Formalni jezici i jezični procesori I

AUTOMATI

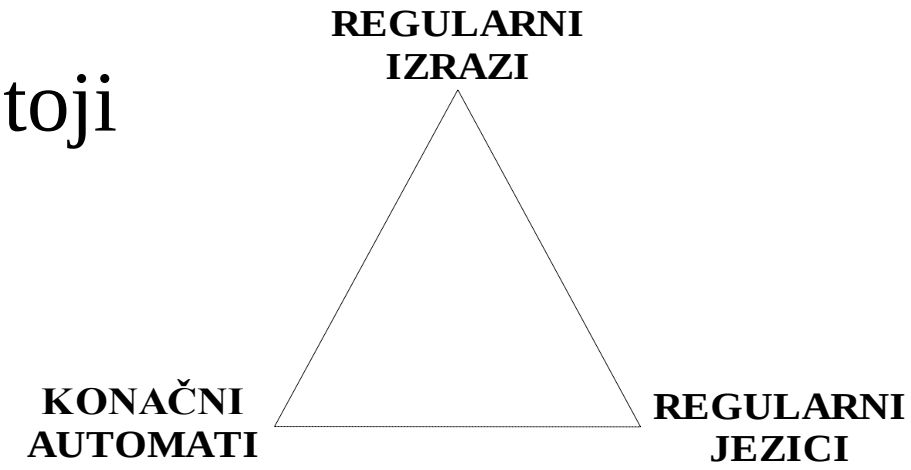
Prof. dr. sc. Sanda Martinčić - Ipšić
smarti@inf.uniri.hr

Automati

- Konačni automati
  - ***DKA –*** deterministički konačni automat
  - ***NKA*** – nedeterministički konačni automat
  - ***ε - NKA*** – nedeterm. konačni automat s ε prijelazima
  - konačni automati s izlazom
    - Mooreov automat
    - Mealyev automat

Odnos automata, regularnih izraza i regularnih jezika

- jezik je regularan ako postoji konačni automat koji ga prihvaća
- za svaki regularni jezik moguće je izgraditi konačni automat koji ga prihvaća
  - regularni izrazi opisuju regularne jezike



Deterministički konačni automat

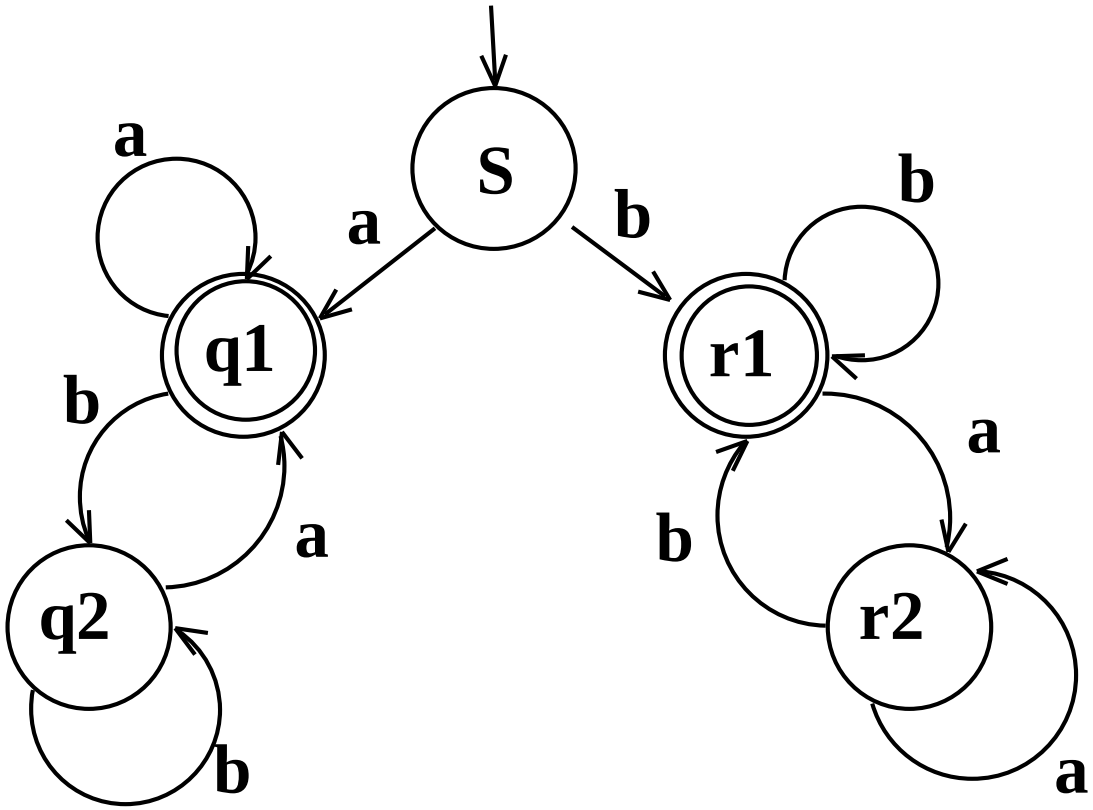
dka = (Q, Σ, δ, q_0, F) gdje je

- Q – konačni skup stanja
- Σ - konačni skup ulaznih znakova
- $\delta : Q \times \Sigma \rightarrow Q$; funkcija prijelaza
  - novo stanje = δ (staro stanje, ulazni znak), novo stanje je jednoznačno određeno
- $q_0 \in Q$; početno stanje
- $F \subseteq Q$, skup prihvatljivih stanja

- prijeđe li konačni automat nakon svih pročitanih znakova niza x iz početnog stanja u jedno od prihvatljivih stanja niz x se prihvaća

PRIMJER I:

- prihvaća nizove koji počinju i završavaju istim znakom
- prihvaća: a, b, aa, bb, aba, bab, bbbab, bbaaaabab.....
- ne prihvaća, ab, ba, bbba, ...



Q ={S, q1,q2,r1,r2}

Σ={a,b}

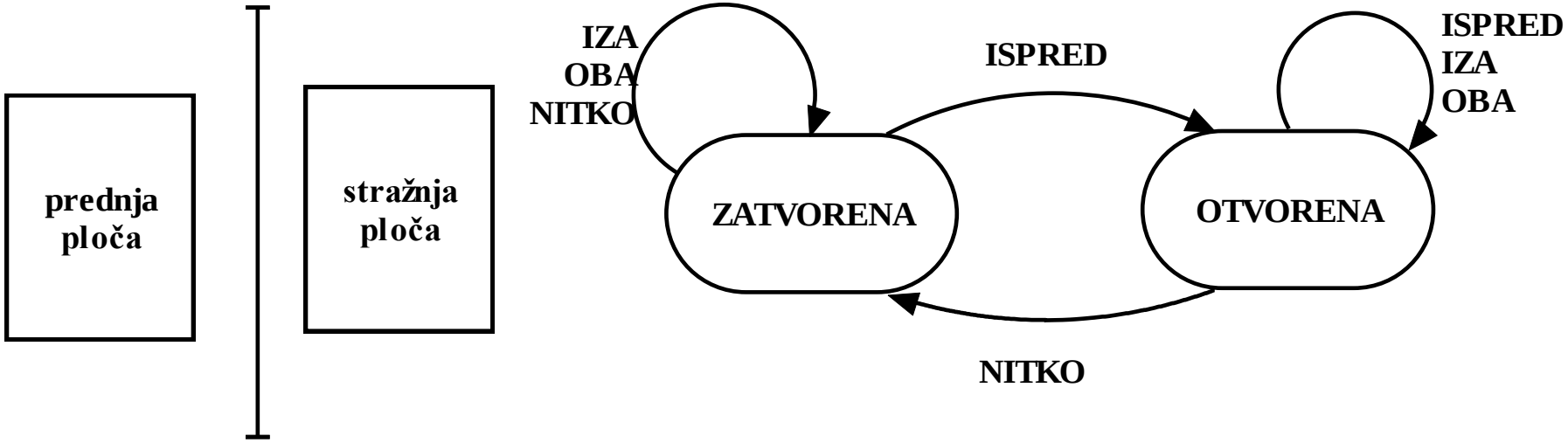
q0 =S

F={q1,r1}

δ: funkcija prijelaza

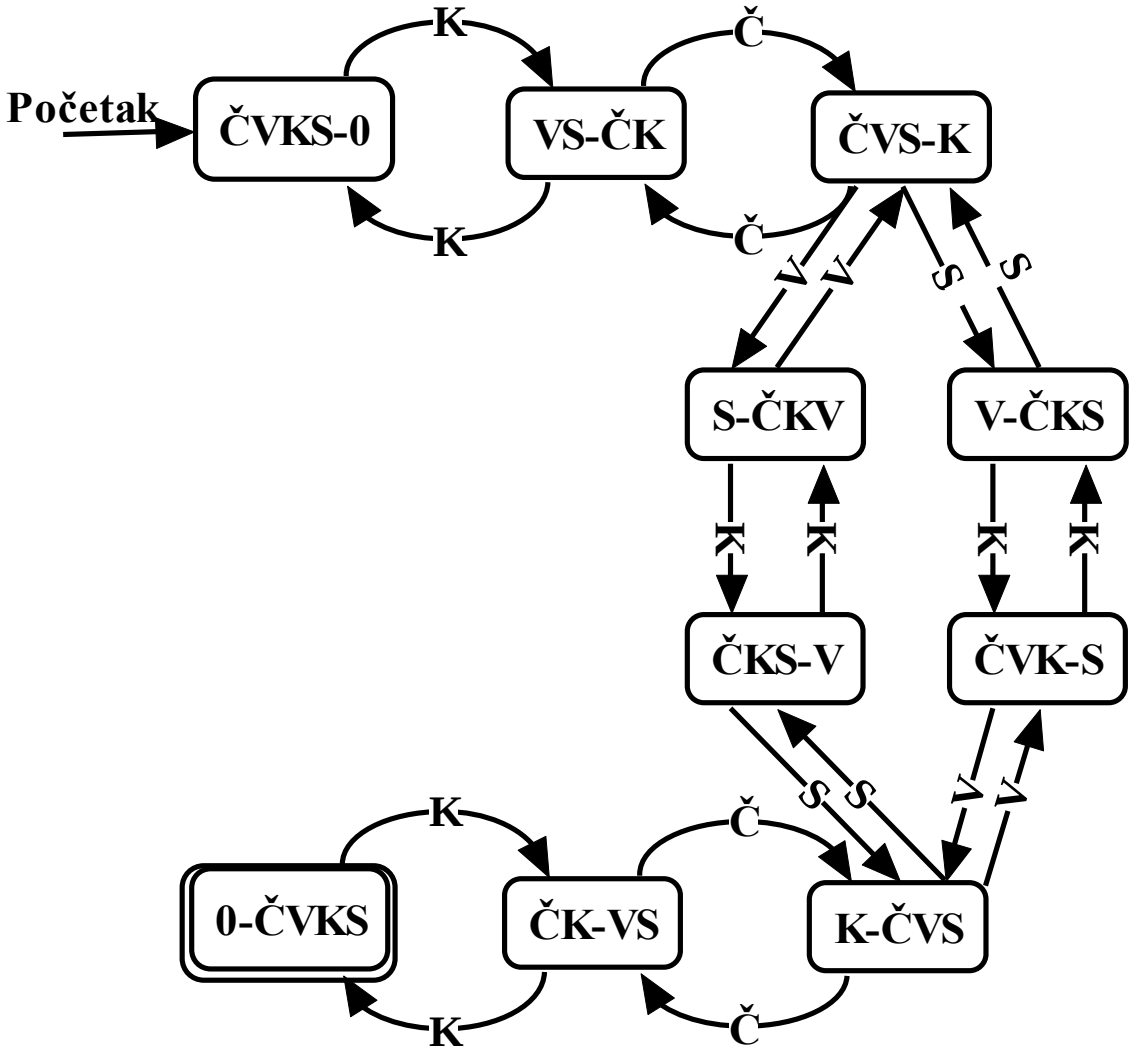
δ	a	b	
→S	q1	r1	0
q1	q1	q2	1
q2	q1	q2	0
r1	r2	r1	1
r2	r2	r1	0

