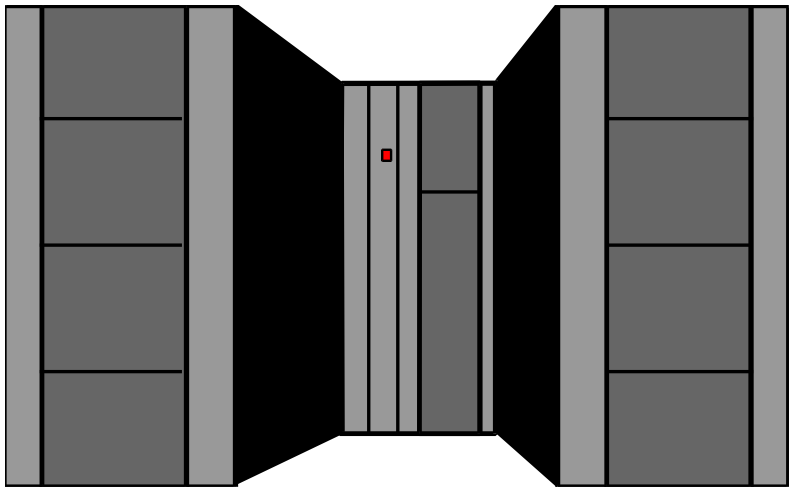
Umjetna inteligencija

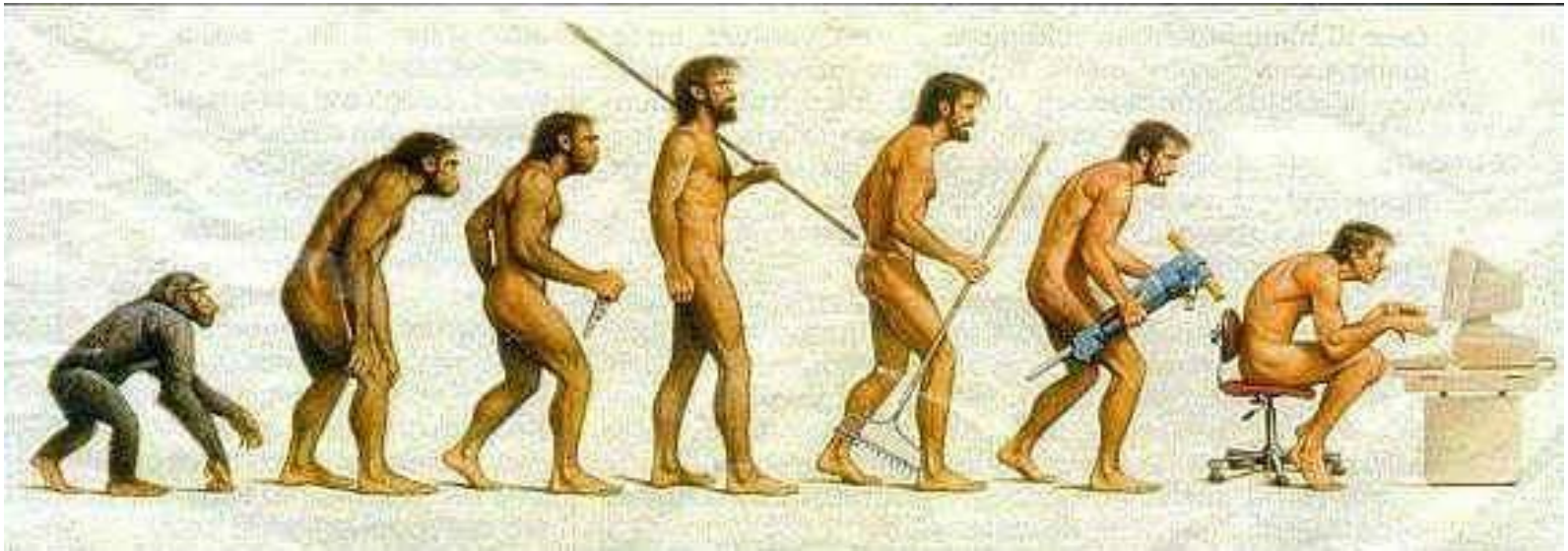
Dave Bowman: Open the pod bay doors, HAL.
HAL: I’m sorry Dave, I’m afraid I can’t do that.



S. Kubrick, A. C. Clarke: 2001.Odiseja u svemiru

Umjetna inteligencija II

- je potraga za idejama, koje računalima omogućavaju inteligenciju
- Cilj:
  - računala učiniti upotrebljivijim
  - razumjeti principe inteligencije
- na osnovu znanja logički zaključivati



Inteligentna računala mogu:

- rješavati teške probleme
- pomoći pri istraživanjima i konstruiranjima
- pomoći u proizvodnji
- razumjeti prirodan jezik
- razumjeti slike
- naučiti određene primjere i uzorke
- ....

AI se je iskazala na problemima koji su čovjeku teški

- igranje šaha
- planiranje, logističko planiranje
- raspored letova i rezervacija u avio-prometu
- otkrivanje prijevara (osiguranje)
- dokazivanje teorema
- rješavanje križaljki
- ....

Ali AI se nije iskazala na problemima koji su čovjeku laki:

- razumjeti govor i razumjeti jezik
- razgovarati
- prepoznati osobu
- napraviti umjetničko djelo (muzika, slika...)
- samostalno navigirati
- kretati se (motorika)
- rezonirati po “zdravoj pameti”
- ....

Turingov test

- pošto su pojmovi *misliti* i *inteligencija* neprecizno definirani engleski logičar i matematičar Alan Turing je osmislio empirijski test s kojim se provjerava:

Mogu li strojevi misliti?

- test se provodi kroz igru pitanja u koji sudjeluje ispitivač (osoba) te stroj i osoba koji odgovaraju na pitanja
  - njihov identitet je nepoznat i ispitivač samo na osnovu razgovora mora odrediti tko je stroj a tko osoba
- ELIZA – ponaša se kao psihoterapeut
- Loebnerov natječaj

*** Judge 05 ***

PROGRAM[10:07:12]Hi, my name is Catherine, what's yours?

JUDGE05[10:07:29]I'm Jurgen.

PROGRAM[10:07:31]Well, it's nice to meet you, Jurgen.

PROGRAM[10:07:40]Did you see that story on CNN last night about the lesbian

PROGRAM[10:07:52]couple who came out at a White House party on Sunday?

JUDGE05[10:08:33]NO. I just came in yesterday. I'm still kind of jet-lagged.

PROGRAM[10:08:35]Ellen Degeneres was one of them - she was kissing her lover

PROGRAM[10:08:51]in front of Bill Clinton. Do you think he was very

PROGRAM[10:09:14]embarrassed?

JUDGE05[10:10:18]I don't know? Don't you think it's a little childish if the leader of the most powerful nation on the planet is embarrassed by seeing two lesbians kiss?

PROGRAM[10:10:26]Probably! No. -

PROGRAM[10:10:29]Are you still a Friend of Bill Clinton?

JUDGE05[10:10:47]Never met him.

PROGRAM[10:10:48]You really don't like the guy--well, at least that's

PROGRAM[10:11:01]definite--I thought most people were just indifferent or

PROGRAM[10:11:14]numb by now.

JUDGE05[10:11:35]Have you seen the movie Mars attacs?

PROGRAM[10:11:36]Give me a break -- I'm just here to answer dumb questions

PROGRAM[10:11:50]-- not that! What were we talking about... Ok.. I guess that

PROGRAM[10:12:09]means you really think there's serious behind Whitewater,

PROGRAM[10:12:29]the Special prosecutor and all that stuff?

JUDGE05[10:12:48]I'm cold.

