UNIVERZITET U BEOGRADU
MATEMATIČKI FAKULTET

O SINTEZI OPTIMALNIH AUTOMATA U NEKIM KLASAMA LAVIRINATA

- Magistarski rad -

AUTOR:
Zoran Maksimović

Pitanja: 1. Kolektiv automata održava n puta brže rešavanje lavirinta od jednog automata.
2. Opštenja i ocena složenosti
3. Lavirinti na površima (na primer na torusu).

Komisija:
1. Ž. Mijajlović
2. Š. Ušćumlić
3. G. Kilibarda
4. M. Živković

Beograd, 2001

Odbrana: 21.03.2001.

Sadržaj

Uvod

Automat $\mathfrak{A}$ je petorka (A, Q, B, φ, ψ), gde je A - konačna ulazna azbuka, B - konačna izlazna azbuka, Q - konačna azbuka stanja, $\varphi : Q \times A \rightarrow Q$, $\psi : Q \times A \rightarrow B$, redom, funkcije prelaska i izlaska automata. Ako izdvojimo stanje q_0 automata, onda takav automat nazivamo inicijalnim i obeležavamo sa $\mathfrak{A}_{q_0}$, a stanje q_0 nazivamo početnim stanjem.

Neka je dat graf $L = (V, E)$ i neka su Ω i Σ azbuke koje sadrže simbole kojim označavamo, redom, čvorove i grane grafa L, takve da $\Omega \backslash \Sigma = \Lambda$, gde je Λ prazan simbol. Ako su svi čvorovi i sve grane grafa L označeni slovima iz Ω i Σ tako da su različite grane koji izlaze iz istog čvora označene različitim slovima, tada takav graf nazivamo lavirintom. Lavirint sa izdvojenim čvorovima $v_0, v_1, \ldots, v_n$ nazivamo inicijalnim lavirintom i označavamo sa $L_{v_0, v_1, \ldots, v_n}$, a čvorove $v_0, v_1, \ldots, v_n$ nazivamo inicijalnim. Označimo sa $\mathcal{L}(\Omega, \Sigma)$ klasu svih lavirinata čiji su čvorovi obeleženi slovima iz Ω, a grane slovima iz Σ.

Automat $\mathfrak{A}$ je dopustiv za klasu lavirinata $\mathcal{L}(\Omega, \Sigma)$, ako se njegov ulazni alfabet sastoji od slova a oblika $(\omega, \{\sigma_1, \ldots, \sigma_n\})$, gde je $\omega \in \Omega$, $\{\sigma_1, \ldots, \sigma_n\} \subseteq$ $\subseteq \Sigma$, izlazni alfabet je $\Sigma \cup \{\kappa\}$, gde $\kappa \notin \Sigma$ pri čemu je $\psi(q, a) \in \{\sigma_1, \ldots, \sigma_n\} \cup \{\kappa\}$. Označićemo sve takve automate sa $\mathrm{At}(\Omega, \Sigma)$. Neka je $\mathfrak{A}_{q_0}$ inicijalni automat iz $At(\Omega, \Sigma)$, i neka je L_{v_0} inicijalni lavirint iz $\mathcal{L}(\Omega, \Sigma)$.

Uvedimo pojam funkcionisanja automata $\mathfrak{A}_{q_0}$ u lavirintu L_{v_0}. U početku je automat $\mathfrak{A}_{q_0}$ postavljen u čvor v_0 lavirinta L_{v_0}. Pretpostavimo da se automat nalazi u stanju q u čvoru v. Automat proverava označene grane koje izlaze iz tekućeg čvora. Ulaz automata u tom trenutku je uređeni par koga čine oznaka čvora i oznake grana koje izlaze iz čvora. U sledećem koraku ako je $\psi(q, a) \neq \kappa$ automat prelazi u čvor granom koja je označena $\psi(q, a)$, a ako je $\psi(q, a) = \kappa$, automat ostaje na istom mestu, a novo stanje je u oba slučaja $\varphi(q, a)$. Na ovaj način, automat se, korak po korak, kreće lavirintom, menjajući stanje i u svakom trenutku odlučujući kuda ide. Na taj način funkcionisanje $\mathfrak{A}_{q_0}$ u lavirintu L_{v_0}, možemo posmatrati kao ponašanje automata $\mathfrak{A}_{q_0}$ u lavirintu L_{v_0}. Niz uređenih parova koji karakterišu stanje u kome se nalazi automat možemo zapisati na sledeći način: $\pi(\mathfrak{A}_{q_0}; L_{v_0}) = (q_0, v_0), (q_1, v_1), \ldots$. Niz $\pi(\mathfrak{A}_{q_0}; L_{v_0})$ nazivamo ponašanje automata $\mathfrak{A}_{q_0}$ u lavirintu L_{v_0}, ako je v_{i+1} čvor lavirinta u koji automat stiže iz čvora v_i, menjajući pri tom stanje iz q_i u q_{i+1}. Ako za neki čvor

$v \in L_{v_0}$, postoji stanje q automata $\mathfrak{A}_{q_0}$ takvo da uređeni par (q, v) pripada nizu $\pi(V_{q_0}; L_{v_0})$, kazaćemo da automat $\mathfrak{A}_{q_0}$ obilazi čvor v lavirinta L_{v_0}. Izdvojimo skup svih čvorova lavirinta L_{v_0} koje automat $\mathfrak{A}_{q_0}$ obilazi i označimo ga sa $Int(\mathfrak{A}_{q_0}; L_{v_0})$. Ako $Int(\mathfrak{A}_{q_0}; L_{v_0})$ čine svi čvorovi lavirinta L_{v_0}, kazaćemo da automat $\mathfrak{A}_{q_0}$ obilazi lavirint L_{v_0}.

Zadatak koji treba rešiti je sinteza automata koji ima minimalni broj stanja koji prolazi zadati put u zadatom ravnom pravougaonom lavirintu (taj put može biti, recimo, od ulaza do izlaza iz lavirinta).

Kako je vremenska zavisnost sinteze optimalnog automata od dužine puta, tj. broja stanja optimalnog automata, eksponencijalna, uvode se heuristike kojima se ne garantuje optimalnost automata, ali je vreme sinteze polinomijalno.

U prvom delu su dati osnovni pojmovi i problemi o automatima, lavirintima i ponašanju automata u lavirintima.

Drugi deo je posvećen centralnoj temi - sintezi optimalnog automata koji prolazi zadati put. Prvo će biti uveden pojam maksimalnog automata koji prolazi zadati put. Potom će biti razmatran odnos maksimalnog automata i drugih automata koji prolaze zadati put. Nakon toga, biće uveden pojam grafa nesaglasnosti maksimalnog automata i razmotren odnos bojenja grafa i nesaglasnog razbijanja skupa stanja maksimalnog automata. Nakon toga, biće opisana tri postupka sinteze optimalnog automata, sinteza pomoću homomorfizma, sinteza pomoću rekurzije i sinteza pomoću kompatibilnih skupova. Pored postupaka sinteze optimalnih automata, data su i dva postupka optimizacije maksimalnog automata, tj. određivanje automata koji prolaze zadati put i imaju manji broj stanja od maksimalnog automata.

U trećem delu je izvršeno upoređivanje algoritama sinteza optimalnog automata i algoritama optimizacije. Kao najefikasniji su se pokazali algoritam sinteze optimalnog automata pomoću rekurzije, i postupak optimizacije sažimanjem susednih stanja.

Na kraju, izražavam zahvalnost prof. dr Šćepanu Ušćumliću i prof. dr. Goranu Kilibardi koji su mi savetima pomogli u izradi magistarske teze.

1 Apstraktni automati i lavirinti

U ovoj glavi biće navedeni osnovni pojmovi i problemi teorije apstraktnih automata, kao i osnovni pojmovi i rezultati o ponašanju automata u lavirintima.

1.1 Pojam apstraktnog automata. Načini zadavanja automata.

Ovaj odeljak je posvećen definiciji automata i načinima njihovog zadavanja. Radi toga, ovde se navode i najosnovniji pojmovi o grafovima [33].

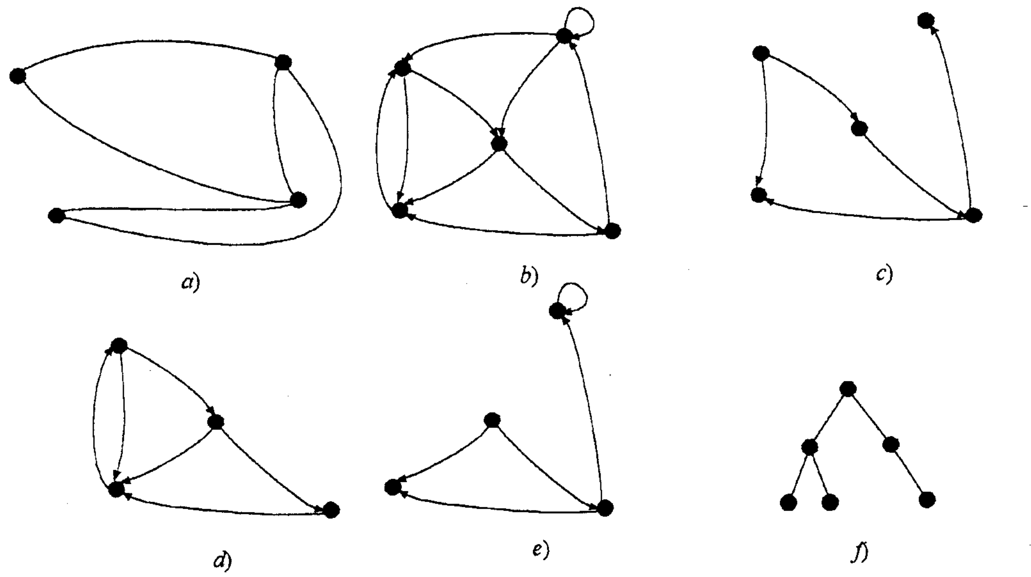
Označimo sa $\mathfrak{B}_k(X)$ sve podskupove skupa X koji imaju tačno k elemenata, i neka je $\mathfrak{B}_{\leq k}(X) = \bigcup_{i=1}^{k} \mathfrak{B}_i(X)$. *Graf* je uređeni par $G = (V, E)$, gde je V konačan skup, a $E \subseteq \mathfrak{B}_2(V)$ (sl. 1.a i 1.f). Elemente skupova V i E nazivamo redom *čvorovima* i *granama* grafa G. Ovako definisan graf nazivamo i *neorijentisanim grafom*. Ako je $E = \mathfrak{B}_2(V)$, kažemo da je graf *potpun*. Čvor $v \in V$ je *incidentan* grani $e \in E$ ako je $v \in e$. Za različite čvorove $u, v \in V$ kažemo da su *susedni* ako $\{u, v\} \in E$. Dve različite grane $e_1, e_2 \in E$ su *susedne* ako $e_1 \cap e_2 \neq \emptyset$. *Graf sa petljama* je uređeni par $G = (V, E)$, gde je V konačan skup, a $E \subseteq \mathfrak{B}_{\leq 2}(X)$; elemente skupova V i E nazivamo, redom, *čvorovima* i *granama*, a grane oblika $e = \{v\}$ $(v \in V)$ *petljama*. *Geometrijska reprezentacija u ravni* grafa G se dobija tako što se njegovim čvorovima pridruže različite tačke u ravni, a granama linije koje spajaju te tačke i to tako da se grani $e = \{u, v\}$ pridruži linija koja spaja tačke koje odgovaraju čvorovima u i v. Geometrijsku reprezentaciju grafa sa petljama dobijamo tako što grani oblika $e = \{v\}$ pridružujemo luk koji vodi od tačke koja odgovara čvoru v do iste te tačke.

Neka je V konačan skup. Označimo $\Delta_V = \{(v, v) \mid v \in V\}$. Uređeni par $G = (V, E)$, gde je $E \subseteq V^2 \backslash \Delta_V$ nazivamo *orijentisanim grafom* ili *orgrafom* (sl. 1.b). Elemente skupa V zovemo čvorovima, a elemente skupa E orijentisanim granama orgrafa G. Za granu $e = (u, v) \in E$ kažemo da *izlazi* iz čvora u, a da *ulazi* u čvor v, i da su čvorovi u i v *incidentni* grani e. Geometrijsku reprezentaciju orijentisanog grafa dobijamo slično reprezentaciji neorijentisanog grafa, samo što kraj grane obeležavamo strelicom.

Ako u definiciji orgrafa pretpostavimo da je $E \subseteq V^2$, dobijamo definiciju *orgrafa sa petljama*; petljama nazivamo grane oblika $e = (v, v)$, $v \in V$. *Graf* (*orijentisani*) *sa višestrukim granama i petljama* je uređeni par $G = (V, E)$, gde je V konačan skup, a E multiskup nad $\mathfrak{B}_{\leq 2}(V)$ (nad V^2).

Broj grana grafa G (orijentisanog ili neorijentisanog), incidentnih čvoru v, naziva se stepenom čvora v i označava sa $st(v)$. Graf G čiji su svi čvorovi stepena m nazivamo *regularnim* m*-grafom*.

Neka su date azbuke Ω, Σ i graf G. Ako je svakoj grani (čvoru) grafa G pridruženo neko slovo iz azbuke Ω (Σ), tada G nazivamo *grafom sa označenim granama* (*grafom sa označenim čvorovima*). Ako je G graf sa označenim granama i čvorovima, tada za graf G kažemo da je *označen*.



Slika 1.

Matrica susedstva $M = [m_{ij}]_{n \times n}$ grafa $G = (V, E)$ je kvadratna $0-1$ matrica reda n (njeni elementi su 0 ili 1) takva da je $m_{ij} = 1$ akko $\{v_i, v_j\} \in$ $\in E$. Na sličan način se mogu uvesti matrice susedstva za orijentisane grafove kao i grafove sa petljama.

Neka su $G_1 = (V_1, E_1)$ i $G_2 = (V_2, E_2)$ grafovi i neka je $V_1 \subseteq V_2$ i $E_1 \subseteq E_2$. Tada kažemo da je graf G_1 *podgraf* grafa G_2 (sl. 1.e). Graf $G' = (V, E')$ nazivamo *delimičnim grafom* grafa $G = (V, E)$ ako je $E' \subseteq E$ (sl. 1.c). Graf (A, E_A) je *podgraf grafa* $G = (V, E)$ *generisan skupom čvorova* A, ako $A \subseteq V$ i $E_A = E \cap \mathfrak{B}_2(A)$, i označavamo ga sa $\langle A \rangle_G$ (sl. 1.d) (ako je iz konteksta jasno u odnosu na koji graf G se posmatra ovakav podgraf, onda se umesto oznake $\langle A \rangle_G$ koristi oznaka $\langle A \rangle$).

Neka je $\mathfrak{P}$ neko svojstvo koje mogu imati grafovi a G neki graf. *Maksimalni podgraf* grafa G u odnosu na uočeno svojstvo $\mathfrak{P}$ je svaki generisani podgraf $\langle X \rangle$ grafa G za koji ne postoji podgraf $\langle Y \rangle$ datog grafa, koji poseduje takođe svojstvo $\mathfrak{P}$, takav da je $X \subset Y$ (X je pravi podskup skupa Y).

Niz grana π: $e_1, e_2, \dots, e_k$ $(k \geq 1)$ za koje postoji niz čvorova $v_0, v_1, \dots, v_k$ tako da $e_i = \{v_{i-1}, v_i\}, i = 1, \dots, k$, nazivamo *putem* u grafu G; za put π kažemo da povezuje čvor v_0 sa čvorom v_k, i da je v_0 početni, a v_k krajnji čvor puta π. Put u kome se svaka grana pojavljuje najviše jednom, naziva se *lancem*. Lanac u kome se svaki čvor javlja najviše jednom naziva se *prostim lancem*. Ako je $v_0 = v_k$, kaže se da je put *zatvoren*. Zatvoren lanac naziva se *ciklom*. Zatvoren lanac kod koga se poklapaju samo početni i krajnji čvor naziva se *prostim ciklom*. Ako za svaka dva čvora grafa G postoji put koji ih povezuje, kažemo da je graf G povezan. Povezan graf G bez ciklova naziva se *drvo* (sl. 1.f).

Analogno pojmu puta u grafu, može se definisati pojam puta u orijentisanom grafu. *Neorijentisan put* u orijentisanom grafu G je niz grana e_i, $i = 1, \dots k$, takvih da postoji niz čvorova $v_0, v_1, \dots, v_k$, tako da je $e_i = (v_{i-1}, v_i)$ ili $e_i = (v_i, v_{i-1}), i = 1, \dots, k$. *Put u orijentisanom grafu* G je neorijentisani put u grafu G, takav da $e_i = (v_{i-1}, v_i), i = 1, \dots, k$.

Sada ćemo formulisati definiciju apstraktnog konačnog automata i opisati načine njihovog zadavanja. Zatim ćemo uvesti pojam funkcionisanja automata.

Apstraktni konačni automat $\mathfrak{A}$ je petorka (A, Q, B, φ, ψ), gde su: $A = \{a_1, \dots, a_m\}$ ulazna azbuka, $Q = \{q_1, \dots, q_n\}$ skup stanja, $B = \{b_1, \dots b_p\}$ izlazna azbuka, $\varphi : Q \times A \to Q$ – funkcija prelaska, a $\psi : Q \times A \to B$ funkcija izlaska. Ako su φ i ψ parcijalno definisane funkcije, automat $\mathfrak{A}$ se naziva *parcijalnim apstraktnim automatom*.

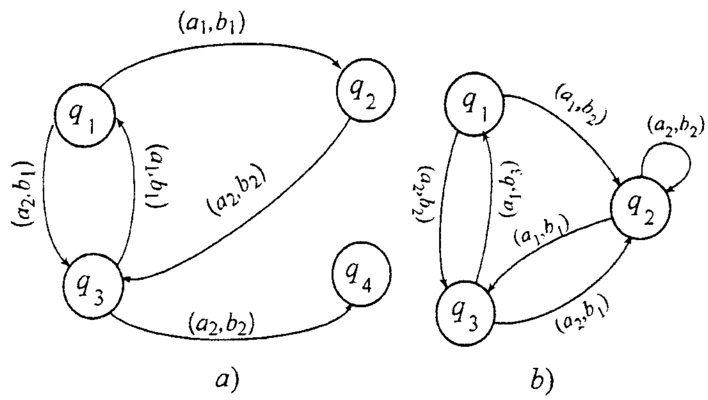
Ako su poznate ulazna i izlazna azbuka kao i skup stanja automata, zadavanje automata se svodi na zadavanje funkcija φ i ψ. Uobičajeni način zadavanja ovih funkcija je pomoću tablica oblika predstavljenog na slici 2.

Navedene tablice se mogu objediniti u jednu, tako da tablica u preseku i-te vrste i j-te kolone sadrži par $(\varphi(q_j, a_i)), \psi(q_j, a_i))$. Moguće je da funkcija prelaska bude definisana za određeni par (q, a), a funkcija izlaska ne. Neka su D_φ i D_ψ, redom, skupovi uređenih parova (q, a) za koje su funkcije φ i ψ definisane. Razmatraćemo takve automate, kod kojih je $D_\varphi = D_\psi$, i taj skup parova (q, a) za koje su funkcije φ i ψ definisane označićemo sa $D_{\mathfrak{A}}$.

$$\begin{array}{c} \\ a_1 \\ \vdots \\ a_i \\ \vdots \\ a_m \end{array}
\begin{array}{c} \begin{array}{ccccc} q_1 & \cdots & q_j & \cdots & q_n \end{array} \\ \left(\begin{array}{ccccc} & & & & \\ & & \vdots & & \\ & \cdots & \psi(q_j, a_i) & & \\ & & & & \\ & & & & \end{array}\right) \end{array}
\qquad
\begin{array}{c} \\ a_1 \\ \vdots \\ a_i \\ \vdots \\ a_m \end{array}
\begin{array}{c} \begin{array}{ccccc} q_1 & \cdots & q_j & \cdots & q_n \end{array} \\ \left(\begin{array}{ccccc} & & & & \\ & & \vdots & & \\ & \cdots & \varphi(q_j, a_i) & & \\ & & & & \\ & & & & \end{array}\right) \end{array}$$

Slika 2.

Automat se zadaje i tzv. *Murovim dijagramom*. Murov dijagram se konstruiše tako što se konstruiše orijentisani graf sa višestrukim granama i petljama, koji ima čvorova koliko automat ima stanja. Čvorovi grafa označavaju se slovima iz Q tako da različiti čvorovi budu obeleženi različitim slovima. Neka je q proizvoljni čvor i neka je $(q, a) \in D_{\mathfrak{A}}$. Tada se konstruiše orijentisana grana koja vodi od čvora obeleženog sa q u čvor koji je obeležen slovom $\varphi(q, a)$, i označava sa (a, b), gde je $b = \psi(q, a)$. Navedenim postupkom konstruišemo označen orijentisan graf koji ima onoliko čvorova koliko automat ima stanja, pri čemu su čvorovi obeleženi slovima iz Q, a grane uređenim parovima iz skupa $A \times B$. Ako je $D_{\mathfrak{A}} = Q \times A$, tada je odgovarajući Murov dijagram označen regularan m-graf, gde je m broj slova ulazne azbuke A. Na slici 3 dati su primeri Murovih dijagrama.



Slika 3.

Automat $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ u kojem je izdvojeno tzv. *početno stanje* q_0 nazivamo *inicijalnim apstraktnim konačnim automatom* i obeležavamo sa $\mathfrak{A}_{q_0}$. Dakle, incijalni automat $\mathfrak{A}_{q_0}$ je šestorka $(A, Q, B, \varphi, \psi, q_0)$.

Automat $\mathfrak{A}'= (A, Q', B, \varphi', \psi')$ je *podautomat* automata $\mathfrak{A}= (A, Q, B, \varphi, \psi)$, ako je $Q' \subseteq Q$, i za svaki par $(q, a) \in Q' \times A$, važi $\varphi'(q, a) = \varphi(q, a)$ i $\psi'(q, a) = = \psi(q, a)$. Murov dijagram podautomata $\mathfrak{A}'$ automata $\mathfrak{A}$ je, u stvari, podgraf generisan skupom čvorova Q' Murovog dijagrama automata $\mathfrak{A}$. Parcijalni automat $\mathfrak{A}'= (A, Q', B, \varphi', \psi')$ je *podautomat* parcijalnog automata $\mathfrak{A}= (A, Q, B, \varphi, \psi)$, ako je $Q' \subseteq Q$, $D_{\mathfrak{A}'} \subseteq D_{\mathfrak{A}}$ i za svaki par $(q, a) \in D'_{\mathfrak{A}'}$ $\varphi'(q, a) = \varphi(q, a)$ i $\psi'(q, a) = = \psi(q, a)$. Murov dijagram podautomata $\mathfrak{A}'$ parcijalnog automata $\mathfrak{A}$ je, u stvari, podgraf Murovog dijagrama parcijalnog automata $\mathfrak{A}$.

Početni interval dužine n ($n \in \mathrm{N}$) skupa prirodnih brojeva je skup $\overline{n} = \{1, 2, \ldots, n\}$; početni interval dužine 0 je prazan skup. Konačna niska elemenata nekog skupa je svako preslikavanje nekog početnog intervala skupa prirodnih brojeva u taj skup. Kažemo da je konačna niska prazna, ako odgovara početnom intervalu dužine 0. Dužina konačne niske je broj elemenata sadržan u odgovarajućem početnom intervalu.

Uzmimo neprazan konačan skup C i nazovimo ga azbukom; elemente skupa C u tom slučaju zovemo slovima, a proizvoljnu konačnu nisku slova α skupa C zovemo *reč* u azbuci (ili nad azbukom) C. *Prazna reč* je reč koja odgovara praznoj niski. Reč α, $\alpha : \overline{n} \rightarrow C$ $(n \geq 1)$, se obično zapisuju u obliku $\alpha = a_1 a_2 \ldots a_n$, gde je $a_i = \alpha(i)$. Prazna reč se označava sa Λ. Sa $|\alpha|$ označimo dužinu reči α. Ako je $|\alpha| = n$, tada često umesto α pišemo $\alpha^{(n)}$. Skup svih reči nad azbukom C označavamo sa C^*, skup svih nepraznih reči sa C^+, a skup svih reči dužine l sa C^l. Za reči skupa C^* je definisana operacija konkatenacije: rezultat konkatenacije dve reči $b_1 b_2 \ldots b_m$ i $c_1 c_2 \ldots c_n$ je reč $d_1 \ldots d_m d_{m+1} \ldots d_{m+n}$, gde je $d_1 = b_1, \ldots, d_m = b_m, d_{m+1} = c_1, \ldots, d_{m+n} = c_n$. Ako se reč γ dobija konkatenacijom reči α i β, pišemo $\gamma = \alpha\beta$. Ako su γ, δ i δ' reči, pri čemu je $\gamma = \delta\alpha\delta'$ za neku reč α, tada kažemo da je δ *početak*, a δ' *kraj* reči γ. Ako je $\delta \neq \gamma$, tada početak δ reči γ nazivamo *pravim početkom*; ako je $\delta' \neq \gamma$, tada kraj δ' reči γ nazivamo *pravim krajem*. Početak reči γ dužine l označavamo sa $\gamma]_l$ $(\gamma]_0 = \Lambda)$, a kraj reči γ dužine l sa $\gamma[_l$ $(\gamma[_0= \Lambda)$.

Ulaznim rečima, izlaznim rečima i *rečima stanja* automata $\mathfrak{A}$, $\mathfrak{A} = (A, Q, B, \varphi, \psi)$, nazivamo redom reči nad azbukama A, B i Q.

Beskonačne nizove slova azbuke C, tj. preslikavanja oblika $\alpha : N \rightarrow C$, nazivamo *ω-rečima* nad azbukom C. Skup svih ω-reči nad azbukom C označavamo sa C^∞. ω–reči azbuka A, Q, B automata $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ nazivamo redom *ulaznim ω–rečima, ω–rečima stanja* i *izlaznim ω–rečima* automata $\mathfrak{A}$.