Primjer II: automatska vrata

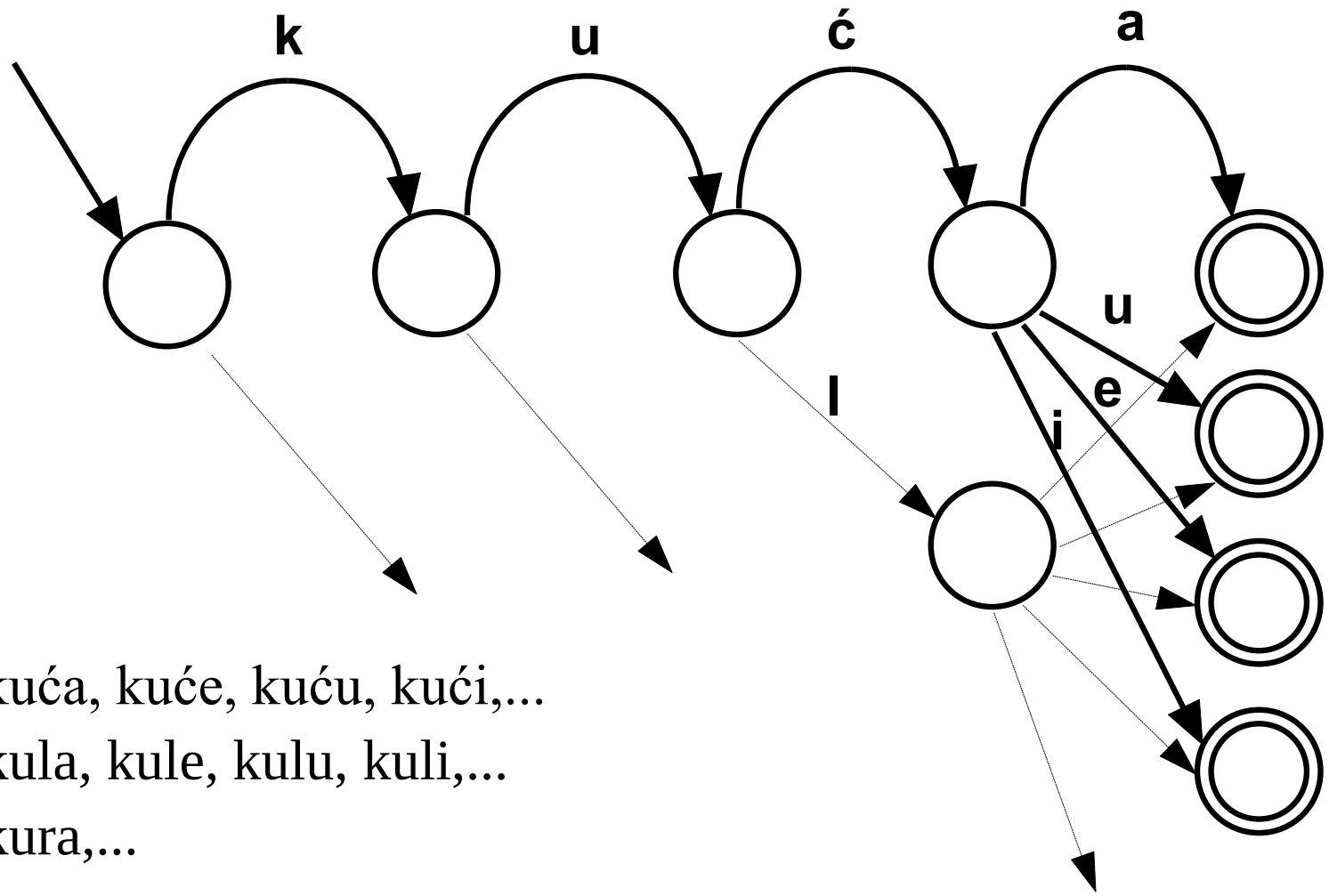


	NITKO	ISPRED	IZA	OBA
ZATVORENA	ZAT	OTVOR	ZAT	ZAT
OTVORENA	ZAT	OTVOR	OTVOR	OTVOR

Primjer III: čovjek, vuk, koza i salata



Primjer IV: morfologija

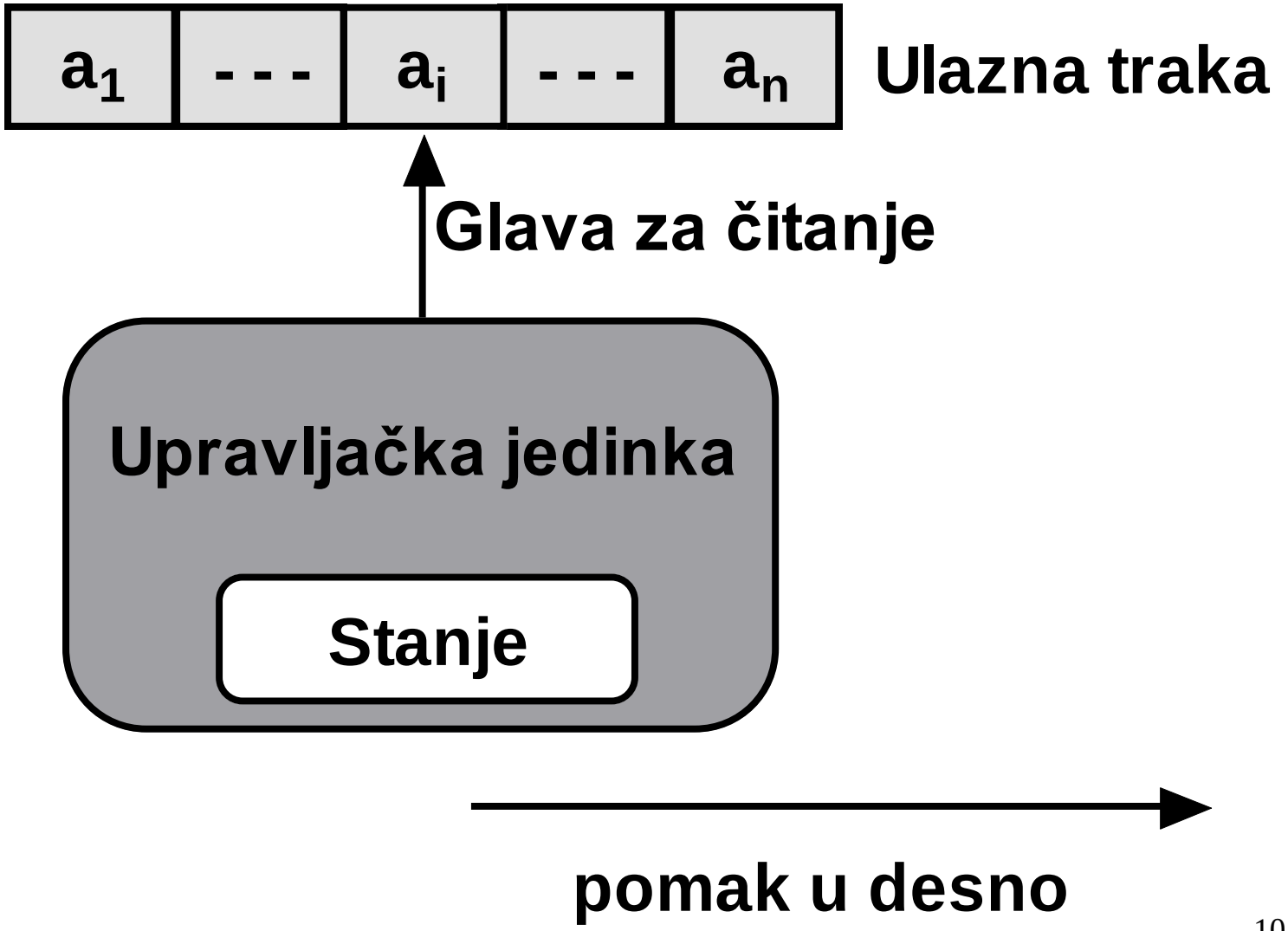


- kuća, kuće, kuću, kući,...
- kula, kule, kulu, kuli,...
- kura,...
- kuma,...

WEB

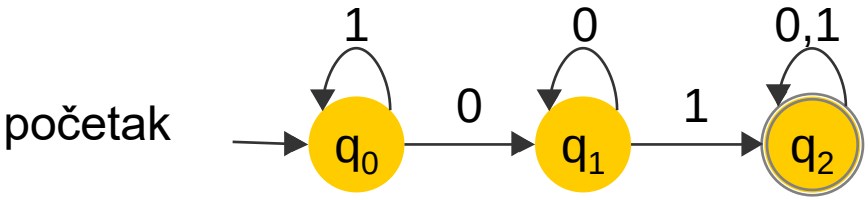
- za regularni izraz prikazuje nedeterministički (NKA) i deterministički (DKA) konačni automat
  - http://osteele.com/tools/reanimator/?detectflash=false

Model konačnog automata



Deterministički konačni automat: primjer

DKA, koji prihvaća sve nizove nula i jedinica koje sadrže podniz 01



tablica prijelaza

	0	1
→ q_0	q_1	q_0
q_1	q_1	q_2
*q_2	q_2	q_2

Formalni postupak prihvaćanja niza

- postupak prihvaćanja simbola raširimo na postupak prihvaćanja niza znakova koji nas iz stanja q dovedu u novo stanje p u tu svrhu uvodimo novu funkciju $\delta': Q \times \Sigma^* \rightarrow Q$, gdje je
  - Σ^* skup svih mogućih nizova; uključujući i prazan niz ε
  - $x,y,w,z \in \Sigma^*$ - nizovi ulaznih znakova i $a,b \in \Sigma$ -ulazni znakovi
  - $p,q \in Q$ stanja te vrijedi:
    1) $\delta'(q, \varepsilon) = q$,
    2) te za bilo koji niz ulaznih znakova w i bilo koji ulazni znak a vrijedi:

       $\delta'(q, wa) = \delta(\delta'(q, w), a)$; gdje je $w \in \Sigma^*$ i $a \in \Sigma$

dokaz: w= ε: $\delta'(q, a) = \delta(\delta'(q, \varepsilon), a) = \delta(q, a)$

Definicija: prihvaćanja niza

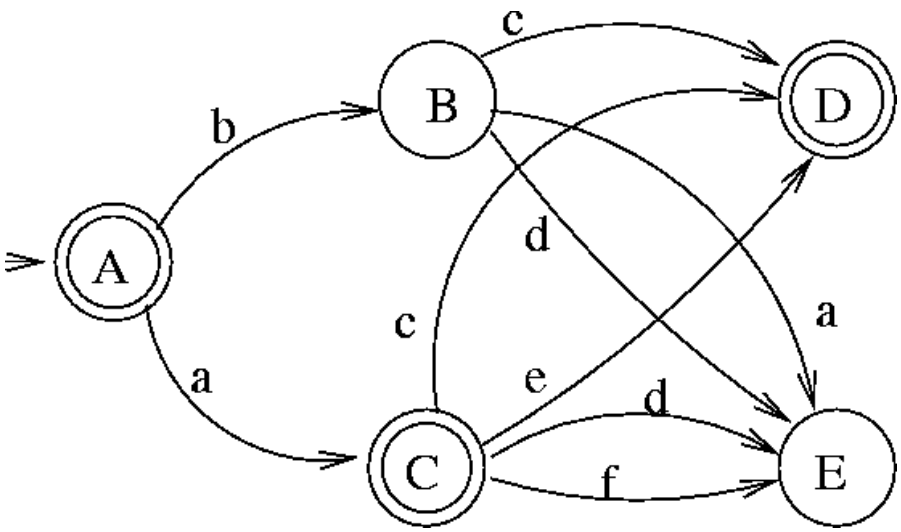
- Deterministički konačni automat **dka = (Q, Σ, δ, q_0, F)** prihvaća niz **x** ako je **$\delta(q_0, x)=p$,** za neki $p \in F$
- DKA prihvaća skup nizova
  - $L(dka)=\{x| \delta(q_0, x)\in F\}$, koji je podskup skupa svih mogućih nizova
  - $L(dka) \subseteq \Sigma^*$
- nizove koji nisu element skupa L(DKA): DKA **NE** prihvaća
- DKA dijeli skup Σ^* na skup koji prihvaća prijelazom u jedno od prihvatljivih stanja, te na skup koji ne prihvaća jer se DKA nakon posljednjeg pročitanog znaka ne nalazi u prihvatljivu stanju