PROGRAM[10:12:49]But how do you actually feel about him, now in the number

PROGRAM[10:13:01]two term?

http://www.cs.flinders.edu.au/Research/AI/LoebnerPrize/

Loebnerov test 2009

- http://www.chatbots.org/awards/announcement/loebner_prize_2009_video_report/

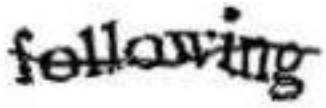
- 2015: http://www.aisb.org.uk/events/loebner-prize

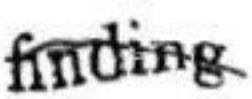
Human or Computer? Take This Test

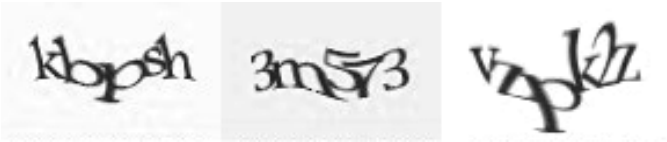
- http://www.nytimes.com/2002/12/10/science/human-or-computer-take-this-test.html

CAPTCHA (wiki)

- or **Captcha** (/kæptʃə/) is a type of challenge-response test used in computing to ensure that **the response is not generated by a computer**.
  - one computer (a server) is asking a user to complete a simple test which the computer is able to generate and grade.
  - other computers are unable to solve the CAPTCHA, any user entering a correct solution is presumed to be human.
- it is sometimes described as a **reverse Turing test,** because it is administered by a machine and targeted to a human, in contrast to the **standard Turing test** that is typically administered by a human and targeted to a machine.
  - requires that the user type letters or digits from a distorted image that appears on the screen.








Demo

- http://www.youtube.com/watch?v=cYw2ewoO6c4

- http://ironphoenix.org/tril/tm/

Formalni jezici i jezični procesori I

TURINGOV STROJ

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

TS i rekurzivno prebrojivi jezici

- jezik je rekurzivno prebrojiv ako i samo ako postoji Turingov stroj (TS) koji ga prihvaća
- za bilo koji rekurzivno prebrojiv jezik moguće je izgraditi TS i obratno

Alan Turing

- engleski matematičar logičar i kriptograf
- postavio je formalne koncepte teorije računarstva:
  - Turingov stroj
  - Turingov test (propituje temeljna pitanja umjetene inteligencije)
- radio je na razvoju Mark I računala
- za vrijeme 2. svjetskog rata razbijao je njemačke kriptirane poruke: ENIGMA
  - razvio je elektromehanički stroj za kriptoanalizu (razbija enigmine poruke)

Alan Turing II

- model univerzalnog računala: Turingov stroj
  - univerzalno izvođenje algoritma
- apstraktni stoj (matematički model) koji se sastoji od:
  - glave za čitanje koja se pomiče lijevo-desno
  - beskonačne trake za pohranjivanje podataka
  - i programa koji određuje što se mora izvesti
  - može riješiti bilo koji problem koji mogu riješiti današnja računala (uz uvjet neograničenog vremena za izvođenja i neograničene memorije)

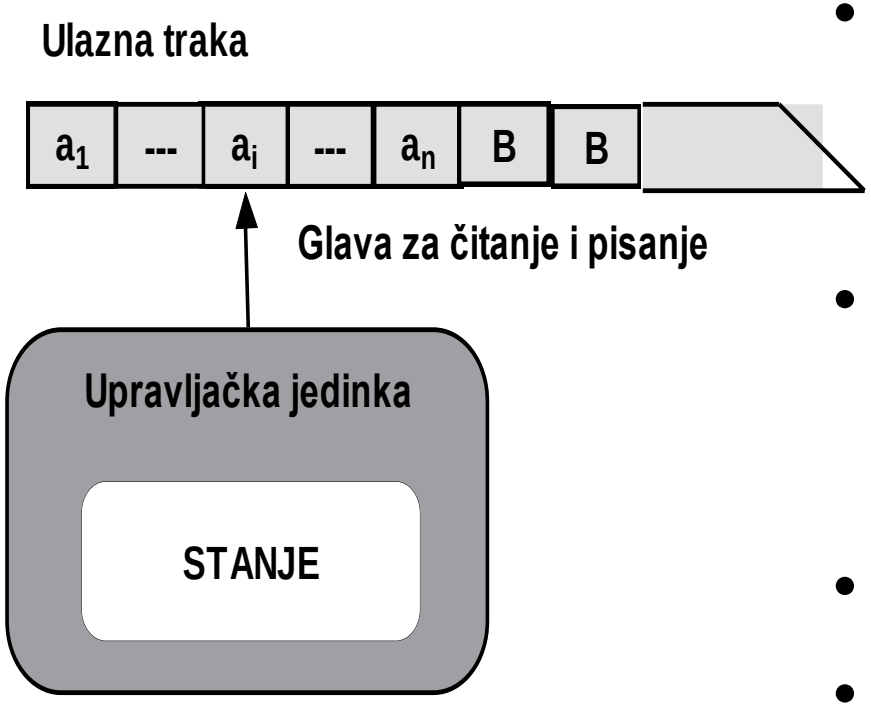
Turingov stroj TS

- TS ima iste mogućnosti računanja kao bilo koje digitalno računalo
  - određene probleme ne možemo riješiti TS-om
  - izvan teorijskih granica izračunljivosti
- predstavlja najopćenitiji matematički model računanja
  - prihvaćanje niza
  - zapis na traci
  - računanje cjelobrojnih funkcija

Gramatika neograničenih produkcija i Turingov stroj

- Gramatika neograničenih produkcija i Turingov stroj su **istovjetni**
  - prihvaćaju klasu rekurzivno prebrojivih jezika
- jezik je rekurzivno prebrojiv neovisan ako i samo ako postoji Turingov stroj koji ga generira
- Type0: Gramatika neograničenih produkcija

Model TS



- upravljačka jedinka nalazi se u jednom od konačnog broja stanja
- TS nakon čitanja znaka sa ulazne trake upisuje novi znak za traku
- glava se miče lijevo-desno
- traka ima početak ali ne i završetak – beskonačna
- na početku je u n krajnje lijevih ćelija zapisan niz w
- prazne ćelije: B