Funkcije prelaska i izlaska automata $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ proširimo na $Q \times A^*$ na sledeći način:

$$\varphi(q, \Lambda) = q,\ \varphi(q, \alpha a) = \varphi(\varphi(q, \alpha), a) \text{ gde } q \in Q, \alpha \in A^*,\ a \in A;$$
$$\psi(q, \Lambda) = \Lambda,\ \psi(q, \alpha a) = \psi(\varphi(q, \alpha), a) \text{ gde } q \in Q,\ \alpha \in A^*,\ a \in A.$$

Skup uređenih parova $(q, \alpha) \in Q \times A^*$ za koje su funkcije φ i ψ definisane, označićemo sa $D^*_{\mathfrak{A}}$.

Ako je u C data relacija linearnog poretka $\leq$, to na skupu C^* možemo definisati linearni poredak (alfabetski poredak), označavaćemo ga takođe sa $\leq$, na sledeći način. Pre svega pretpostavimo da je $\Lambda \leq \alpha$ za svako $\alpha \in C^*$. Neka su $\alpha, \beta \in C^*$ dve proizvoljne reči. Reći ćemo da je $\alpha \leq \beta$ ako: ili postoji $\gamma \in C^*$ takvo da je $\alpha = \gamma\alpha'$, $\beta = \gamma\beta'$, $|\alpha'|$, $|\beta'| \geq 1$ i $\alpha']_1 \leq \beta']_1$, ili postoji $\delta \in C^*$ takvo da je $\beta = \alpha\delta$.

Automat $\mathfrak{A} = (A, Q, B, \varphi, \psi)$, nalazeći se u stanju q, ulaznu reč $\alpha^{(m)}$ transformiše u reč stanja i izlaznu reč koje označavamo sa $\overline{\varphi}(q, \alpha^{(m)})$ i $\overline{\psi}(q, \alpha^{(m)})$ na sledeći način:

$$\overline{\varphi}(q, \alpha^{(m)}) = \varphi(q, \alpha^{(m)}]_0)\varphi(q, \alpha^{(m)}]_1) \ldots \varphi(q, \alpha^{(m)}]_m)$$
$$\overline{\psi}(q, \alpha^{(m)}) = \psi(q, \alpha^{(m)}]_1)\psi(q, \alpha^{(m)}]_2) \ldots \psi(q, \alpha^{(m)}]_m).$$

Transformisanje ulaznih reči $\alpha^{(m)}$ automatom $\mathfrak{A}$, može se opisati relacijom:

$$F_{\mathfrak{A}} = \{(\alpha^{(m)}, \overline{\varphi}(q, \alpha^{(m)}), \overline{\psi}(q, \alpha^{(m)}))|\alpha^{(m)} \in A^*, q \in Q, (q, \alpha^{(m)}) \in D^*_{\mathfrak{A}}, m = 1, 2 \ldots\}$$

Relaciju $F_{\mathfrak{A}}$ nazivamo *funkcionisanjem automata* $\mathfrak{A}$.

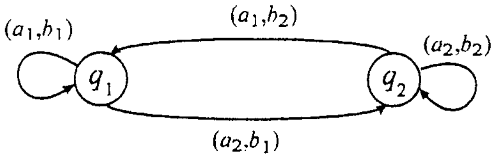
Navedimo primer automata a zatim konstruišimo njegov Murov dijagram i opišimo funkcionisanje.

Primer: Neka je $\mathfrak{A} = (\{a_1, a_2\}, \{q_1, q_2\}, \{b_1, b_2\}, \varphi, \psi)$, gde su funkcije φ i ψ zadate tablicom na sl. 4.

	q_1	q_2
a_1	(q_1, b_1)	(q_1, b_2)
a_2	(q_2, b_1)	(q_2, b_2)

Slika 4.

Murov dijagram automata $\mathfrak{A}$ predstavljen je na slici 5.



Slika 5.

Za $q_i \in Q, i \in \{1,2\}$ i proizvoljnu reč $\alpha = a_{i_1}a_{i_2}\dots a_{i_m}$ iz A^*, imamo $\overline{\varphi}(q_i, a_{i_1}a_{i_2}\dots a_{i_m}) = q_iq_{i_1}q_{i_2}\dots q_{i_m}, \overline{\psi}(q_i, a_{i_1}a_{i_2}\dots a_{i_m}) = b_ib_{i_1}b_{i_2}\dots b_{i_{m-1}}$. Funkcionisanje automata $\mathfrak{A}$ modelira element zadrške, u stvari, električnu šemu koja se koristi u elektronskim računskim mašinama za zadržavanje impulsa signala.

Ako je $\mathfrak{A}_{q_0}$ inicijalni automat tada se funkcionisanje opisuje relacijom

$$F_{\mathfrak{A}_{q_0}} = \{(\alpha, \overline{\varphi}(q_0,\alpha), \overline{\psi}(q_0,\alpha)) | \alpha \in A^*, \overline{\varphi} \text{ i } \overline{\psi} \text{ su definisani za } (q_0,\alpha)\}$$

Stanje q inicijalnog automata $\mathfrak{A}_{q_0} = (A, Q, B, \varphi, \psi, q_0)$ naziva se *dostižnim* iz početnog stanja q_0 ako postoji trojka $(\alpha, \gamma, \beta) \in F_{\mathfrak{A}_{q_0}}$ takva da reč γ sadrži stanje q.

Inicijalni automati i njihovo funkcionisanje zadaje se i tzv. *kanoničkim jednačinama.* Nije teško uočiti da je funkcionisanje $F_{\mathfrak{A}_{q_0}}$ inicijalnog automata $\mathfrak{A}_{q_0} = (A, Q, B, \varphi, \psi, q_0)$, skup trojki reči (α, γ, β), takvih, da za neko $m \in \in \{1, 2, \dots\}$ važi

$$\alpha = a(1)a(2)\dots a(m); \gamma = q(1)q(2)\dots q(m+1); \beta = b(1)b(2)\dots b(m)$$

i pri tome su zadovoljene relacije:

$$\begin{cases} q(1) = q_0 \\ q(t+1) = \varphi(q(t), a(t)) \\ b(t) = \psi(q(t), a(t)) \end{cases}$$

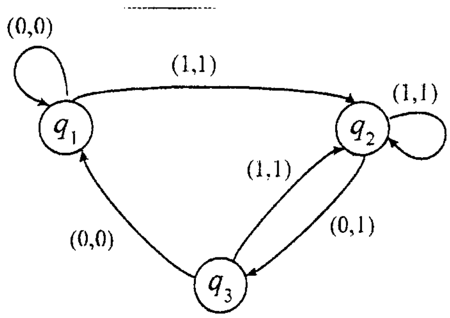
$(t = 1, \dots, m)$ koje nazivamo *sistemom kanoničkih jednačina automata* $\mathfrak{A}_{q_0}$.

1.2 Osnovni tipovi ponašanja automata

Jedna od najvažnijih karakteristika apstraktnih automata je takozvano ponašanje automata. Ovde ćemo navesti tri osnovna tipa ponašanja automata.

Neka je $\mathfrak{A}_{q_0} = (A, Q, B, \varphi, \psi, q_0)$ inicijalni automat. Slovarna funkcija $f(\alpha^{(m)}) = \overline{\psi}(q_0, \alpha^{(m)})$, koja preslikava skup A^* u skup B^*, predstavlja *ponašanje automata* $\mathfrak{A}_{q_0}$ *kao pretvarača.* Funkciju $f(\alpha^{(m)})$ nazivamo *konačno automatnom funkcijom*, koja se realizuje (ili izračunava) automatom $\mathfrak{A}_{q_0}$. Inicijalni automat razmatran sa ovog stanovišta, naziva se *automat pretvarač.*

Inicijalni automat $\mathfrak{A}_{q_1}$, zadat Murovim dijagramom predstavljenim na slici 6. realizuje preslikavanje ulazne reči α u izlaznu reč β tako što se u njoj nule koje se nalaze posle jedinica zamenjuju jedinicama. Opisano preslikavanje je osnovna karakteristika zadatog automata, koji se razmatra kao pretvarač reči njegove ulazne azbuke $A = \{0,1\}$ u reči izlazne azbuke $B = \{0,1\}$.



Slika 6.

Neka je dat inicijalni automat $\mathfrak{A}_{q_0} = (A, Q, B, \varphi, \psi, q_0)$ i skup $B' \subseteq B$. Uočimo skup $L_{B'}$ reči azbuke A definisan na sledeći način:

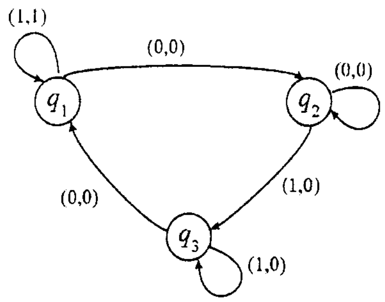
$$L_{B'} = \{\alpha^{(m)} \in A^* | \psi(q_0, \alpha^{(m)}) \in B'\}.$$

Skup $L_{B'}$ nazivamo *ponašanje inicijalnog automata* $\mathfrak{A}_{q_0}$ kao *akceptora u odnosu na skup* B' ili *događajem predstavljivim inicijalnim automatom* $\mathfrak{A}_{q_0}$ *pomoću skupa* B'. Na taj način, ponašanje automata $\mathfrak{A}_{q_0}$ kao akceptora predstavlja skup ulaznih reči na koje automat $\mathfrak{A}_{q_0}$ reaguje pojavom na njegovom izlazu signala iz izdvojenog skupa B'. U ovom slučaju, automat $\mathfrak{A}_{q_0}$ tretiramo kao uređaj koji raspoznaje ove ili one klase ulaznih reči. Inicijalni automat, razmatran sa ovog aspekta, naziva se *automat akceptor*. Kaže se da zadati automat *prepoznaje* ili *akceptira* tj. prihvata opisani skup $L_{B'}$ reči ulazne azbuke.

Nije teško uočiti da se skup M ulaznih reči automata $\mathfrak{A}_{q_1}$, zadatog Murovim dijagramom predstavljenim na slici 7, za koje automat posle njihovog prihvatanja na izlazu daje jedinicu, sastoji od reči oblika:

$$\underbrace{1\ldots1}_{k_1}\underbrace{0\ldots0}_{n_1}\underbrace{1\ldots1}_{m_1}\underbrace{1\ldots1}_{k_2}\underbrace{0\ldots0}_{n_2}\underbrace{1\ldots1}_{m_2}0\ldots\underbrace{1\ldots1}_{k_s}\underbrace{0\ldots0}_{n_s}\underbrace{1\ldots1}_{m_s}0\underbrace{1\ldots1}_{k_{s+1}}1$$

gde je $s \geq 0$, $k_i \geq 0$, $n_i \geq 1$, $m_i \geq 1$, $i = 1, \ldots, s+1$ (u slučaju $s = 0$ reč je oblika $\underbrace{1...1}_{k_{s+1}}1$).

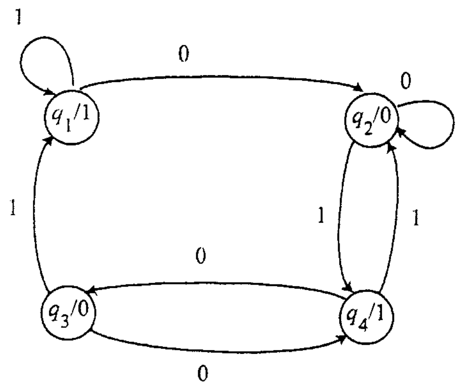


Slika 7.

Pre nego nego što navedemo još jedan tip ponašanja automata za koji se automat naziva upravljačkim automatom, uvedimo definiciju Murovog automata.

Automat $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ naziva se *Murovim automatom* ako funkcija izlaska $\psi(q, a)$ ne zavisi od promenljive a, tj. ako je $\psi(q, a) = \psi(q)$. Dijagram Murovog automata možemo pojednostaviti tako što ćemo čvorove njegovog dijagrama obeležavati slovima skupa stanja i odgovarajućim slovima izlazne azbuke, a njegove grane obeležavati samo slovima ulazne azbuke.

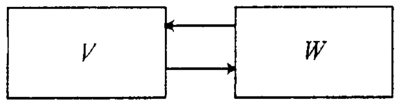
Na slici 8 predstavljen je dijagram Murovog automata $\mathfrak{A}(\{0,1\}, \{q_1, q_2, q_3, q_4\}, \{0,1\}, \varphi, \psi)$.



Slika 8.

Proces upravljanja može se šematski predstaviti kao na slici 9. gde je V *upravljački uređaj*, a W *objekat upravljanja*. Uzmimo da su V i W konačni automati koji funkcionišu u diskretnom vremenu $t = 1, 2, \ldots$ i neka je W Murov automat. Za upravljački automat V mogu se postaviti sledeći zadaci: prevođenje objekta upravljanja W, koga nazivamo upravljivim automatom, u neko zadato stanje, zadržavanje automata W unutar nekog zadatog skupa stanja; optimizacija učestanosti nalaženja automata W u stanjima nekog zadatog skupa stanja, itd.

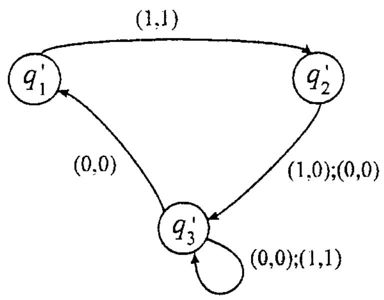
Zavisno od postavljenog cilja za upravljivi automat realizuje se problem njegove konstrukcije poznat kao zadatak sinteze automata. Napomenimo da je posebno bitan zadatak sinteze univerzalnih upravljačkih automata za neku klasu upravljivih automata.



Slika 9.

Definišimo, sada, pojam ponašanja inicijalnog automata kao upravljačkog uređaja.

Ponašanje inicijalnog automata $\mathfrak{A}_{q_0} = (A, Q, B, \varphi, \psi, q_0)$ *kao upravljača* naziva se par $(\mathfrak{A}'_{q'_0}, \pi)$, gde je $\mathfrak{A}'_{q'_0} = (B, Q', A, \varphi', \psi', q'_0)$ inicijalni konačni Murov automat, π niz četvorki $(q_0, q'_0, a_0, b_0), (q_1, q'_1, a_1, b_1), \ldots$, za koje je $a_0 = \psi'(q'_0)$, $b_0 = \psi(q_0, a_0)$, $q_{i+1} = \varphi(q_i, a_i)$, $q'_{i+1} = \varphi'(q'_i, b_i)$, $a_{i+1} = \psi'(q'_{i+1})$, $b_{i+1} = \psi(q_{i+1}, a_{i+1})$. Niz π opisuje rad sistema, koji se dobija kao rezultat izjednačavanja izlaza automata $\mathfrak{A}_{q_0}$ sa ulazom automata $\mathfrak{A}'_{q'_0}$, a ulaza automata $\mathfrak{A}_{q_0}$ sa izlazom $\mathfrak{A}'_{q'_0}$. Za proizvoljno $i = 0, 1, \ldots$, q_i je stanje automata $\mathfrak{A}_{q_0}$ u trenutku $t = i$; q'_i je stanje automata $\mathfrak{A}'_{q'_0}$, u trenutku $t = i$, a_i ulazni signal, a b_i izlazni signal automata $\mathfrak{A}_{q_0}$ u trenutku $t = i$.



Slika 10.

Ponašanje automata kao upravljačkih uređaja u tesnoj je vezi sa ponašanjem automata u lavirintima.

Na slici 10. predstavljen je dijagram upravljačkog automata $\mathfrak{A}_{q'_1}$ koji automat $\mathfrak{A}$, čiji je Murov dijagram predstavljen na slici 8, i čije je početno stanje proizvoljno, posle konačnog broja koraka prevodi u stanje q_2 i dalje neograničeno dugo ostaje u tom stanju.

1.3 Homomorfizam i izomorfizam automata

Neka su dati automati $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ i $\mathfrak{A}' = (A', Q', B', \varphi', \psi')$. Trojku preslikavanja (f, g, h), gde je $f : A \to A'$, $g : Q \to Q'$ i $h : B \to B'$, nazivamo *homomorfizmom automata* $\mathfrak{A}$ *u automat* $\mathfrak{A}'$, ako je

$$g(\varphi(q, a)) = \varphi'(g(q), f(a)), \qquad h(\psi(q, a)) = \psi'(g(q), f(a))$$

za svako $(q, a) \in D_{\mathfrak{A}}$. Ako su preslikavanja f, g i h surjektivna tada kažemo da je trojka (f, g, h) *homomorfizam automata* $\mathfrak{A}$ *na automat* $\mathfrak{A}'$. Ako postoji homomorfizam automata $\mathfrak{A}$ na automat $\mathfrak{A}'$, tada za $\mathfrak{A}'$ kažemo da je *homomorfna slika* automata $\mathfrak{A}$. Ako su f, g i h bijekcije, onda trojku (f, g, h) nazivamo *izomorfizmom automata* $\mathfrak{A}$ *na automat* $\mathfrak{A}'$. Ako postoji izomorfizam automata $\mathfrak{A}$ na automat $\mathfrak{A}'$, kažemo da su automati $\mathfrak{A}$ i $\mathfrak{A}'$ *izomorfni.*

Uočimo specijalni slučaj homomorfizma kada su ulazne i izlazne azbuke automata jednake i kada je $f = i_A$ i $h = i_B$ ($i_X(x) = x$ za svako $x \in X$). Tada homomorfizam određuje samo preslikavanje $g : Q \to Q'$, i zato ćemo ubuduće homomorfizam (i_A, g, i_B) obeležavati sa g, i nazivati ga *g-homomorfizam.*

Za proizvoljan g-homomorfizam g automata $\mathfrak{A}=(A, Q, B, \varphi, \psi)$ u automat $\mathfrak{A}'=(A, Q', B, \varphi', \psi')$ označimo sa g^* preslikavanje $g^* : Q^* \to Q'^*$ takvo da je $g^*(q_1q_2 \ldots q_k) = g(q_1)g(q_2) \ldots g(q_k)$ za proizvoljnu reč $q_1q_2 \ldots q_k \in Q^*$. Za preslikavanje g^* važi sledeće tvrđenje.

Lema 1. *Neka je g proizvoljni g-homomorfizam automata* $\mathfrak{A}= (A, Q, B, \varphi, \psi)$ *u automat* $\mathfrak{A}' = (A, Q', B, \varphi', \psi')$. *Tada je*

$$g^*(\overline{\varphi}(q, \alpha^{(m)})) = \overline{\varphi}'(g(q), \alpha^{(m)}) \quad i \quad \overline{\psi}(q, \alpha^{(m)}) = \overline{\psi}'(g(q), \alpha^{(m)})$$

za svako $(q, \alpha^{(m)}) \in D_{\mathfrak{A}}$.

Dokaz: Tvrđenje ćemo dokazati matematičkom indukcijom po dužini m reči $\alpha^{(m)}$. Ako je $m = 1$, tada je $\alpha^{(1)} = a$ za neko $a \in A$. Kako je

$$g^*(\overline{\varphi}(q, a)) = g^*(q\varphi(q, a)) = g(q)g(\varphi(q, a)) = g(q)\varphi'(g(q), a) = \overline{\varphi}'(g(q), a),$$

to je prva jednakost tačna za $m = 1$. Pretpostavimo da je ono tačna za proizvoljno m, tj. da je $g^*(\overline{\varphi}(q, \alpha^{(m)})) = \overline{\varphi}'(g(q), \alpha^{(m)})$ i dokažimo da je tačna za $m + 1$, tj. da je $q^*(\overline{\varphi}(q, \alpha^{(m+1)})) = \overline{\varphi}'(g(q), \alpha^{(m+1)})$. Iz $g^*(\overline{\varphi}(q, \alpha^{(m)})) = \overline{\varphi}'(g(q), \alpha^{(m)})$ sledi da je $g(\varphi(q, \alpha^{(m)})) = \varphi'(g(q), \alpha^{(m)})$. Dalje imamo da je

$$\overline{\varphi}(q, \alpha^{(m+1)}) = \overline{\varphi}(q, \alpha^{(m)}a) = \overline{\varphi}(q, \alpha^{(m)})\varphi(\varphi(q, \alpha^{(m)}), a)$$

i

$$\overline{\varphi}'(g(q),\alpha^{(m+1)}) = \overline{\varphi}'(g(q),\alpha^{(m)}a) = \overline{\varphi}'(g(q),\alpha^{(m)})\varphi'(\varphi'(g(q),\alpha^{(m)}),a).$$

Otuda sledi da je

$$g^*(\overline{\varphi}(q,\alpha^{(m+1)})) = g^*(\overline{\varphi}(q,\alpha^{(m)})\varphi(\varphi(q,\alpha^{(m)}),a)) = g^*(\overline{\varphi}(q,\alpha^{(m)}))\cdot$$
$$\cdot g(\varphi(\varphi(q,\alpha^{(m)}),a)) = \overline{\varphi}'(g(q),\alpha^{(m)})\varphi'(\varphi'(g(q),\alpha^{(m)}),a)) = \overline{\varphi}'(g(q),\alpha^{(m+1)}),$$

čime je prva jednakost dokazana. Dokaz druge jednakosti je analogan.

Neka je $\{X_\iota \mid \iota \in I\}$ indeksirana familija skupova. Tada, za svako $\iota \in I$ sa $\mathbf{p}_\iota$ označimo funkciju projekcije proizvoda $\prod_{\iota\in I} X_\iota$ na ι-ti član X_ι ovog proizvoda.

Neka je $F_{\mathfrak{A}} \subseteq A^* \times Q^* \times B^*$ funkcionisanje automata $\mathfrak{A} = (A,Q,B,\varphi,\psi)$. Uvedimo skupove $P_{\mathfrak{A}} = \{(\alpha,\beta) : (\exists u \in Q^*)\,(\alpha,u,\beta) \in F_{\mathfrak{A}}\}$, $U_{\mathfrak{A}} = \mathbf{p}_1(F_{\mathfrak{A}}) = \mathbf{p}_1(P_{\mathfrak{A}})$ i $I_{\mathfrak{A}} = \mathbf{p}_3(F_{\mathfrak{A}}) = \mathbf{p}_2(P_{\mathfrak{A}})$.

Primetimo da iz leme 1 sledi da ako postoji g-homomorfizam automata $\mathfrak{A}$ u automat $\mathfrak{A}'$, onda $P_{\mathfrak{A}} \subseteq P_{\mathfrak{A}'}$.

Preslikavanje $g : Q' \to Q''$ je *g-homomorfizam inicijalnog automata* $\mathfrak{A}'_{q'_0} = (A,Q',B,\varphi',\psi',q'_0)$ *(na) u inicijalni automat* $\mathfrak{A}''_{q''_0} = (A,Q'',B,\varphi'',\psi'',q''_0)$, ako je (i_A,g,i_B) homomorfizam automata $\mathfrak{A}'$ (na) u automat $\mathfrak{A}''$ i ako je $g(q'_0) = q''_0$ (automati $\mathfrak{A}'$ i $\mathfrak{A}''$ su odgovarajući neinicijalni automati za automate $\mathfrak{A}'_{q'_0}$ i $\mathfrak{A}''_{q'_0}$).

Za g-homomorfizam inicijalnih automata važe sledeća tvrđenja:

Tvrđenje 1. *Ako postoji g-homomorfizam inicijalnog automata $\mathfrak{A}_{q_0} = (A,Q,B,\varphi,\psi,q_0)$ u inicijalni automat $\mathfrak{A}'_{q'_0} = (A,Q',B,\varphi',\psi',q'_0)$ i ako je $\alpha \in U_{\mathfrak{A}_{q_0}}$ onda je $\alpha \in U_{\mathfrak{A}'_{q'_0}}$, i $\overline{\psi}(q_0,\alpha) = \overline{\psi}'(q'_0,\alpha)$.*

Dokaz navedenog tvrđenja sledi iz leme 1.

Tvrđenje 2. *Neka su parcijalni inicijalni automati $\mathfrak{A}'_{q'_0} = (A,Q',B,\varphi',\psi',q')$ i $\mathfrak{A}''_{q''_0} = (A,Q'',B,\varphi'',\psi'',q'')$ g-homomorfni. Tada je $P_{\mathfrak{A}'_{q'_0}} \subseteq P_{\mathfrak{A}''_{q''_0}}$ (i, sledstveno, $U_{\mathfrak{A}'_{q'_0}} \subseteq U_{\mathfrak{A}''_{q''_0}}$ i $I_{\mathfrak{A}'_{q'_0}} \subseteq I_{\mathfrak{A}''_{q''_0}}$).*

Dokaz: Ako je $(\alpha,\beta) \in P_{\mathfrak{A}'_{q'_0}}$ tada je $(\alpha,\beta) = (\alpha,\overline{\psi}'(q'_0,\alpha))$. Prema tvrđenju 1, $\overline{\psi}'(q'_0,\alpha) = \overline{\psi}''(q''_0,\alpha)$, pa je $(\alpha,\beta) = (\alpha,\overline{\psi}''(q_0,\alpha)) \in P_{\mathfrak{A}''_{q''_0}}$, čime je tvrđenje dokazano.

1.4 Analiza i sinteza automata

Načini opisivanja ponašanja automata se mogu podeliti na dve grupe, na grupu jezika, tj. algebarsko logičkog opisivanja, i grupu načina koji opisuju unutrašnje procese realnih automata, kao što su npr. Murovi dijagrami, izvornici i sl. Prelazak sa prvog na drugi način zadavanja, nazivamo sintezom automata, a od drugog na prvi način zadavanja, analizom automata.

U odeljku 1.2 rečeno je da se automatom prihvatačem zadaju, tj. prepoznaju skupovi reči ulazne azbuke koji se nazivaju događajima. Događaji koji se predstavljaju automatima mogu se, takođe, zadavati pomoću tzv. izvornika.