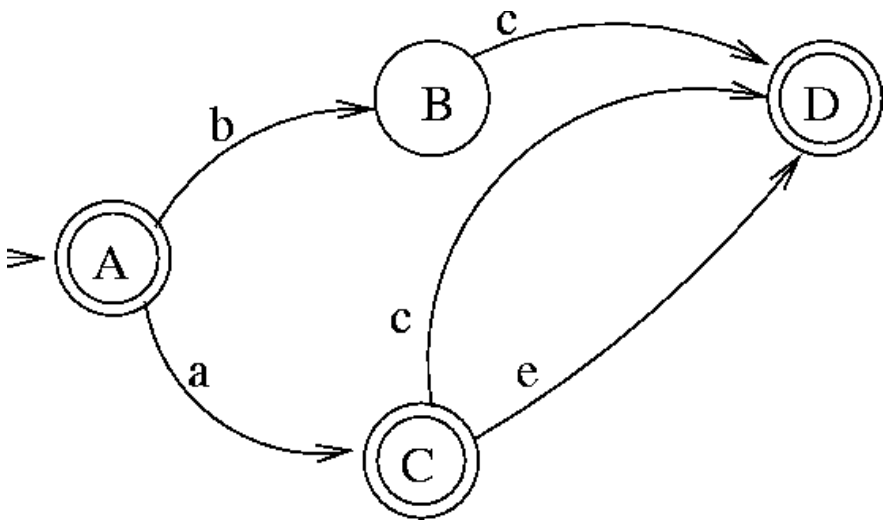
Minimizacija DKA

- za svaki DKA je moguće izgraditi beskonačno mnogo drugih DKA koji prihvaćaju isti jezik, zbog učinkovitosti tražimo DKA s čim manjim brojem stanja
- ***Def.:***

  Za regularan jezik L moguće je izgraditi DKA M koji ima **manji ili jednak broj stanja** od bilo kojeg drugog DKA M´ koji prihvaća isti jezik L.

PRIMJER MINIMIZACIJE





Istovjetnost stanja

- **stanje p** DKA $M = (Q, \Sigma, \delta, q_0, F)$ je istovjetno stanju **p´** DKA $M' = (Q', \Sigma, \delta', q_0', F')$ ako i samo ako DKA M u stanju p **prihvaća isti skup nizova** kao i DKA M´ u stanju p´
- za bilo koji niz $w \in \Sigma^*$ važi: $\delta(p , w) \in F \wedge \delta'(p' , w) \in F'$ ili $\delta(p , w) \notin F \wedge \delta'(p' , w) \notin F'$
- relacija istovjetnosti je tranzitivna
  - ako su stanja p i q te q i r istovjetna onda su p i r istovjetna
- ***Istovjetnost automata DKA:***
  - DKA M i N su istovjetni ako i samo ako su **istovjetna njihova početna stanja**

Primjer:

	c	d	
p_1	p_1	p_4	0
p_2	p_3	p_5	1
p_3	p_5	p_1	0
p_4	p_2	p_3	1
p_5	p_2	p_3	1

	c	d	
p_1	p_1	X	0
p_2	p_3	X	1
p_3	X	p_1	0
X	p_2	p_3	1
X	p_2	p_3	1

	c	d	
p_1	p_1	X	0
p_2	p_3	X	1
p_3	X	p_1	0
X	p_2	p_3	1

- istovjetna stanja se označe istim zajedničkim X
- istim znakom X označe se sva prijelazi istovjetnih stanja u tablici prijelaza
- od svih istovjetnih stanja ostane samo stanje X

Ispitivanje istovjetnosti stanja p i q

- Ispitujemo dva uvjeta:
  - ***podudarnost:*** oba stanja moraju bit prihvatljiva ili oba neprihvatljiva
    - $(p \in F \wedge q \in F)$ ili $(p \notin F \wedge q \notin F)$
  - ***napredovanje:*** za bilo koji ulazni znak a vrijedi da su stanja $\delta(p, a)$ i $\delta(q, a)$ istovjetna
- 3 algoritma za određivanje istovjetnosti:
  - ispitivanjem 2 uvjeta
  - dijeljenjem skupa stanja po uvjetu podudarnosti (u grupe)
  - traženjem neistovjetnih stanja

Pojednostavljanje DKA minimizacijom

- cilj smanjiti broj stanja zadanog DKA
- grupa istovjetnih stanja zamjeni se jedinstvenim stanjem na slijedeći način:
  - istovjetna stanja se označe zajedničkim imenom
  - sve oznake istovjetnih stanja u funkciji prijelaza δ označe se novim zajedničkim imenom (tabela prijelaza)
  - u skupu stanja Q se ostavi jedno od istovjetnih stanja a ostala se izuzmu

Određivanje istovjetnosti ispitivanjem dvaju uvjeta

- za svaki par stanja DKA ispituju se uvjeti podudarnosti i napredovanja (algoritam je neučinkovit)
- *1. korak*: tablica ispitivanja istovjetnosti za svaki ulazni znak ima svoj ulaz a u recima parove stanja koje ispitujemo
- *2. korak*: ispitujemo uvjet podudarnosti za sva stanja u tablici
  - ako uvjet nije ispunjen $\Rightarrow$ par stanja iz prvog retka nije podudaran, stanemo
  - inače odredimo nove parove stanja i nastavimo ispitivanje
- *3. korak*: za novi par stanja
  - ako su ista nema akcije
  - ako su različita, ali postoji zapis u prethodnim recima nema akcije
  - ako su različita, i NE postoji prethodni zapis, upišemo novi par u novi redak tablice
- *4. korak*: ako u 3. koraku nema novog para onda je par iz 1. retka ISTOVJETAN,
  - kao i svi ostali ispitani parovi (uvjet napredovanja) inače pređi na korak 2

Određivanje istovjetnosti dijeljenjem u podskupove po podudarnosti

- *korak 1*: skup stanja podijelimo u dva podskupa:
  - u 1. grupi sva stanja prihvatljiva $p_i \in F$ a u 2. sva stanja neprihvatljiva $q_i \notin F$
- *korak 2*: primjeni se algoritam na podjelu Π
  - za (sve grupe stanja G_j u podjeli Π) {
    - podjeli grupu G_j na podskupove tako da su p i q iz G_j u istom podskupu ako i samo ako za bilo koji ulazni znak a vrijedi:

    $(\delta(p, a) \in G_i \wedge \delta(q, a) \in G_i)$ gdje je G_i jedna od grupa stanja podjele Π

    // za različite ulazne znakove a grupe G_i mogu biti različite
    - označi novu podjelu skupa stanja Π_{nova} }
- *korak 3*:
  - ako podjela ostaje ista $\Pi_{nova} = \Pi$ onda stop, stanja u istim grupama su istovjetna
  - inače $\Pi_{nova} \neq \Pi$ nastavi s korakom 2 dok uvjet $\Pi_{nova} = \Pi$ nije ispunjen

Određivanje istovjetnosti traženjem neistovjetnih stanja

- označi sve parove (p, q) za koje vrijedi $p \in F$ i $q \notin F$;
- za (bilo koji par različitih stanja $(p,q) \in (F \times F)$ ili $(p,q) \notin (F \times F)$){

  ako (za neki znak a par ($\delta(p, a)$, $\delta(q, a)$) jest označen) {

  označi (p,q);

  rekurzivno označi sve neoznačene parove u listi koja je pridružena paru (p,q) i sve ostale parove u listama koje su pridružene parovima označenim u ovom koraku;}

  inače {

  za (svi znakovi a) {

  ako ($\delta(p, a) != \delta(q, a)$)

  stavi (p,q) u listu koja je pridružena paru ($\delta(p, a)$, $\delta(q, a)$)

  }}}

Nedohvatljiva stanja

- stanje p DKA $M = (Q, \Sigma, \delta, q_0, F)$ je nedohvatljivo ako ne postoji niti jedan niz $w \in \Sigma^*$ za koji vrijedi $p = \delta(q_0 ,w)$
- odbacivanjem nedohvatljivih stanja dobije se istovjetan DKA s manjim brojem stanja
- Određivanje dohvatljivih stanja u listi DS:
  - u listu dohvatljivih stanja DS upiše se početno stanje q_0;
  - listu DS proširi se skupom stanja $\{p \mid p = \delta(q_0 ,a)$ za sve $a \in \Sigma\}$;
  - za sva stanja $q_i \in$ DS proširi se lista skupom stanja

    $\{p \mid p = \delta(q_i ,a)$, stanje p nije prethodno upisano u listu, za sve $a \in \Sigma\}$;
  - ako se lista DS proširi novim stanjem ponavlja se od koraka 3.

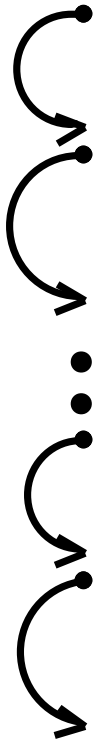
DKA s minimalnim brojem stanja

- **Algoritam**: minimalni DKA dobije se odbacivanjem
  - nedohvatljivih stanja **(1.)** i
  - istovjetnih stanja **(2.)**
  - **PAZI redosljed**
- ne postoji niti jedan drugi DKA koji prihvaća isti jezik a ima manji broj stanja
- istovjetnost minimalnog DKA s originalnim dokazuje se pomoću dokaza o istovjetnosti njihovih početnih stanja
  - algoritma ispitivanjem uvjeta podudarnosti i napredovanja

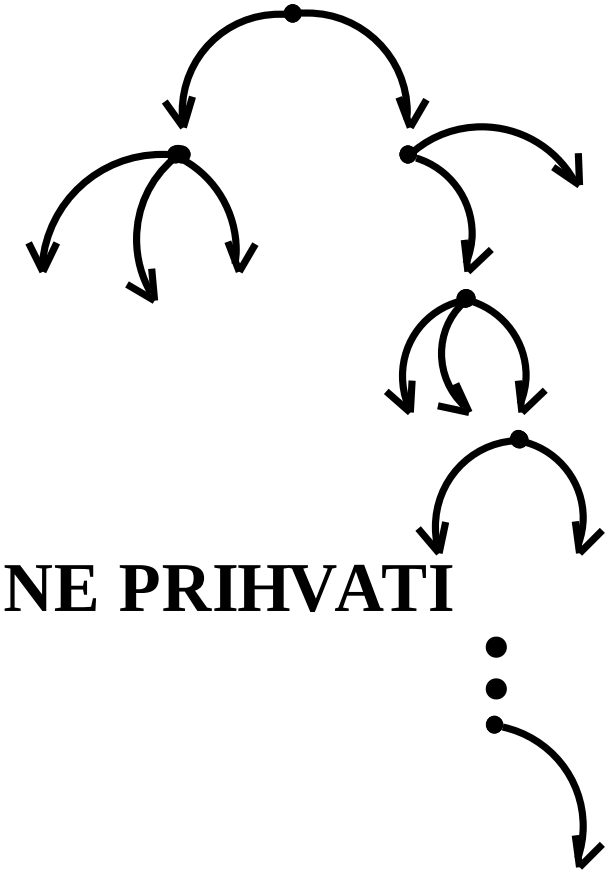
Nedeterminizam

- izvođenje paralelnih procesa
  - ako jedan od procesa dođe do prihvatljivog stanja svi se procesi prihvate
- UNIX: fork
- stablo mogućnosti, grananja
  - počinjemo u korijenu, grananje na svakom koraku
  - ako je jedna od grana u prihvatljivom stanju onda se prihvaća rad automata