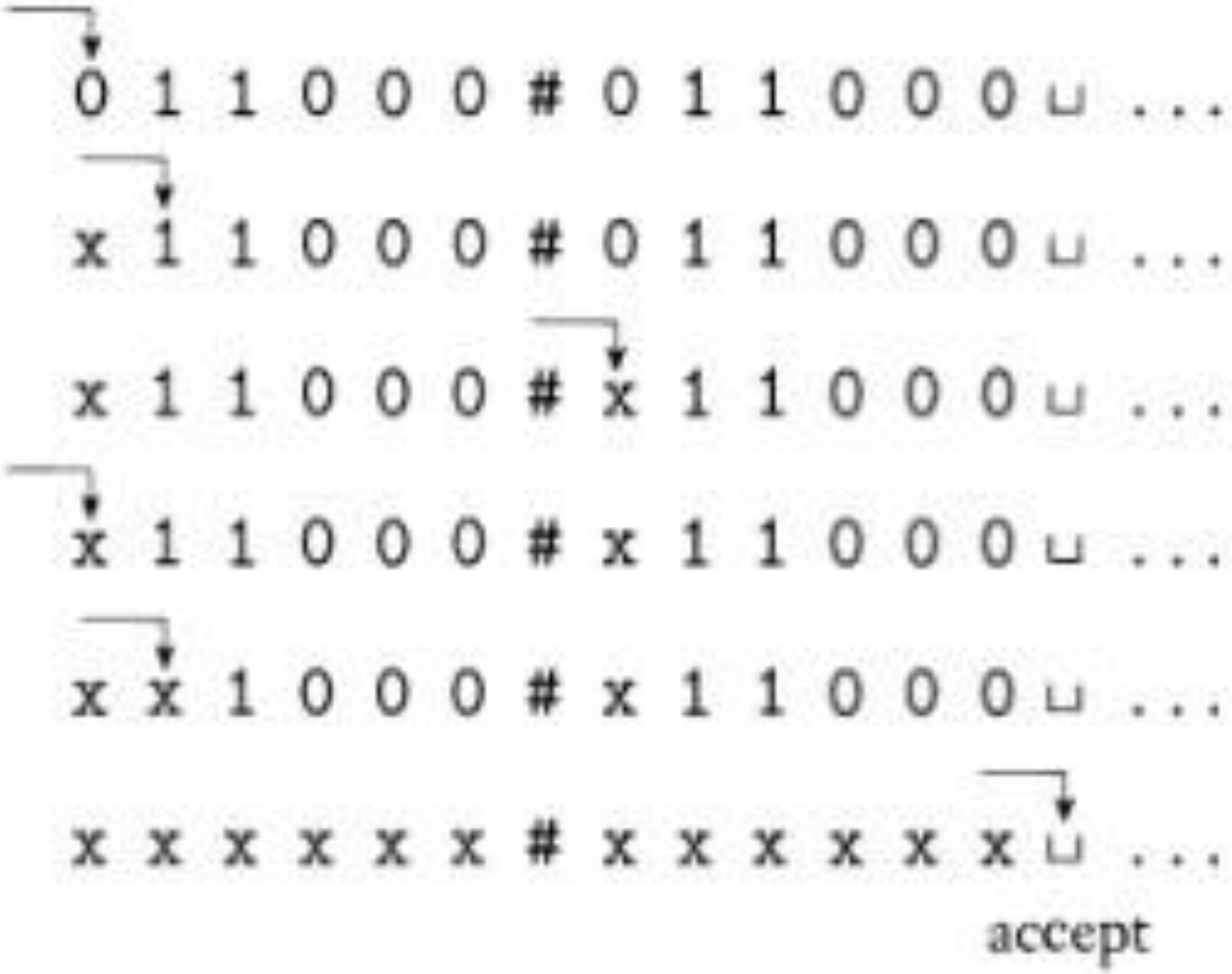
Rad TS

- na osnovu:
  - stanja i
  - pročitanog znaka na traci
- TS odlučuje
  - u koje novo stanje prelazi upravljačka jedinica
  - koji znak se zapiše na traku umjesto pročitanog znaka
  - u koju stranu se miče glava za čitanje i pisanje

Primjer rada TS

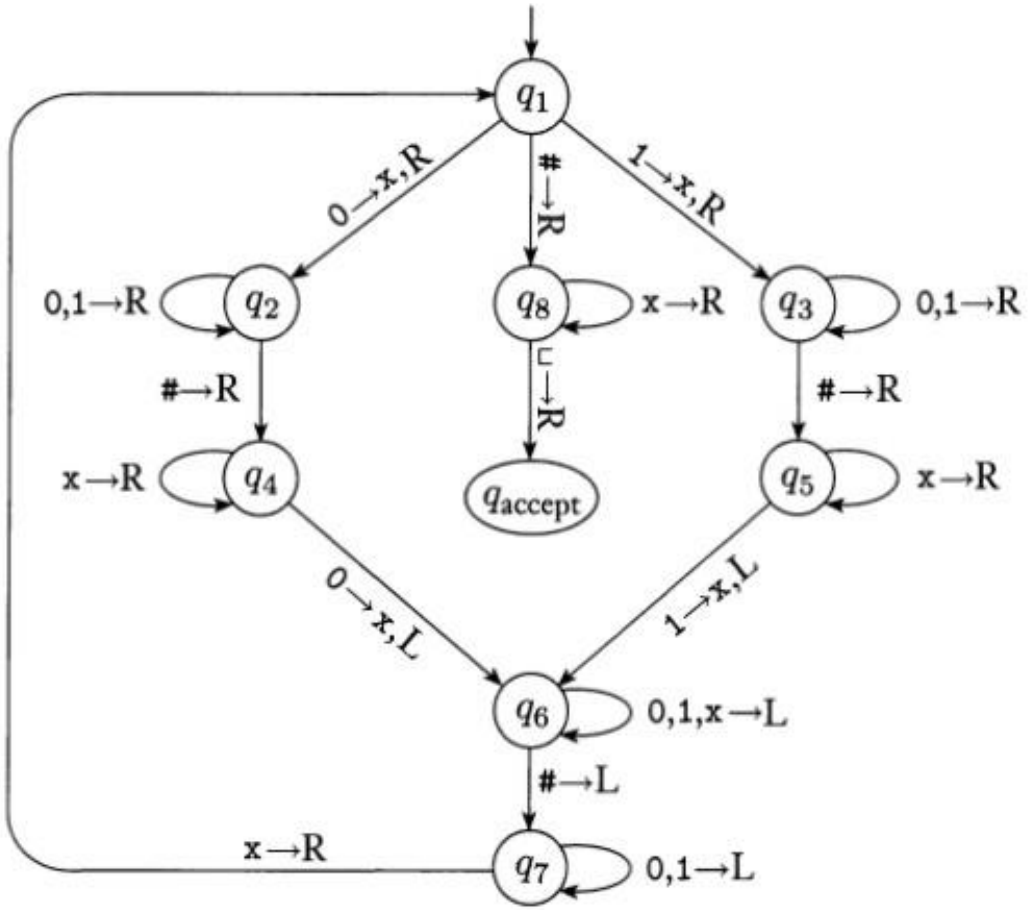
- je li ulazni niz iz jezika $B=\{w\#w \mid w \in \{0,1\}^* \}$
- ulazni niz prolazimo naprijed-natrag
  - najprije provjerimo je li samo jedan znak # u nizu
  - prvi znak s lijeva označimo s X / oznakom da smo ga obradili i provjerimo je li prvi znak nakon # jednak
  - ukoliko je i njega označimo s X
  - postupak ponavljamo dok s lijeve strane od # nemamo same X
  - zatim provjerimo desno stranu ako je sve X prihvatimo inače odbacimo

Primjer rada TS II



Primjer rada TS III

- $0 \rightarrow x, R$ na prijelazu iz q1 u q2 znači ako u q1 pročitano 0 na traku zapišemo x i pomaknemo glavu u desno i pređemo u stanje q2



Formalna definicija TS

- uređena sedmorka **ts=(Q, Σ, Γ, δ, q_0, B, F)**
  - Q – konačni skup stanja
  - Γ- konačni skup znakova trake
  - $B \in \Gamma$ – znak za oznaku prazne ćelije
  - $\Sigma \subseteq (\Gamma - \{B\})$ - konačni skup ulaznih znakova
  - $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$; funkcija prijelaza
    - gdje su L i R oznake za pomak glave u lijevo L ili desno R
  - $q_0 \in Q$; početno stanje
  - $F \subseteq Q$, skup prihvatljivih stanja

Formalna definicija TS II

- Sipser: uređena sedmorka

  ts=(Q, Σ, Γ, δ, q_0, **F-prihvatljiva** st., **R-neprihvatljiva** stanja)

  – TS mora završiti ili u prihvatljivom ili u neprihvatljivom stanju
  – ili radi u beskonačnost

Funkcija prijelaza TS

- $\delta : Q \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$; funkcija prijelaza
  - funkcija prijelaza **može biti nedefinirana** za određene argumente
- $\delta(q,V)=(p,Z,W)$ određuje da TS iz stanja q čitanjem znaka V prelazi u stanje p, na traku zapisuje znak Z (umjesto postojećeg znaka V) te se pomiče u lijevo ili desno ovisno o W

Primjer TS

ts=(Q ={q0, q1,q2,q3,q4}, {0,1}, {0,1,X,Y,B}, δ, q0, B, {q4})

δ(q0,0)=(q1,X,R) δ(q1,1)=(q2,Y,L) δ(q0,Y)=(q3,Y,R)

δ(q1,0)=(q1,0,R) δ(q2,X)=(q0,X,R) δ(q1,Y)=(q1,Y,R)

δ(q2,0)=(q2,0,L) δ(q2,Y)=(q2,Y,L)

δ(q3,Y)=(q3,Y,R)

δ(q3,B)=(q4,B,R)