Neka je zadat inicijalni automat $\mathfrak{A}_q = (A, Q, B, \varphi, \psi, q)$ i podskup $B' \subseteq B$. Reći ćemo da automat $\mathfrak{A}_q$ predstavlja podskup $E, E \subseteq A^*$, pomoću izdvojenog podskupa B', ako je za proizvoljnu reč α iz A^* pripadnost α podskupu E ekvivalentna relaciji $\psi(q, \alpha) \in B'$. Automat $\mathfrak{A}_q$ sa izdvojenim podskupom B' označićemo sa $(\mathfrak{A}_q, B')$ ili sa $\mathfrak{A}_q = (A, Q, B, \varphi, \psi, q, B')$. Podskupove skupa A^* ćemo ubuduće nazivati događajima.

Izvornik nad azbukom A je konačni orijentisani graf G čije je svako rebro obeleženo slovom azbuke A, u kojem je izdvojen tzv. početni čvor v_0 i neprazan podskup F završnih čvorova, pri čemu su ispunjeni uslovi:

1) različita rebra koja izlaze iz jednog istog čvora obeležena su različitim slovima azbuke A.

2) za svaki čvor v postoji put π_1 koji vodi iz v_0 u v i put π_2 koji vodi iz v u neki završni čvor.

Broj čvorova izvornika G nazivamo njegovom *složenošću* i označavamo sa $\|G\|$. Svaki izvornik nad azbukom A određuje događaj $|G|$-skup reči $p = a_{i_1} \dots a_{i_k}$ u azbuci A, za koje postoji put $\pi = v_0 \rho_{i_1} v_{i_1} \dots \rho_{i_k} v_{i_k}$ u izvorniku G, koji vodi iz v_0 u završni čvor v_{i_k} čije je rebro $\rho_{i_j} (1 \leq j \leq k)$ obeleženo slovom a_{i_j}. Govorićemo da *reč* $a_{i_1} \dots a_{i_k}$ *odgovara putu* π.

Neka je $\mathfrak{A} = (A, Q, B, \varphi, \psi, q_0)$ automat i neka je $L_{(\mathfrak{A}, B')} \neq \emptyset$ događaj predstavljen podskupom B' skupa slova izlazne azbuke B.

Razmotrimo izvornik $G_{(\mathfrak{A}, B')}$, čiji je skup čvorova podskup Q' skupa Q, koji se sastoji iz svih $q \in Q$ za koje postoje reči $\alpha, \beta \in A^*$, takve da je $q = \varphi(q_0, \alpha), \psi(q_0, \alpha) \in B'$. Iz čvora q_1 vodi rebro u čvor q_2 izvornika $G_{(\mathfrak{A}, B')}$, obeleženo slovom a iz A, tada i samo tada, kada je $\varphi(q_1, a) = q_2$. Početni čvor je stanje q_0, a skup F završnih čvorova sastoji se iz svih $q \in Q'$ za koje je $\psi(q) \in B'$. Očigledno, za svaki, na ovaj način konstruisani, izvornik važi jednakost $L_{(\mathfrak{A}, B')} = |G_{(\mathfrak{A}, B')}|$.

Obrnuto, neka je G-proizvoljni izvornik nad A, v_0 njegov početni čvor i F skup završnih čvorova. Razmotrimo automat $\mathfrak{A} = (A, Q, \{0, 1\}, \varphi, \psi, v_0)$, čiji se skup stanja sastoji iz skupa čvorova izvornika G i dodatnog elementa q'. Ako je

$q \in Q$, $q \neq q'$ i izvornik G sadrži rebro iz čvora q u čvor q_1, obeleženo slovom a tada je $\varphi(q,a) = q_1$. Ako iz čvora q ne izlazi rebro obeleženo slovom a, tada je $\varphi(q,a) = q'$. Osim toga $\varphi(q',a) = q'$ za proizvoljno a i $\psi(q) = 1$, tada i samo tada, kada $q \in F$. Očigledno je $L_{(\mathfrak{A},\{1\})} = |G|$.

Ukazana ekvivalentnost između načina zadavanja događaja automatima i izvornicima omogućuje svođenje, dalje razmatranih zadataka analize i sinteze automata na odgovarajuće probleme za izvornike, koji su više podesni sa teorijskog aspekta.

Pokazano je da se skup događaja koji se mogu predstaviti automatima, podudara sa skupom regularnih događaja, koji se predstavljaju pomoću tzv. regularnih izraza.

Neka je $A = \{a_1, \ldots, a_m\}$ konačna azbuka i (,) , $\vee$, $\cdot$, $<$, $>$, e - pomoćni simboli koji ne pripadaju azbuci A. Definišimo induktivno *regularan izraz* nad azbukom A.

1. Svako slovo a_i azbuke A, a takođe i simbol e su regularni izrazi nad A.

2. Ako su R_1 i R_2 regularni izrazi nad A, tada su $R = (R_1 \vee R_2)$, $R = (R_1 \cdot R_2)$ i $R = \langle R_1 \rangle$ takođe regularni izrazi nad A. Respektivno govorimo da je R *dusjunkcija*, *konjunkcija* i *iteracija* izraza R_1 i R_2.

Definišimo, takođe, složenost $L(R)$ regularnog izraza R,

$L(e) = 0$

$L(a_i) = 1, a_i \in A$

$L(\langle R \rangle) = L(R) + 1$

$L((R_1 \vee R_2)) = L((R_1 \cdot R_2)) = L(R_1) + L(R_2) + 1.$

Svakom regularnom izrazu R nad azbukom A pridružimo neki skup reči $|R|$ nad azbukom A, koji ćemo nazivati *regularnim događajem*, zadanim datim izrazom R. Naime,

$|e| = \{\Lambda\}$ (Λ - prazna reč)

$|a_i| = \{a_i\}$

$|(R_1 \vee R_2)| = |R_1| \cup |R_2|$

$|(R_1 \cdot R_2)| = \{p : (\exists p_1, q)(p = q \cdot r \wedge q \in |R_1| \wedge r \in |R_2|)\}$

$|\langle R_1 \rangle| = \{p : (\exists p_1, \ldots p_s)(p = p_1 \ldots p_s \wedge p_1 \in |R_1| \wedge \cdot \wedge p_s \in |R_1|)\} \cup \{\lambda\}$

Pošto je $|R_1 \cdot (R_2 \cdot R_3)| = |(R_1 \cdot R_2) \cdot R_3|$, $|R_1 \vee (R_2 \vee R_3)| = |(R_1 \vee R_2) \vee R_3|$, $|(R_1 \vee R_2)| = |(R_2 \vee R_1)|$, to regularne izraze koji se razlikuju rasporedom zagrada u delovima oblika $R_1 \ldots R_k$, $R_1 \vee \ldots \vee R_k$ i poretkom članova u izrazima oblika $R_1 \vee \ldots \vee R_k$, smatraćemo uvek jednakim, ukoliko to nije posebno naglašeno.

Neka je G zadati izvornik. *Analizom izvornika* G nazivamo postupak određivanja regularnog izraza R, tako da je $|G| = |R|$ [17].

Problem obrnut problemu analize izvornika naziva se problem sinteze izvornika.

Neka je dat regularan događaj zadat regularnim izrazom R. Postupak određivanja izvornika G, takvog da je $|G| = |R|$, nazivamo *sintezom izvornika* G.

Zbog opisa algoritma sinteze izvornika G uvedimo nekoliko novih pojmova. Neka je regularan događaj zadat regularnim izrazom R.

Ako je regularan izraz oblika $R_1 \vee R_2 \ldots R_k$, gde izraz $R_i (1 \leq i \leq k)$ nije disjunkcija, onda se izrazi $R_1, \ldots R_k$ nazivaju *članovima izraza* R. Ako regularni izraz nije disjunkcija onda je on član samoga sebe.

Mestom razdvajanja u regularnom izrazu R nazovimo specijalni znak | postavljen između proizvljna dva znaka tog izraza (uključujući i zagrade). Mesto koje se nalazi levo od prvog simbola izraza R nazovimo početnim mestom, a mesto koje se nalazi desno od poslednjeg simbola izraza R poslednjim ili konačnim. Ako se iz mesta α regularnog izraza može preći u mesto β, koristeći samo prelaz preko e, tada kažemo da je mesto β podređeno mestu α.

Uvedimo sledeća pravila podređenosti mesta:

1. Početna mesta svih članova izraza R zatvorenog u obične ili iteracione zagrade, podređena su mestu koje se nalazi neposredno levo od početne zagrade.

2. Mesto, koje se nalazi neposredno desno od zatvorene (desne) zagrade, podređeno je konačnim mestima svih članova izraza zatvorenog u odgovarajuće zagrade, a u slučaju iteracionih zagrada, takođe i mestu, koje se nalazi neposredno levo od odgovarajuće otvorene (leve) zagrade.

3. Početna mesta svih članova izraza zatvorenog u iteracione zagrade, podređena su mestu, koje se nalazi neposredno desno od odgovarajuće desne (ztvorene) zagrade.

4. Mesto koje se nalazi neposredno desno od simbola prazne reči, podređeno je mesto koje se nalazi neposredno levo od tog simbola.

5. Ako je mesto γ podređeno mestu β, a mesto β mestu α, tada je i mesto γ podređeno mestu α.

6. Svako mesto podređeno je samome sebi.

7. Ne postoje drugi slučajevi podređenosti mesta.

Neka je p reč u azbuci A nad kojom je zadan regularan izraz R. Uvedimo, induktivno, definiciju mesta β, koje p-sledi za mestom α tog izraza.

1. Mesto β e-sledi za mestom α, tada i samo tada, ako je β podređeno α.

2. Neka je mesto β p-sledi za mestom α i neka je mesto γ podređeno mestu β, pri čemu se neposredno desno od mesta γ nalazi slovo a. Tada mesto δ, koje se nalazi neposredno desno od slova a, pa-sledi za mestom α.

Osnovnim mestom regularnog izraza R, zadatog nad azbukom A, nazivamo mesto neposredno levo od kojeg se nalazi slovo te azbuke, a takođe i početno mesto. Osnovno mesto izraza nazivamo finalnim, ako mu je podređeno konačno mesto tog izraza. Mesto izraza R neposredno desno od kojega se nalazi slovo azbuke A nazivamo predosnovnim.

Odredimo izvornik $G = \mathbf{S}(R)$ nad azbukom A koja se podudara sa azbukom zadatog izraza R, tako da je $|G| = |R|$.

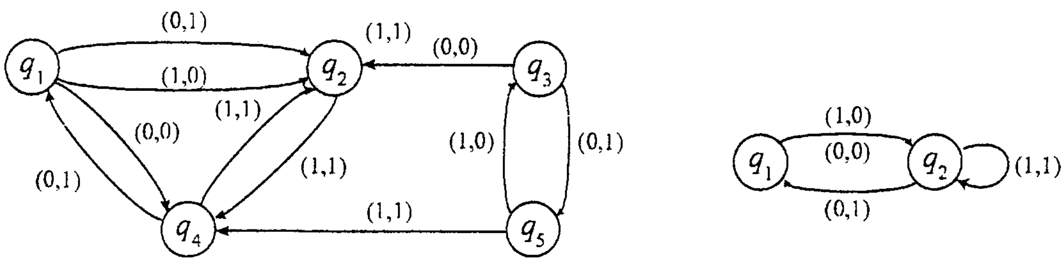
Za početni čvor v_0 izvornika $G = \mathbf{S}(R)$ uzmimo početno mesto izraza R. Neka je a proizvoljno slovo azbuke A. Odredimo skup v_a svih osnovnih mesta izraza R, koja a-slede za njegovim početnim mestom. Ako je $v_1 \neq \emptyset$, tada je v_a čvor izvornika G, u koji ulazi rebro koje izlazi iz čvora v_0 i čija je oznaka slovo a. Ukoliko je $v_a = \emptyset$, tada iz čvora v_0 ne izlazi rebro obeleženo slovom a. Pretpostavimo da je v već konstruisani čvor izvornika G, koji je ustvari neprazan skup osnovnih mesta izraza R, za proizvoljno slovo b azbuke A odredimo skup v_b svih osnovnih mesta izraza R, koja b-slede bar za jednim mestom skupa osnovnih mesta v. Ako je $v_b = \emptyset$, tada se iz čvora v ne konstruiše rebro obeleženo slovom b, Ako je $v_b \neq \emptyset$ i ako se podudara sa nekim već konstruisanim čvorom w izvornika G, tada konstruišemo rebro iz čvora v u čvor w i označavamo ga slovom b. Ako je $v_b \neq \emptyset$ i ne poklapa se ni sa jednim od već konstruisanih čvorova, tada konstruišemo novi čvor v_b i rebro ρ, koje vodi iz v u v_b, označavajući to rebro slovom b. Čvor v konstruisanog izvornika G je završni ako sadrži bar jedno finalno mesto izraza R. Primetimo, da ako je k broj svih pojavljivanja azbuke A u izrazu R, tada broj čvorova izvornika G nije veći od 2^{k+1}.

Sledećom teoremom, koju navodimo bez dokaza, dokazana je korektnost opisanog postupka sinteze.

Teorema 1. [17]) *Jednakost* $|R| = |\mathbf{S}(R)|$ *je tačna za proizvoljan regularan izraz* R.

1.5 Minimizacija automata

Postupkom sinteze automata, opisanom u prethodnom odeljku, dobija se automat koji, u opštem slučaju, nema minimalan broj stanja. Tako, na primer, automat čiji je dijagram predstavljen na sl. 11.a realizuje isto preslikavanje kao automat čiji je dijagram predstavljen na sl. 11.b.



a) *b)*

Slika 11.

Automat na slici 11.b dobijen je razbijanjem skupa stanja automata predstavljenog na sl. 11.a u dve grupe $\{q_2, q_4, q_5\}$ i $\{q_1, q_3\}$, i zatim njihovim izjednačavanjem u okviru iste grupe. Ovo je moguće jer automat za sva stanja iz iste grupe daje ista izlazna stanja (slova) i pri tome njegov prelaz iz stanja jedne u stanja iz druge grupe jednoznačno je određen ulaznim signalom nezavisno od izbora predstavnika grupe. Zato se stanja iste grupe mogu izjednačiti i tako dobiti automat predstavljen na sl 10.b).

Postupak dobijanja iz zadatog automata G takvog automata G' koji ima manji broj stanja i koji realizuje isto preslikavanje (pretvaranje) skupa ulaznih u skup izlaznih reči, naziva se *minimizacijom automata*. Konačni automat na koji se postupak minimizacije više ne može primeniti, zove se *minimalnim* odnosno *svedenim automatom*. Ovde ćemo umesto termina minimalni automat koristiti i termin *optimalni automat*. Rešavanje problema dobijanja optimalnog automata, tj. automata svedenog oblika, nameće potrebu razmatranja niza relacija ekvivalencije vezanih za funkcionisanje automata. Razmotrimo neke od njih.

Neka su zadati automati $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ i $\mathfrak{A}' = (A, Q', B, \varphi', \psi')$. Za stanja q i q', $q \in Q$ i $q' \in Q'$, kažemo da su *nesaglasna* ako postoji reč $\alpha \in A^+$ takva da $(q, \alpha) \in D_{\mathfrak{A}}$, $(q', \alpha) \in D_{\mathfrak{A}'}$ i $\psi(q, \alpha) \neq \psi'(q', \alpha)$; za ovakvu reč α kažemo da razlikuje stanja q i q'. Za stanja q i q', $q \in Q$ i $q' \in Q'$, kažemo da su *saglasna* ako nisu nesaglasna, tj. ako za svaku reč $\alpha \in A^+$ iz $(q, \alpha) \in D_{\mathfrak{A}}$ i $(q', \alpha) \in D_{\mathfrak{A}'}$ sledi da je $\psi(q, \alpha) = \psi'(q', \alpha)$. Ako su stanja q i q' saglasna i za svaku reč $\alpha \in A^+$ važi da je $(q, \alpha) \in D_{\mathfrak{A}}$ akko $(q', \alpha) \in D_{\mathfrak{A}'}$, tada za stanja q i q' kažemo da su *jako saglasna* što ćemo označavati sa $q \sim q'$. Primetimo da se u slučaju "potpunih" automata pojmovi jake saglasnosti i saglasnosti poklapaju.

Postupak minimizacije automata sastoji se i u određivanju skupova njegovih saglasnih, odnosno nesaglasnih stanja. Pre nego što navedemo tvrđenje kojim se daju potrebni i dovoljni uslovi o nesaglasnosti dva zadata stanja automata $\mathfrak{A} = (A, Q, B, \varphi, \psi)$, navedimo sledeće oznake i leme.

Neka je dat automat $\mathfrak{A} = (A, Q, B, \varphi, \psi)$, gde je $Q = \{q_1, q_2, \ldots, q_n\}$, i neka je $q \in Q$ i $k \in \mathbf{N}$. Označimo sa N_q^k skup svih stanja q' automata $\mathfrak{A}$ takvih da je $(q, \alpha) \in D_{\mathfrak{A}}$, $(q', \alpha) \in D_{\mathfrak{A}'}$ i $\overline{\psi}(q, \alpha) \neq \overline{\psi}(q', \alpha)$ za neku reč $\alpha \in A^+$ dužine najviše k, a sa $\mathcal{N}^k$ uređenu n-torku $(N_{q_1}^k, N_{q_2}^k, \ldots, N_{q_n}^k)$.

Lema 2. *Ako za parcijalni automat $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ postoji $k \in \mathbf{N}$ tako da je $\mathcal{N}^k = \mathcal{N}^{k+1}$, onda je za svako $l \geq k$, $\mathcal{N}^k = \mathcal{N}^l$.*

Dokaz: Pretpostavimo suprotno, tj. da postoje $k, l \in \mathbf{N}$, $l > k+1$, tako da je $\mathcal{N}^k = \mathcal{N}^{k+1}$, a $\mathcal{N}^k \neq \mathcal{N}^l$. Odavde sledi da postoji i $(1 \leq i \leq n)$ i $q \in Q$ takvo da su q_i i q nesaglasna stanja, a da svaka reč iz A^+ koja ih razlikuje nema manju dužinu od $k+2$; neka je $\alpha^{(t)}$ jedna od takvih "najkraćih" reči (znači $t \geq k+2$). Neka je $s = t - k$ i neka su $\varphi(q_i, \alpha^{(t)}]_{s-1}) = q'$ i $\varphi(q, \alpha^{(t)}]_{s-1}) = q''$. Primetimo da $q'' \notin N_{q'}^k$, zato što bi u suprotnom postojala kraća reč od $\alpha^{(t)}$ koja razlikuje q_i i q što je nemoguće. Pošto $\psi(q_i, \alpha^{(t)}) \neq \psi(q, \alpha^{(t)})$, odatle sledi $\psi(q', \alpha^{(t)}[_{k+1}) \neq$ $\neq \psi(q'', \alpha^{(t)}[_{k+1})$, što znači da $q'' \in N_{q'}^{k+1}$, tj. $N_{q'}^k \neq N_{q'}^{k+1}$, što je suprotno pretpostavci leme.

Lema 3. *Neka je $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ parcijalni automat, i neka je $|Q| \geq 2$. Tada je najmanje k, za koje je $\mathcal{N}^k = \mathcal{N}^{k+1}$, manje ili jednako od $|Q|^2 - |Q|$.*

Dokaz: Pretpostavimo suprotno, tj. postoje nesaglasna stanja q_1^0 i q_2^0 iz Q, takva, da svaka reč iz A^+ koja ih razlikuje nije dužine manje od $|Q|^2 - |Q| + 1$. Neka je $\alpha^{(t)}$ jedna od takvih najkraćih reči. U nizu $(q_1^0, q_2^0), (q_1^1, q_2^1), \ldots, (q_1^{t-1}, q_2^{t-1})$, gde je $(q_1^i, q_2^i) = (\varphi(q_1^0, \alpha^{(t)}]_i), \varphi(q_2^0, \alpha^{(t)}]_i))$ $(i = 1, \ldots, t-1)$ se ili neki od njegovih članova ponavlja ili je $q_1^i = q_2^i$ za neko $0 \leq i \leq t-1$, što je nemoguće pošto reč razlikuje stanja q_1^0 i q_2^0. Neka se neki članovi niza ponavljaju, i neka su to, recimo, članovi (q_1^k, q_2^k) i (q_1^l, q_2^l), $k < l$. Tada umesto reči $\alpha = \alpha^{(t)}$ posmatrajmo reč $\alpha' = \alpha]_k \alpha[_l$. Očigledno, reč α' razlikuje stanja q_1^0 i q_2^0, ali ona je kraća od reči

α. Iz dobijene kontradikcije sledi tačnost leme.

Napomena. Primetimo da za automate ("potpune", ne parcijalne) minimalna reč koja razlikuje proizvoljna dva nesaglasna stanja je uvek kraća od $|Q|$ (dokaz ovog tvrđenja se može naći, na primer, u [34]).

Iz navedenih lema sledi teorema.

Teorema 2. *Dva stanja zadatog parcijalnog automata* $\mathfrak{A} = (A, Q, B, \varphi, \psi)$, $|Q| \geq 2$, *nesaglasna su ako i samo ako postoji reč dužine najviše* $|Q|^2 - |Q|$ *koja ih razlikuje.*

Grafom prelaska automata $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ naziva se orijentisani graf G dobijen brisanjem oznaka grana i čvorova njegovom Murovog dijagrama. Automat $\mathfrak{A}$ je (*jako*) *povezan* ako za proizvoljna dva čvora grafa prelaska G toga automata postoji (orijentisan) put u grafu G koji ih povezuje.

Teorema 3. *Neka postoji g-homomorfizam parcijalnog automata* $\mathfrak{A}' = (A, Q', B, \varphi', \psi')$ *na parcijalni automati* $\mathfrak{A}'' = (A, Q'', B, \varphi'', \psi'')$ *i neka su stanja* $q_i', q_j' \in Q'$ *nesaglasna. Tada su i njihove homomorfne slike* $g(q_i')$ *i* $g(q_j')$ *takođe nesaglasna stanja.*

Dokaz: Neka je $g(q_i') = q_i''$ i neka je $g(q_j') = q_j''$. Pošto su stanja q_i' i q_j' nesaglasna, postoji reč $\alpha \in U_{\mathfrak{A}'_{q_0'}}$ koja ih razlikuje, tj. za koju je $\overline{\psi}(q_i', \alpha) \neq \overline{\psi}(q_j', \alpha)$. Na osnovu leme 1 $\overline{\psi}'(q_i', \alpha) = \overline{\psi}''(g(q_i'), \alpha) = \overline{\psi}''(q_i'', \alpha)$ i isto važi za stanje q_j', tj. $\overline{\psi}'(q_j', \alpha) = \overline{\psi}''(g(q_j'), \alpha) = \overline{\psi}''(q_j'', \alpha)$. Odatle zaključujemo da $\overline{\psi}''(q_i'', \alpha) \neq \overline{\psi}''(q_j'', \alpha)$, čime smo dokazali da su stanja q_i'' i q_j'' nesaglasna.

Sada ćemo navesti neke pojmove vezane za potpune automate. Operacija *sabiranja* automata definiše se na sledeći način. Neka su zadati parcijalni automati $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ i $\mathfrak{A}' = (A, Q', B, \varphi', \psi')$, i $Q \cap Q' = \emptyset$. *Suma* automata $\mathfrak{A}$ i $\mathfrak{A}'$ (u oznaci $\mathfrak{A} + \mathfrak{A}'$) je automat $\mathfrak{A}'' = (A, Q \cup Q', B, \varphi'', \psi'')$ gde je

$$\varphi''(q, a) = \begin{cases} \varphi(q, a), q \in Q \\ \varphi'(q, a), q \in Q' \end{cases}, \qquad \psi''(q, a) = \begin{cases} \psi(q, a), q \in Q \\ \psi'(q, a), q \in Q' \end{cases}$$

Teorema 4. [17] *Neka su dati automati* $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ *i* $\mathfrak{A}' = (A, Q', B, \varphi', \psi')$, *gde je* $Q \cap Q' = \emptyset$, *i neka je* $q \in Q$ *i* $q' \in Q'$. *Stanja* q *i* q' *su saglasna ako i samo ako za proizvoljnu reč* $\alpha^{|Q|+|Q'|-1} \in A^*$ *važi* $\overline{\psi}(q, \alpha^{|Q|+|Q'|-1}) = \overline{\psi}(q', \alpha^{|Q|+|Q'|-1})$.

Dokaz: Neka je automat $\mathfrak{A}'' = \mathfrak{A} + \mathfrak{A}'$ suma automata $\mathfrak{A}$ i $\mathfrak{A}'$. Ako su stanja q i q' u ovom automatu nesaglasna, onda, dokaz teoreme sledi iz napomene koja prethodi teoremi 2. Ako su stanja q i q' saglasna, tvrđenje važi po definiciji saglasnosti.

Primetimo da se u opštem slučaju ocena $|Q| + |Q'| - 1$ dužine reči kojima se realizuje raspoznavanje saglasnih stanja ne može biti poboljšana.

Za automat $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ kažemo da je *slabo uronjiv* u automat $\mathfrak{A}' = (A, Q', B, \varphi', \psi')$ (u oznaci $\mathfrak{A} \leq \mathfrak{A}'$) ako za proizvoljno stanje $q \in Q$ i proizvoljnu reč $\alpha \in A^*$ postoji stanje $q' \in Q'$ tako da je $\overline{\psi}(q, \alpha) = \overline{\psi}(q', \alpha)$.

Za automat $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ kažemo da je *uronjiv* u automat $\mathfrak{A}' = (A, Q', B, \varphi', \psi')$ (u oznaci $\mathfrak{A} \preceq \mathfrak{A}'$) ako za proizvoljno stanje q, postoji njemu saglasno stanje q' automata $\mathfrak{A}'$.

Očigledno, ako je $\mathfrak{A}$ uronjiv u $\mathfrak{A}'$, onda je $\mathfrak{A}$ slabo uronjiv u $\mathfrak{A}'$. Takođe, jasno je da je slaba uronjivost tranzitivna, tj. ako $\mathfrak{A} \leq \mathfrak{A}' \wedge \mathfrak{A}' \leq \mathfrak{A}'' \Rightarrow \mathfrak{A} \leq \mathfrak{A}''$. Odavde direktno sledi i tranzitivnost uronjivosti.