Determinizam / NEdeterminizam



PRIHVATI
ili NE PRIHVATI



PRIHVATI

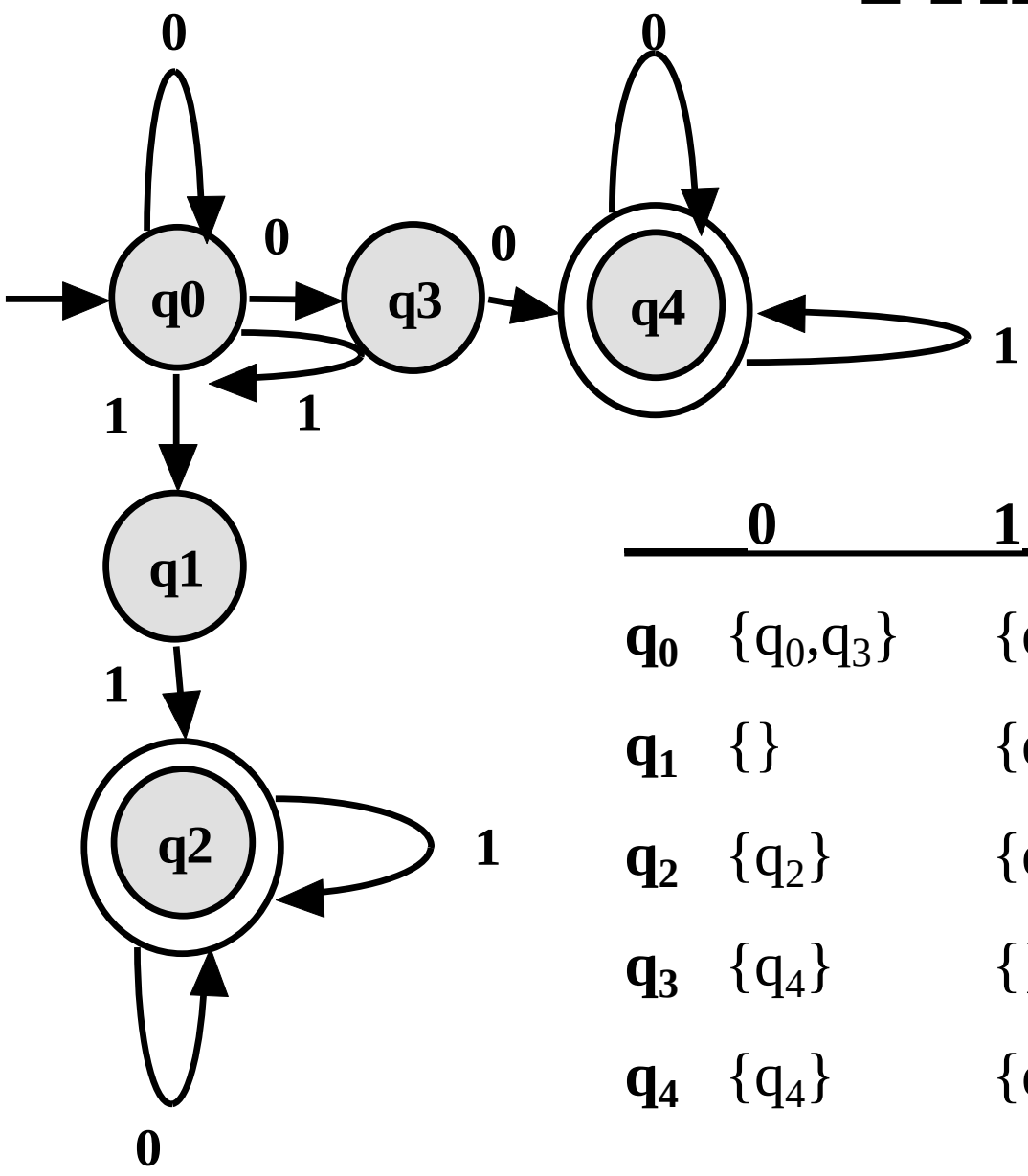
NKA – nedeterministički konačni automat

- nova funkcija prijelaza $\delta(q, a)=\{p_1, p_2, ...\}$
- za bilo koji NKA N moguće je izgraditi DKA D koji prihvaća isti jezik L(N)=L(D)

nka = (Q, Σ, δ, q_0, F) gdje je

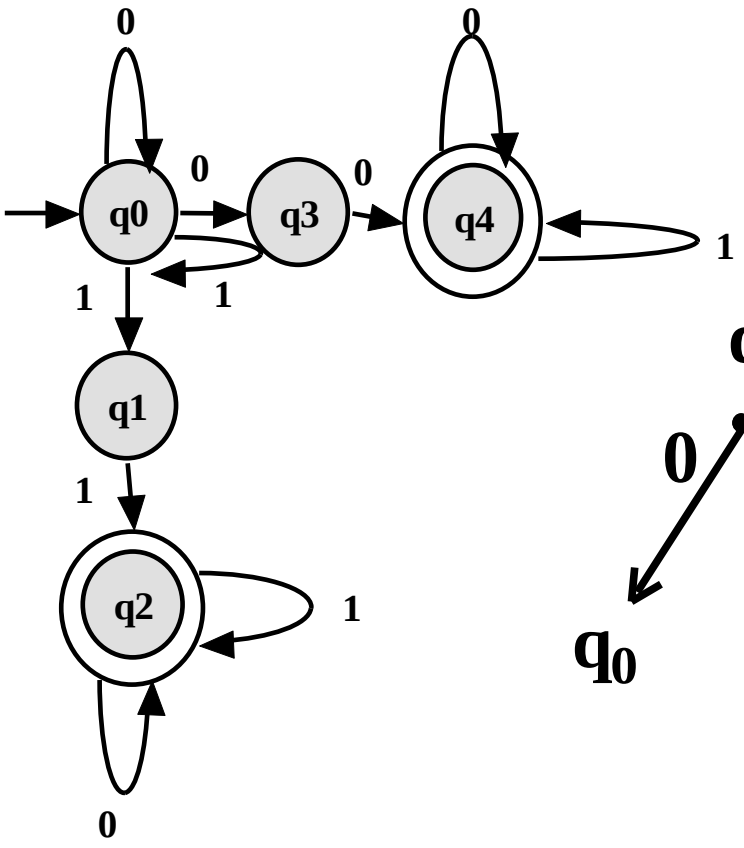
- Q – konačni skup stanja
- Σ - konačni skup ulaznih znakova
- $\delta : Q \times \Sigma \rightarrow 2^Q$; funkcija prijelaza
  - gdje je 2^Q skup svih podskupova skupa stanja Q
- $q_0 \in Q$; početno stanje
- $F \subseteq Q$, skup prihvatljivih stanja

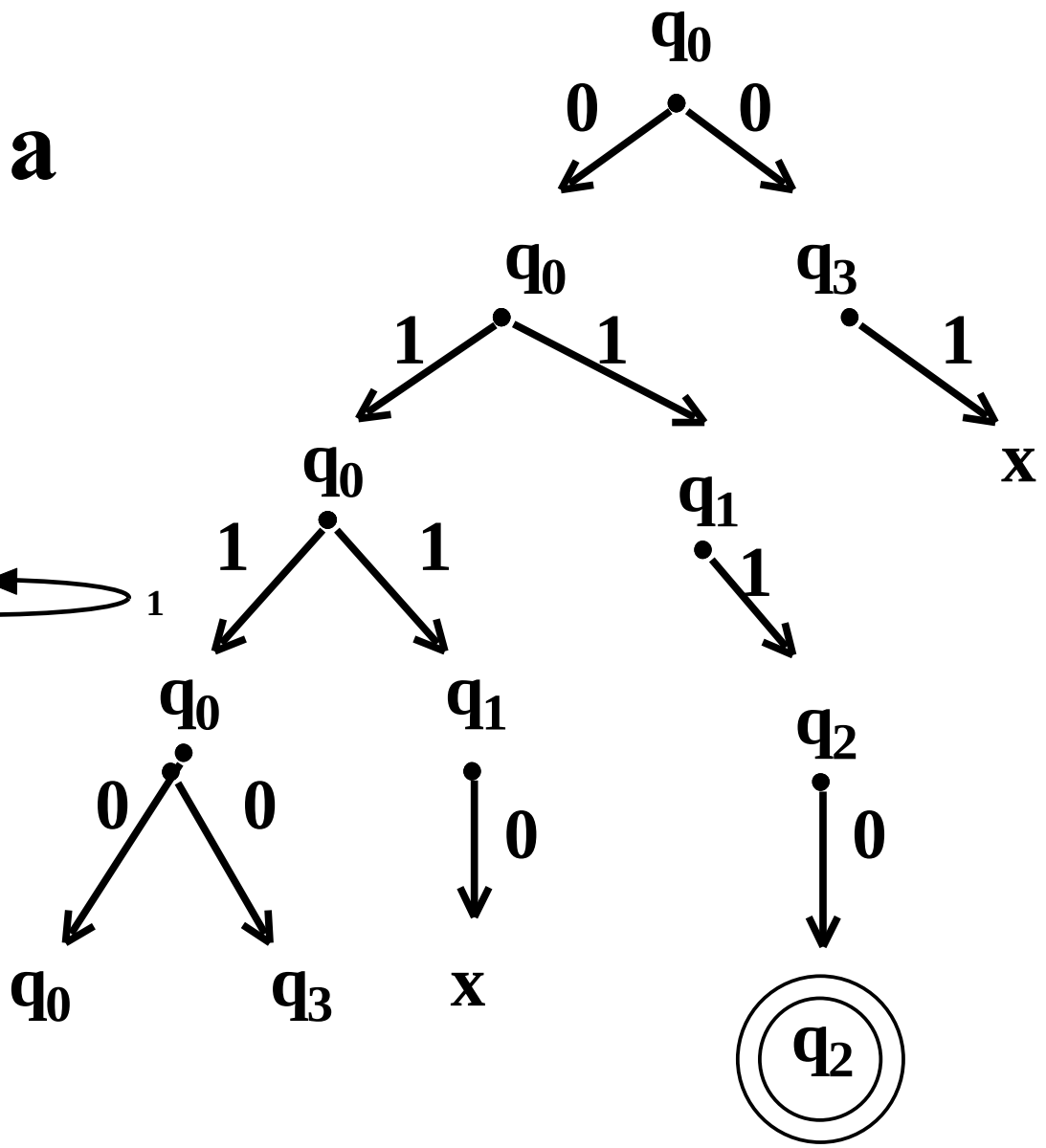
Primjer NKA



	0	1	
q_0	$\{q_0,q_3\}$	$\{q_0,q_1\}$	0
q_1	$\{\}$	$\{q_2\}$	0
q_2	$\{q_2\}$	$\{q_2\}$	1
q_3	$\{q_4\}$	$\{\}$	0
q_4	$\{q_4\}$	$\{q_4\}$	1

Stablo prihvaćanja niza 0110





Prihvaćanje niza DKA vs. NKA

- za bilo koji niz postoji samo jedan slijed prijelaza DKA iz početnog stanja q0 u stanje p= δ(q0,w), ako slijed prijelaza završi u prihvatljivu stanju niz w se prihvaća
- za neki niz w NKA može imati više od jednog slijeda prijelaza, provjeravaju se svi slijedovi prijelaza i ako postoji barem jedan slijed prijelaza iz q0 u jedno od prihvatljivih stanja niz w se prihvaća
- primjer: niz 01001

0 1 0 0 1

q0→q0 →q0 →q0 →q0 →q0

0↓ 1↓ 0↓ 0↓ 1↓

q3 q1 q3 q3 q1

0↓ 1

q4 →**q4**

Prihvaćanje jezika

- NKA M = (Q, Σ, δ, q_0, F) prihvaća jezik L(M)

- **L(M)={w δ(q_0, w) sadrži barem jedno stanje iz F}**

$$\delta(p, w) = \bigcup_{q \in P} \delta(q, w)$$

za bilo koji skup stanja P⊆Q.