B- oznaka praznog mjesta na traci

Primjer TS II

- prihvaćanje niza 0011

Traka lijevo od glave	Stanje	Traka desno od glave	Prijelaz
ε	**q0**	**0011BBB...**	δ(q0,0)=(q1,X,R)
X	q1	011BBB...	δ(q1,0)=(q1,0,R)
X0	q1	11BBB...	δ(q1,1)=(q2,Y,L)
X	q2	0Y1BBB...	δ(q2,0)=(q2,0,L)
ε	**q2**	**X0Y1BBB...**	δ(q2,X)=(q0,X,R)
X	q0	0Y1BBB...	δ(q0,0)=(q1,X,R)
XX	q1	Y1BBB...	δ(q1,Y)=(q1,Y,R)
XXY	q1	1BBB...	δ(q1,1)=(q2,Y,L)
XX	q2	YYBBB...	δ(q2,Y)=(q2,Y,L)
X	q2	XYYBBB...	δ(q2,X)=(q0,X,R)
XX	q0	YYBBB...	δ(q0,Y)=(q3,Y,R)
XXY	q3	YBBB...	δ(q3,Y)=(q3,Y,R)
XXYY	q3	BBB...	δ(q3,B)=(q4,B,R)
XXYYB	q4	BB...	q4 prihvatljivo, i nema daljnjih prijelaza nz w 0011 se prihvaća

Primjer TS II

- prihvaćanje niza 011

Traka lijevo od glave	Stanje	Traka desno od glave	Prijelaz
ε	q0	011BBB...	δ(q0,0)=(q1,X,R)
X	q1	11BBB...	δ(q1,1)=(q2,Y,L)
ε	q2	XY1BBB...	δ(q2,X)=(q0,X,R)
X	q0	Y1BBB...	δ(q0,Y)=(q3,Y,R)
XY	q3	1BBB...	q3 nije iz F i nema daljnjih prijelaza niz se ne prihvaća

Primjer TS III

- prihvaćanje niza 001

Traka lijevo od glave	Stanje	Traka desno od glave	Prijelaz
ε	q0	001BBB...	δ(q0,0)=(q1,X,R)
X	q1	01BB...	δ(q1,0)=(q1,0,R)
X0	q1	1BB...	δ(q1,1)=(q2,Y,L)
X	q2	0YBB..	δ(q2,0)=(q2,0,L)
ε	q2	X0YBB..	δ(q2,X)=(q0,X,R)
X	q0	0YBB..	δ(q0,0)=(q1,X,R)
XX	q1	YBB...	δ(q1,Y)=(q1,Y,R)
XXY	q1	BB..	δ(q2,0)=(q2,0,L)
			q1 nije iz F i nema daljnjih prijelaza niz 001 se ne prihvaća

Prihvaćanje niza TS

- prihvaćanje niza: prijelaz među konfiguracijama TS
- konfiguracija TS je trojka: α1 q α2
  - α1 sadržaj ćelija lijevo od glave
  - q stanje upravljačke jedinke
  - α2 sadržaj ćelija desno od glave
- TS na slijedećem koraku čita krajnje lijevi znak iz α2
- TS može na 4 različita načina prelaziti između konfiguracija
  - prijelaz: lijevo ili desno,
  - slijedeći znak je: znak ili prazan znak

Konfiguracije TS I

- TS se nalazi u konfiguraciji: $\mathbf{X_1X_2..\ X_{i-1}q\ X_iX_{i+1}..\ X_n}$

- zadana je funkcija prijelaza TS: $\mathbf{\delta(q, X_i) = (p, Y, L)}$
  - ako je **i-1=n** => glava je postavljena na praznu ćeliju, i desno od glave nema znakova (sve prazno), TS stane
  - ako je **i=1** => glava je postavljena krajnje lijevo, nema više pomaka u lijevo na praznu ćeliju, TS stane
  - ako je **i>1** => moguć je prijelaz u novu konfiguraciju

$$\mathbf{X_1X_2..\ X_{i-1}q\ X_iX_{i+1}..\ X_n > X_1X_2..\ X_{i-2}p\ X_{i-1}YX_{i+1}..\ X_n}$$

glava u lijevo

Konfiguracije TS II

- zadana je funkcija prijelaza TS: $\delta(q, X_i) = (p, Y, R)$
  - moguć je prijelaz u novu konfiguraciju

$$X_1X_2..\ X_{i-1}\,q\ X_iX_{i+1}..\ X_n > X_1X_2..\ X_{i-1}\,Y\,p\ X_{i+1}..\ X_n$$

glava u desno

  - ako je **i-1=n** => onda je $X_iX_{i+1}..\ X_n$ prazan niz i TS zapiše **Y** na kraj niza, TS stane

Primjer rada TS I

- prihvaćanje niza 0011

Traka lijevo od glave	Stanje	Traka desno od glave	Prijelaz
ε	**q0**	**0011BBB...**	δ(q0,0)=(q1,X,R)
X	q1	011BBB...	δ(q1,0)=(q1,0,R)
X0	q1	11BBB...	δ(q1,1)=(q2,Y,L)
X	q2	0Y1BBB...	δ(q2,0)=(q2,0,L)
ε	**q2**	**X0Y1BBB...**	δ(q2,X)=(q0,X,R)
X	q0	0Y1BBB...	δ(q0,0)=(q1,X,R)
XX	q1	Y1BBB...	δ(q1,Y)=(q1,Y,R)
XXY	q1	1BBB...	δ(q1,1)=(q2,Y,L)
XX	q2	YYBBB...	δ(q2,Y)=(q2,Y,L)
X	q2	XYYBBB...	δ(q2,X)=(q0,X,R)
XX	q0	YYBBB...	δ(q0,Y)=(q3,Y,R)
XXY	q3	YBBB...	δ(q3,Y)=(q3,Y,R)
XXYY	q3	BBB...	δ(q3,B)=(q4,B,R)
XXYYB	q4	BB...	q4 prihvatljivo, i nema daljnjih prijelaza nz w 0011 se prihvaća

Konfiguracije: prihvaćanje niza 0011

- $q_0 0011 \vdash X q_1 011 \vdash X0 q_1 11 \vdash X q_2 0Y1 \vdash q_2 X0Y1 \vdash X q_0 0Y1 \vdash XX q_1 Y1 \vdash XXY q_1 1 \vdash$