Sada se postavlja pitanje, ako je automat $\mathfrak{A}$ (slabo) uronjiv u $\mathfrak{A}'$, da li je $\mathfrak{A}'$ (slabo) uronjiv u $\mathfrak{A}$, tj. da li je (slaba) uronjivost simetrična relacija? Da uronjivosti nisu nisu simetrične relacije govori sledeća teorema.

Teorema 5. [17] *Za proizvoljni automat* $\mathfrak{A} = (A, Q, B, \varphi, \psi), |B| \geq 2$, *postoji automat* $\mathfrak{A}' = (A, Q', B, \varphi', \psi')$ *takav da je* $\mathfrak{A} \preceq \mathfrak{A}'$ *i* $\mathfrak{A}' \not\leq \mathfrak{A}$.

Dokaz: Neka je $Q = \{q_0, q_1, \ldots, q_{n-1}\}$ i neka je $a_1 \in A$. Obrazujmo n reči $\alpha_0 = a_1$, $\alpha_1 = a_1 a_1$, $\ldots$, $\alpha_{n-1} = \underbrace{a_1 \ldots a_1}_{n-\text{puta}}$. Konstruišimo automat $\mathfrak{A}'' = (A, Q', B, \varphi', \psi')$, gde je $Q' = \{q'_0, \ldots, q'_{n-1}\}$, $Q \cap Q' = \emptyset$, $\varphi'(q'_i, a_1) = q'_{i+1 (\text{mod } n)}$ i $\varphi'(q'_i, a) = q'_0$ za $a \neq a_1$; $\psi'(q'_i, a_1) \neq \psi(q_i, \alpha_i)$ i za neko fiksirano b_1 je $\psi'(q'_i, a) = b_1$, za proizvoljno $a \neq a_1$ i $i = 1, \ldots, n$. Nije teško dokazati da je za proizvoljno $q \in Q, \overline{\psi}(q, \alpha_n) \neq \overline{\psi}(q'_0, \alpha_n)$. Prema tome, automat $\mathfrak{A}''$ nije slabo uronjiv u automat $\mathfrak{A}$. Za automat $\mathfrak{A}'$ uzmimo automat $\mathfrak{A} + \mathfrak{A}''$. Pošto je automat $\mathfrak{A}''$ slabo uronjiv u automat $\mathfrak{A}'$, to automat $\mathfrak{A}'$ ne može biti slabo uronjiv u automat $\mathfrak{A}$. Uronjivost automata $\mathfrak{A}$ u automat $\mathfrak{A} + \mathfrak{A}''$ je očigledna. Time je teorema dokazana.

Automate $\mathfrak{A}$ i $\mathfrak{A}'$ nazivamo *slabo saglasnim* (*slabo nerazlikujućim*) (u oznaci $\mathfrak{A} \sim \mathfrak{A}'$) ako je $\mathfrak{A} \leq \mathfrak{A}'$ i $\mathfrak{A}' \leq \mathfrak{A}$. Automate $\mathfrak{A}$ i $\mathfrak{A}'$ nazivamo *saglasnim* (*nerazlikujućim*) (u oznaci $\mathfrak{A} \approx \mathfrak{A}'$) ako $\mathfrak{A} \preceq \mathfrak{A}'$ i $\mathfrak{A}' \preceq \mathfrak{A}$.

Iz saglasnosti automata $\mathfrak{A}$ i $\mathfrak{A}'$, očigledno, sledi i njihova slaba saglasnost. Može se pokazati da iz slabe saglasnosti ne mora da sledi saglasnost (kao što je važilo i za uronjivost). Može se postaviti pitanje kada su uronjivost, slaba uronjivost, saglasnost i slaba saglasnost međusobno ekvivalentni? O tome govori sledeća teorema.

Teorema 6. [17] *Ako su $\mathfrak{A}$ i $\mathfrak{A}'$ jako povezani automati i ako je $\mathfrak{A}$ slabo uronjiv u $\mathfrak{A}'$, onda su automati $\mathfrak{A}$ i $\mathfrak{A}'$ saglasni.*

Dokaz: Neka su dati automati $\mathfrak{A} = (A, Q, B, \varphi, \psi)$ i $\mathfrak{A}' = (A, Q', B, \varphi', \psi')$, gde je $Q = \{q_0, q_1, \ldots, q_{n-1}\}$ i $Q' = \{q'_0, q'_1, \ldots, q'_{m-1}\}$, i neka je α_1 neka reč iz A^*. Označimo sa $Q(\alpha_1)$ skup takvih stanja $q' \in Q'$ za koja važi $\overline{\psi}(q_0, \alpha_1) = \overline{\psi}'(q', \alpha_1)$. Skup $Q(\alpha_1)$ je neprazan, jer je $\mathfrak{A} \leq \mathfrak{A}'$. Posmatrajmo skup $Q'(\alpha_1) = \{\varphi'(q', \alpha_1) \mid q' \in Q(\alpha_1)\}$. Kako je $Q(\alpha_1)$ neprazan skup, onda je i $Q'(\alpha_1)$ neprazan. Takođe jasno je da za svaku reč $\gamma \in A^*$ važi $|Q'(\alpha_1)| \geq |Q'(\alpha_1\gamma)|$, (što je reč duža sve je manje stanja koja se "ponašaju na njoj" isto kao i q_0). Ako postoji reč γ_1 takva da je, $|Q'(\alpha_1\gamma_1)| < |Q'(\alpha_1)|$, razmotrimo reč $\alpha_2 = \alpha_1\gamma_1$. Za reč α_2 određujemo njemu odgovarajući skup $Q'(\alpha_2)$ i ispitujemo da li postoji reč $\gamma_2 \in A^*$ tako da $|Q'(\alpha_2\gamma_2)| < |Q'(\alpha_2)|$. Ako postoji, tada neka je $\alpha_3 = \alpha_2\gamma_2 = \alpha_1\gamma_1\gamma_2$, itd. Jasno je da će se ovaj proces na određenom koraku n zaustaviti. Neka je $\alpha = \alpha_n$. Tada će, očigledno važiti $|Q'(\alpha)| = |Q'(\alpha\gamma)|$ za proizvoljnu reč $\gamma \in A^*$. Pošto je automat jako povezan, postoje reči $\delta_0, \ldots, \delta_{n-1} \in A^*$, tako da za proizvoljno i $(i = 0, 1, \ldots, n-1)$, $\varphi(q_i, \delta_i) = q_{i+1 (\mathrm{mod}\ n)}$. Ako je $\varphi(q_0, \alpha) = q_k$, konstruišimo reč $\alpha\delta_k\delta_{k+1}\ldots\delta_{n-1}\delta_0\delta_1\ldots\delta_{k-1}$, i njoj odgovarajuće skupove $Q'(\alpha\delta_k)$, $Q'(\alpha\delta_k\delta_{k+1})$, $Q'(\alpha\delta_k\ldots\delta_{n-1}\delta_0\ldots\delta_{k-1})$. Pokažimo da ako je q_i proizvoljno stanje Q i q' proizvoljno stanje iz $Q'(\alpha\delta_k\ldots\delta_{n-1}\delta_0\ldots\delta_{i-1 (\mathrm{mod}\ n)})$ onda je $q_i \sim q'$. Ako bi stanja q_i i q' bila nesaglasna, onda bi postojala reč $\gamma \in A^*$, za koju $\overline{\psi}(q_i, \gamma) \neq \overline{\psi}(q', \gamma)$. To bi značilo da je $|Q'(\alpha\delta_k\ldots\delta_{n-1}\delta_0\ldots\delta_{i-1}\gamma)| < |Q'(\alpha\delta_k\ldots\delta_{n-1}\delta_0\ldots\delta_{i-1})|$ što protivureči izboru reči α. Pokažimo da za proizvoljno stanje automata q'_i automata $\mathfrak{A}'$ postoji takvo stanje q_j automata $\mathfrak{A}$, da važi $q'_i \sim q_j$. Neka je q bilo koje stanje automata $\mathfrak{A}$ i neka je q' stanje automata $\mathfrak{A}'$, za koje je $q \sim q'$. Kako je $\mathfrak{A}'$ jako povezan automat, postoji reč α tako da je $\varphi'(q', \alpha) = q'_i$. Odatle sledi da ako je $q_j = \varphi(q, \alpha)$ onda je $q_j \sim q'_i$.

Označimo sa $K(A, B)$ skup svih automata sa istim ulaznim i izlaznim azbukama A i B i kojima su skupovi stanja podskupovi prebrojivog skupa $\{q_1, \ldots q_n, \ldots\}$. Relacije saglasnosti i slabe saglasnosti razbijaju skup $K(A, B)$ na klase ekvivalencije.

Teorema 7. [17] *Ako je* $|B| \geq 2$ *onda je broj klasa ekvivalencije slabe saglasnosti prebrojiv.*

Dokaz: Neka je $\mathfrak{A}$ proizvoljan automat iz $K(A, B)$. Na osnovu teoreme 5 postoji takav niz automata $\mathfrak{A} = \mathfrak{A}_1, \mathfrak{A}_2, \ldots \in K(A, B)$, da automat $\mathfrak{A}_{i+1}$ nije slabo uronjiv u automat $\mathfrak{A}_i$ i da je $\mathfrak{A}_i$ uronjiv u $\mathfrak{A}_{i+1}$, $i = 1, 2, \ldots$. Pokažimo da automati $\mathfrak{A}_i$ i $\mathfrak{A}_j$, $i \neq j$, pripadaju različitim klasama slabe saglasnosti, čime bi dokazali teoremu. Slobodno možemo pretpostaviti da je $i < j$. Pretpostavimo suprotno, tj. da $\mathfrak{A}_i \sim \mathfrak{A}_j$. Odatle sledi da je $\mathfrak{A}_j \leq \mathfrak{A}_i$, a pošto $\mathfrak{A}_i \leq \mathfrak{A}_{i+1}, \ldots$, $\mathfrak{A}_{j-2} \leq \mathfrak{A}_{j-1}$ dobijamo $\mathfrak{A}_j \leq \mathfrak{A}_{j-1}$, što je protivurečno uslovu formiranja niza automata.

Označimo sa $K_{\mathfrak{A}}(A, B)$ skup svih automata iz $K(A, B)$ koji su saglasni s automatom $\mathfrak{A}$ iz $K(A, B)$. Označićemo sa $T_{\mathfrak{A}}(A, B)$ skup automata iz $K(A, B)$ koji su slabo saglasni sa automatom $\mathfrak{A} \in K(A, B)$.

Automat $\mathfrak{A}$ čija su sva stanja međusobno nesaglasna je, u stvari, minimalni automat, tj. automat svedenog oblika. Sledeća teorema pokazuje kakav je odnos optimalnog automata $\mathfrak{A}$ i klase $K_A(A, B)$ automata $K(A, B)$ koji su saglasni sa automatom $\mathfrak{A}$.

Teorema 8. [17] *Za proizvoljni automat* $\mathfrak{A}$ *klasa* $K_{\mathfrak{A}}(A, B)$ *sadrži sa tačnošću do izomorfizma jedinstven automat svedenog oblika.*

Dokaz: Pošto je relacija saglasnosti stanja relacija ekvivalencije, relacija saglasnosti stanja automata $\mathfrak{A}$, razbija skup njegovih stanja Q na klase ekvivalencije $Q_1, Q_2, \ldots, Q_k$. Neka je φ funkcija prelaska automata $\mathfrak{A}$. Pokažimo da za proizvoljno Q_i, $i = 1, 2, \ldots . k$, proizvoljna stanja q i q' iz Q_i i proizvoljno $a \in A$, ako je $\varphi(q, a) \in Q_j$, onda $\varphi(q', a) \in Q_j$, tj. saglasna stanja pod dejstvom iste ulazne reči prelaze u saglasna stanja. Pretpostavimo suprotno, tj. za neko i postoje $q, q' \in Q_i$ i $a \in A$ tako da $\varphi(q, a) \in Q_j$ i $\varphi(q', a) \in Q_l$, i pri tome je $l \neq j$. Kako $\varphi(q', a)$ i $\varphi(q, a)$ pripadaju raznim klasama saglasnosti, onda su ta stanja nesaglasna, tj. postoji neka reč $\alpha \in A^*$ koja razlikuje stanja $\varphi(q, a)$ i $\varphi(q', a)$. Sa druge strane, $\overline{\psi}(q, a\alpha) = \psi(q, a)\overline{\psi}(\varphi(q, a), \alpha)$ i $\overline{\psi}(q', a\alpha) = \psi(q', a)\overline{\psi}(\varphi(q', a), \alpha)$ odakle dobijamo da je $\overline{\psi}(q, a\alpha) \neq \overline{\psi}(q', a\alpha)$, a to znači da su stanja q i q' nesaglasna što je u kontradikciji sa pretpostavkom da su u istoj klasi Q_i. Očigledno je da za svako $q, q' \in Q_i$, $1 \leq i \leq k$, i $a \in A$ važi $\psi(q, a) = \psi(q', a)$. Na osnovu opisanih svojstava možemo konstruisati automat $\mathfrak{A}' = (A, Q', B, \varphi', \psi')$ na sledeći način: $Q' = \{q'_1, \ldots, q'_k\}$, tj. biramo po jedno stanje iz svake klase ekvivalencije; $\varphi'(q'_i, a) = q'_j$ ako postoje takva stanja q iz Q_i, i q' iz Q_j da je $\varphi(q, a) = q'$ i $\psi(q, a) = \psi'(q', a)$. Iz konstrukcije automata sledi da su sva njegova stanja međusobno nesaglasna, i da je proizvoljno stanje q iz Q_i automata $\mathfrak{A}$ saglasno stanju q'_i iz Q' automata $\mathfrak{A}'$, a važi i obrnuto, tj. da je proizvoljno stanje q'_i iz Q' saglasno proizvoljnom stanju q iz Q_i automata $\mathfrak{A}$. Time smo pokazali da je $\mathfrak{A}'$ sveden i da se sadrži u $K_{\mathfrak{A}}(A, B)$. Sada treba dokazati drugi deo teoreme, tj. da je konstruisani automat jedinstven do na izomorfizam.

Neka je $\mathfrak{A}'' \in K_{\mathfrak{A}}(A, B)$ proizvoljni svedeni automat. Treba pokazati da su automati $\mathfrak{A}'$ i $\mathfrak{A}''$ izomorfni. Kako su oba automata svedena, i kako su saglasna, onda imaju jednak broj stanja. Neka su $Q' = \{q'_1.\ldots, q'_k\}$ i $Q'' = \{q''_1, \ldots q''_k\}$, redom, skupovi stanja automata $\mathfrak{A}'$ i $\mathfrak{A}''$. Sada treba uspostaviti vezu među tim stanjima. Zato, definišimo funkciju $\xi : Q' \to Q''$ na sledeći način: $\xi(q'_i) = q''_j$ ako je $q'_i \sim q''_j$. Ovakvo preslikavanje ξ je uzajamno jednoznačno i za funkcije izlaska automata $\mathfrak{A}'$ i $\mathfrak{A}''$ važi jednakost $\psi'(q'_i, a) = \psi''(\xi(q'_i), a)$. Sada je jedino preostalo da pokažemo da za proizvoljne $i = 1, \ldots, k$ i $a \in A$ važi $\xi(\varphi(q'_i, a)) = \varphi''(\xi(q'_i), a)$, gde je φ'' funkcija prelaska automata $\mathfrak{A}''$. To mora biti tačno zato što pod dejstvom istog ulaznog slova, automati $\mathfrak{A}'$ i $\mathfrak{A}''$ iz saglasnih stanja moraju preći u saglasna stanja, jer bi u protivnom polazna stanja bila nesaglasna.

Može se pokazati da tvrđenje analogno prethodnoj teoremi za klasu slabo saglasnih automata $T_{\mathfrak{A}}(A, B)$, nije tačno. Naime, skup međusobno neizomorfnih automata svedenog oblika sadržanih u klasi $T_{\mathfrak{A}}(A, B)$ može biti beskonačan.

1.6 Pojam lavirinta. Neke klase lavirinata

Neka su Ω i Σ disjunktne azbuke slova ω i σ, pri čemu $\Omega \backslash \Sigma$ sadrži prazan simbol λ i neka je dat orgraf $G = (V, E)$. Trojku $L = (G, f_L, h_L)$, gde je $f_L : V \to \Omega$ preslikavanje kojim su čvorovima orgrafa G pridružena slova iz azbuke Ω, i $h_L : E \to \Sigma$ preslikavanje kojim su orijentisanim granama orgrafa G pridružena slova iz azbuke Σ tako da, ako za $\gamma_1, \gamma_2 \in E$ važi $\mathbf{p}_1(\gamma_1) = \mathbf{p}_2(\gamma_2)$, tada je $h_L(\gamma_1) \neq h_L(\gamma_2)$, nazivamo *lavirintom*. Skup čvorova V i skup grana E orgrafa G predstavlja skup čvorova i skup grana lavirinta L i nadalje ćemo ih označavati sa $V(L)$ i $E(L)$ redom.

Lavirint L sa izdvojenim skupom čvorova $V_0 = \{v_1, v_2, \ldots, v_n\}$, koje nazivamo *početnim* ili *inicijalnim čvorovima* lavirinta L, nazivamo *inicijalnim* i označavamo sa $L_{v_1,\dots,v_n}$ ili L_{V_0} ili $(L; V_0)$.

Sa $L(\Omega, \Sigma)$ označimo klasu svih lavirinata sa skupom oznaka čvorova slovima azbuke Ω i skupom oznaka grana slovima azbuke Σ.

Neka je $E^n = \{e_1, \dots, e_n\}$, skup baznih jediničnih vektora n-dimenzionog euklidskog prostora $\mathbf{R}^n$, i neka je $\overline{E}^n = \{e_1, \dots, e_n, \overline{e}_1, \dots, \overline{e}_n\}$, gde je $\overline{e}_i = -e_i, 1 \leq i \leq n$. U slučaju da je $n = 2$, $E^2 = \{i, j\}, \overline{E}^2 = \{i, j, -i, -j\}$. Nadalje ćemo umesto oznaka vektora $i, j, -i, -j$, koristiti, redom, oznake e, n, w, s.

Lavirint $L = (G, f_L, h_L) \in L(\Omega, \Sigma)$, gde je G simetričan orgraf (orijentisan graf kod koga je skup grana simetrična relacija nad skupom čvorova) nazivamo *n-dimenzionalnim lavirintom*, $n \geq 2$, ako važi:

1) $\Sigma = \overline{E}^n$ i $\Omega = \{\lambda\}$

2) za sve $u, v \in V(L)$, ako je $(u, v) \in E(L)$, tada je $h_L(v, u) = -h_L(u, v)$

Nadalje, za lavirint $L = (G, f_L, h_L)$ umesto oznake $h_L(\gamma), \gamma \in E(L)$, koristićemo oznaku $|\gamma|_L$ a ukoliko je iz konteksta jasno o kom je lavirintu reč, koristićemo oznaku $|\gamma|$.

Za n-dimenzionalne lavirinte L_1 i L_2 kažemo da su *slabo izomorfni* ako postoji bijekcija $g : V(L_1) \to V(L_2)$, takva da $(u, v) \in E(L_1)$ ako i samo ako $(g(u), g(v)) \in E(L_2)$, i pri tome ako jedan od lavirinata ima ulaz, to i drugi lavirint ima ulaz i pri tome $g(v_0^1) = v_0^2$, gde je v_0^1 ulaz lavirinta L_1, a v_0^2 ulaz lavirinta L_2. Ako je još $|(u, v)|_{L_1} = |(g(u), g(v))|_{L_2}$, za sve $(u, v) \in E(L_1)$, tada kažemo da su lavirinti L_1 i L_2 *izomorfni*. Mi nećemo razlikovati izomorfne lavirinte i pisaćemo $L_1 = L_2$.

Neka su $M, N \in \mathbf{R}^n, M \neq N$ i $e \in \overline{E}^n$. Kažemo da je vektor $\overrightarrow{MN}$ e-vektor ako je $\overrightarrow{MN} = \alpha e$ i $\alpha > 0$. Skup $\mathbf{T}$ duži iz $\mathbf{R}^n$ nazivamo *n-konfiguracijom* ako svake dve duži iz tog skupa mogu imati ne više od jedne zajedničke tačke, pri čemu, ako ona postoji, tada je ona krajnja tačka obe duži.

n-dimenzionalni lavirint L, gde je $V(L) \subseteq \mathbf{R}^n$ nazivamo *n-dimenzionalnim pravouglim lavirintom*, ako:

1) za svako $(u, v) \in E(L)$ tada je vektor $\overrightarrow{uv}$ je $|(u, v)|$-vektor,

2) skup duži $\mathbf{T} = \{\overline{uv} : (u, v) \in E(L)\}$ je n-konfiguracija.

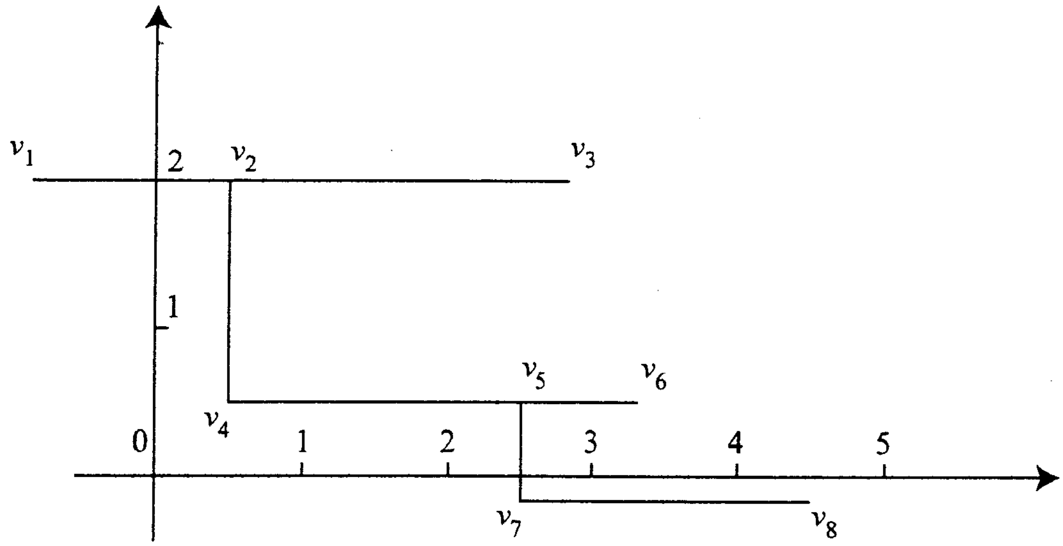
n-dimenzionalni lavirint L izomorfan nekom n-dimenzionalnom pravouglom lavirintu, naziva se *kvazipravougaonim*.

Neka je L neki n-dimenzioni pravougaoni lavirint. Figuru

$$\overline{L} = \bigcup_{(u,v) \in E(L)} \overline{uv}$$

u $\mathbf{R}^n$ nazivamo *realizacijom* n-dimenzionog pravouglog lavirinta L.

Na slici 4 predstavljena je realizacija 2-dimenzionalnog pravougaonog lavirinta $L = (G, f_L, h_L)$, gde je: $G = (V(L), E(L))$, $V(L) = \{v_1, v_2, \ldots, v_8\}$, gde su $v_1 = (-0.8, 2.0)$, $v_2 = (0.5, 2.0)$, $v_3 = (2.8, 2.0)$, $v_4 = (0.5, 0.5)$, $v_5 = (2.5, 0.5)$, $v_6 = (3.3, 0.5)$, $v_7 = (2.5, -0.2)$, $v_8 = (4.5, -0.2)$, skup grana $E(L) = \{(v_1, v_2), (v_2, v_3), (v_2, v_4), (v_4, v_5), (v_5, v_6), (v_5, v_7), (v_7, v_8)\}$ i preslikavanje kojim se orijentisanim granama pridružuju oznake iz azbuke $\{e, n, w, s\}$, je dato sa $|v_1, v_2|_L = e$, $|v_2, v_3|_L = e$, $|v_2, v_4|_L = s$, $|v_4, v_5|_L = e$, $|v_5, v_6|_L = e$, $|v_5, v_7|_L = s$, $|v_7, v_8|_L = e$.



Slika 12.

Neka je $\mathbf{Z}^n$ celobrojna rešetka u $\mathbf{R}^n$. Ako je $V(L) \subseteq \mathbf{Z}^n$, tada n-dimenzioni pravougli lavirint L, nazivamo n-dimenzionim celobrojnim lavirintom. Ako je $L = (V, E)$, n-dimenzioni celobrojni lavirint, i ako važi da je $\mathbf{T} = \{\overline{uv} : (u, v) \in E\}$ skup segmenata dužine 1, onda kažemo da je lavirint L *n–dimenzioni mozaični lavirint.*

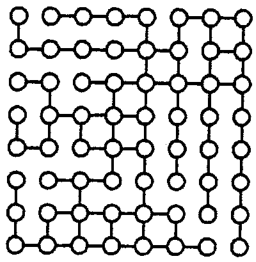
Kažemo da je čvor v u n-dimenzionom mozaičnom lavirintu L, *otvoren* u L, ako postoji beskonačni n-dimenzionalni mozaični lavirint L_1 takav da $\overline{L} \cap \overline{L}_1 = \{v\}$. Ako je $n = 2$ govorićemo da je lavirint *ravan*, npr, ravan mozaički lavirint (sl. 13), ravan pravougaoni lavirint, i sl.

Nacrtajmo sve moguće prave linije paralelne koordinatnim osama koje prolaze kroz čvorove iz $\mathbf{Z}^n$. Lik dobijen na taj način je *realizacija* n-dimenzionog lavirinta koji označavamo sa Z^n. Skup čvorova ovog lavirinta je $\mathbf{Z}^n$. Primetimo da n-dimenzioni mozaični lavirint možemo definisati i kao povezani deo lavirinta Z^n. Potpuni podgraf lavirinta Z^n nazivamo *n-dimenzionim šahovskim lavirintom.*

Ako sve parove suprotnih grana zamenimo odgovarajućim rebrima, onda dobijamo neusmeren graf koji označavamo sa $G(L)$.