Konstrukcija DKA iz NKA

- NKA $M = (Q, \Sigma, \delta, q_0, F)$ i DKA $M' = (Q', \Sigma', \delta', q_0', F')$
- DKA je istovjetan NKA ukoliko prihvaća isti jezik $L(M) = L(M')$
  - $Q' = 2^Q$, skup svih podskupova NKA Q, $[p_0, p_1,..p_i]$, $p_k \in Q$
  - F' = skup svih stanja $[p_0, p_1,..p_i]$, $\exists p_k \in F' | p_k \in F$ (barem 1 prihvatljivo)
  - $q_0' = [q_0]$ početno stanje
  - $\delta'([p_0, p_1,..p_i], a) = [r_0, r_1,..r_i]$, ako i samo ako je

    $\delta(\{p_0, p_1,..p_i\},a) = \{r_0, r_1,..r_i\}$, gdje je $a \in \Sigma$
- dobiveni DKA se minimizira, budući da su mnoga njegova stanja nedohvatljiva

NKA s ε prijelazima

- konačni automat mijenja stanje bez učitanog ulaznog znaka ε-prijelaz
- Nedeterministički konačni automat s ε-prijelazima: ε-NKA

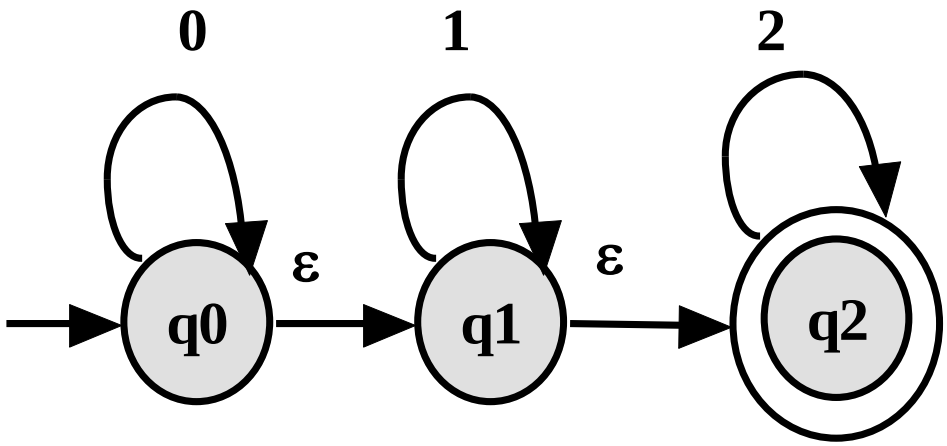
- problem: prijelaz iz početnog u prihvatljivo stanje samo ε-prijelazima

NKA s ε prijelazima

ε-nka = (Q, Σ, δ, q_0, F) gdje je

- Q – konačni skup stanja
- Σ - konačni skup ulaznih znakova
- $\delta : Q \times (\Sigma \cup \{\varepsilon\}) \rightarrow 2^Q$; funkcija prijelaza
- $q_0 \in Q$; početno stanje
- $F \subseteq Q$, skup prihvatljivih stanja

Primjer: ε-NKA



	0	1	2	ε	
q_0	$\{q_0\}$	$\emptyset$	$\emptyset$	$\{q_1\}$	0
q_1	$\emptyset$	$\{q_1\}$	$\emptyset$	$\{q_2\}$	0
q_2	$\emptyset$	$\emptyset$	$\{q_2\}$	$\emptyset$	1

prihvaća jezik: $L=\{0^n1^m2^l \mid n,m,l \geq 0\}$

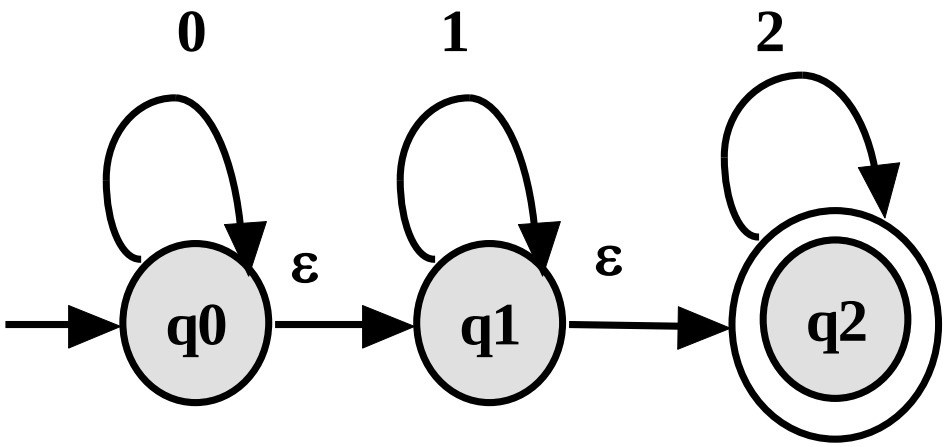
ε-okruženje

- **ε-okruženje**(q)={p | ε-NKA prelazi iz stanja q u stanje p isključivo putem ε-prijelaza **ili** stanje p je stanje q}
- za skup stanja P: ε-okruženje(P) = $\bigcup_{q \in P} \varepsilon - \text{okruženje}\ (q)$

gdje je $P \subseteq Q$

  – svako stanje je u svom ε-okruženje

Primjer: ε-NKA



	0	1	2	ε	
q_0	$\{q_0\}$	$\varnothing$	$\varnothing$	$\{q_1\}$	0
q_1	$\varnothing$	$\{q_1\}$	$\varnothing$	$\{q_2\}$	0
q_2	$\varnothing$	$\varnothing$	$\{q_2\}$	$\varnothing$	1

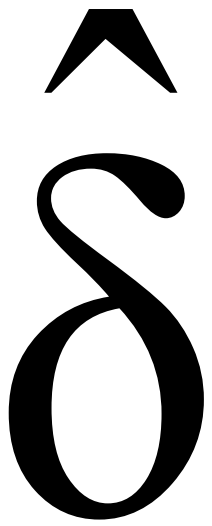
prihvaća jezik: $L=\{0^n1^m2^l \mid n,m,l \geq 0\}$

ε-okruženje(q0)={q0, q1, q2}
ε-okruženje(q1)={q1, q2}
ε-okruženje(q2)={q2}

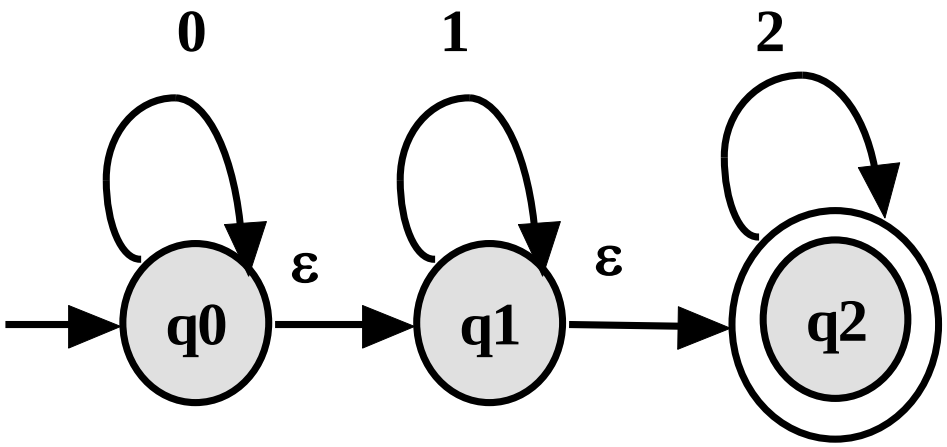
Pretvaranje ε-NKA u NKA

- proširenje funkcije δ s $\hat{\delta} : Q \times \Sigma^* \rightarrow 2^Q$
- $\hat{\delta}(q,w)$ određuje skup stanja za koje postoji slijed prijelaza (i ε-prijelaza) iz stanja q za niz w i definira se pomoću funkcije ε-okruženja
  - $\hat{\delta}(q, \varepsilon)$= ε-okruženje{q}, gdje je ε prazan niz
  - $\hat{\delta}(q,wa)$= ε-okruženje{P}, P={p | za neko stanje r iz $\hat{\delta}(q,w)$, p je iz $\delta(r,a)$}, $w \in \Sigma^*$, $a \in \Sigma$
- ε-nka M= (Q, Σ, δ, q_0, F) prihvaća jezik L(M)={w| $\hat{\delta}(q_0,w)$ sadrži barem jedno stanje iz F}

delta kapa (delta kvaka)



Primjer: ε-NKA



	0	1	2	ε	
q_0	$\{q_0\}$	$\varnothing$	$\varnothing$	$\{q_1\}$	0
q_1	$\varnothing$	$\{q_1\}$	$\varnothing$	$\{q_2\}$	0
q_2	$\varnothing$	$\varnothing$	$\{q_2\}$	$\varnothing$	1

prihvaća jezik: $L=\{0^n1^m2^l \mid n,m,l \geq 0\}$

ε-okruženje(q0)={q0, q1, q2}
ε-okruženje(q1)={q1, q2}
ε-okruženje(q2)={q2}

Pretvaranje ε-NKA u NKA

- ε-nka $M = (Q, \Sigma, \delta, q_0, F)$ daje istovjetni nka $M' = (Q', \Sigma', \delta', q_0', F')$:
  - $Q' = Q$;
  - $q_0' = q_0$;
  - $F' = F \cup \{q_0\}$ ako ε-okruženje(q_0) sadrži barem jedno stanje iz F inače $F' = F$;
  - $\delta'(q,a) = \hat{\delta}(q,a)$; $\forall a \in \Sigma$ i $\forall q \in Q$

Definicija $\hat{\delta}$

1. $\hat{\delta}$ (q, ε)= ε-okruženje{q}, gdje je ε prazan niz

2. $\hat{\delta}$ (q,wa)= ε-okruženje{P}

   – P={p | p je iz δ ($\hat{\delta}$ (q,w), a)}
     - w ∈ Σ*, a ∈ Σ

$\hat{\delta}$ i δ

- raširimo δ s $\hat{\delta}$
  - očekujemo da će $\hat{\delta}(q,w)$ dati sva stanja do kojih se može stići iz q do p po putu w, uključno s ε-prijelazima
    - kao računanje koje čvorove je moguće doseći u grafu
  - ε-okruženje definira sva stanja p koja je moguće doseći iz q samo putem ε-prijelaza
- $\hat{\delta} \neq \delta$
  - $\hat{\delta}(q,a)$ sadrži sva stanja p koja je moguće doseći iz q **pomoću a i ε-prijelaza**
  - $\delta(g,a)$ sadrži sva stanja p koja je moguće doseći iz g **samo pomoću a**

Definicija $\hat{\delta}$

1. $\hat{\delta}$ (q, ε)= ε-okruženje{q}, gdje je ε prazan niz

2. $\hat{\delta}$ (q,wa)= ε-okruženje{P}

   – P={p | p je iz δ ($\hat{\delta}$ (q,w), a)}
     - w ∈ Σ*, a ∈ Σ

Primjer DKA u Prologu I

- zapis s predikatima

```
% početno stanje
start(q0).
% prihvatljivo stanje
final(q2).
% prijelazi
% prijelaz(ishodišno_stanje,znak,ciljno_stanje)
transition(q0, a, q1).
transition(q1, b, q1).
transition(q1, c, q2).
```

Primjer DKA u Prologu II

```
% rad automata
prihvati(Znakovi) :-
start(PocetnoStanje),   %počinjemo u početnom stanju
prihvati(Znakovi, PocetnoStanje). %za ulazni niz Znakovi
                                  rekurzivno provjerimo da li
                                  ga DKA prihvaća

prihvati([], Stanje):-  %ako je niz Znakovi u Stanju prazan
final(Stanje).     %provjeri da li je Stanje prihvatljivo

prihvati([Znak|Znakovi], Stanje):- %Za Znak iz niza Znakovi
                                    i Stanje
transition(Stanje, Znak, SljedeceStanje),%napravi prijelaz
prihvati(Znakovi, SljedeceStanje).  %provjeri da li je
                                    SlijedeceStanje prihvatljivo
```

Rad DKA u Prologu III

```
?- prihvati([a, b, b, c]).
Yes

?- prihvati([a, b, b, c, b]).
No
```