  $XX q_2 YY \vdash X q_2 XYY \vdash XX q_0 YY \vdash XXY q_3 Y \vdash$

  $XXYY q_3 \vdash XXYY\mathbf{B} q_4$

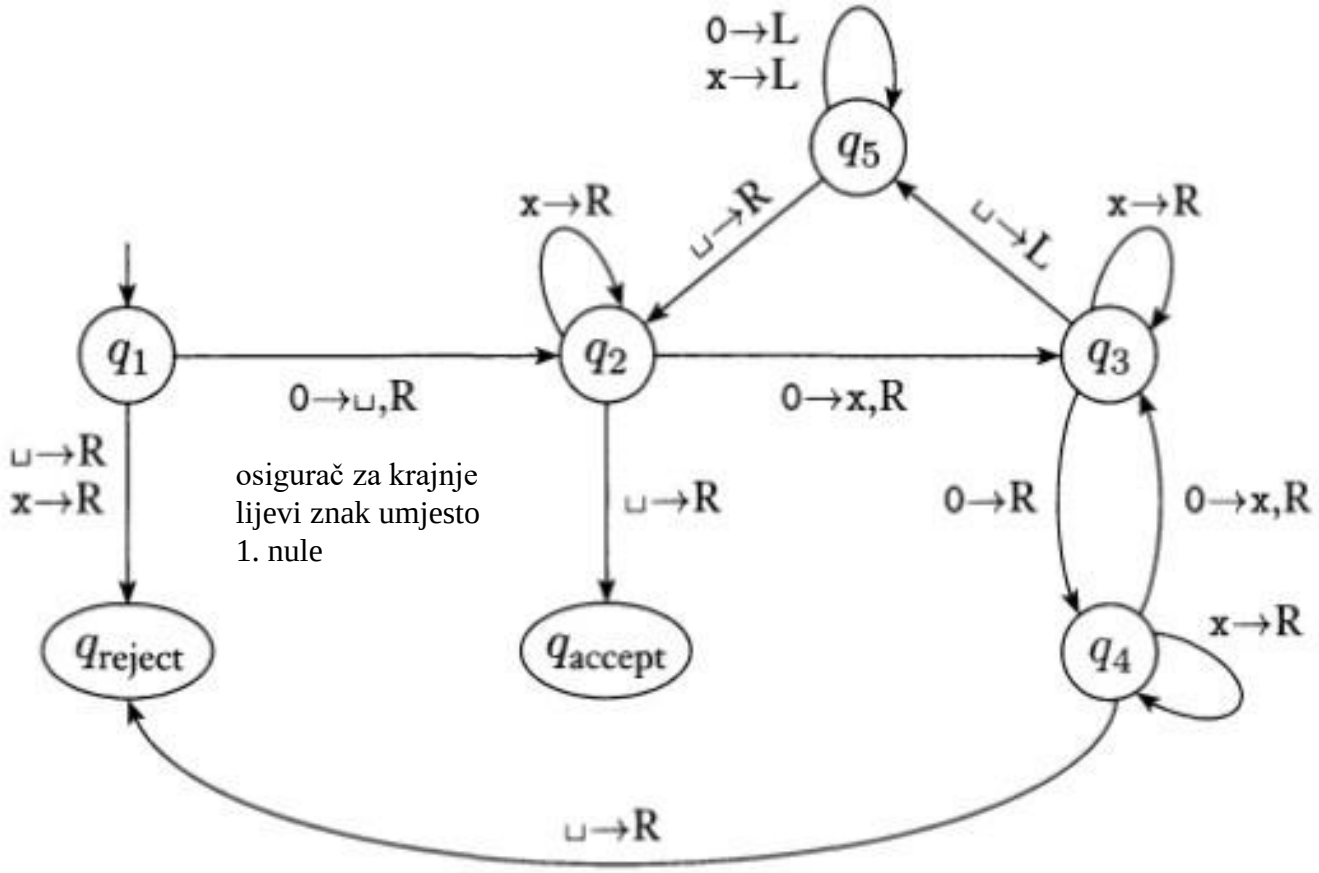
Primjer 1: nizovi duljine 2^n

$$A = \{0^{2^n} \mid n \geq 0\}$$

- postupak:
  - najprije pređi preko svih 0
  - ako je samo jedna 0: prihvati
  - ako je neparan broj 0: odbaci
  - u sljedećem prolazu:
    - prepolovi broj 0
    - ponovi postupak
      - ako je jedna
      - ako je neparan broj

Primjer 1: nizovi duljine 2^n -II

- $0 \rightarrow \sqcup, R$ na prijelazu iz q1 u q2 znači ako u q1 pročitano 0 na traku zapišemo blank i pomaknemo glavu u desno i pređemo u stanje q2

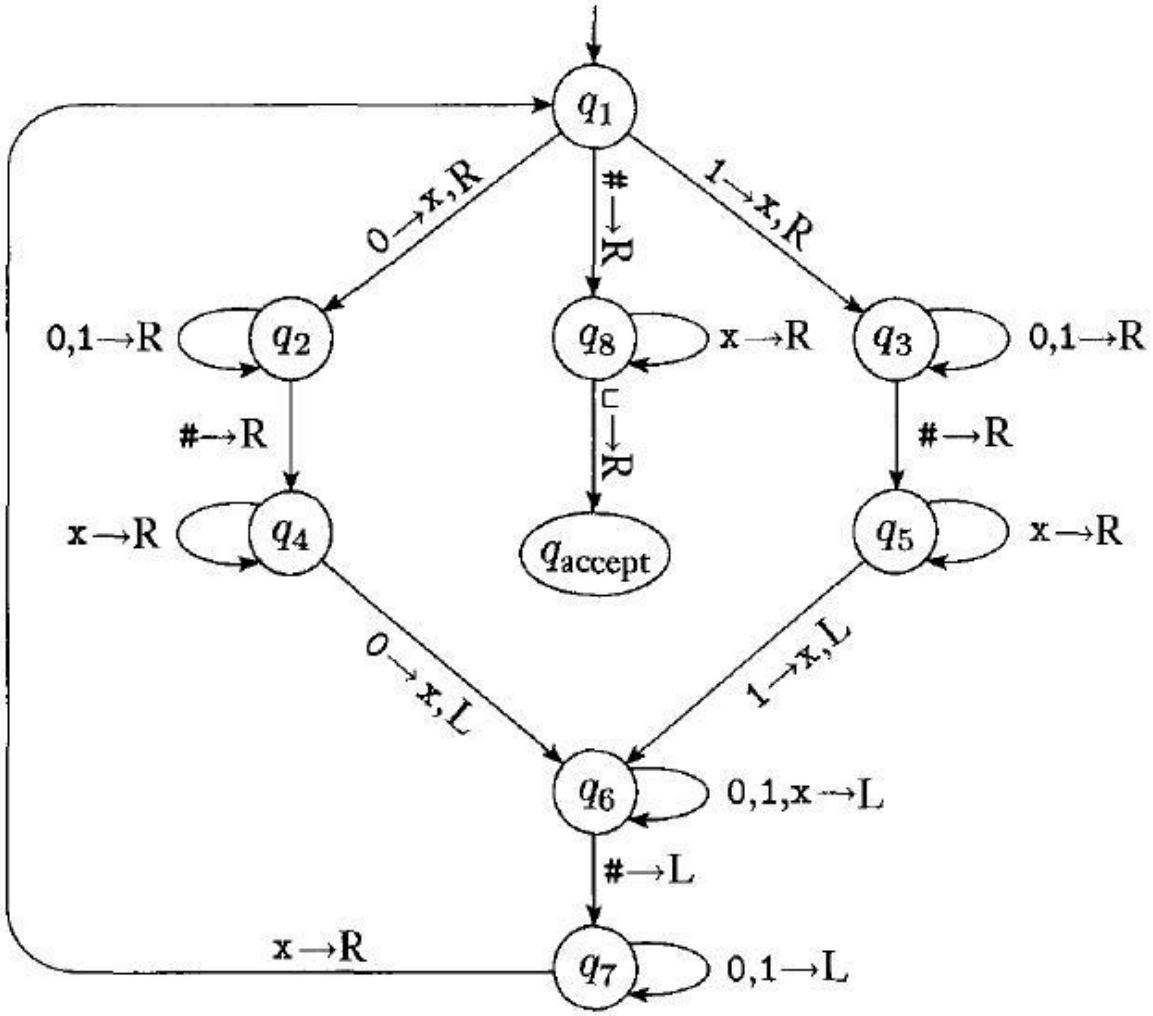


Primjer 1: nizovi duljine 2^n -III

- konfiguracije

$q_1 0000$
$\sqcup q_2 000$
$\sqcup x q_3 00$
$\sqcup x 0 q_4 0$
$\sqcup x 0 x q_3 \sqcup$
$\sqcup x 0 q_5 x \sqcup$
$\sqcup x q_5 0 x \sqcup$
$\sqcup q_5 x 0 x \sqcup$
$q_5 \sqcup x 0 x \sqcup$
$\sqcup q_2 x 0 x \sqcup$
$\sqcup x q_2 0 x \sqcup$
$\sqcup x x q_3 x \sqcup$
$\sqcup x x x q_3 \sqcup$
$\sqcup x x q_5 x \sqcup$
$\sqcup x q_5 x x \sqcup$
$\sqcup q_5 x x x \sqcup$
$q_5 \sqcup x x x \sqcup$
$\sqcup q_2 x x x \sqcup$
$\sqcup x q_2 x x \sqcup$
$\sqcup x x q_2 x \sqcup$
$\sqcup x x x q_2 \sqcup$
$\sqcup x x x \sqcup q_{\text{accept}}$

Primjer 2: nizovi w#w



Primjer: $\{a^i b^j c^k \mid i \times j = k\}$

- postupak
  - pregledaj ulazni niz je li oblika $a^* b^* c^*$
    - ako nije odbaci
  - vrati glavu na početak niza
  - pronađi prvi a, označi i pređi ostale a-jeve dok ne dođeš do b-a
    - vrti se između b-jeva i c-jeva tako da u svakom prolazu eliminiraš po jedan b i jedan c dok se ne riješiš svih b-jeva
  - pronađi sljedeći a – i vrati sve zamjenske znakove na b (c ostavi s X-om)
    - ponovi prethodni postupak
  - nakon što smo prešli sve znakove a i ukoliko nema više c
    - prihvati inače
    - odbaci (c-jeva ima više i ne vrijedi $i \times j = k$)