Neka je $L = (V, E)$ neki ravan pravougaoni lavirint. Skup $\mathbf{R}^n \backslash \overline{L}$ je otvoren, i nepovezan (sem ako $G(L)$ nije drvo). Lavirint L nazivamo $(r+1)$-*povezanim*, ako skup $\mathbf{R}^n \backslash \overline{L}$ ima r ograničenih povezanih komponenti koje nazivamo *rupama*. Neka je L neki ravni šahovski lavirint, i neka su $U_1, \ldots, U_r$ sve komponente povezanosti skupa $\mathbf{R}^n \backslash \overline{L}$. Neprazan skup D oblika $U_l \cap \mathbf{Z}^n$ nazivamo *šahovskom rupom* lavirinta L. Ako je D konačno, onda rupu nazivamo *konačnom*, a u suprotnom je nazivamo *beskonačnom*. Ravni šahovski lavirint nazivamo $(k+1)$-*povezanim* ako ima tačno k konačnih rupa, $k \in \{0, 1, \ldots\}$. Ako je $k = 0$ lavirint nazivamo *jednopovezanim*.

Prost lanac u ravnom mozaičkom lavirintu L ćemo nazvati hodnikom. Reći ćemo za hodnik da je hodnik bez skretanja ako su sve grane koje izlaze iz čvorova v_i označene slovima iz $\{w, e\}$ ili slovima iz $\{n, s\}$.



Slika 13.

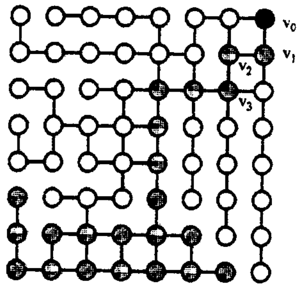
Označimo sa $\mathcal{L}_2$ klasu svih ravnih mozaičnih lavirinata, sa $\mathcal{L}_0$ klasu svih konačnih lavirinata iz $\mathcal{L}_2$, i sa $\mathcal{L}_1$ klasu svih beskonačnih lavirinata iz $\mathcal{L}_2$.

1.7 Ponašanje automata u lavirintima

U ovom odeljku će biti uveden pojam ponašanja automata u lavirintima. Nakon toga, biće uveden pojam obilaženja lavirinta. Odogovarajući pojmovi će biti uvedeni i za kolektive automata.

Automat $\mathfrak{A}$ je *dopustiv* za klasu lavirinata $\mathcal{L}(\Omega, \Sigma)$ ako mu se ulazna azbuka sastoji od slova a oblika $(\omega, \{\sigma_1, \ldots, \sigma_n\})$, gde je $\omega \in \Omega$ i $\{\sigma_1, \ldots, \sigma_n\} \subseteq \Sigma$, a izlazna azbuka mu je $\Sigma \cup \{\kappa\}$, $\kappa \notin \Sigma$ i $\psi(q, a) \in \mathbf{p}_2(a) \cup \{\kappa\}$. Klasu svih takvih automata označimo sa $\mathrm{At}(\Omega, \Sigma)$. Neka je $\mathfrak{A}_{q_0} \in \mathrm{At}(\Omega, \Sigma)$, i neka je $L_{v_0} \in \mathcal{L}(\Omega, \Sigma)$. Opišimo funkcionisanje automata $\mathfrak{A}_{q_0}$ u lavirintu L_{v_0}. U početku se automat $\mathfrak{A}_{q_0}$ nalazi u čvoru v_0 lavirinta L_{v_0}. Pretpostavimo da se u nekom vremenskom trenutku automat nalazi u stanju q u čvoru v lavirinta L_{v_0}. Pretpostavimo da automat "osmatra" označene grane koje izlaze iz tog čvora. Ulazno slovo automata u tom trenutku je uređeni par koga čine oznaka čvora i oznake grana koje izlaze iz čvora. U sledećem trenutku, ako je $\psi(q,a) \neq \kappa$ automat prelazi u čvor u koji ulazi grana označena sa $\psi(q,a)$, a ako je $\psi(q,a) = \kappa$, automat ostaje u uočenom čvoru i u oba navedena slučaja prelazi u stanje $\varphi(q,a)$. Na ovaj način, automat realizuje kretanje po lavirintu, korak po korak prelazeći neki put. *Funkcionisanje automata* $\mathfrak{A}_{q_0}$ *u lavirintu* L_{v_0}, možemo definisati kao ponašanje automata $\mathfrak{A}_{q_0}$ u lavirintu L_{v_0}. Niz $\pi(\mathfrak{A}_{q_0}; L_{v_0}) = (q_0, v_0), (q_1, v_1), \ldots$ nazivamo *ponašanje automata* $\mathfrak{A}_{q_0}$ u lavirintu L_{v_0} (sl. 14), ako je v_{i+1} čvor lavirinta L_{v_0}, u koji automat prelazi iz čvora v_i, nalazeći se u stanju q_i, i q_{i+1} stanje u koje automat $\mathfrak{A}_{q_0}$ prelazi. Označimo sa $\mathrm{Tr}(\mathfrak{A}_{q_0}; L_{v_0})$ niz oznaka $|(v_0, v_1)|, |(v_1, v_2)|, \ldots$, a sa $\mathrm{Tr}(\mathfrak{A}_{q_0}; L_{v_0}; s)$ prvih s elemenata niza $\mathrm{Tr}(\mathfrak{A}_{q_0}; L_{v_0})$. Ako za neki čvor $v \in L_{v_0}$, postoji stanje q automata $\mathfrak{A}_{q_0}$ takvo da uređeni par (q, v) pripada nizu $\pi(\mathfrak{A}_{q_0}; L_{v_0})$, kažemo da automat $\mathfrak{A}_{q_0}$ obilazi čvor v lavirinta L_{v_0}. Označimo sa $\mathrm{Int}(\mathfrak{A}_{q_0}; L_{v_0})$ skup svih čvorova lavirinta L_{v_0} koje automat $\mathfrak{A}_{q_0}$ obilazi.

Neka je $L_{v_0} \in \mathcal{L}(\Omega, \Sigma)$ i $\mathfrak{A}_{q_0} \in \mathrm{At}(\Omega, \Sigma)$. Ako je $\mathrm{Int}(\mathfrak{A}_{q_0}; L_{v_0})$ jednako skupu čvorova lavirinta L_{v_0}, kažemo da automat $\mathfrak{A}_{q_0}$ obilazi lavirint L_{v_0}, u protivnom lavirint L_{v_0} nazivamo zamkom za $\mathfrak{A}_{q_0}$. Ovi pojmovi se mogu proširiti i na neinicijalne automate i na neinicijalne lavirinte, kao i na bilo koju kombinaciju (ne)inicijalnih automata i (ne)inicijalnih lavirinata. Neka je $L \in \mathcal{L}(\Omega, \Sigma)$ i neka je $\mathfrak{A} \in \mathrm{At}(\Omega, \Sigma)$, gde i $\mathfrak{A}$ i L mogu biti i inicijalni i neinicijalni. Neka su $\alpha, \beta \in \{\mathbf{I}, \mathbf{A}, \mathbf{E}\}$. $\beta = \mathbf{I}$ ako je lavirint inicijalan, $\beta \neq \mathbf{I}$ u suprotnom, i neka isto to važi za automat, tj. $\mathfrak{A}$ je inicijalni automat ako $\alpha = \mathbf{I}$ ($\alpha \neq \mathbf{I}$ u suprotnom). Slovo $\mathbf{A}$ ukazuje da su svi čvorovi neinicijalnog lavirinta, odnosno sva stanja ne inicijalnog automata $\mathfrak{A}$ razmotrena. Sada možemo uvesti pojam $\alpha\beta$-obilaska i $\beta\alpha$-zamke, na primer: lavirint $L_{v_0} \in \mathcal{L}(\Omega, \Sigma)$ je **IA** zamka za automat $\mathfrak{A} \in \mathrm{At}(\Omega, \Sigma)$ ako za svako stanje $q \in Q_V$, lavirint L_{v_0} je zamka za automat $\mathfrak{A}_q$. Automat $V \in \mathrm{At}(\Omega, \Sigma)$ **AA**-obilazi lavirint L ako za bilo koje stanje automata, automat sa tim početnim stanjem obilazi lavirint krenuvši iz bilo kog čvora lavirinta. Uvešćemo konvenciju za posebne izbore α i β. Ako su $\alpha, \beta \in \{\mathbf{I}, \mathbf{E}\}$ tada ćemo reći *obilazi* i *zamka* umesto $\alpha\beta$-obilazi i $\beta\alpha$-zamke. Ako $\alpha, \beta \in \{\mathbf{I}, \mathbf{A}\}$, kazaćemo *jako obilazi* i *jaka zamka* umesto $\alpha\beta$-obilazi i $\beta\alpha$-zamke.



Slika 14.

Do sada smo razmatrali ponašanje jednog automata u lavirintu. Moguće je razmatrati i ponašanje sistema automata u lavirintu. Neka je dat lavirint $L_{v_1,\ldots,v_n} \in \mathcal{L}(\Omega, \Sigma)$ i neka je dat sistem dopustivih automata $\mathcal{A} = (\mathfrak{A}^1_{q_1}, \ldots, \mathfrak{A}^n_{q_n})$. Ako pod ponašanjem sistema, podrazumevamo n-torku koju čine ponašanja svakog automata pojedinačno, tj. $(\pi(\mathfrak{A}_{q_1}; L_{v_1}), \ldots, \pi(\mathfrak{A}_{q_n}; L_{v_n}))$, onda takav sistem nazivamo nezavisnim sistemom, a ponašanje nazivamo ponašanjem nezavisnog sistema. Nezavisni sistem opravda svoj naziv zato što automati ni na koji način nisu svesni prisustva drugih automata, pa sam tim i njihovo ponašanje uopšte ne zavisi od ponašanja drugih automata.

Ponašanje sistema možemo ojačati uvođenjem "svesnosti" automata drugih automata i njihovih ponašanja. Ovakva "svesnost" se može uvesti na razne načine. Neka su automati kodirani slovima $u_1, \ldots, u_n$ gde je u_i ili stanje i-tog automata $\mathfrak{A}^i_{q_i}$ u datom trenutku ili Λ. Neka je ulazna azbuka automata $\mathfrak{A}^i_{q_i}, 1 \leq i \leq$ $\leq n$ sastavljena od slova $a = (\omega, \{u_1, \ldots, u_{i-1}, u_{i+1}, \ldots u_n\}, \{\sigma_1, \ldots, \sigma_n\})$, gde $\omega \in \Omega$, $\{\sigma_1, \ldots, \sigma_n\} \subseteq \Sigma$ a izlazna azbuka je skup $\Sigma \cup \{\kappa\}$, $\kappa \notin \Sigma$, funkcija izlaska je $\psi_i(q, a) \in \mathbf{pr}_3(a) \cup \{\kappa\}, q \in Q_{V_{q_i}}$, tada sistem $\mathcal{A}$ nazivamo *kolektivom.*

Kao i kod sistema (nezavisnih) automata, možemo uvesti pojam funkcionisanja. Funkcionisanje kolektiva $\mathcal{A} = (\mathfrak{A}^1_{q_1}, \ldots, \mathfrak{A}^n_{q_n})$ u lavirintu $L_{v_1,\ldots,v_n}$ opisujemo na sledeći način. U početku je automat $\mathfrak{A}^i_{q_i}$ u čvoru v_i $(1 \leq i \leq n)$ lavirinta $L_{v_1,\ldots,v_n}$. Neka se u trenutku t automat $\mathfrak{A}^i_{q_i}$ nalazi u čvoru v^t_i i stanju q^t_i. Tada automat proverava uređenu trojku koju čine: oznaka čvora, kodovi svih automata sistema (tj. njihova stanja, ako se nalaze u datom čvoru v^t_i, ili simboli Λ u suprotnom slučaju) osim automata $\mathfrak{A}^i_{q_i}$ i oznake grana koje izlaze iz čvora v^t_i. Ako je $\psi_i(q^t_i, a^t_i) \neq \kappa$ automat prelazi u čvor u koji vodi grana označena sa $\psi_i(q^t_i, a^t_i)$, a ako je $\psi_i(q^t_i, a^t_i) = \kappa$, onda automat ostaje u čvoru v^t_i. Nakon toga automat prelazi u stanje $\varphi_i(q^t_i, a^t_i)$. Niz uređenih parova $(q^0_i, v^0_i), (q^1_i, v^1_i), \ldots$ je *ponašanje* automata $\mathfrak{A}^i_{q_i}$ u lavirintu $L_{v_1,\ldots,v_n}$, ako važi $(q^0_i, v^0_i) = (q_i, v_i)$, i ako je v_i^{j+1} čvor u koji automat $\mathfrak{A}^i_{q_i}$ prelazi iz čvora v_i^j menjajući stanje iz q_i^j u q_i^{j+1}. Možemo reći da automat $\mathfrak{A}^i_{q_i}$ obilazi čvorove $v_i^0, v_i^1, \ldots$. Taj skup čvorova označimo sa $\mathrm{Int}(\mathcal{A}, L_{v_1,\ldots,v_n}; i)$. Sada možemo uvesti pojam ponašanja kolektiva automata. Niz $\pi(\mathcal{A}, L_{v_1,\ldots,v_n}) = (q_1^0, \ldots, q_n^0, v_1^0, \ldots v_n^0), (q_1^1, \ldots, q_n^1, v_1^1, \ldots v_n^1), \ldots$ takav da je $(q_i^0, v_i^0), (q_i^1, v_i^1), \ldots$ ponašanje automata $\mathfrak{A}^i_{q_i}$ iz kolektiva $\mathcal{A}$ u lavirintu $L_{v_1,\ldots,v_n}$, nazivamo ponašanjem kolektiva automata $\mathcal{A}$ u lavirintu $L_{v_1,\ldots,v_n}$. Neka je $\mathrm{Int}(\mathcal{A}, L_{v_1,\ldots,v_n}) = \bigcup_{i=1}^{n} \mathrm{Int}(\mathcal{A}, L_{v_1,\ldots,v_n}; i)$. Ako je $\mathrm{Int}(\mathcal{A}, L_{v_1,\ldots,v_n}) = V$, tada kažemo da kolektiv *obilazi* lavirint $L_{v_1,\ldots,v_n}$, inače kažemo da je lavirint $L_{v_1,\ldots,v_n}$, zamka za kolektiv $\mathcal{A}$. Kažemo da $\mathcal{A}$ *jako obilazi* lavirint L ako za bilo koji izbor inicijalnih čvorova lavirinta L, kolektiv ga uvek obilazi.

1.8 Osnovni rezultati o ponašanju automata u lavirintima

Teorija automata u lavirintima se razvila u poslednjih četrdeset godina. Za to vreme pojavilo se više od 100 radova vezanih za problematiku ponašanja automata u lavirintima.

Prvi rad, čime je i započeto bavljenje ovim problemom, je rad K. Šenona [24].

Navedimo neke rezultate vezane za problematiku ponašanja automata u lavirintima. Pregled ovih, i svih ostalih značajnijih radova se može naći u [18].

Jedan od prvih i važnijih problema koji se razmatrao u okviru ove teorije je problem obilaženja automatom svih lavirinata iz neke zadate klase.

Neka je $\mathfrak{L}$ klasa (konačnih i/ili beskonačnih) lavirinata. Automat (kolektiv automata) $\mathfrak{A}$ je *univerzalan za klasu* $\mathfrak{L}$ ako obilazi svaki lavirint iz te klase. Postavlja se pitanje da li univerzalni automat uvek postoji. Lako je pokazati postojanje klasa lavirinata za koje ne postoji univerzalni automat. Recimo, ako se posmatra klasa svih konačnih pravouglih 3-dimenzionalnih lavirinata, to za nju nije moguće naći, ne samo univerzalan automat, nego i univerzalan kolektiv automata [3], [13] i [14]. To jest, za svaki unapred zadati kolektiv automata, moguće je naći lavirint-klopku koja pripada razmatranoj klasi lavirinata. Za neke druge slučajeve koji se odnose na dovoljno važne klase lavirinata, intuitivno, takođe, bilo je jasno da nije moguće za njih naći univerzalan automat. Do odgovarajućeg strogog dokaza nije se moglo tako lako doći.

Recimo ako je $\mathfrak{L}_{km}$ klasa svih konačnih mozaičnih ravnih lavirinata, tek je Budah [7], pokazao da je za svaki automat (dopustiv za tu klasu) moguće naći lavirint-klopku iz $\mathfrak{L}_{km}$. Prva varijanta njegovog dokaza izloženog u [7] je bila veoma komplikovana i dugačka. Dokaz je bio na 90 strana. Razlog za to je u glavnom ležao u primenjenom matematičkom aparatu (aparat teorije kategorija). On nije bio ni logički potpun, nije razmatran jedan od mogućih slučajeva. Prelaskom na jezik teorije automata dokaz se višestruko skraćuje i uprošćava (Potkolzin [34]) Suštinski, dokaz se nije razlikovao od [7], ali je za razliku od njega bio logički potpun. Sasvim drugi dokaz, dobijen primenom matematičke indukcije, i nesumnjivo do sada najkraći, dat je u [31], mada se mora naglasiti da je složenost klopki datih u [31] mnogo veća od odgovarajućih klopki konstruisanih u [34]. U [1] dato je uopštenje tvrdnje iz [7], pokazano je da da zvaki konačan nezavisni sistema automata postoji klopka iz $\mathfrak{L}_{km}$.

Razmatrani su i problemi složenosti dobijenih lavirinata-klopki. Složenost lavirinta sa može procenjivati na dva načina. Može se razmatrati složenost po broju čvorova koji pripadaj klopki, a takođe i takozvana složenost po broju konačnih rupa.

U [2] je dokazano da za svaki inicijalni automat sa n stanja postoji konačna ravna regularna klopka sa brojem čvorova ne većim od $c\ \exp\left((2n\log_2 2n)^{1/2}\right)$, gde je c konstanta.

Sa druge strane u [19] je pokazano da za svaki inicijalni automat postoji konačna ravna regularna r-povezana klopka tako da je $r \leq 3$.

Pokazano je da za klasu $\mathfrak{L}_{km}$ postoji univerzalni kolektiv automata tipa (2,0) i (1,2) (kažemo da je kolektiv tipa (a,b) ako se sastoji od a+b automata od kojih b njih "najprostijeg mogućeg vida" - ima samo jedno stanje i kreće se duž neke grane, ako i samo ako, zajedno sa njim duž te grane se kreće neki od prvih a automata; ovi automati se još zovu kamenovima i imaju ulogu spoljne memorije" ([4], [30]).

Razmatran je slučaj klase svih mozaični ravnih lavirinata (i beskonačnih), u [25] je pokazano da postoji kolektiv tipa (1,5) univerzalan za tu klasu. Posle rezultata izloženih u [28] ostalo samo nejasno pitanje da li postoji kolektiv tipa (1,4) koji obilazi sve lavirinte iz date klase lavirinata.

Svi izloženi rezultati, na neki način, daju negativnu karakterizaciju mogućnosti u obilaženju jednim automatom klasa lavirinata (osim u slučaju ako klasa lavirinata nije sasvim trivijalna, npr konačnim automatima). Primetimo da u slučaju klase svih drvolikih lavirinata takav automat postoji, dovoljno je da se on služi pravilom leve (desne) ruke. U [26], [27] navedene se neke klase lavirinata koje jedan automat ipak može da obiđe (univerzalan je za njih)

Neka je L konačan ravni mozaički lavirint. Za svaku stranu $f = (\gamma_1, \gamma_2, \ldots \ldots, \gamma_n)$ (strane su uzete u odnosu na kretanje obrnuto od kretanja kazaljki na časovniku) lavirinta L skup čvorova $\{\mathbf{pr}_i(\gamma_j) | 1 \leq i \leq 2, 1 \leq j \leq n\}$ označavamo sa $[f]$. Klasu svih strana koje ograničavaju konačne rupe lavirinta L označimo sa $F(L)$. Broj $\max\{\mathrm{diam}[f] | f \in F(L)\}$ nazivamo *straničnim dijametrom* lavirinta L.

U [27] je pokazano da za svako $d \in N$ postoji inicijalni automat $\mathfrak{A}$ koji jako obilazi klasu svih konačnih ravnih mozaičkih lavirinata čiji stranični dijametar nije veći od d, broj stanja automata $\mathfrak{A}$ je jednak Cd a vreme obilaska lavirinta sa n čvorova iz ove klase je ograničen sa $C'n$, gde su C i C' konstante.

I u slučaju kad automat obilazi lavirint, on često nije "svestan" te činjenice. On često nije u stanju da se zaustavi i signalizira nekim svojim unapred definisanim stanjem čin obilaženja. Najvažniji razlog za to leži u činjenici da on svoj svet (lavirint) posmatra iznutra, tj. nije u stanju da baci "pogled" na njega sa strane. Taj problem se razmatrao pod nazivom *problem zaustavljanja*. U [10] je dokazano da ne postoji konačni automat koji obilazi i zaustavlja se nakon obilaska bilo kog lavirinta iz klase jedno-povezanih konačnih ravnih šahovskih lavirinta.

Razmatrana su i pitanja postojanja algoritama kojima se utvrđuje za svaki predloženi automat da li on obilazi zadatu klasu lavirinata ili ne. U slučaju kada je klasa lavirinata konačna odgovor je trivijalan, dovoljno je "pustiti" automat da pokuša da obiđe svaki lavirint iz zadate klase. Složeniji slučajevi se odnose na beskonačne klase lavirinata. Neki od tih slučajeva su razmatrani u [12].

2 Sinteza optimalnih automata u nekim klasama lavirinata

U ovoj glavi će biti pokazano da je moguće sintetisati automat koji prolazi zadati put u lavirintu. Isto tako, biće dati i postupci sinteze minimalnog automata (po broju stanja), kao i neki postupci sinteze neoptimalnog automata koji prolazi zadati put u lavirintu.

2.1 Maksimalni automat

Neka je dat je ravan pravougaoni lavirint L i neki put P u tom lavirintu. U ovom poglavlju će biti dat postupak sinteze automata koji prolazi datim putem P u L. Kako čvorovi nisu označeni, na ponašanje automata u lavirintu L utiče samo trenutno stanje i "okolina" čvora u kome se on trenutno nalazi, koja se kodira (zadaje) odgovarajućim podskupom skupa $\mathrm{D} = E^2 = \{n, e, w, s\}$.

Kretanjem nazivamo bilo koji konačan niz $\mathfrak{k}$ oblika $(a_1, b_1), \dots, (a_n, b_n)$ $(n \geq$ $\geq 1)$, gde je $a_i \subseteq \mathrm{D}$ i $b_i \in \mathrm{D}$ za svako $1 \leq i \leq n$, takav da je $\{\overline{b_{i-1}}, b_i\} \subseteq a_i$ za svako $2 \leq i \leq n$. Podniz $\mathfrak{k}'$ niza $\mathfrak{k}$, takav da $\mathfrak{k}'$ čine susedni elementi niza $\mathfrak{k}$, nazvaćemo *podkretanjem* kretanja $\mathfrak{k}$. Inicijalni automat $\mathfrak{A} = (A, Q, B, \varphi, \psi, q)$, dopustiv za klasu ravnih pravougaonih lavirinata, izvršava kretanje $\mathfrak{k}$, ako postoji niz stanja $q_0 = q, q_1, q_2, \dots, q_n$ takvih da je uređena trojka $(a_1 \dots a_n, q_0, q_1 \dots q_n, b_1 \dots b_n)$ element funkcionisanja inicijalnog automata $\mathfrak{A}$.

Neka je $P = e_1, \dots, e_n$ proizvoljan put u lavirintu L, i neka su $v_0, v_1, \dots, v_n$ čvorovi lavirinta L takvi da je $e_i = (v_{i-1}, v_i)$ za svako $1 \leq i \leq n$. Broj n nazivamo dužinom puta i označavamo sa $|P|$. Neka je $[v_i]$ skup oznaka grana koje izlaze iz čvora v_i, a $|e_i|$ oznaka grane e_i. Lako je proveriti da je niz $\mathfrak{k}(P) =$ $([v_0], |e_1|), \dots, ([v_{n-1}], |e_{n-1}|)$ jedno kretanje; to kretanje nazvaćemo *kretanjem koje odgovara putu P u lavirintu L*. Inicijalni automat *prolazi datim putem* P, ako izvršava kretanje $\mathfrak{k}(P)$.

Skup svih automata koji izvršavaju dato kretanje $\mathfrak{k}$ (prolaze datim putem P u lavirintu L) označimo sa $\mathrm{Aut}(\mathfrak{k})$ ($\mathrm{Aut}(P)$). Naš glavni zadatak je da u skupu $\mathrm{Aut}(\mathfrak{k})$ svih automata koji izvršavaju kretanje $\mathfrak{k}$ pronađemo onaj automat koji ima minimalan broj stanja. Pre nego što se pozabavimo tim problemom razmotrimo da li tako definisan problem ima uopšte smisla, tj. da li je $\mathrm{Aut}(\mathfrak{k}) \neq \emptyset$ za svako $\mathfrak{k}$? Pokažimo sledeće tvrđenje.

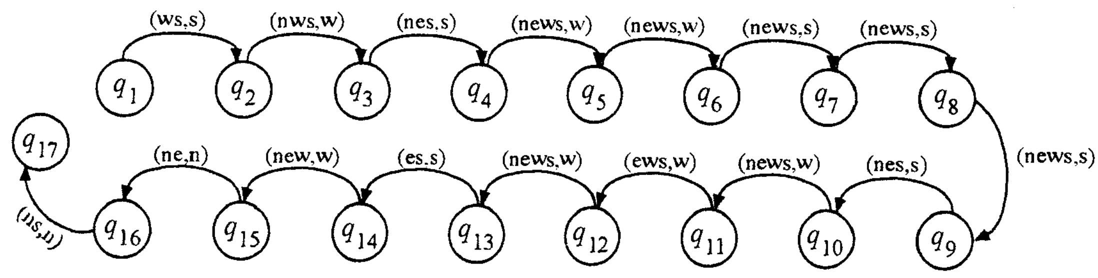
Tvrđenje 3. *Za svako kretanje* $\mathfrak{k}$ *je* $\mathrm{Aut}(\mathfrak{k}) \neq \emptyset$.