Primjer ε-NKA u Prologu II

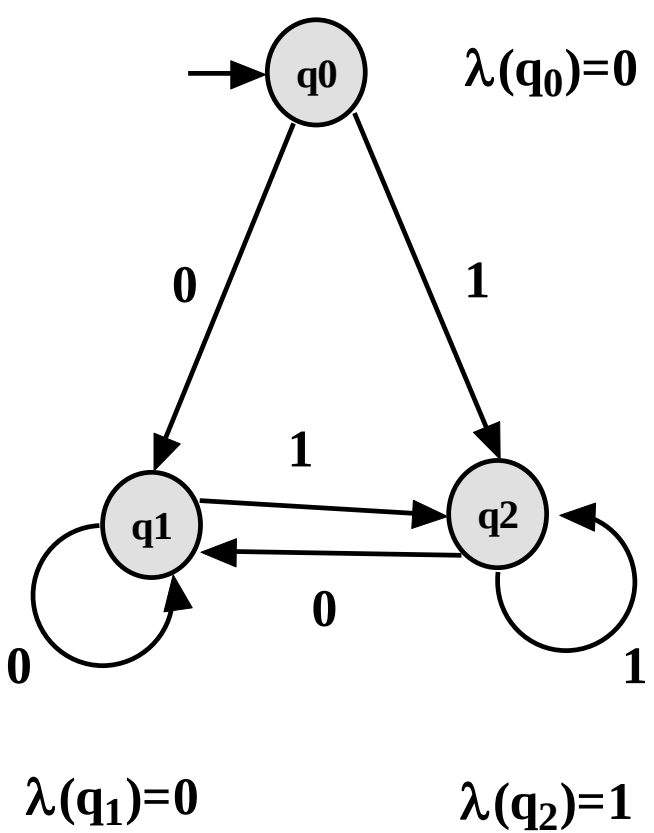
```
prihvati([], Stanje):-
final(Stanje).

prihvati([Znak|Znakovi], Stanje):-
transition(Stanje, Znak, SljedeceStanje),
prihvati(Znakovi, SljedeceStanje).

prihvati(Znakovi, Stanje):-
epsilon(Stanje, SljedeceStanje),
prihvati(Znakovi, SljedeceStanje).
```

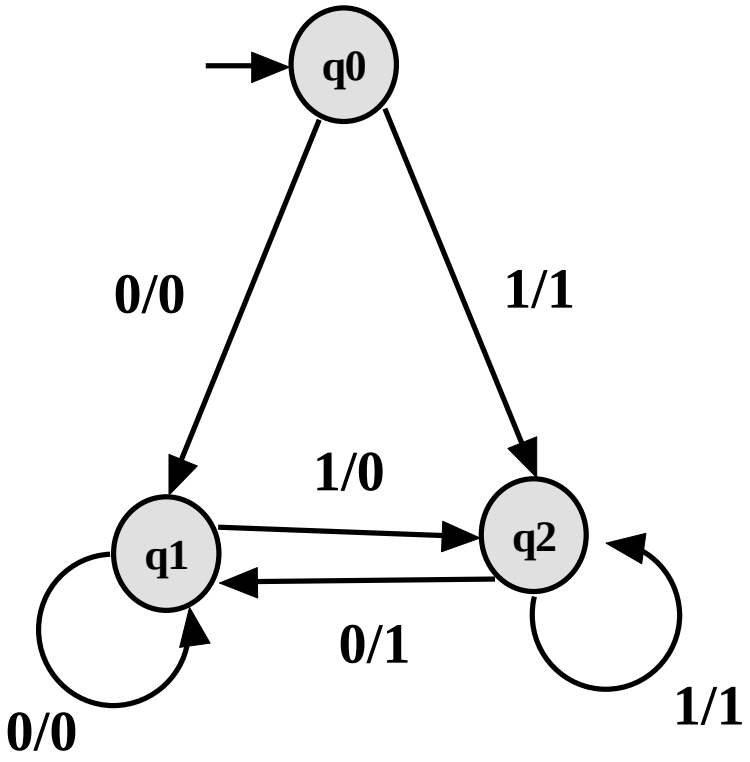
Konačni automati s izlazom

- izlaz je funkcija stanja automata:
  - Mooreov automat
- izlaz je funkcija stanja automata i ulaznog znaka:
  - Mealyev automat
- Mooreov i Mealyev automat su istovjetni ukoliko za bilo koji ulazni niz daju jednake izlazne nizove
- za bilo koji MoDKA moguće je izgraditi istovjetni MeDKA i obrnuto
- Skriveni Markovljev model

Primjeri

- **MOOREov automat**
- **MEALYjev automat**





Mooreov automat MoDKA

MoDKA = (Q, Σ, Δ, δ, λ, q_0) gdje je

- Q – konačni skup stanja
- Σ - konačni skup ulaznih znakova
- Δ - konačni skup izlaznih znakova
- $\delta : Q \times \Sigma \rightarrow Q$; funkcija prijelaza
- $\lambda: Q \rightarrow \Delta$; funkcija izlaza
- $q_0 \in Q$; početno stanje

Mealyev automat MeDKA

MeDKA = (Q, Σ, Δ, δ, λ,q_0) gdje je

- Q – konačni skup stanja
- Σ - konačni skup ulaznih znakova
- Δ - konačni skup izlaznih znakova
- $\delta : Q \times \Sigma \rightarrow Q$; funkcija prijelaza
- $\lambda: Q \times \Sigma \rightarrow \Delta$; funkcija izlaza
- $q_0 \in Q$; početno stanje

Pretvaranje
MoDKA → MeDKA

- za prazni niz ε MoDKA=M ima izlaz, MeDKA=M´ nema
  - istovjetnost:Mo i Me: $bT_{M'}(w)=T_M(w)$; w niz ulaznih znakova
    - b –izlaz MoDKAza prazni niz $b= \lambda(q_0)$,
    - $T_{M'}(w)$- izlaz MeDKA i
    - $T_M(w)$ je izlaz MoDKA
- $\lambda'(q, a)=\lambda(\delta(q,a)), \forall q \in Q, \forall a \in \Sigma$
- $Q= Q'$; $\Sigma= \Sigma'$; $\Delta= \Delta'$; $\delta= \delta'$; $q_0= q_0'$

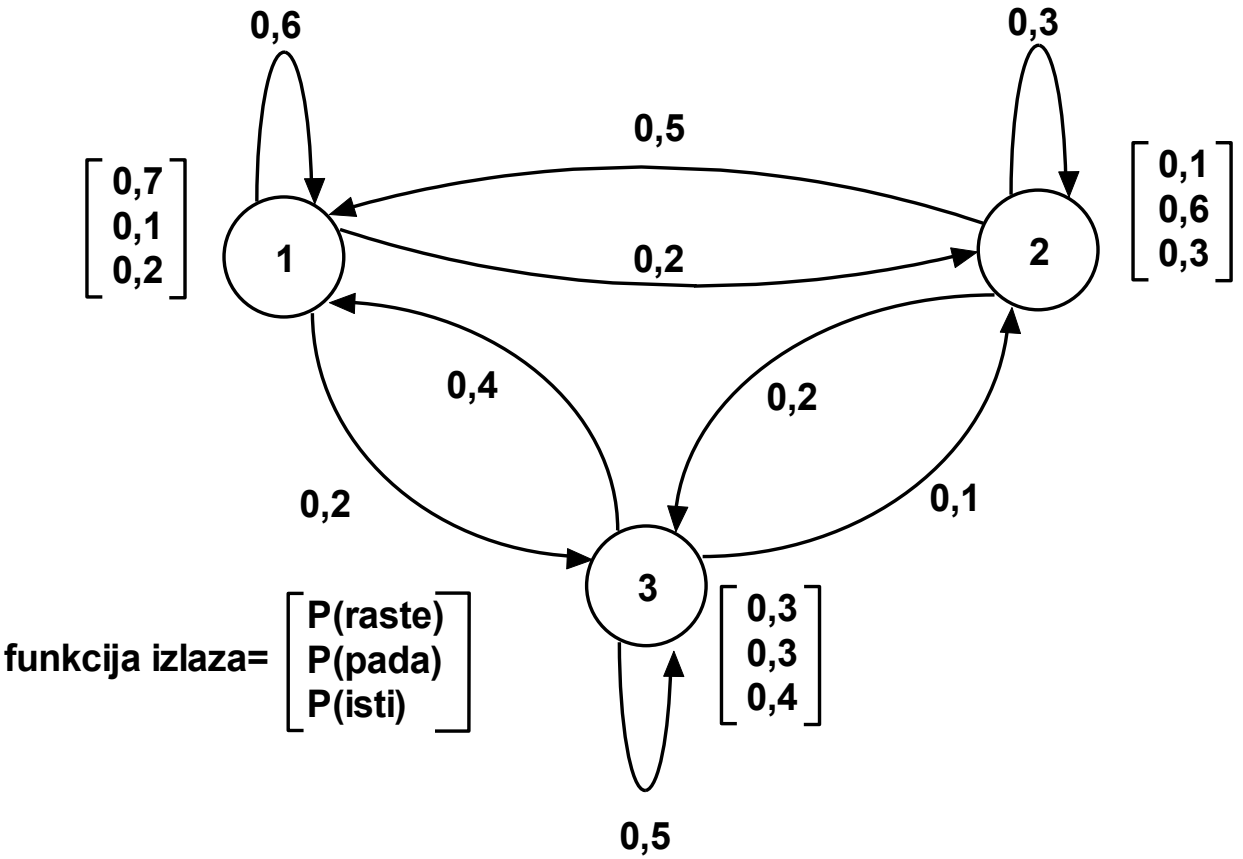
Pretvaranje
MeDKA $\rightarrow$ MoDKA

- $Q' = Q \times \Delta$; $[q, b] \in Q'$, $q \in Q$, $b \in \Delta$
- $q_0' = [q_0, b_0]$; b_0 proizvoljni element iz Δ
- $\delta'([q, b], a) = [\delta(q, a), \lambda(q, a)]$
- $\lambda'([q, b]) = b$; $q \in Q$, $b \in \Delta$

Primjena automata s izlazom

- govorne tehnologije
  - raspoznavanje govora
    - za govorni signal zapisujemo tekst
  - sinteza govora
    - iz teksta generiramo govorni signal
  - nadzor nad vođenjem dijaloga
- modeliranje mrežnih protokola
- modeliranje naprava
  - inteligentne sobe, kuće
- ....

HMM za DowJones index



- indeks u svakom stanju može porasti, pasti ili ostati nepromijenjen
- dakle odnos događaja i stanja više nije 1:1,
- znači promatrač na osnovu opaženog izlaznog simbola (npr. indeks raste), ne zna u kojem se stanju proces nalazi. To je svojstvo **prikrivenosti modela**.

Izbor kuglice iz 3 posude

O={crvena, plava, žuta, zelena}

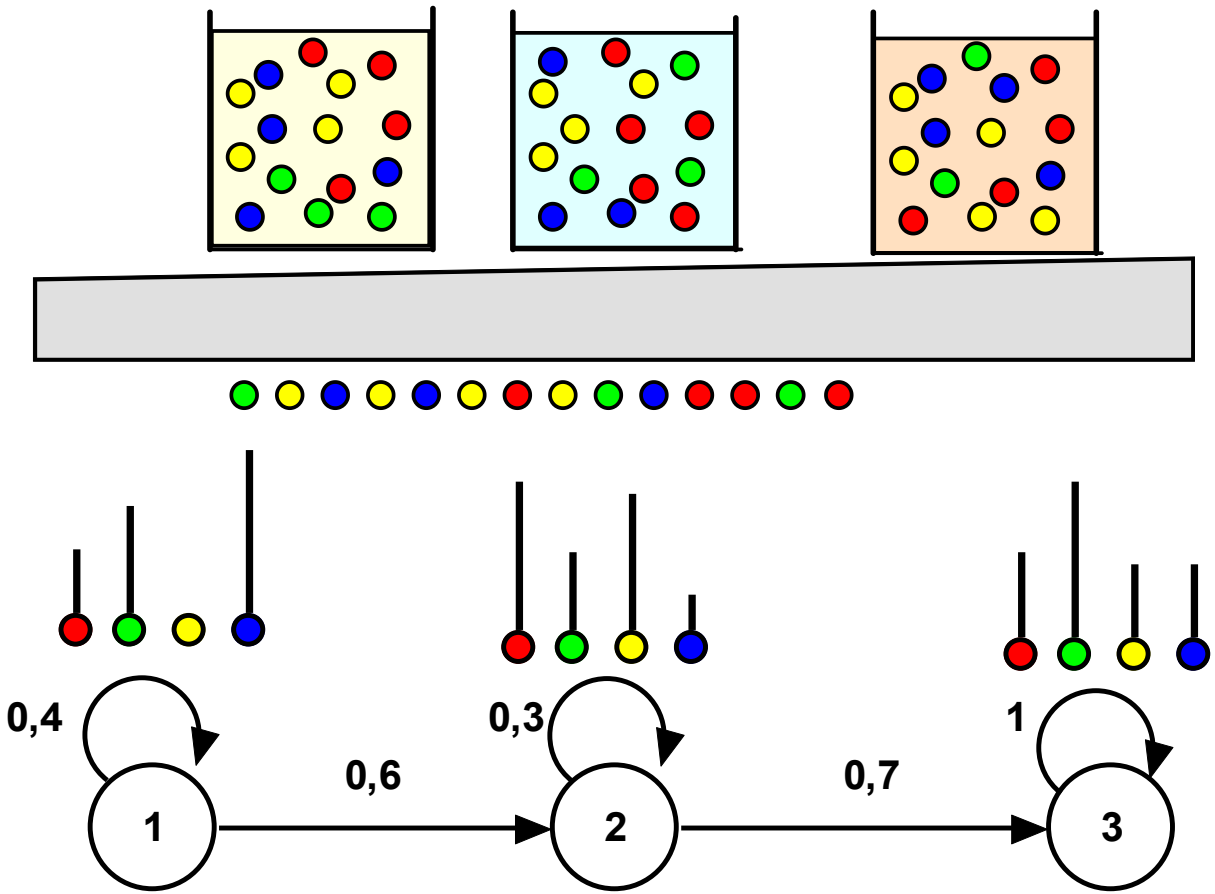
P(crvena)=$b_1(1)$ P(crvena)=$b_2(1)$ P(crvena)=$b_3(1)$