Prihvaćanje jezika TS

- TS prihvaćaju rekurzivno prebrojive jezike (***Turing-decidable)***
  - **prebrojivi**: moguće je izgraditi TS koji ispisuje (nabraja, broji) sve nizove jezika
  - **rekurzivni**: TS radi u petlji (ne staje)
- kad TS čita ulazni niz radi akcije
  - prihvaća (*accept*), odbija (*reject*) ili radi u petlji (*loop*) (rekurzija)
  - ***deciders*****-** TS koji ili prihvaća ili odbija jezik ali uvijek staje s radom (*decision to accept or to reject*)

Računanje cjelobrojnih funkcija

- cijeli broj se predstavi nizom **0**:
  - broj nula = cijeli broj
  - cijeli broj 000..0 i≥0 : $\mathbf{0^i}$
- ako funkcija ima k argumenata i_1, i_2, ..i_k oni su odvojeni znakom 1: $\mathbf{0^{i1}10^{i2}1..10^{ik}}$
- Rad
  - TS ne stane
  - TS se zaustavi, na traci je 0^m: vrijednost funkcije je **m**, bez obzira u kojem stanju se TS na kraju nalazi

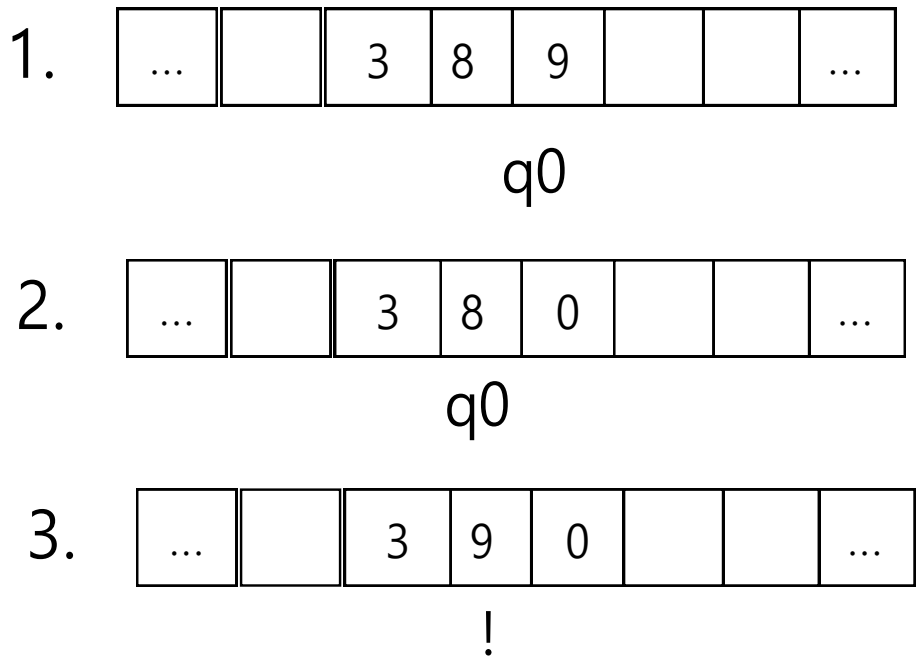
Primjeri cjelobrojnih algoritama na TS

- prijelaz na n+1 decimalu broja n decimalnog broja (I)
- pretvaranje unarnog zapisa u decimalni (II)
- paran i neparan broj jedinica (III)
- zbrajanje (IV)

Primjer I-I prijelaz n+1 decimala

- ulazna abeceda ={0,1,2,3,4,5,6,7,8,9} i prazan znak Λ
- stanja={q0, !} **q0** – radno stanje **!**-prihvatljivo stanje/zaustavljanje
- broj n se zapisuje s n+1 decimalnim mjestom

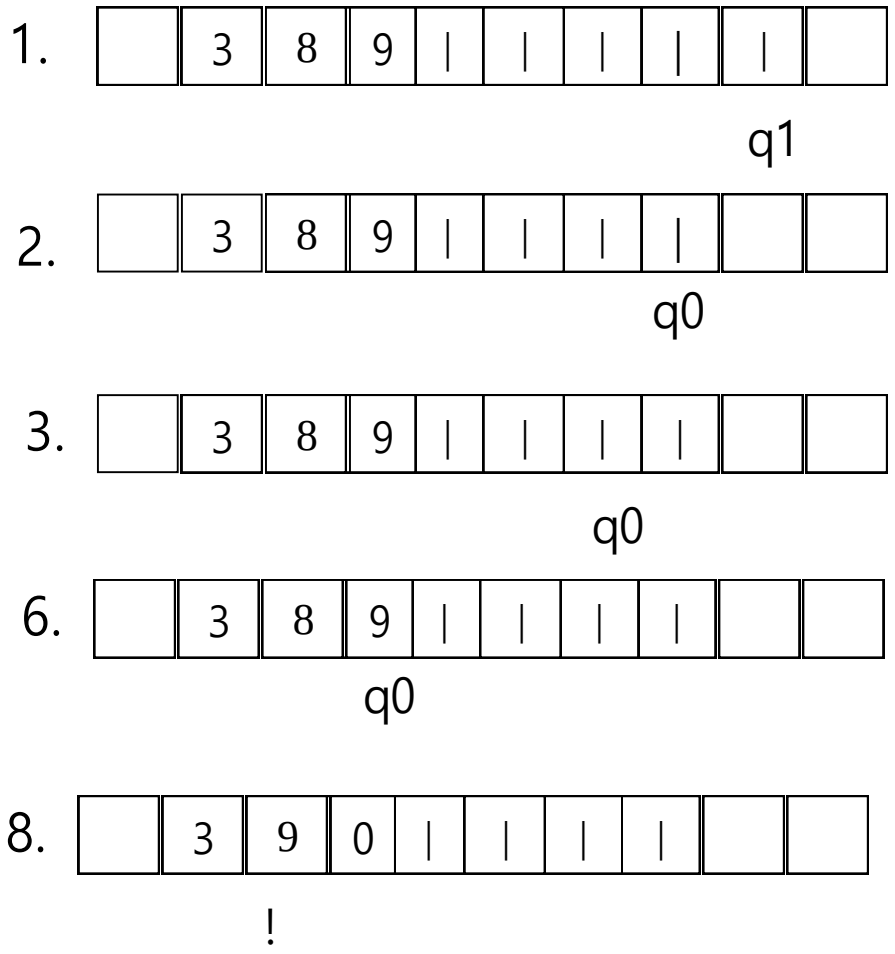
	q0
0	1!
1	2!
2	3!
3	4!
4	5!
5	6!
6	7!
7	8!
8	9!
9	0L
Λ	1!



Primjer I-II

- ulazna abeceda ={0,1,2,3,4,5,6,7,8,9,|} i prazan znak Λ
- stanja={q0, q1, !}
- crtice se preskaču

