledeći način:
- Čvor •: prva pozicija desnog naslednika je sledeća pozicija za sve iz poslednje pozicije levog naslednika.
- Čvor *: prva pozicija naslednika je sledeća pozicija za sve iz poslednje pozicije naslednika.

poz.	sled. pozicija
1	{4}
2	{3}
3	{2,4}
4	{5}
5	/

- Stanja automata ⇒ skupovi pozicija listova sintaksnog stabla.
- Startno stanje = *prva_pozicija* korena stabla
- Stanja prihvatanja su ona koja sadrže poziciju markera kraja.

Vrsta tabele prelaza, koja je prikazana na slici, za određeno stanje popunjava se razmatrajući pojedine pozicije iz skupa pridruženog tom stanju. Na primer, za stanje {1, 2, 4}:

- popunjavanje tabele prelaza: vrsta {1, 2, 4}, ulaz A

{1, 2 , 4}
↓ izdvojiti samo pozicije koje se odnose na simbol A
{1}
↓ za svaku poziciju proračunati sledeću poziciju pa unija
{4} – upisati u tabelu prelaza

	A	B	C	D	
→ {1, 2, 4}	{4}	{3}	{}	{5}	0
{4}	{}	{}	{}	{5}	0
{3}	{}	{}	{2, 4}	{}	0
{}	{}	{}	{}	{}	0
{5}	{}	{}	{}	{}	1
{2, 4}	{}	{3}	{}	{5}	0

DODATNI ZADACI IZ ZBIRKE KOJI MOGU DA SE URADE: 1.1.2; 1.1.3; 1.1.5; 1.2.3 (deo koji nije rađen na vežbama); 1.2.4; 1.2.5; 1.2.6; 1.3.4; 1.4.1; 1.4.4;