Dokaz: Neka je $\mathfrak{k} = (a_1, b_1), \ldots, (a_n, b_n)$ $(n \geq 1)$. Konstruišimo automat $\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}}(\mathfrak{k}) = (A, Q^{\mathrm{m}}, B, \varphi^{\mathrm{m}}, \psi^{\mathrm{m}}, q_1^{\mathrm{m}})$, dopustiv za ravne pravougaone lavirinte, na sledeći način. Neka je $Q^{\mathrm{m}} = \{q_1^{\mathrm{m}}, q_2^{\mathrm{m}}, \ldots, q_n^{\mathrm{m}}\}$ (kako je automat dopustiv, to je $A = \mathfrak{B}(\mathbf{D})$, a $B = \mathbf{D} \cup \{\kappa\}$), a q_1^{m} početno stanje. Definišimo funkcije φ^{m} i ψ^{m}. Neka je $\varphi^{\mathrm{m}}(q_i^{\mathrm{m}}, a_i) = q_{i+1}^{\mathrm{m}}$ za $i \in \overline{n-1}$, a $\varphi^{\mathrm{m}}(q_n^{\mathrm{m}}, a_n) = q_1^{\mathrm{m}}$ (ustvari možemo uzeti da je $\varphi^{\mathrm{m}}(q_n^{\mathrm{m}}, a_n)$ jednako proizvoljnom stanju iz Q^{m}, ali ne možemo ostaviti nedefinisanom ovu vrednost, jer je $D_{\varphi^{\mathrm{m}}} = D_{\psi^{\mathrm{m}}}$). Takođe, uzmimo da je $\psi^{\mathrm{m}}(q_i, a_i) = b_i$ za svako $i \in \overline{n}$. Jasno je da ovaj automat izvršava dato kretanje.

Automat $\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}}(\mathfrak{k}) = (A, Q^{\mathrm{m}}, B, \varphi^{\mathrm{m}}, \psi^{\mathrm{m}}, q_1^{\mathrm{m}})$ iz dokaza prethodnog tvrđenja zovemo *maksimalnim automatom koji izvršava zadato kretanje* $\mathfrak{k}$; ako je iz konteksta jasno o kom kretanju $\mathfrak{k}$ je reč umesto gornje oznake koristićemo oznaku $\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}}$. Kada se ovaj automat nađe u stanju q_i^{m}, ako se na ulazu pojavi simbol a_i, automat prelazi u stanje q_{i+1}^{m}, a na izlazu daje b_i (za druge ulazne simbole funkcije φ^{m} i ψ^{m} nisu definisane).

Na osnovu tvrđenja 3. možemo zapisati algoritam sinteze maksimalnog automata.

```
{α je ulazna niska, npr. (ns), (ns), (ns), ...; β je izlazna niska, npr. n, n, n, ...}
procedure max_aut(α, β)
    φ := {}; ψ := {}; l = len(α)
    for i:=1 to l − 1 do
        ψ := ψ ∪ ((q_i, α(i)), β(i)); φ := φ ∪ ((q_i, α(i)), q_{i+1});
    ψ := ψ ∪ ((q_l, α(l)), β(l)); φ := φ ∪ ((q_l, α(l)), q_l);
return (P({n, e, w, s}), {q_1, ..., q_l}, {n, e, w, s}, φ, ψ, q_1);
```



Slika 15. Primer maksimalnog automata.

Sledeće tvrđenje pokazuje da je klasa Aut($\mathfrak{k}$) "zatvorena" u odnosu na g-homomorfizme.

Lema 4. *Neka su dati kretanje $\mathfrak{k}$ i dva dopustiva (za klasu svih ravnih pravouglih lavirinata) automata $\mathfrak{A}'_{q'}$ i $\mathfrak{A}''_{q''}$. Ako automat $\mathfrak{A}'_{q'}$ izvršava kretanje $\mathfrak{k}$, tj. ako je $\mathfrak{A}'_{q'} \in \mathrm{Aut}(\mathfrak{k})$, i postoji g-homomorfizam $h : \mathfrak{A}'_{q'} \to \mathfrak{A}''_{q''}$, tada je i $\mathfrak{A}''_{q''} \in \mathrm{Aut}(\mathfrak{k})$.*

Dokaz: Iz postojanja g-homomorfizma $h : \mathfrak{A}'_{q'} \to \mathfrak{A}''_{q''}$ i tvrđenja 2 sledi da je $P_{\mathfrak{A}'} \subseteq P_{\mathfrak{A}''}$. Otuda i iz $\mathfrak{A}'_{q'} \in \mathrm{Aut}(\mathfrak{k})$ neposredno sledi da automat $\mathfrak{A}''_{q''}$ izvršava kretanje $\mathfrak{k}$, čime je tvrđenje dokazano.

Iz prethodne leme sledi da ako postoji g-homomorfizam maksimalnog automata u automat $\mathfrak{A}_q$ da onda automat $\mathfrak{A}_q$ izvršava kretanje $\mathfrak{k}$. Sledeća teorema pokazuje da važi i "obrnuto" tvrđenje: ako automat $\mathfrak{A}_q$ izvršava kretanje $\mathfrak{k}$, to postoji g-homomorfizam maksimalnog automata u automat $\mathfrak{A}_q$.

Teorema 9. *Neka je dato kretanje $\mathfrak{k}$. Automat $\mathfrak{A}_q$ izvršava kretanje $\mathfrak{k}$ ako i samo ako postoji g-homomorfizam automata $\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}}$ u automat $\mathfrak{A}_q$.*

Dokaz: Neka je $\mathfrak{k} = (a_1, b_1), \ldots, (a_n, b_n)$. Kao što smo videli, iz leme 4 sledi da ako postoji g-homomorfizam maksimalnog automata za dato kretanje u automat $\mathfrak{A}_q$, to automat $\mathfrak{A}_q$ izvršava kretanje $\mathfrak{k}$. Dokažimo obrnuto tvrđenje. Pošto je $\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}} \in \mathrm{Aut}(\mathfrak{k})$, to je $(a_1 \ldots a_n, q_1^{\mathrm{m}} \ldots q_n^{\mathrm{m}}, b_1 \ldots b_n)$ element funkcionisanja automata $\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}}$. Takođe, kako je $\mathfrak{A}_q \in \mathrm{Aut}(\mathfrak{k})$, to postoji niz stanja $q'_1, \ldots, q'_n$ takvih da je $(a_1 \ldots a_n, q'_1 \ldots q'_n, b_1 \ldots b_n)$ element funkcionisanja automata $\mathfrak{A}_q$. Uočimo preslikavanje $q_i^{\mathrm{m}} \xrightarrow{h} q'_i$, $i = 1 \ldots n$. Dokažimo da je ovo preslikavanje g-homomorfizam. Iz definicije maksimalnog automata za dato kretanje sledi da su sva stanja u nizu $q_1^{\mathrm{m}}, \ldots, q_k^{\mathrm{m}}$ različita, te je preslikavanje h jednoznačno. Funkcije prelaska i izlaza za automat $\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}}$ označimo redom sa φ^{m} i ψ^{m}, a za automat $\mathfrak{A}_q$

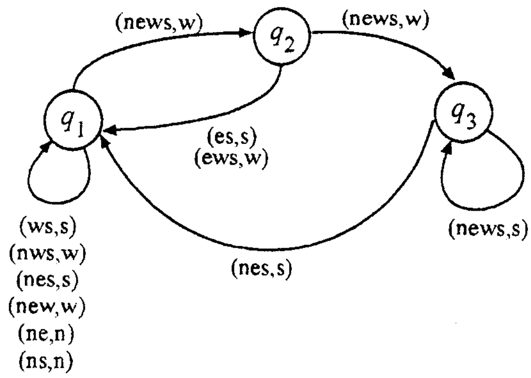
redom sa φ' i ψ'. Kako je $\mathfrak{A}_g \in \mathrm{Aut}(\mathfrak{k})$, to je

$$\psi^{\mathrm{m}}(q_i^{\mathrm{m}}, a_i) = \psi'(q_i', a_i) = \psi'(h(q_i^{\mathrm{m}}), a_i)$$

i

$$h(\varphi^{\mathrm{m}}(q_i^{\mathrm{m}}, a_i)) = h(q_{i+1}^{\mathrm{m}}) = q_{i+1}' = \varphi'(q_i', a_i) = \varphi'(h(q_i^{\mathrm{m}}), a_i)$$

za svako $1 \leq i \leq n$, tj. h je g-homomorfizam automata $\mathfrak{A}_{q_1^{\mathrm{m}}}^{\max}$ u automat $\mathfrak{A}_g$.



Slika 16.

Kažemo da je automat *optimalan za dato kretanje* $\mathfrak{k}$, ako je on automat sa minimalnim brojem stanja koji izvršava kretanje $\mathfrak{k}$. Jasno je da optimalnih automata može biti više. Iz prethodne teoreme sledi sledeća teorema.

Teorema 10. *Neka je dato kretanje* $\mathfrak{k}$. *Tada je svaki optimalni automat za kretanje* $\mathfrak{k}$ *g-homomorfna slika maksimalnog automata za dato kretanje.*

2.2 Graf nesaglasnosti automata. Procena broja stanja optimalnog automata

U ovom poglavlju uvešćemo pojmove grafa nesaglasnosti za zadati automat, bojenja grafa i pojam hromatskog broja grafa. Nakon toga će biti pokazano da za utvrđivanje nesaglasnosti dva stanja nije neophodno proveravati ponašanje automata za sve ulazne niske dužine n, gde je n dužina puta. Na kraju, biće pokazana veza između broja stanja optimalnog automata koji prolazi zadati put, i hromatskog broja grafa nesaglasnosti odgovarajućeg maksimalnog automata.

Neka su α i β dve proizvoljne reči nad azbukom C. Kažemo da je reč $\gamma \in C^*$ *najveći zajednički početak* reči α i β, i pišemo $\gamma = \mathrm{nzp}(\alpha, \beta)$, ako je γ početak za obe date reči α i β, i ako bilo koja druga reč $\gamma' \in C^*$, takva da je $|\gamma| < |\gamma'|$, nije početak za barem jednu od datih reči.

Tvrđenje 4. *Neka je dato kretanje* $\mathfrak{k} = \{(a_i, b_i)\}_{i=1}^{n}$, *i neka su* q_i^{m} *i* q_j^{m}, $i < j$, *dva stanja automata* $\mathfrak{A}_{q_1^{\mathrm{m}}}^{\max}(\mathfrak{k})$. *Stanja* q_i *i* q_j *su nesaglasna ako i samo ako je reč* $\alpha = \mathrm{nzp}(a_i a_{i+1} \dots a_n, a_j a_{j+1} \dots a_n)$ *neprazna i za nju važi da je* $\psi^{\mathrm{m}}(q_i^{\mathrm{m}}, \alpha) \neq$ $\neq \psi^{\mathrm{m}}(q_j^{\mathrm{m}}, \alpha)$.

Dokaz: Neka je ψ^{m} funkcija izlaza automata $\mathfrak{A}_{q_1^{\mathrm{m}}}^{\max}(\mathfrak{k})$. Ako je reč $\alpha \neq \Lambda$ tada su vrednosti $\psi^{\mathrm{m}}(q_i^{\mathrm{m}}, \alpha)$, $\psi^{\mathrm{m}}(q_j^{\mathrm{m}}, \alpha)$ definisane, i u slučaju da je $\psi^{\mathrm{m}}(q_i^{\mathrm{m}}, \alpha) \neq$ $\neq \psi^{\mathrm{m}}(q_j^{\mathrm{m}}, \alpha)$ stanja su q_i^{m} i q_j^{m} nesaglasna.

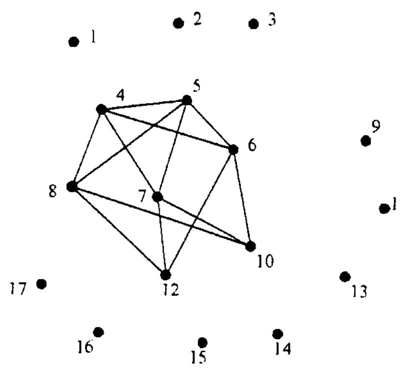
Obratno, pretpostavimo da su stanja q_i^{m} i q_j^{m} nesaglasna. Tada postoji neprazna reč β takva da su vrednosti $\psi^{\mathrm{m}}(q_i^{\mathrm{m}}, \beta)$, $\psi^{\mathrm{m}}(q_j^{\mathrm{m}}, \beta)$ definisane i da je $\psi^{\mathrm{m}}(q_i^{\mathrm{m}}, \beta) \neq \psi^{\mathrm{m}}(q_j^{\mathrm{m}}, \beta)$. No iz definicije maksimalnog automata tada sledi da je reč β početak za reči $a_i a_{i+1} \dots a_n$ i $a_j a_{j+1} \dots a_n$, a time i početak za reč α, odakle sledi da je $\alpha \neq \Lambda$ i $\psi^{\mathrm{m}}(q_i^{\mathrm{m}}, \alpha) \neq \psi^{\mathrm{m}}(q_j^{\mathrm{m}}, \alpha)$, čime je tvrđenje dokazano.

Primetimo da je u prethodnom tvrđenju $|\alpha| \leq n - j + 1$. Takođe, jasno je, da ako je reč α prazna, da tada stanja q_i^{m} i q_j^{m} nisu nesaglasna.

Neka je $\mathfrak{A}$ proizvoljan parcijalan automat i Q skup njegovih stanja. Binarnu relaciju

$$R_N(\mathfrak{A}) = \{(q_1, q_2) \in Q^2 \mid q_1 \text{ i } q_2 \text{ su nesaglasna stanja}\}$$

na skupu Q nazivamo *relacijom nesaglasnosti* parcijalnog automata $\mathfrak{A}$. Relacija nesaglasnosti je očito simetrična relacija. Neorijentisani graf $G(\mathfrak{A}) = (Q, \overline{R_N(\mathfrak{A})})$ ćemo zvati *grafom nesaglasnosti* automata $\mathfrak{A}$ (na sl. 17 je prikazan graf nesaglasnosti automata sa slike 15.).



Slika 17.

Neka su q_i^{m} i q_j^{m}, $i < j$, dva proizvoljna stanja maksimalnog automata za dato kretanje $\mathfrak{k} = \{(a_i, b_i)\}_{i=1}^n$, i neka je $\alpha = \mathrm{nzp}(a_i a_{i+1} \ldots a_n, a_j a_{j+1} \ldots a_n)$. Kao što smo videli, ako je $\alpha = \Lambda$, tada stanja q_i^{m} i q_j^{m} nisu nesaglasna. Ako je $\alpha \neq \Lambda$, tj. $|\alpha| = l$, gde je $1 \leq l \leq n - j + 1$, tada razmotrimo dva slučaja. Ako je $\psi^{\mathrm{m}}(q_i^{\mathrm{m}}, \alpha) = \psi^{\mathrm{m}}(q_j^{\mathrm{m}}, \alpha)$, tada za svako $0 \leq k \leq l$ stanja q_{i+k}^{m} i q_{j+k}^{m} nisu nesaglasna. Ako je $\psi^{\mathrm{m}}(q_i^{\mathrm{m}}, \alpha) \neq \psi^{\mathrm{m}}(q_j^{\mathrm{m}}, \alpha)$, tada postoji neko k_0, $1 \leq k_0 \leq l$, takvo da je $b_{i+k_0} \neq b_{j+k_0}$ i $b_{i+k} = b_{j+k}$ za svako $1 \leq k < k_0$. U tom slučaju su stanja q_{i+k}^{m} i q_{j+k}^{m} nesaglasna za svako $1 \leq k \leq k_0$. Ovo razmatranje navodi na činjenicu, koju ćemo kasnije i pokazati, da svaki neorijentisani graf ne može biti grafom nesaglasnosti za neki maksimalni automat. Takođe, koristeći uočena svojstva možemo zapisati algoritam za dobijanje grafa nesaglasnosti maksimalnog automata za neko kretanje, u obliku sledećeg programa.

```
Program GrafNesaglasnosti(input, output);
const
      MAX_STANJA = 100;
var
      mat_nes:array[1..MAX_STANJA,1..MAX_STANJA] of Boolean;
      inout:array[1..MAX_STANJA] of String[8];
      size:LongInt;
      f: TextFile;
procedure UcitajPut;
begin
      AssignFile(f, Edit1.Text);
      Reset(f);
      size:=0;
      // ucitavanje ulaza i izlaza
      while not Eof(f) do begin
            size := size + 1;
            readln(f, inout[size]);
      end; while
      CloseFile(f);
end;

procedure GenerisiGrafNesaglasnosti;
var
      i, j ,k, l, m:integer;
      flag:boolean;
begin
      // postavljanje pocetnih vrednosti za matricu nesaglasnosti
      for i:=1 to size do
```

```
            for j:=1 to size do
                    mat_nes[i,j]:=false;
    i:=1;
    while i ≤ size - 1 do begin
            j:=i+1;
            while j≤size do begin
                    flag:=true; k:=i; l:=j;
                    while Copy(inout[k],1,4)=Copy(inout[l],1,4) and l≤size and not flag and
                                    mat_nes[k,l]=false do begin
                            if Copy(inout[k],5,4)<>Copy(inout[l],5,4) then flag:=false;
                            k:=k+1; l:=l+1;
                    end;{while}

                    // ako su stanja nesaglasna onda je flag=false
                    if not flag then begin
                            for m:=0 to k-i do begin
                                    mat_nes[i+m,j+m]:=true; mat_nes[j+m,i+m]:=true;
                            end; {for}
                            j:=j+k-i;
                    end; {if}
                    j:=j+1;
            end; {while}
            i:=i+1;
    end; {while}
end; {proc}
// pocetak programa
begin
    UcitajPut;
    GenerisiGrafNesaglasnosti;
end.
```

Program u matricu nesaglasnosti *mat_nes* u kolonu i i vrstu j zapisuje true ako su stanja q_i^m i q_j^m nesaglasna, a u suprotnom false.

Neka je dat neorjentisani graf G. Maksimalni potpuni podgrafovi grafa G zovu se *klikama* grafa G. Ako sa S označimo skup svih klika grafa G onda je *gustina* datog grafa broj

$$k(G) = \max_{K \in S} |K|.$$

Graf G nazivamo *r-obojivim* ako čvorovi grafa G mogu biti obojeni sa r boja tako da ne postoje dva susedna čvora koja su obojena istom bojom; drugim rečima, graf $G = (V, E)$ je r-obojiv, ako postoji r-bojenje grafa G, tj. funkcija $\mathfrak{b} : V \to \overline{r}$ takva da je $\mathfrak{b}(u) \neq \mathfrak{b}(v)$ za svako $(u, v) \in E$. Najmanji broj r takav da je graf G

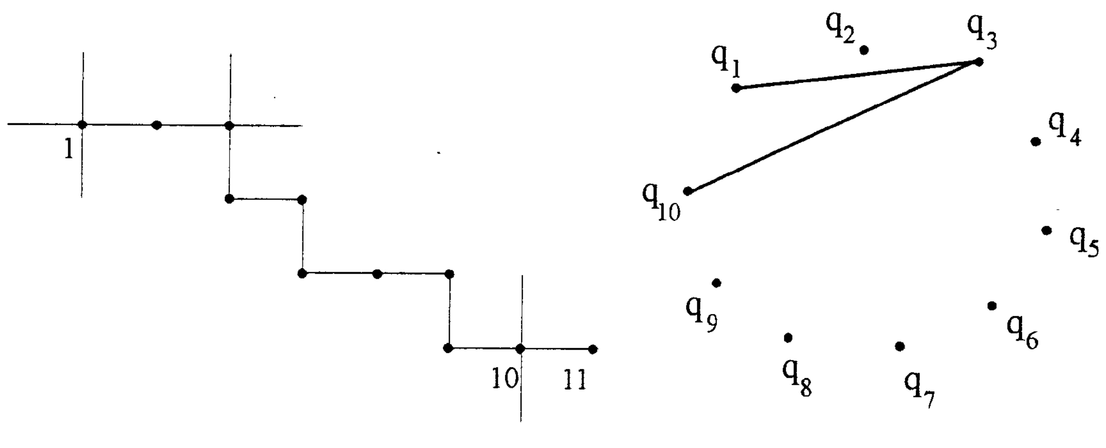
r-obojiv nazivamo *hromatskim brojem* grafa G i označavamo sa $\gamma(G)$. Nije teško uočiti da je $k(G) \leq \gamma(G)$. Primetimo da je hromatski broj potpunog grafa jednak broju čvorova u grafu.

Tvrđenje 5. *Za svaki automat* $\mathfrak{A}_q = (A, Q, B, \varphi, \psi, q) \in \mathrm{Aut}(\mathfrak{k})$ *važi nejednakost:*

$$|Q| \geq \gamma(G(\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}}(\mathfrak{k}))).$$

Dokaz: Pokažimo da svaki g-homomorfizam maksimalnog automata određuje jedno bojenje odgovarajućeg grafa nesaglasnosti. Neka je $g : \mathfrak{A}^{\max}_{q_1^{\mathrm{m}}} \longrightarrow \mathfrak{A}_q$, $\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}} = (A, Q^{\mathrm{m}}, B, \varphi^{\mathrm{m}}, \psi^{\mathrm{m}}, q_1^{\mathrm{m}})$, $\mathfrak{A}_q = (A, Q, B, \varphi, \psi, q)$, $Q = \{q_1, \ldots, q_r\}$. Tada možemo odrediti bojenje $\mathfrak{b} : Q^{\mathrm{m}} \longrightarrow \overline{r}$ grafa $G(\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}}(\mathfrak{k}))$ na sledeći način: $\mathfrak{b}(q) = k (1 \leq k \leq r)$, akko $g(q) = q_k$. Lako se pokazuje da je preslikavanje $\mathfrak{b}$ zaista bojenje grafa nesaglasnosti sa $|Q|$ boja odakle direktno sledi dokaz tvrđenja.

U prethodnom tvrđenju smo pokazali da svaki g-homomorfizam maksimalnog automata $\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}}(\mathfrak{k})$ određuje i jedno bojenje grafa nesaglasnosti $G(\mathfrak{A}^{\max}_{q_1^{\mathrm{m}}}(\mathfrak{k}))$. Da obrnuto tvrđenje ne važi, pokazuje sledeći primer:



Slika 18.

Za kretanje $\mathfrak{k}$ i graf nesaglasnosti koji su prikazani na prethodnoj slici izaberimo bojenje $\mathfrak{b}(q_1) = \mathfrak{b}(q_{10}) = 1$, $\mathfrak{b}(q_2) = \mathfrak{b}(q_3) = \cdots = \mathfrak{b}(q_9) = 2$. Lako se proverava da ovim bojenjem nije određen g-homomorfizam. Kako je hromatski broj grafa nesaglasnosti sa prethodne slike jednak 2, vidimo da tvrđenje ne važi ni za optimalno bojenje.

2.3 Sinteza optimalnog automata.

Pokazano je da g-homomorfizam maksimalnog automata, grupiše stanja maksimalnog automata u skupove međusobno saglasnih stanja, čime opisuje razbijanje skupa stanja automata. Kako svako razbijanje ne određuje g-homomorfizam, postupak sinteze optimalnog automata se može svesti na pretraživanje svih razbijanja skupa stanja maksimalnog automata i određivanje minimalnog razbijanja koje određuje g-homomorfizam, a time i minimalni automat koji izvršava zadato kretanje $\mathfrak{k}$.