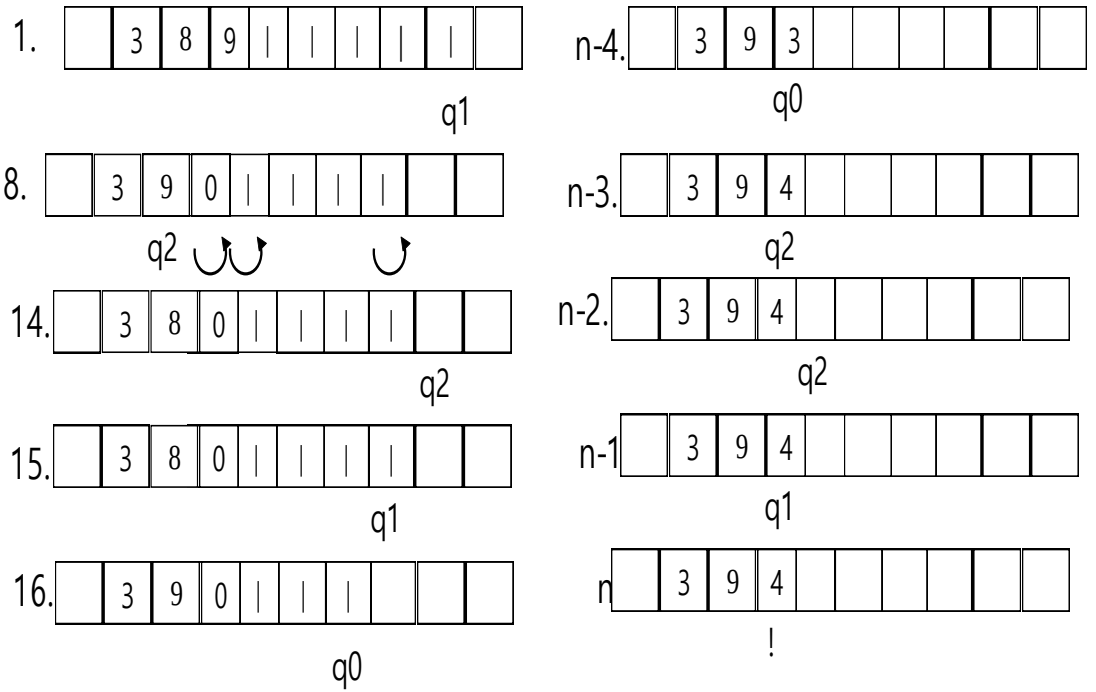
	q0	q1
0	1!	
1	2!	
2	3!	
3	4!	
4	5!	
5	6!	
6	7!	
7	8!	
8	9!	
9	0L	
Λ	1!	
|	L	ΛLq0



Primjer II -zadatak prebroji crtice

- konačan niz ||||| bez razmaka upisan u polja– slog crtica
- ulazna abeceda ={0,1,2,3,4,5,6,7,8,9,|} i prazan znak Λ
- stanja={q0, q1,q2, !}
- 389 + 5 crtica=394

	q0	q1	q2
0	1 q2	!	D
1	2 q2	!	D
2	3 q2	!	D
3	4 q2	!	D
4	5 q2	!	D
5	6 q2	!	D
6	7 q2	!	D
7	8 q2	!	D
8	9 q2	!	D
9	0L	!	D
Λ	1 q2	!	Lq1
|	L	ΛLq0	D



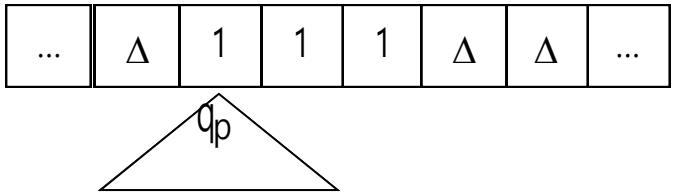
Primjer III-I
paran i neparan broj jedinica

T - neparan broj jedinica

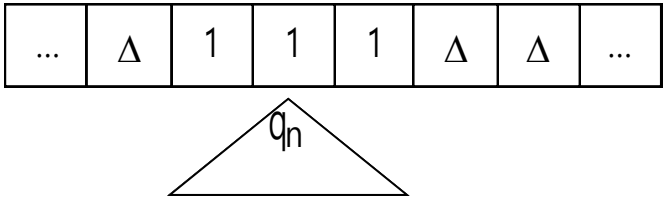
N - paran broj jedinica

	q_p	q_n
1	$1Dq_n$	$1Dq_p$
Δ	NR!	TR!

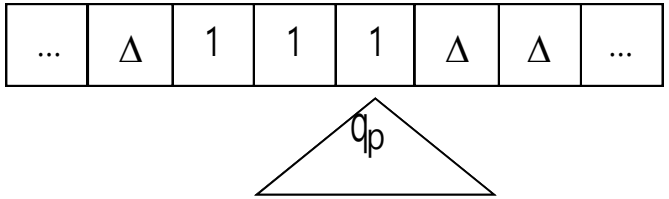
1.



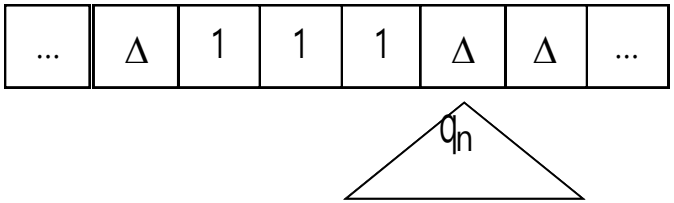
2. $q_p \rightarrow 1Rq_n$



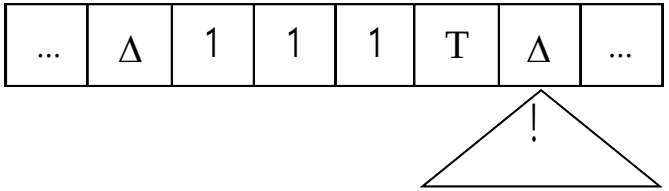
3. $1q_n \rightarrow 1Rq_p$



4. $1q_p \rightarrow 1Rq_n$



5. $\Delta q_n \rightarrow TR!$



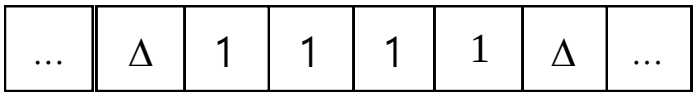
Primjer III-II

T - neparan broj jedinica

N - paran broj jedinica

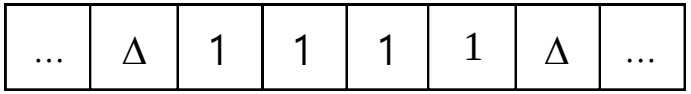
	q_p	q_n
1	$1Dq_n$	$1Dq_p$
Δ	NR!	TR!

1.



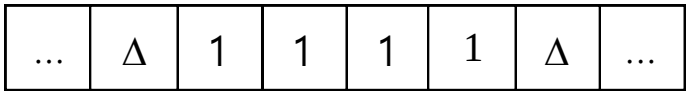
q_p

2.



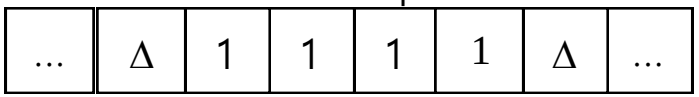
q_n

3.



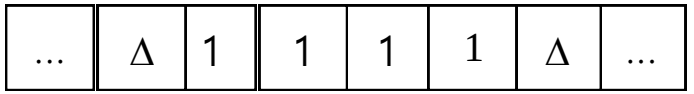
q_p

4.



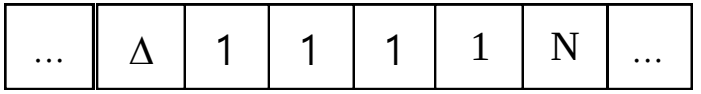
q_n

5.



q_p

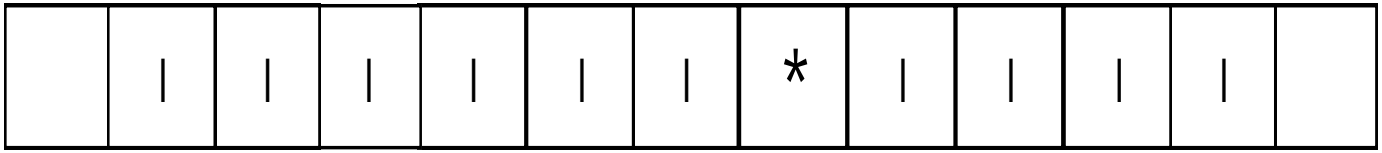
6.

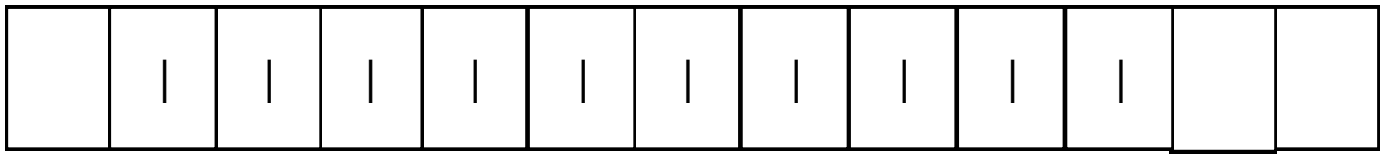


!