P(plava)=$b_1(2)$ P(plava)=$b_2(2)$ P(plava)=$b_3(2)$

P(žuta)=b1(3) P(žuta)=$b_2(3)$ P(žuta)=$b_3(3)$

P(zelena)=$b_1(4)$ P(zelena)=$b_2(4)$ P(zelena)=$b_3(4)$

HMM za izbor kuglice iz posude



Definicija SMM-a

- Diskretni skriven Markovljev model je definiran:
  - **Izlaznom abecedom** $O=\{o_1, o_2,..,o_M\}$, gdje je M broj simbola u abecedi;
  - **Skupom stanja** $S=\{s_1,..,s_N\}$, gdje je N broj stanja modela;
  - Matricom **vjerojatnosti prijelaza** $\boldsymbol{A=\{a_{ij}\}}$, gdje je a_{ij} vjerojatnost prijelaza iz stanja i u stanje j zapisana izrazom $a_{ij=}P(s_t=j|s_{t-1}=i)$;
  - Matricom **vjerojatnosti izlaznih simbola** $\mathbf{B=\{b_i(k)\}}$, gdje je $b_i(k)$ vjerojatnost pojave simbola o_k u stanju i;
    - Ako je $X=(X_1,..,X_t)$ izlazni niz procesa do trenutka t, i ako je slijed stanja $S=(s_1,s_2,..,s_t)$ koje je proces pritom zauzeo prikriven onda se $b_i(k)$ može definirati kao $b_i(k)=P(X_t=o_k|s_t=i)$,
  - **Vektorom početnih vjerojatnosti** $\prod=\{\pi_i\}$; gdje je π_i vjerojatnost da se model u trenutku t_0 nalazi u stanju s_i za koju vrijedi $\pi_i=P(s_0=i)$ za $1 \le i \le N$

Definicija SMM-a II

- SMM Φ je trojka $\Phi = (A, B, \Pi)$
  - A matrica vjerojatnosti prijelaza,
  - B matrica vjerojatnosti izlaza i
  - Π matrica početne distribucije vjerojatnosti
- Vjerojatnosti prijelaza a_{ij} moraju zadovoljavati uvjet:

$$\sum_{j=1}^{N} a_{ij} = 1, \qquad a_{ij} \geq 0, \forall i, j$$

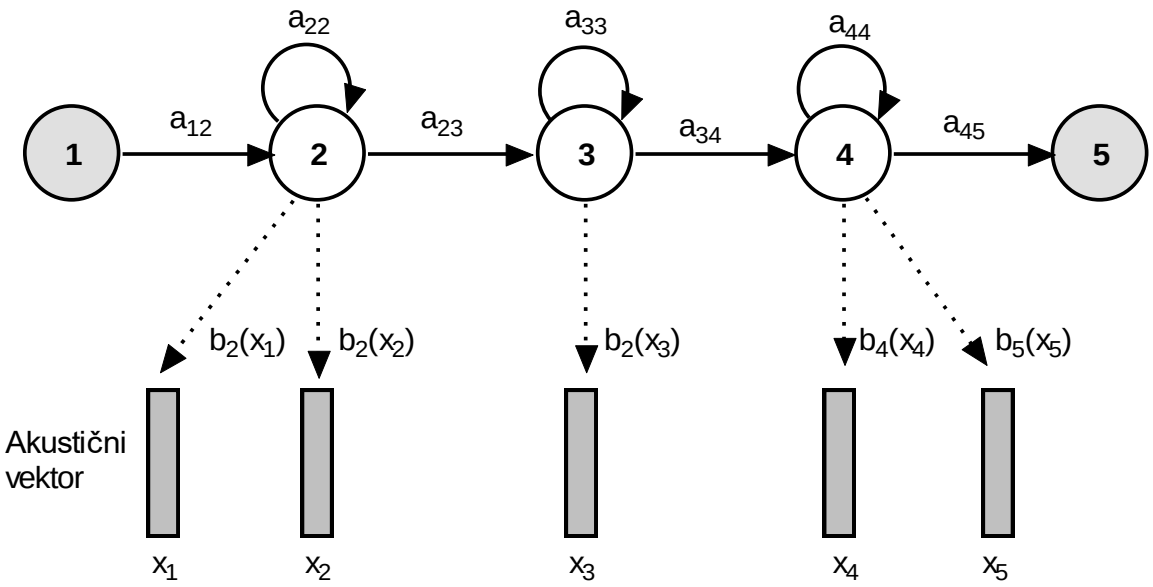
- Vjerojatnosti izlaznih simbola $b_i(k)$ moraju zadovoljavati uvjet:

$$\sum_{k=1}^{M} b_i(k) = 1, b_i(k) \geq 0, \forall i, k$$

- Vjerojatnosti π_i moraju zadovoljavati uvjet:

$$\sum_{i=1}^{N} \pi_i = 1, \qquad \pi_i \geq 0, \forall i$$

SMM govora

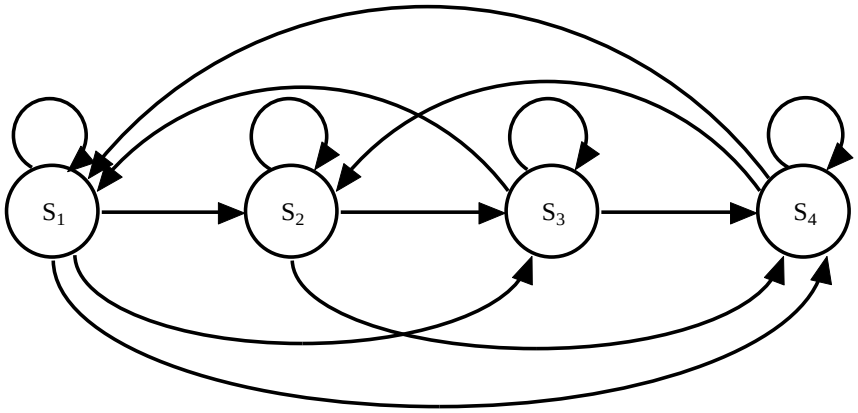


- linearni SMM-i s Gaussovim kontinuiranim funkcijama gustoća vjerojatnosti
- pet stanja (5/3):
  - **drugo, treće i četvrto**: modeliraju spektralne značajke pojedinog monofona na početku, sredini i kraju intervala izgovora
  - **prvo i zadnje:** neemitirajuće (povezivanje modela u složenije)

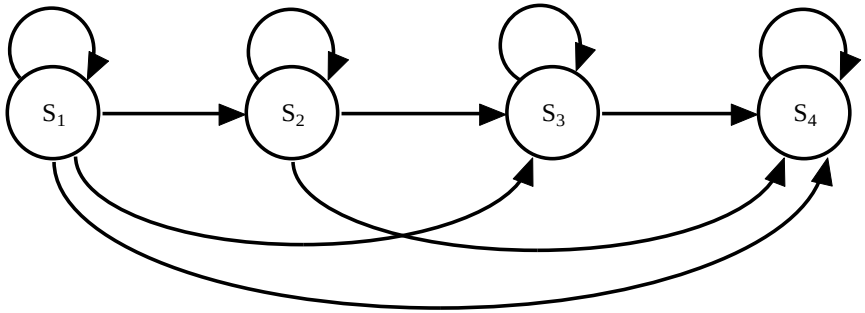
Rad diskretnog SMM-a

1. **INICIJALIZACIJA**	ODABERI POČETNO STANJE $S_1=I$ S OBZIROM NA POČETNU DISTRIBUCIJU VJEROJATNOSTI Π_I. POSTAVI VRIJEME $T=1$.
2. **GENERIRANJE IZLAZNOG SIMBOLA**	S OBZIROM NA VJEROJATNOST IZLAZNOG ZNAKA $B_I(K)$ ODABERI IZLAZNI ZNAK O_K U STANJU S_I.
3. **PRIJELAZ**	PRIJEĐI U NOVO STANJE $S_{T+1}=J$ U SKLADU S VJEROJATNOSTIMA PRIJELAZA A_{IJ} IZ STANJA S_I
4. **ZAVRŠETAK**	POVEĆAJ T ZA 1 AKO JE $T < T$ VRATI SE NA 2 INAČE ZAVRŠI.

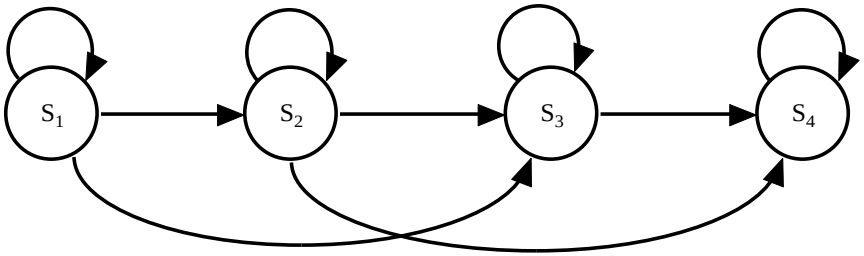
Ergodički (a), lijevo-desni (b), Bakisov (c) i linearan (d) SMM.