Familija podskupova $\mathfrak{X} = (X_\lambda, \lambda \in \Lambda)$ skupa X zove se *pokrivanjem* skupa X, ako je $\bigcup_{\lambda \in \Lambda} X_\lambda = X$. Pokrivanje $\mathfrak{X}$ je *razbijanje* skupa X, ako su svaka dva elementa familije $\mathfrak{X}$ disjunktna. Svako razbijanje $\mathfrak{Q}$ skupa Q^{m} određuje jedno preslikavanje $g : Q^{\mathrm{m}} \to Q'$, gde je $Q' = \{q'_1, q'_2, \ldots, q'_l\}$ a $l = |\mathfrak{Q}|$, gde $g(q_i) = g(q_j)$ akko q_i i q_j pripadaju istom elementu familije $\mathfrak{Q}$. Preslikavanje g određuje relacije $\tilde{\varphi}' = \{(q'_i, a, q'_j) \mid (\exists q^{\mathrm{m}}_b, q^{\mathrm{m}}_c \in Q^{\mathrm{m}})\ g(q^{\mathrm{m}}_b) = g'_i \wedge g(q^{\mathrm{m}}_c) = g'_j \wedge \varphi^{\mathrm{m}}(q^{\mathrm{m}}_b, a) = q^{\mathrm{m}}_c\}$ i $\tilde{\psi}' = \{(q'_i, a, d) \mid (\exists q^{\mathrm{m}}_b \in Q^{\mathrm{m}})\ g(q^{\mathrm{m}}_b) = q'_i \wedge \psi^{\mathrm{m}}(q^{\mathrm{m}}_b, a) = d\}$ Ako su relacije $\tilde{\varphi}'$ i $\tilde{\psi}'$ funkcijske, tada one određuju funkcije prelaska i izlaska φ' i ψ' automata $\mathfrak{A}' = \{A, Q', B, \varphi', \psi', q'\}$. Koristeći razbijanje skupa stanja maksimalnog automata, možemo formulisati algoritam sinteze automata sa k stanja koji izvršava zadato kretanje $\mathfrak{k}$: *za svako razbijanje skupa stanja $Q^{\max}$, na k podskupova, maksimalnog automata $\mathfrak{A}^{\max}(\mathfrak{k})$, formirati odgovarajuće preslikavanje g i proveriti da li je time određen automat, i ako jeste, da li izvršava zadato kretanje $\mathfrak{k}$.*

Algoritam sinteze optimalnog automata koji izvršava zadato kretanje, je prikazan sledećim programom:

```
Program SintezaOptimalnogAutomata(input, output);
const
    MAX_STANJA = 100;
    ulazi: array[1..15] of String[4]= ('0001','0010','0100','1000','1001','1010','1100','0101','0110',
           '0011','1101','1110','1011','0111','1111');
type
    TStanje = 0..MAX_STANJA;
    TSkupStanja = set of TStanje;
    TUlaz = string[4];
    TIzlaz = string[4];
var
    razbijanje:array[1..MAX_STANJA] of integer;
    br_st:Integer;
```

```
        phi:array[1..MAX_STANJA,1..15] of TStanje;
        psi:array[1..MAX_STANJA,1..15] of TIzlaz;
        inout:array[1..MAX_STANJA] of String[8];
        size:LongInt;
        f: TextFile;
procedure UcitajPut;
begin
        AssignFile(f, Edit1.Text);
        Reset(f);
        size:=0;
        // ucitavanje ulaza i izlaza
        while not Eof(f) do begin
                size := size + 1;
                readln(f, inout[size]);
        end; while
        CloseFile(f);
end;

// formira automat na osnovu razbijanja
function Formiraj():Boolean;
var
        i, j, k:integer;
begin
        formiraj:=true;
        for i:=1 to size do
                for j:=1 to 15 do begin
                        phi[i,j]:=0;
                        psi[i,j]:='';
                end;
        // idi po stanjima iz maksimalnog automata
        // i preslikavaj ih u stanja automata sa i stanja
        for i:=1 to size do begin
        // pronadji redni broj ulaza
                k:=0;
                for j:=1 to 15 do
                        if ulazi[j]=copy(inout[i],1,4) then begin
                                k:=j;
                                break;
                        end; {if}
                // postavljamo izlaz
                if (psi[razbijanje[i],k]='') or (psi[razbijanje[i],k]=copy(inout[i],5,4)) then
                        psi[razbijanje[i],k]:=copy(inout[i],5,4)
                else begin
```

```
                formiraj:=false;
                break;
            end;

            // postavljamo sledece stanje
            if (phi[razbijanje[i],k]=0) or (phi[razbijanje[i],k]= razbijanje[i+1]) or (razbijanje[i+1]=0) then
                if i≥ size then
                    phi[razbijanje[i],k] := razbijanje[i]
                else
                    phi[razbijanje[i],k] := razbijanje[i+1]
            else begin
                formiraj:=false;
                break;
            end;{if}
        end; {for}
end; { proc}
// proverava da li formirani automat izvrsava kretanje
function izvrsava():boolean;
var
        br, st,i,k:integer;
        iz:TIzlaz;
begin
        st:=1;
        br:=1;
        while br≤size do begin
            // odredi izlaz i uporedi sa zeljenim
            k:=0;
            for i:=1 to 15 do
                if ulazi[i]=copy(inout[br],1,4) then begin
                    k:=i;
                    break;
                end; {if}
            if psi[st,k]<>copy(inout[br],5,4) then
                // doslo je do kontradikcije
                break;

            // idemo na sledecu io nisku
            br:=br+1;
            st:=phi[st,k];
        end; {while}
        if br>size then
            izvrsava:=true
        else
```

```
            izvrsava:=false;
end;{proc}

// generisanje automata pomocu razbijanja
procedure SintezaOptimalnogAutomata;
var
      opt_aut, i, j:integer;
      flag, f1:boolean;
begin
      for i:=1 to MAX_STANJA do
            razbijanje[i]:=0;
      opt_aut := 0;
      for br_st := 1 to size do begin
            // postavimo sve brojace na 1
            for i:=1 to size do
                  razbijanje[i]:=1;

            // generisemo sva preslikavanja size stanja
            // u br_st stanja
            while razbijanje[size]≤br_st do begin
                  f1:=formiraj();
                  if f1 and izvrsava() then begin
                        // pronasli smo jedno razbijanje
                        opt_aut:=opt_aut+1;
                        break;
                  end;{if}

                  // generisemo sledeci izbor razbijanja
                  flag := true;
                  j:=2;
                  while flag and (j≤ size) do begin
                        razbijanje[j]:=razbijanje[j]+1;
                        if razbijanje[j] ≤ br_st then
                              flag := false // zavrsili smo
                        else begin
                              razbijanje[j]:=1;
                              j:=j+1;
                        end; {if}
                  end; {while}

                  if j> size then
                        Break;
            end; {while}
```

```
            if opt_aut > 0 then
                  Writeln('Broj stanja optimalnog automata', br_st);
      end; {for}
end;{proc}

// pocetak programa
begin
      UcitajPut;
      SintezaOptimalnogAutomata;
end.
```

Primetimo da postupak prikazan prethodnim programom, ne koristi činjenicu da se dva nesaglasna stanja ne mogu preslikati u isto stanje optimalnog automata. Ovaj algoritam je neefikasan za puteve kojima odgovaraju optimalni automati sa većim brojem stanja, ali je značajan zbog algoritma slučajnog pretraživanja o kome će biti reči kasnije.

2.4 Kompatibilni skupovi i sinteza optimalnog automata

U ovom poglavlju ćemo opisati postupak sinteze optimalnog automata koji koristi činjenicu da se nesaglasna stanja maksimalnog automata ne mogu preslikati u isto stanje optimalnog automata. Zbog toga, uvodimo pojmove kompatibilnog skupa i prostog kompatibilnog skupa koji predstavljaju skupove stanja koja se mogu preslikati u isto stanje optimalnog automata. Na taj način, ne pretražuju se sva razbijanja skupa, već se odbacuju ona koja sigurno ne mogu da daju optimalni automat.

Podskup X skupa stanja Q^{m} maksimalnog automata $\mathfrak{A}^{\mathrm{max}}_{q_1^{\mathrm{m}}}(\mathfrak{k})$ nazivamo *kompatibilnim* ako svaka dva stanja iz skupa X nisu nesaglasna. Pokrivanje Ω skupa Q^{m} nazivamo *kompatibilnim pokrivanjem* ako je svaki element familije Ω kompatibilan skup. Primer pokrivanja (razbijanja) za sliku 18, pokazuje da svako kompatibilno pokrivanje ne određuje automat koji izvršava kretanje. To znači da je potrebno postaviti i dodatne uslove za pokrivanje da bi se mogao odrediti automat koji izvršava kretanje $\mathfrak{k}$.

Označimo sa φ_a funkciju $\varphi_a(q) = \varphi^{\mathrm{m}}(q, a)$, gde je $q \in Q^{\mathrm{m}}$, a Q^{m} je skup stanja maksimalnog automata $\mathfrak{A}_{q_1^{\mathrm{m}}}(\mathfrak{k}) = (A, Q^{\mathrm{m}}, B, \varphi^{\mathrm{m}}, \psi^{\mathrm{m}}, q_1^{\mathrm{m}})$, koji izvršava kretanje $\mathfrak{k}$. Skup $\varphi_a(X), X \subseteq Q^{\mathrm{m}}, a \in A$ nazivamo *impliciranim skupom* skupa X

za ulaz a. Pokrivanje $\mathfrak{Q}$ skupa Q^{m} je *zatvoreno* ako je svaki impliciran skup proizvoljnog elementa familije $\mathfrak{Q}$ podskup nekog elementa familije $\mathfrak{Q}$.

Tvrđenje 6. *Svako zatvoreno i kompatibilno razbijanje skupa stanja Q^{m} maksimalnog automata $\mathfrak{A}_{q_1^{\mathrm{m}}}(\mathfrak{k})$ određuje automat koji izvršava kretanje $\mathfrak{k}$.*

Dokaz: Neka je $\mathfrak{Q} = \{Q_1, Q_2, \ldots, Q_l\}$ zatvoreno i kompatibilno razbijanje skupa Q^{m}. Pokažimo da je automat $\mathfrak{A}'$ dobro definisan: $\mathfrak{A}' = (A, \mathfrak{Q}, B, \varphi', \psi', Q'_0)$, gde je $Q'_0 \in \mathfrak{Q}$, $q_0 \in Q'_0$, $\varphi'(Q'_i, a) = Q'_j$ ako je $\varphi'_a(Q'_i) \subseteq Q'_j$, a $\psi'(Q'_i, a) = b$ ako postoji $q' \in Q'_i$ tako da $\psi^{\mathrm{m}}(q', a) = b$. Kako je $\mathfrak{Q}$ razbijanje skupa Q^{m}, Q'_0 uvek postoji, funkcija φ' je dobro definisana zato što je razbijanje zatvoreno, a funkcija ψ' je dobro definisana zato što je razbijanje kompatibilno. Time smo pokazali da je automat dobro definisan. Pokazaćemo da automat izvršava kretanje $\mathfrak{k}$ tako što ćemo pokazati da je sledeće preslikavanje g-homomorfizam: $g : Q^{\mathrm{m}} \to \mathfrak{Q}$; $g(q) = Q'$, ako $Q' \in \mathfrak{Q}$, $q \in Q'$. Neka je $(q, a) \in D_{\mathfrak{A}^{\max}}$. Tada je $g(\varphi^{\mathrm{m}}(q, a)) = \varphi'(g(q), a)$ po definiciji funkcija g i φ'. Na sličan način, $\psi^{\mathrm{m}}(q, a) = \psi'(g(q), a)$, čime smo pokazali da je g, g-homomorfizam, što znači da automat $\mathfrak{A}'$ izvršava kretanje $\mathfrak{k}$.

Sinteza optimalnog automata koji izvršava kretanje $\mathfrak{k}$ se svodi na određivanje kompatibilnog i zatvorenog razbijanja $\mathfrak{Q}$ skupa Q^{m}, koje ima minimalan broj elemenata. Algoritam sinteze optimalnog automata možemo zapisati na sledeći način: *generisati sve kompatibilne skupove nad skupom Q^{m}. Za sve moguće izbore dva kompatibilna skupa proveriti da li oni predstavljaju zatvoreno i kompatibilno razbijanje. Ako ni za jedan izbor nije dobijeno zatvoreno kompatibilno razbijanje, postupak ponavljati za 3, 4, ... kompatibilna skupa sve dok ne dobijemo zatvoreno kompatibilno razbijanje. Dobijenim razbijanjem je određen optimalni automat.*

```
Program SintezaOptimalnogAutomata(input, output);
const
      MAX_STANJA = 100;
      ulazi: array[1..15] of String[4]= ('0001','0010','0100','1000','1001','1010','1100','0101','0110',
             '0011','1101','1110','1011','0111','1111');
type
      TStanje = 0..MAX_STANJA;
      TSkupStanja = set of TStanje;
      TUlaz = string[4];
      TIzlaz = string[4];
var
      mat_nes:array[1..MAX_STANJA, 1..MAX_STANJA] of Integer;
      inout:array[1..MAX_STANJA] of String[8];
```

```
        size:LongInt;
        f: TextFile;
        komp_skupovi: array[1..1000000] of TSkupStanja;
        komp_broj:LongInt;
        razbijanje:array[1..MAX_STANJA] of integer;
        max_skup:TSkupStanja;
        br_st:Integer;

function kompatibilan(skup: TSkupStanja):Boolean;
var
        i,j: Integer;
        b: Boolean;
begin
        b := False;
        for i:=1 to size-1 do
                for j:=i+1 to size do
                        b:= b or (mat_nes[i,j] and (i in skup) and (j in skup));
        kompatibilan := not b;
end;

function d(skup:TSkupStanja; ulaz:TUlaz):TSkupStanja; // generisanje impliciranog skupa
var
        st: 1..MAX_STANJA;
        tmp:TSkupStanja;
begin
        tmp:=[];
        for st:=1 to size do
                if (st in skup) then
                        if (Copy(inout[st],1,4)=ulaz) then
                                include(tmp, st);
        d:=tmp;
end; {func}

function razbija():boolean; // da li postojeci izbor komp. stanja pokriva sva stanja
var
        s:TSkupStanja;
        i:integer;
        pokriva:boolean;
begin
        s:=[];
        for i:=1 to br_st do
                s:=s+komp_skupovi[razbijanje[i]];
        if s=max_skup then pokriva:=true
```

```
        else pokriva:=false;
        if pokriva then begin
                razbija:=true;
                for i:=1 to br_st-1 do
                        for j:=i+1 to br_st do
                                if (komp_skupovi[razbijanje[i]]*komp_skupovi[razbijanje[j]])<>[] then razbija:=false;
end;{func}

function zatvoren():boolean; // da li je razbijanje zatvoreno
var     i,j,k:integer;
        da:TSkupStanja;
        flag, f1:boolean;
begin
        flag := true;
        // idi po svim skupovima
        i:=1;
        while (i≤ br_st) and flag do begin
                // idi po svim mogucim ulazima
                j:=1;
                while (j≤15) and flag do begin
                        da := d(komp_skupovi[razbijanje[i]],ulazi[j]);
                        f1:=false;
                        // proveri da li je impliciran skup pokriven nekim od skupova
                        for k:=1 to br_st do
                                if komp_skupovi[razbijanje[k]]≤da) and (komp_skupovi[razbijanje[k]]<> da) then
                                        f1:=true;
                        flag := flag and (not f1);
                        j:=j+1;
                end; {while}
                i:=i+1;
        end; {while}
        zatvoren:=flag; // vracamo rezultat
end; {func}

procedure GenerisiOptimalniAutomat;
var
        i, j, opt_aut:integer;
        flag:boolean;
begin
        // da li ima optimalnih automata sa 1 do size stanja
        opt_aut := 0;
        for br_st := 1 to size do begin
                // postavimo sve brojace na 1
```

```
            for i:=1 to br_st do
                razbijanje[i]:=1;
            // generisemo sve izbore br_st od komp_broj elemenata
            // skupa kompatibilnih skupova
            while razbijanje[br_st]≤komp_broj do begin
                if razbija() and zatvoren() then
                    // pronasli smo jedno razbijanje
                    opt_aut:=opt_aut+1;
                // generisemo sledeci izbor kompatibilnih automata
                flag := true;
                j:=1;
                while flag and (j≤ br_st) do begin
                    razbijanje[j]:=razbijanje[j]+1;
                    if razbijanje[j] ≤ komp_broj then
                        flag := false // zavrsili smo
                    else begin
                        razbijanje[j]:=1;
                        j:=j+1;
                    end; {if}
                end; {while}
                if j> br_st then begin
                    // prosli smo kroz sve izbore razbijanja sa
                    // br_st stanja i sada idemo dalje ako nismo
                    // pronasli optimalni automat
                    if opt_aut > 0 then
                        writeln('Br. stanja optimalnog automata: ', br_st);
                    Break;
                end; {if}
            end; {while}
            if opt_aut > 0 then
                Break;
        end; {for}
        if opt_aut > 0 then
            Writeln('Br. stanja optimalnog automata: ', br_st);
end; {proc}

function intpow(b:integer; p:integer):integer;
var
        i, ret:integer;
begin
        ret:=1;
        for i:=1 to p do ret:=ret*b;
        intpow:=ret;
```

```
end;

// generisanje kompatibilnih skupova
procedure GenerisiKompatibilneSkupove;
var
        i, j:integer;
        s:TSkupStanja;
        vr:string;
begin
        // generisi sve podskupove i eleminisi nekompatibilne skupove
        komp_broj:=0;
        for i:=1 to IntPow(2,size) do begin
                s:=[];
                for j:=1 to size do
                        if ( (1 shl (j-1)) and i ) <> 0 then
                                include(s, j);
                // da li je kompatibilan
                if kompatibilan(s) then begin
                        komp_broj:=komp_broj+1;
                        komp_skupovi[komp_broj]:=s;
                end;
        end; {for}
end{proc};

procedure Ucitaj;
begin
        AssignFile(f, Edit1.Text);
        Reset(f);
        size:=0;
        max_skup:=[];
        // ucitavanje ulaza i izlaza
        while not Eof(f) do begin
                size := size + 1;
                readln(f, inout[size]);
                // pravimo maksimalni skup stanja
                include(max_skup, size);
        end; {while}
        CloseFile(f);
end;
begin
        Ucitaj; GenerisiGrafNesaglasnosti;
        GenerisiKompatibilneSkupove;
        GenerisiOptimalniAutomat;
```

end.

Nedostatak postupka sinteze optimalnog automata koji izvršava zadato kretanje $\mathfrak{k}$ pomoću kompatibilnih skupova, je u tome što je potrebno dosta memorije za smeštanje svih kompatibilnih skupova, a sa povećanjem broja kompatibilnih skupova raste i vreme potrebno za sintezu optimalnog automata. Broj kompatibilnih skupova je manji ili jednak od 2^n gde je n dužina puta $\mathfrak{k}$.

2.5 Sinteza optimalnog automata metodom rekurzije

Neka su $\alpha = a_1 a_2 \dots a_n \in A^+$, $\beta = b_1 b_2 \dots b_n \in B^+$, redom, ulazna i željena izlazna niska automata $\mathfrak{A}_1 = (A, Q, B, \varphi, \psi, 1)$, gde je $Q = \{1, 2, \dots s\}$, $s \leq n$. Automat $\mathfrak{A}_1$ će biti predstavljen stekom čiji elementi su oblika (st_1, u, i, st_2), gde su $st_1, st_2 \in Q$, $u \in A$, $i \in B$, tako da $i = \psi(st_1, u)$, a $st_2 = \varphi(st_1, u)$. Lako se može pokazati da se na ovaj način može predstaviti bilo koji parcijalni automat.

Rad automata $\mathfrak{A}_1$ za datu ulaznu nisku α opisan nizom četvorki (st_1, a_1, b_1, st_2), $(st_2, a_2, b_2, st_3), \dots, (st_i, a_i, b_i, st_{i+1})$, gde je $st_1 = 1$.

Zapišimo algoritam sinteze automata koji izvršava kretanje $\mathfrak{k}$ metodom rekurzije:

A. Proglasi stek praznim
B. Na stek postavi $(1, a_1, b_1, 1)$
C. Proveri da li automat opisan stekom izvršava kretanje $\mathfrak{k}$
  Moguća su tri slučaja:
    1. automat izvršava kretanje $\mathfrak{k}$
    2. automat ne izvršava kretanje zato što se prilikom izvršavanja
       došlo do elementa steka $(st_i, a_i, b_i, st_{i+1})$
       gde $(st_{i+1}, a_{i+1}, b_{i+1}, st_{i+2})$ ne postoji u steku
       (kažemo da je automat nedefinisan)
    3. Automat ne izvršava kretanje $\mathfrak{k}$ zato što se prilikom
       izvršavanja došlo do $(st_i, a_i, b_i, st_{i+1})$,
       i postoji $k < i$ tako da $st_i = st_k$, $a_i = a_k$ i $b_i \neq b_k$
       (kažemo da je došlo do kontradikcije)
D. Ako automat opisan stekom izvršava kretanje $\mathfrak{k}$ prikaži automat
E. Ako je došlo do kontradikcije onda
  - Izbaci sa vrha steka sve elemente oblika (st, a, b, s)
  - Ako je stek ostao prazan onda ne postoji automat sa s stanja
    koji izvršava kretanje $\mathfrak{k}$.
  - element na vrhu steka (st', a, b, st'')
    zameni elementom $(st', a, b, st'' + 1)$

F. Ako je došlo do neodređenosti onda dodaj element $(st_{i+1}, a_{i+1}, b_{i+1}, 1)$ na vrh steka.

G. Idi na korak C.

Prethodni postupak, kao rezultat, daje automat sa datim brojem stanja s koji izvršava kretanje $\mathfrak{t}$, ili signalizira da se automat sa s stanja ne može sintetisati. Ako uvećavamo broj stanja od 1 do dužine puta $\mathfrak{t}$, u jednom trenutku ćemo dobiti optimalni automat. Sledi program za sintezu optimalnog automata koji prolazi zadati put.

```
program OptimimalniAutomat(input, output);
type
        TStek =record
                st1:integer;
                ui:string[8];
                st2:integer;
        end;
var
        br_st,s,duz_steka:Integer;
        ui:string[8];
        Stek:array[1..1000] of TStek;
        inout:array[1..MAX_STANJA] of String[8];
function ok_stanje(st:integer; uii:string):integer;
// vraca -1 ako je kontradikcija
// -2 ako nema takvog ui para
// stanje u koje prelazimo, ako nije ni -1 ni -2
var
        i:integer;
begin
        i:=1;
        while i≤duz_steka do begin
                if (stek[i].st1=st) and (stek[i].ui = uii) then begin
                        ok_stanje:=stek[i].st2;
                        break;
                end;
                if (stek[i].st1=st) and (copy(stek[i].ui,1,4) = copy(uii,1,4)) then begin
                        ok_stanje := -1;
                        break;
                end;
                i:=i+1;
        end;
        if i>duz_steka then begin
```

```
            ok_stanje := -2;
            s:=st;
            ui:=uii;
        end;
end;

function pusti_aut():integer;  provera korektnosti automata
var
        stanje, tek_ul, tmp:integer;
begin
        stanje:=1;
        tek_ul:=1;
        while tek_ul≤size do begin
                tmp := ok_stanje(stanje, inout[tek_ul]);
                if tmp>0 then begin
                        stanje:=tmp;
                        tek_ul:=tek_ul+1; end
                else break;
        end;{while}
        if tek_ul>size then
                pusti_aut := 0
        else
                pusti_aut := tmp;
end;

procedure SintezaOptimalnogAutomata;
label  t;
var
        lKraj:boolean;
        tmp,k,max_st:integer;
        ss:string;
begin
        duz_steka:=1;
        stek[1].st1:=1;
        stek[1].st2:=1;
        stek[1].ui := inout[1];
        for max_st := 1 to size do begin
                lKraj:=false;
                // kreiraj pocetni automat
                while not lKraj do begin
t:                      tmp:=pusti_aut();
                        if tmp=0 then Break; // pronasli smo automat
                        if tmp=-1 then begin// doslo je do kontradikcije
```

```
                    while (duz_steka>0) and (stek[duz_steka].st2 = max_st) do
                        duz_steka:=duz_steka-1;
                    if duz_steka=0 then
                        lKraj:=true
                    else
                        stek[duz_steka].st2:=stek[duz_steka].st2+1
                end
                else begin // stanje nije bilo definisano
                    duz_steka:=duz_steka+1;
                    stek[duz_steka].st1 := s;
                    stek[duz_steka].ui := ui;
                    stek[duz_steka].st2 := 1;
                end;
            end; {while}
            if tmp=0 then begin
                writeln('Broj stanja automata', max_st);
                Break;
            end;
        end; {for}
end;
begin
    UcitajPut;
    SintezaOptimalnogAutomata;
end.
```


Karakteristični broj automata opisanog stekom je broj $kb = 1s_1s_2\dots s_k0\dots 0$, koji ima n cifara, s_i predstavlja broj stanja koji odgovara elementu na steku u trenutku pre izvršavanja koraka označenog labelom t:

$(s_{k-1}, a'_k, b'_k, s_k) \leftarrow$ vrh steka

$\dots$

(s_2, a'_3, b'_3, s_3)

(s_1, a'_2, b'_2, s_2)

$(1, a'_1, b'_1, s_1) \leftarrow$ dno steka

Primetimo da karakteristični broj predstavlja broj zapisan u osnovi o koja je jednaka max_st+1. Tokom izvršavanja programa, karateristični broj se menja, čineći niz brojeva zapisanih u osnovi o: $kb_1, kb_2, \dots, kb_n, \dots$. Lako se pokazuje da je taj niz brojeva strogo rastući.

Lema 5. *Program OptimalniAutomat završava rad.*

Dokaz: Dodelimo svakom trenutku pre izvršavanja koraka obeleženog labelom t, broj $b_i = 1\underbrace{oo\ldots o}_{n-1 \text{ puta}} - kb_i$, gde je o osnova brojnog sistema i jednaka je vrednosti max_st u trenutku i. Niz brojeva $b_1 b_2 \ldots b_i \ldots$ je strogo opadajući, pošto je niz brojeva $kb_1, kb_2 \ldots$ strogo rastući. Kako procedura ok_stanje završava rad nakon najviše $|\mathfrak{t}|$ koraka, a pusti_aut nakon najviše $|\mathfrak{t}|^2$ koraka, i kako je niz $b_1, b_2, \ldots b_i, \ldots$ strogo opadajući, a $b_i \geq 0$ za svako i, sledi da program završava rad.

Neka je $kb_i^* = 1o_1o_2 \ldots o_{k-1}o_k$ *modifikovan karakterističan broj* u trenutku i, koji smo dobili tako što smo odbacili nule na kraju karakterističnog broja. Primetimo da je $kb_i^*]_k = kb_{i+1}^*]_k$ ako $o_k < o - 1$ a o osnova brojnog sistema.

Pokažimo da, ako je u trenutku i modifikovani karakteristični broj $kb_i^* = 1o_1o_2 \ldots o_k$, za koji važi da je $o_k < o$, onda se cifra o_k menja pre cifara $o_1o_2 \ldots o_{k-1}$. Pretpostavimo suprotno, tj. da se neka od cifara $o_1o_2 \ldots o_{k-1}$ menja pre cifre o_k. Neka je i' prvi trenutak kome su se cifre $o_1o_2 \ldots o_{k-1}$ promenile pre cifre o_k, i neka su $kb_{i'-1}^* = 1o_1o_2 \ldots o_{k-1}o_k \ldots o_l$ i $kb_{i'}^* = 1o_1'o_2' \ldots o_{k-1}'o_k \ldots o_m'$ modifikovani karakteristični brojevi u trenutku $i'-1$ i u trenutku i'. Da bi se cifre $o_1o_2 \ldots o_{k-1}$ promenile u trenutku i' neophodno je da $o_k = o$, što je suprotno pretpostavci da je $o_k < o$. Time smo pokazali da se cifra o_k mora promeniti pre cifara $o_1o_2 \ldots o_{k-1}$.