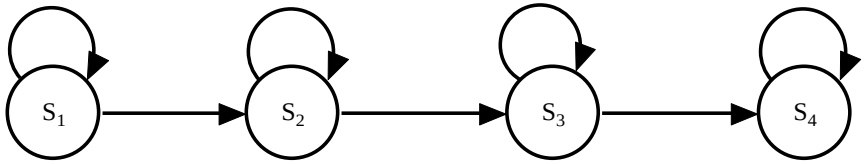
(a)



(b)

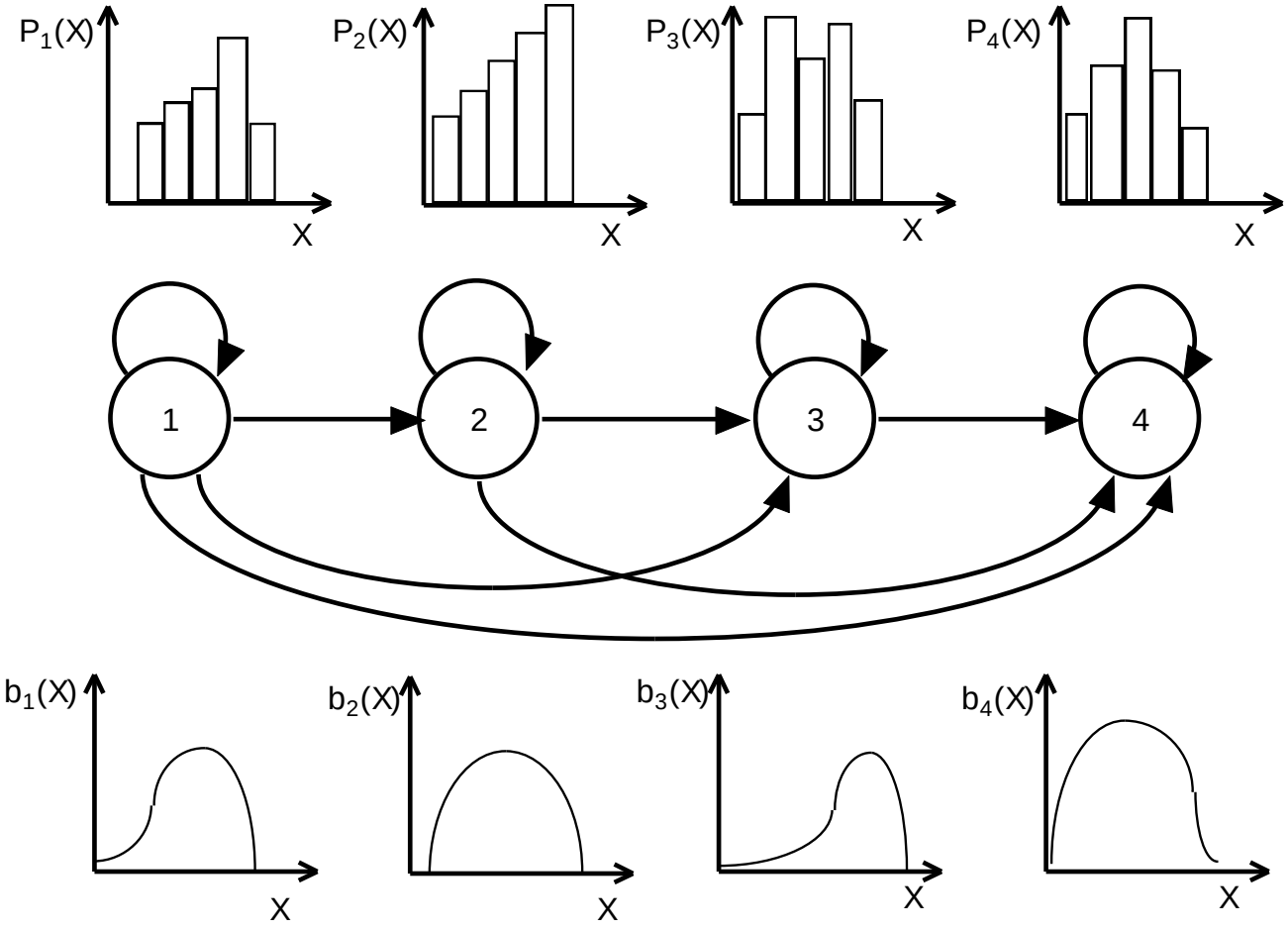


(c)



(d)

Diskretni i kontinuirani lijevo-desni SMM s četiri stanja



Struktura monofonskog SMM-a fonema /a/

- SMM je označen oznakama <BEGINHMM> i <ENDHMM>,
- naziv SMM-a oznakom ~h "a",
- broj stanja oznakom <NUMSTATES>,
- matrica vjerojatnosti prijelaza među stanjima oznakom <TRANSP>,
- stanje oznakom <STATE>,
- a srednje vrijednosti i varijance svakog stanja oznakama <MEAN> i <VARIANCE>,
- duljina i vrsta vektora značajki oznakom <VECSIZE>.
- Broj Gaussovih mješavina kao i težinski faktor označeni su oznakama <NUMMIXES> i <MIXTURE>,

~o
<STREAMINFO> 1 39
<VECSIZE> 39<NULLD><MFCC_E_D_A><DIAGC>
~h "a"
<BEGINHMM>
<NUMSTATES> 5

<STATE> 2
<MEAN> 39
-2.899339e+00 -1.308102e+01 -3.859741e+00 -5.578077e+00 1.886920e+00 -4.362893e+00 -2.831639e+00 -3.419966e+00 -2.572051e+00 -1.476051e+00 -7.433438e+00 -5.603982e+00 2.250297e+01 1.341190e-01 9.331121e-02 -1.717419e-02 7.486364e-02 2.749340e-01 2.601945e-01 1.839875e-01 -7.917245e-02 -1.172790e-01 -2.418435e-02 -1.248855e-01 -1.869368e-01 -4.074142e-02 1.452143e-02 4.144624e-01 2.558722e-01 1.368404e-01 -2.115643e-01 -1.215054e-01 -1.398575e-01 -2.350915e-01 -3.453482e-02 1.724735e-01 2.212539e-01 7.645249e-02 -4.765237e-02
<VARIANCE> 39
7.932775e+00 1.786971e+01 2.486252e+01 3.243413e+01 5.059038e+01 7.262206e+01 5.125904e+01 5.875118e+01 5.318542e+01 5.299713e+01 7.454996e+01 6.636087e+01 1.888268e-01 4.979259e-01 1.093826e+00 1.196317e+00 1.334101e+00 2.379372e+00 2.407022e+00 2.504204e+00 2.300557e+00 2.166366e+00 2.415282e+00 2.357176e+00 2.133943e+00 7.790518e-03 1.287570e-01 1.924337e-01 2.007127e-01 2.756920e-01 3.355306e-01 3.952366e-01 4.316493e-01 4.681919e-01 4.308535e-01 4.386297e-01 4.333408e-01 3.795784e-01 4.566273e-03
<GCONST> 9.702691e+01

<STATE> 3

<MEAN> 39

-1.063308e+00 -8.613764e+00 -1.878786e+00 -4.538751e+00 -4.183505e-01 -5.903518e+00 -4.784183e+00 -5.141273e+00 -2.289769e+00 1.332671e+00 -5.991622e+00 -5.716003e+00 2.207525e+01 2.338001e-01 2.261684e+00 1.187068e+00 9.443569e-01 -1.995222e-01 -2.176124e-01 3.016089e-01 -5.553427e-01 -4.353977e-02 -6.998805e-03 2.489749e-01 4.853007e-02 -2.211624e-01 -2.503028e-01 2.961237e-01 1.633116e-01 1.384239e-01 6.928017e-02 5.831994e-02 2.347713e-01 1.104801e-01 1.350555e-02 -2.819899e-01 -7.055640e-02 2.466260e-02 -5.488205e-02

<VARIANCE> 39

1.230534e+01 3.025063e+01 3.408766e+01 3.894379e+01 5.465639e+01 8.379287e+01 6.370923e+01 6.947521e+01 6.230185e+01 6.728851e+01 1.027657e+02 9.158002e+01 2.813180e-01 1.143072e+00 1.721416e+00 1.639243e+00 2.308818e+00 3.004421e+00 3.674523e+00 3.588318e+00 4.047483e+00 3.643836e+00 3.721678e+00 3.705328e+00 3.254243e+00 2.812584e-02 2.191565e-01 3.528966e-01 3.090518e-01 3.735814e-01 4.665537e-01 6.717233e-01 6.498726e-01 7.060002e-01 6.189460e-01 6.349556e-01 6.872172e-01 6.801853e-01 5.727982e-03

<GCONST> 1.129312e+02

<STATE> 4

<MEAN> 39

-4.631654e+00 -3.570257e+00 6.052272e-01 -1.415057e+00 1.356118e+00 -3.959465e+00 -3.078352e-01 -2.790277e+00 -1.673839e+00 -2.168639e+00 -6.375071e+00 -5.395334e+00 1.963285e+01 -4.821330e-01 8.893453e-01 4.577440e-01 9.963217e-02 -2.229648e-01 5.198171e-02 2.538395e-01 1.885251e-02 -1.653914e-02 -1.769117e-01 2.049307e-01 1.188798e-01 -2.516664e-01 -1.563174e-01 -3.125495e-01 -1.439878e-01 -1.501855e-01 -2.841547e-02 -9.346209e-03 -1.667996e-01 -7.611429e-02 -1.179364e-01 5.958437e-02 1.258440e-01 8.019649e-02 4.149693e-02

<VARIANCE> 39

2.993065e+01 3.531915e+01 3.261441e+01 3.542360e+01 4.681488e+01 5.671862e+01 4.992723e+01 6.217274e+01 5.416755e+01 5.549760e+01 6.002085e+01 5.498278e+01 4.190552e+00 1.798784e+00 2.654591e+00 2.060711e+00 2.566887e+00 2.778024e+00 3.502275e+00 3.593817e+00 4.457613e+00 3.751519e+00 3.611640e+00 3.788476e+00 3.337752e+00 1.572320e-01 2.755345e-01 4.279337e-01 3.574103e-01 4.509982e-01 5.092507e-01 6.075901e-01 6.385270e-01 7.319860e-01 6.780887e-01 6.321148e-01 6.499683e-01 5.677502e-01 1.753231e-02

<GCONST> 1.189476e+02

Matrica prijelaza

<TRANSP> 5
 0.000000e+00 1.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
 0.000000e+00 7.835935e-01 2.164065e-01 0.000000e+00 0.000000e+00
 0.000000e+00 0.000000e+00 5.917447e-01 4.082553e-01 0.000000e+00
 0.000000e+00 0.000000e+00 0.000000e+00 7.245473e-01   2.754527e-01
 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00 0.000000e+00
<ENDHMM>

Tri problema

- problem procjene (evaluacije) vjerojatnosti niza simbola X *(The Evaluation Problem)*
- problem dekodiranja slijeda stanja za dani niz X *(The Decoding Problem)*
- problem ocjene (učenja, optimiranja) parametara PMM $\Phi=(A, B, \pi)$ *(The Learninig/Estimation Problem)*.

Procjena

- problem procjene (evaluacije) vjerojatnosti niza simbola X *(The Evaluation Problem)*
- Ukoliko imamo model $\Phi=(A, B, \pi)$ i izlazni slijed opažanja $X=\{X_1,X_2,..X_T\}$, koja je vjerojatnost da je upravo model Φ generirao ta opažanja $P(X|\Phi)$?
  - izračunati vjerojatnost da je izlazni slijed nastao baš u tom modelu ili koliko je model usklađen s opažanjima.
  - problem određivanja koliko je model prilagođen opaženom izlaznom slijedu, odnosno možemo ga svesti na problem klasifikacije koji od potencijalnih modela najbolje odgovara opaženom izlaznom slijedu
- **algoritmi naprijed i natrag**

Dekodiranje

- Ukoliko imamo model $\Phi=(A, B, \pi)$ i izlazni slijed opažanja $X=\{X_1,X_2,..X_T\}$ koji je najvjerojatniji slijed stanja $S=\{s_1,s_2,..s_T\}$ u kojima je model generirao izlazni slijed?
  - problem prikrivenosti unutarnjeg procesa odnosno - znamo kako se prikriven proces ponaša.
  - otkriti optimalan slijed stanja koji nije nužno i ″pravilan″ slijed stanja
- **Viterbijev algoritam** određuje najvjerojatniji slijed stanja u PMM-u. Traži se slijed stanja $S=\{s_1,s_2..s_T\}$ takav da je vjerojatnost $P(S, X|\Phi)$ maksimalna

Procjena- Učenje

- Ukoliko imamo izlazni slijed opažanja $X=\{X_1,X_2,..X_T\}$ i neki početni model $\Phi=(A, B, \pi)$ kako ćemo odrediti parametre modela $\Phi'=(A',B', \pi')$, koji maksimiziraju produkt vjerojatnost ?
  - Rješenje: na osnovi podataka s kojima učimo model ocijeniti parametre modela, zato se ovaj problem često naziva i problemom učenja.
  - Dakle parametre modela učenjem možemo maksimalno prilagoditi podacima na kojim učimo. To može dovesti do dodatnih problema prevelike prilagodbe podacima koji nastaju zbog premale količine podatak za učenje (***ovefitting***)
- kombinacija **Naprijed-natrag algoritama**
- **Baum–Welchev algoritam** odnosno
- **metoda očekivanja i popravaka** (***Expectation-Modification Method – EM algorithm*).**