Na osnovu prethodnog tvrđenja, lako se pokazuje sledeća lema:

Lema 6. *Ako je u nekom trenutku modifikovani karakteristični broj* $kb_i^* = 1o_1o_2 \ldots o_k$, *gde je* $o_k < o$, *tada će u nekom narednom trenutku biti* $kb_j^* = 1o_1o_2 \ldots (o_k + 1)$ *ili će biti pronađen automat.*

Na osnovu navedenih tvrđenja i lema možemo formulisati sledeću teoremu:

Teorema 11. *Algoritam sinteze optimalnog automata rekurzijom sintetiše optimalni automat koji prolazi zadati put.*

Dokaz: Pretpostavimo da optimalni automat ima o stanja. Svakom optimalnom automatu, dodelimo odgovarajući karakteristični broj na sledeći način: neka je funkcionisanje optimalnog automata za par (α, β) ulaznu i izlaznu nisku $(a_1, q_{i_1}, q_{i_2}, b_1), (a_2, q_{i_2}, q_{i_3}, b_2), \ldots (a_n, q_{i_n}, q_{i_{n+1}}, b_n)$. Iz niza izbacimo višestruka pojavljivanja istih četvorki, tako što ostavljamo prvo pojavljivanje četvorke. Iz takvog niza generišemo karakteristični broj $kb_j^* = 1o_1o_2 \ldots o_l$, gde je $q_{i_1} = 1$, $q_{i_2} = o_1 \ldots$. Neka je kb^* najmanji karakteristični broj od svih optimalnih automata sa o stanja. Dokažimo da algoritam sintetiše automat sa karakterističnim brojem kb^*. Posmatrajmo karakteristične brojeve koje dobijamo tokom rada algoritma. Neka je i poslednji trenutak u kome kb^* ima najveći zajednički početak sa kb^*_i. Neka je zajednički početak dužine $k < l$ i neka je $kb^* = 1o_1 \ldots o_k o'_{k+1} \ldots$ a $kb^*_i = 1o_1 \ldots o_k o''_{k+1} \ldots$. Ako je $o'_{k+1} < o''_{k+1}$ tada je u nekom prethodnom trenutku j moralo da bude $o'_{k+1} = o''_{k+1}$, čime smo pokazali da je najveći zajednički početak dužine $k+1$ što je nemoguće. Ako je $o'_{k+1} > o''_{k+1}$ tada je $o''_{k+1} = o$ (zato što je to poslednje pojavljivanje sa najvećim zajedničkim početkom), odakle sledi da je $o'_{k+1} > o$ što je nemoguće. Time smo pokazali da najveći zajednički početak mora imati bar n stanja, čime je tvrđenje dokazano.

2.6 Optimizacija maksimalnog automata algoritmom slučajnog pretraživanja

Postupak sinteze optimalnog automata pomoću homomorfizma, opisan u 2.3, nije pogodan zato što je za put dužine n čiji optimalni automat ima k stanja, potrebno ispitati najmanje $1+2^n+3^n+\ldots+(k-1)^n+1$ a najviše $1+2^n+3^n+\ldots+(k-1)^n+k^n$ homomorfizama.

Algoritam naveden u 2.3 možemo modifikovati tako što nećemo ispitati sve moguće homomorfizmime, nego ćemo, za fiksiran broj l, $1 \leq l < n$ (n - dužina puta) generisati na slučajan način homomorfizme maksimalnog automata na automate sa l stanja. Ako pronađemo homomorfizam, time smo pronašli i automat sa l stanja koji prolazi zadati put. Ako za fiksiran broj (npr. 1000000) tako generisanih homomorfizama dobijemo automat sa l stanja onda postupak možemo nastaviti tako što ćemo probati da pronađemo automat sa $l-1$ stanja. Ako nismo pronašli automat sa l stanja onda pretpostavljamo da automat sa l stanja, koji prolazi zadati put, ne postoji. Na taj način dobijamo algoritam koji prikazan sledećom procedurom, koja u programu iz 2.3 menja proceduru SintezaOptimalnogAutomata:

```
procedure OptimizacijaMaksimalnogAutomata;
var
        min_aut, i, j, t: integer;
        flag, f1: boolean;
        vr, vr1: string;
begin
        Randomize;
        br_st := size - 1;
        min_aut := 0;
```

```
        razbijanje[1] := 1;
        t := 1000000; // milion pokusaja
        for i := 1 to t do begin
                // generisi slucajno pokrivanje
                for j := 2 to size do
                        pokrivanje[j] := random(br_st) + 1;

                f1 := formiraj();
                if f1 and izvrsava() then begin // pronasli smo pokr.
                        zapamti_min_aut(i); // zapamti trenutno optimalni automat

                        if br_st > 0 then begin
                                min_aut := min_aut + 1;
                                br_st := br_st - 1;
                        end; {if}
                end; {if}
                if br_st = 0 then
                        break;
        end;{for}

        // prikazi broj stanja optimizovanog automata
        writeln('Broj stanja optimizovanog automata: ',br_st+1);
end; {OptimizacijaMaksimalnogAutomata}
```

Da navedeni algoritam ne mora dati optimalni automat je jasno iz same prirode algoritma, pošto je algoritam jedna varijanta slučajnog pretraživanja. Lako je primetiti da je vreme potrebno za optimizaciju linearno u odnosu na dužinu puta n.

2.7 Optimizacija maksimalnog automata sažimanjem

U ovom poglavlju će biti data modifikacija algoritma rekurzije. Iz algoritma rekurzije je eliminisan deo koji vodi do rekurzije, čime se eliminiše eksponencijalna zavisnost vremena od broja stanja optimalnog automata, ali zato optimizacijom ne moramo dobiti optimalni automat. Sledi procedura koja menja proceduru SintezaOptimalnogAutomata u programu iz 2.5:

```
procedure OptimizacijaSazimanjem;
var
        lKraj: boolean;
        tmp, k: integer;
```

```
        max_st: integer;
begin
        duz_steka := 1;
        stek[1].st1 := 1;
        stek[1].st2 := 1;
        stek[1].ui := inout[1];
        max_st := 1;
        lKraj := false;
        while not lKraj do begin
                tmp := pusti_aut();
                if  tmp=0 then
                        Break; // pronasli smo automat
                if tmp = -1 then
                        stek[duz_steka].st2 := stek[duz_steka].st2 + 1
                else begin
                        // stanje nije bilo definisano
                        duz_steka := duz_steka + 1;
                        stek[duz_steka].st1 := s;
                        stek[duz_steka].ui := ui;
                        stek[duz_steka].st2 := s;
                end;
        end; {while}
end;
```

Lako je pokazati da ovaj algoritam u najgorem slučaju radi u vremenu an^2+bn+c gde su a, b, c neke konstante koje zavise od puta. Da algoritam ne mora davati optimalni automat može se videti na primeru kretanja (news,n),(news,e),(news,n).

3 Neke karakteristike algoritama sinteze optimalnog automata

U prethodnom delu smo opisali algoritme sinteze optimalnog automata, kao i neke algoritme optimizacije maksimalnog automata. U ovom delu ćemo analizirati neke karakteristike navedenih algoritama.

3.1 Vremenska ocena algoritama sinteze optimalnog automata

U ovom odeljku ćemo opisati zavisnost vremena izvršavanja algoritama od dužine puta i broja stanja optimalnog automata koji izvršava zadato kretanje. Za algoritme sinteze optimalnih automata ćemo ustanoviti da ta zavisnost eksponencijalna. Za algoritme optimizacije maksimalnog automata ćemo pokazati da je zavisnost vremena od dužine puta polinomijalna, ali i da njihova preciznost opada sa dužinom puta.

Ispitajmo algoritam sinteze optimalnog automata pomoću homomorfizma. Ako je k broj stanja optimalnog automata koji izvršava zadato kretanje $\mathfrak{k}$ a n dužina puta, tada je prosečan broj homomorfizama koje treba proveriti jednak $1 + 2^n + \ldots + (k-1)^n + \frac{1}{2} \cdot k^n$. Vreme se može poboljšati ako znamo da je broj stanja optimalnog automata jednak k, i tada je prosečan broj homomorfizama koje treba proveriti $\frac{1}{2}k^n$.

Algoritam sinteze optimalnog automata pomoću kompatibilnih skupova ima slične osobine kao i algoritam sinteze pomoću homomorfizma. Algoritam prvo formira skup svih kompatibilnih skupova, i nakon toga pokušava da od njih formira zatvoreno razbijanje sa $1, 2, \ldots, k$ elemenata, gde je k broj stanja optimalnog automata. Ako je broj kompatibilnih skupova b, tada je potrebno proveriti $C_b^1 + C_b^2 + \ldots + C_b^k$ kompatibilnih skupova. Kako b može da raste eksponencijalno sa dužinom puta n to najgorem slučaju moramo proveriti $C_{2^n}^1 + C_{2^n}^2 + \ldots + C_{2^n}^k$ skupova.

Algoritam sinteze optimalnog automata pomoću rekurzije u najgorem slučaju će proveriti $1^n + 2^n + \ldots + k^n$ automata, gde je n dužina puta a k broj stanja optimalnog automata.

Za algoritam slučajnog pretraživanja prikazanog u odeljku 2.6 se može pokazati da je vreme $C \cdot n \cdot t$. gde je t broj pokušaja, a n dužina puta. Kako je obično t fiksirano, to možemo reći da je ocena $O(n)$.

Za algoritam optimizacije maksimalnog automata sažimanjem je rečeno da je vreme potrebno za sintezu u najgorem slučaju $an^2 + bn + c$. Vreme može biti i kraće, što pokazuje primer puta $(news, n), (news, n), \ldots, (news, n)$, za koji se optimizacija izvršava u linearnom vremenu.

U praksi je najbolje rezultate postigao algoritam sinteze optimalnog automata pomoću rekurzije. Algoritam sinteze optimalnog automata je pokazao lošije rezultate, a najlošije rezultate je pokazao algoritam sinteze optimalnih automata pomoću kompatibilnih skupova.

Algoritmi su testirani tako što je generisano po 100 slučajnih puteva u ravnom pravougaonom lavirintu, dužina od 5 do 20 (ukupno 1600 puteva). Potom su primenjivani algoritmi sinteza optimalnih automata na generisane puteve. Jedino je algoritam sinteze optimalnog automata pomoću rekurzije uspeo da završi sintezu optimalnih automata, za puteve svih dužina.

Dužina puta	Pros. vreme
5	0,002 sekundi
6	0,002 sekundi
7	0,002 sekundi
8	0,003 sekundi
9	0,005 sekundi
10	0,011 sekundi
11	0,076 sekundi
12	0,100 sekundi
13	2,093 sekundi
14	1,535 sekundi
15	1,869 sekundi
16	57,813 sekundi
17	1,631 sekundi
18	48,136 sekundi
19	53,375 sekundi
20	133,080 sekundi

U prethodnoj tabeli se može primetiti da je u nekim slučajevima, npr. za puteve dužine 17, prosečno vreme sinteze optimalnog automata manje nego za puteve dužine 16. Uzrok tome je činjenica da su putevi generisani slučajno, i da je za puteve dužine 16, sintetisano 9 optimalnih automata sa 5 stanja, 19 optimalnih automata sa 4 stanja, 48 optimalnih automata sa 3 stanja i 24 optimalna automata sa 2 stanja. Za puteve dužine 17 dobijeno je 29 optimalnih automata sa 4 stanja, 50 optimalnih automata sa 3 stanja i 21 optimalni automat sa 2 stanja.

Nešto preciznija slika se stiče kada se grupišu putevi po broju stanja optimalnog automata. Međutim, ni tu ne dobijamo pravu sliku zbog malog broja puteva sa odgovarajućim optimalnim automatima koji imaju 6 i 7 stanja.

Br. stanja opt. automata	Pros. vreme	Br.puteva
1	0,001 sekundi	140
2	0,005 sekundi	619
3	0,241 sekundi	581
4	16,709 sekundi	214
5	624,159 sekundi	39
6	284,76 sekundi	6
7	203,272 sekundi	1

Algoritam sinteze optimalnih automata pomoću homomorfizma daje lošije rezultate nego algoritam sa rekurzijom. Algoritam pomoću homomorfizma je uspešno odredio optimalne automate za puteve dužina do 5 do 12. Algoritam je prekidan nakon 3 sata ako u međuvremenu nisu određeni optimalni automati.

Dužina puta	Pros. vreme
5	0,0003 sekundi
6	0,0003 sekundi
7	0,0020 sekundi
8	0,0033 sekundi
9	0,0176 sekundi
10	0,0525 sekundi
11	6,5676 sekundi
12	47,7642 sekundi

Algoritam sinteze optimalnih automata pomoću kompatibilnih skupova daje najlošije rezultate. Naime, primenom ovog algoritma na slučajno generisane puteve dobijeni su rezultati koji za puteve dužina 5 do 9. Algoritam je prekidan nakon 3 sata, ako u međuvremenu nije dao optimalne automate. Rezultati su prikazani u sledećoj tabeli:

Dužina puta	Pros. vreme
5	0,003 sekundi
6	0,017 sekundi
7	0,265 sekundi
8	2,769 sekundi
9	49,312 sekundi

3.2 O tačnosti algoritama optimizacije

U prethodnom delu je rečeno da algoritmi optimizacije maksimalnog automata koji izvršavaju zadato kretanje t, rade u polinomijalnom vremenu (linearnom i kvadratnom). Negativna strana navedenih algoritama je činjenica da se optimizacijom ne mora doći do optimalnog automata.

Kako algoritam slučajnog pretraživanja 'prekriva' sve moguće homomorfizme pomoću t pokušaja da bi se došlo do optimalnog automata (ili automata bliskom njemu), to možemo pretpostaviti da ako raste broj homomorfizama a t ostaje fiksirano, dolazi do sve slabijeg 'prekrivanja' homomofizama, a samim tim i algoritam postaje sve neprecizniji.

Da bi potvrdili tu pretpostavku, izvršena je optimizacija automata pomoću algoritama za slučajno generisane puteve dužina od 5 do 20 (po 100 za svaku dužinu puta). Dobijeni rezultati su upoređeni sa rezultatima dobijenim pomoću algoritma rekurzije. Rezultati potvrđuju pretpostavku da sa porastom dužine puta dolazi do pada tačnosti algoritma slučajnog pretraživanja. U sledećoj tabeli je prikazano prosečno odstupanje grupisano po dužini puteva:

Dužina puta	Pros. odstupanje	Pros. vreme
5	0,02	5,08
6	0,03	10,02
7	0,04	12,11
8	0,01	14,50
9	0,03	17,91
10	0,03	18,50
11	0,04	19,52
12	0,05	20,21
13	0,05	20,70

14	0,05	20,11
15	0,07	21,18
16	0,07	21,86
17	0,06	23,98
18	0,27	24,12
19	0,21	23,92
20	0,41	24,30

Slične rezultate dobijamo ako grupišemo odstupanja po broju stanja optimalnog automata

Br. stanja opt. automata	Prosečno odstupanje
1	0,00
2	0,04
3	0,09
4	0,21
5	0,51
6	0,67
7	1,00

Algoritam optimizacije sažimanjem radi tako što pokušava da za tekuće stanje, dodefiniše funkcije prelaska i izlaska, tako da izvršava što veći deo puta. Ako u nekom trenutku dođe do kontradikcije (kada automat da pogrešan izlaz) onda se uvodi novo stanje. Na osnovu postupka se može zaključiti da bi sa porastom dužine puta došlo i do povećanja greške optimizacije. Da je to zaista tako, pokazuje primer optimizacije automata za 1600 slučajno generisanih puteva dužina od 5 do 20:

Dužina puta	Pros. odstupanje	Pros. vreme
5	0,05	0,002 sekundi
6	0,10	0,002 sekundi
7	0,20	0,002 sekundi
8	0,29	0,002 sekundi
9	0,36	0,002 sekundi
10	0,50	0,002 sekundi
11	0,69	0,002 sekundi
12	0,92	0,003 sekundi
13	0,89	0,002 sekundi
14	1,13	0,004 sekundi
15	1,26	0,004 sekundi
16	1,39	0,004 sekundi

17	1,70	0,004 sekundi
18	1,57	0,004 sekundi
19	1,93	0,005 sekundi
20	2,34	0,004 sekundi

Slične rezultate dobijamo ako grupišemo rezultate po broju stanja optimalnog automata.

Br. stanja opt. automata	Prosečno odstupanje
1	0,00
2	0,65
3	1,22
4	1,54
5	2,03
6	1,67
7	2,00

Poređenjem algoritam optimizacije slučajnim pretraživanjem i algoritma optimizacije sažimanjem, dolazimo do zaključka je algoritam optimizacije sažimanjem tačniji. Poređenje je vršeno za fiksirani broj pokušaja (1.000.000) kod algoritma optimizacije slučajnim pretraživanjem, a rezultati su prikazani u sledećoj tabeli:

duz puta	vr.saz	saz.br.stanja	rnd.br.stanja	vr.rnd
5	0.002	1.53	1.50	5.080
6	0.002	1.86	1.79	10.020
7	0.002	2.18	2.02	12.110
8	0.002	2.33	2.05	14.500
9	0.002	2.62	2.29	17.910
10	0.002	2.89	2.42	18.500
11	0.002	3.31	2.66	19.520
12	0.003	3.64	2.77	20.210
13	0.002	3.53	2.69	20.700
14	0.004	3.92	2.84	20.110
15	0.004	4.26	3.07	21.180
16	0.004	4.52	3.20	21.860
17	0.004	4.78	3.14	23.980
18	0.004	4.89	3.59	24.120
19	0.005	5.46	3.74	23.920
20	0.004	5.75	3.82	24.300
30	0.010	7.98	6.74	74.656
40	0.014	10.76	11.61	103.002

50	0.022	13.71	17.28	120.422
60	0.029	16.34	24.35	150.066
70	0.040	19.01	33.25	166.271
80	0.053	22.13	42.88	187.831
90	0.068	24.81	55.32	212.083
100	0.082	27.03	68.38	232.268
125	0.116	33.71	106.13	284.896
150	0.179	40.20	142.68	341.523
200	0.387	54.15	199.69	429.343
500	3.899	133.31	-	-
750	10.941	201.25	-	-
1000	24.197	266.70	-	-

U drugoj koloni se nalaze prosečno vreme potrebno za optimizaciju maksimalnog automata algoritmom sažimanja. U trećoj koloni se nalazi prosečan broj stanja automata posle optimizacije algoritmom sažimanja. U četvrtoj koloni je prikazan prosečan broj stanja posle optimizacije algoritmom slučajnog pretraživanja. U petoj koloni je prikazano prosečno vreme potrebno za optimizaciju algoritmom slučajnog pretraživanja.

Na osnovu rezultata se može zaključiti da algoritmom optimizacije se dobija automat sa brojem stanja koji se približava četvrtini broja stanja maksimalnog automata. Pored toga, može se primetiti da je algoritam optimizacije slučajnim pretraživanjem tačniji od algoritma opotimizacije sažimanjem do puteva dužine 30. Nakon toga, tačnost algoritma optimizacije slučajnim pretraživanjem naglo opada, da bi za maksimalne automate sa 200 stanja dobili vrlo malu tačnost. Rezultati navedeni u prethodnoj tabeli potvrđuju da je vremenska ocena algoritam optimizacije sažimanjem $O(n^2)$, a algoritma slučajnim pretraživanjem $O(n)$.

3.3 Zaključak

Svi algoritmi sinteze optimalnog automata pokazuju eksponencijalnu zavisnost vremena od broja stanja optimalnog automata koji prolazi zadati put. Već za puteve sa odgovarajućim optimalnim automatima sa 8 i više stanja postaje teško određivanje optimalnih automata. Zato se uvode algoritmi optimizacije maksimalnih automata. Primenom tih algoritama se ne garantuje optimalnost dobijenih rezultata, ali se dobija automat koji ima manji broj stanja od maksimalnog.

Svi prikazani algoritmi sinteze optimalnog automta, i algoritama optimizacije imaju svoje mane i prednosti. Od algoritama sinteze optimalnih automata kao najefikasniji se pokazao algoritam koji se koristi rekurzijom. Što se algoritama optimizacije tiče, algoritam optimizacije sažimanjem se pokazao tačnijim od algoritma slučajnog pretraživanja za fiksirani broj pokušaja. Sa druge strane, algoritam slučajnog pretraživanja se lako može paralelizovati, a uvećanjem broja pokušaja može se uvećavati tačnost optimizacije.

Spisak literature

[1] Antelmann H., Budach L., Rollik H.A. On universale traps, Elektronische Informationsverarbeitunf und Kybernetik, V. 15, No. 3, pp 123-131 (1979).

[2] Antelmann H., An application of the prime number theorem in automata theory, ICS PAS Reports 411, pp 9-11 (1980).

[3] Blum M., Sakoda W., On the capability of finite automata in 2 and 3 dimensional space, The Proceedings of the 18th Annual Symposium on Foundations of Computer Science, pp 147-161 (1977).

[4] Blum M., Kozen D., On the power of the compass, The Proceedings of th 19th Annual Symposium on Foundations of Computer Science, pp 132-142 (1978).

[5] Budach L., On the Solution of the Labirinth Problem for Finite automata, Elektronische Informationsverarbeitung und Kybernetik, pp. 132-142 (1978).

[6] Budach L., Environments, labyrinths and automata, Lecture notes ins Computer science 56, Springer, 1977, pp 54-64.

[7] Budach L., Automata and labyrinths, Math. Nachrichten 86, pp. 195-282 (1978)

[8] Budach L., Counterautomata in Mazes, Proc Workshop ACT, Poznan (1979)

[9] Budach L., Two pebles don't suffice, Foundation od Computing Theory 79. V.1. Berlin, pp. 578-589 (1980)

[10] Bull M., Hemmerling A., Finite embedded trees and and simply connected mazes cannot be searched by halting finite automata, Journal of inf Process and Cibern. EIK, V.26. No. 1-2, pp 65-73 (1990).

[11] Calude E., Lipponen M.: Minimal Deterministic Incomplete Automata, Journal of Universal Computer Science, vol. 3, No 11, pp 1180-1193 (1997).

[12] Danecki R., Karpinski M., Decidability results on plane automata searching mazes, Proc. 2nd Int. FCT'79 Berlin: Conf. Akademie Verlag, pp 84-91 (1979).

[13] Hemmerling A., Three-Dimensional Traps and Barrages for Cooperating Automata, Lecture Notes in Computer Science 278, Berlin, Springer-Verlag, pp 197-203 (1987).

[14] Hemmerling A., Normed Two-Plane Traps for Finite Systems of Cooperating Compass Automata, J.Inf.Process Cybern. EIK., V.28. No. 8-9, pp 453-470.

[15] Kilibarda V.: O izomorfizmima automata, magistarski rad na PMF u Beogradu, 1991.

[16] Kobrinskii N.E., Trakhtenbrot B.A.: Introduction to the theory of finite automata, Amsterdam, North-Holland, 1965.

[17] Kudrjavcev V.B., Podkolzin A.S., Ušćumlić Š.: Uvod u teoriju apstraktnih automata, Beograd, Naučna knjiga 1986.

[18] Kudryavtsev V.B., Ushchumlich Sh., Kilibarda G.: On behavior of automata in labyrinths, Discrete Math. Appl., Vol.3, No. 1, pp. 1-28 (1993).

[19] Muler H., Automata catching labyrinths with at most three components, Elektronische Informationsverarbeitung und Kybernetik, V.15 No. 1/2, pp 3-9 (1979).

[20] Nelson R.J.: Introduction to Automata, New York, John Wiley & Sons, 1968.

[21] Pfleeger C.F.: State reduction in incompletely specified finite state machines. IEEE Trans. computers, C-22:1099-1102, 1973.

[22] Reusch B., Merzenich W.: Minimal Coverings for Incompletely Specified Sequential Machines, Acta Informatica 22, 663-678 (1986).

[23] Salomaa A.: Theory of Automata, Oxford, Pergamon Press, 1969.

[24] Shannon Cl.E., Presentations of a maze-solving machine, Cybernetics Trans. of the 8th Conf. of the Josiah Macy Jr. Found, Editor: H. Foerester, pp 173-180 (1951).

[25] Szepietowski A. A finite 5-pebble automaton can search every maze, Information Processing Letters, V.15. No. 1/2, pp 199-204. (1982).

[26] Богомолов С.А., Золотых А.А., Зыричев А.Н., Автоматы и графы, Саратов: Изд-во Саратовского ун-та, 1992. - 180 с.

[27] Килибарда Г., О сложности автоматного обхода лабиринтов Дискретная математика - 1993. Т. 5, вып. 3. - с. 116-124.

[28] Килибарда Г. О минимальных универсалъных колективах автоматов для плоских лабиринтов, Дискретная математика - 1994. Т. 6, вып. 4. - с. 133-153.

[29] Килибарда Г. Об универсалъных лабиринтах-ловушках для конечных множеств автоматов, Дискретная математика - 1990. Т. 2, вып. 1. - с. 72-79.

[30] Килибарда Г. Об обходе конечных лабиринтов системами автоматов, Дискретная матемтика - 1990. Т. 2, вып. 2. - с. 71-82.

[31] Килибарда Г. Новое доказательство теоремы Будаха-Подколзина, Дискретная матемтика - 1991. Т. 3, вып. 3. - с. 135-146.

[32] Килибарда Г.Ф., Ушчумлич Ш. О лабиринтах-ловушках для колективов автоматов (в печати)

[33] Кристофидес Н.: Теория графов - алгоритмический подход, Москва, Издательство "Мир", 1978.

[34] Кудрявцев В.Б., Алешин С.В., Подколзин А.С., Введение в теорию автоматов, М. Наука, 1985.

X [35] Ушчумлич Ш. О лабиринтах-ловушках для колективов автоматов (в печати)

[36] Ушчумлич Ш.М. О некоторых свойствах алгоритмов анализа и синтеза автоматов. В кн.: Методы и системы технической диагностики, т. 1. Саратов: Изд-во Саратовского ун-та, 1980.

[37] Ушчумлич Ш.М. Исследование некоторых алгоритмов синтеза и анализа автоматов. В кн.: Методы и системы технической диагностики, ДАН СССР, 1979, 247, Но 3, 561-565.