Primjer IV- Zbrajanje

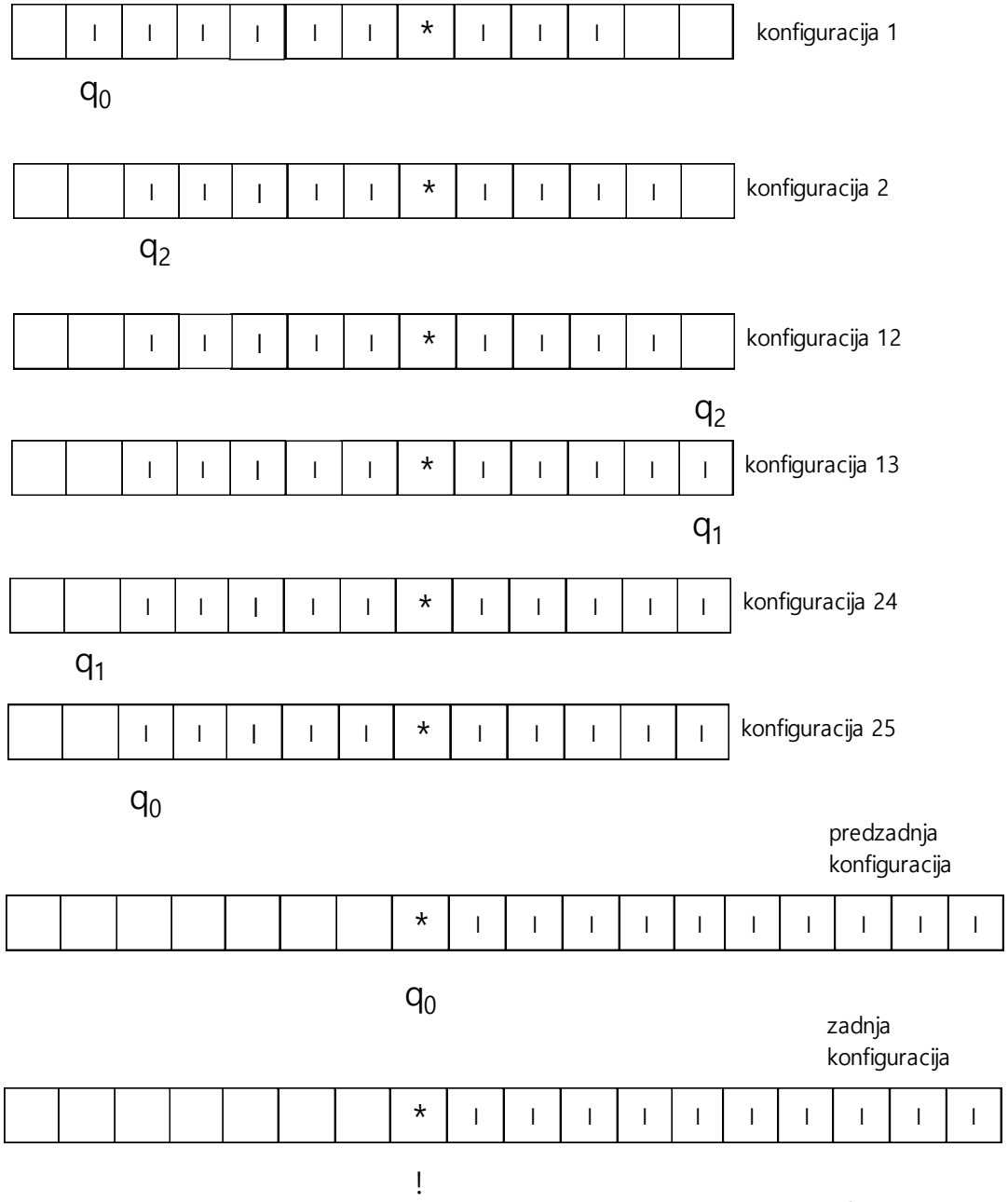
- na traci par brojeva zapisan |
- 6+4 = 10 |





Primjer IV-II –zbrajanje

	q_0	q_1	q_2
I	$\Delta D q_2$	L	D
Δ	D	$D q_0$	$I q_1$
*	Δ!	L	D



Standardni algoritmi na TS

- identični prijepis: E(P)=P
- kopiranje: $Kop_1(P)=P\|P$ ili $Kop_2(P)=P\|P\|P$
- supstitucija: $U^a_\alpha baba=b\alpha b\alpha$
- provjeravanje palindroma
- izdvajanje riječi: Iz(abb||aa||bab)=bab
- preimenovanje svih stanja osim !
- proširenje vanjske abecede
- ...

Primjeri I-IV: Literatura i izvori

- B. A. Trahtenbrot. Što su algoritmi : Algoritmi i računski automati. Moderna matematika, Školska knjiga Zagreb 1978.

- Alan Turing

  http://www.turing.org.uk/turing/

  http://artzia.com/History/Biography/Turing/

Standardni programi na TS

- uvijek s cijelim brojevima
- serijska kompozicija
  - algoritmi A i B: C(P)=B(A(P))
- paralelna kompozicija
  - algoritmi A i B: C(P||H)=A(P)||B(H)
  - teško, TS s 2 trake
- grananje $$C(P) = \begin{cases} A(P) \text{ako vrijedi } \Phi(P) \\ B(P) \text{ako ne vrijedi } \Phi(P) \end{cases}$$
- ponavljanje
  - A(P), A(A(P)), A(A(A(P))) dok vrijedi uvjet Φ(P)

Parcijalne i potpune rekurzivne funkcije

- broj argumenata funkcije $f(i_1, i_2, ..i_k)$ je k
  - nije nužno da su vrijednosti svih argumenata definirane: **parcijalne rekurzivne funkcije**
  - ako su svi argumenti definirani: **potpuna rekurzivna funkcija**
    - primjeri: množenje, n!, 2^{2n}
- parcijalno i potpune rekurzivne funkcije su analogne rekurzivnim jezicima:
  - TS koji za bilo koji ulazni niz stane

Primjer cjelobrojne funkcije: ÷ I

- vrijednost funkcije m÷n:
  - ako je m≥n : m÷n je m-n
  - ako je m<n : m÷n je 0
- argumenti funkcije na traci TS: $\mathbf{0^m10^n}$
- rad:
  - TS zamijeni krajnje lijevi znak 0 s B: $\mathbf{B0^{m-1}10^n}$
  - glava TS se pomakne u desno preko m-1 nula do 1
  - potraži prvu slijedeću nulu i zamijeni je 1: $\mathbf{B0^{m-1}110^{n-1}}$
  - glava TS promijeni smjer: pomakne se u lijevo do B
  - glava TS promijeni smjer: pomakne se u desno

Primjer cjelobrojne funkcije: ÷ II

- rad (nastavak):
  - TS zamijeni krajnje lijevi 0 znak s B: $\mathbf{BB0^{m-2}110^{n-1}}$
  - glava TS se pomakne u desno preko m-2 nula do 1
  - prijeđe preko 1 i potraži prvu slijedeću nulu i zamijeni je 1: $\mathbf{BB0^{m-2}1110^{n-2}}$
  - glava TS promijeni smjer: pomakne se u lijevo do B
- i ponovi se cijeli postupak dok **slučaj 1**:
  - micanjem u desno preko jedinica TS ne nađe 0 nego B
    - znači m≥n (svih n nula je postalo 1): $\mathbf{B^{n+1}0^{m-n-1}1^{n+1}}$
  - zatim se miče u lijevo (nastavak)

Primjer cjelobrojne funkcije: ÷ III

- ili micanjem u lijevo
  - mijenja n+1 jedinica 1 u B
  - prijeđe preko m-n-1 nula
  - mijenja prvi B u nulu (krajnje desni)
  - na traci je m-n nula: $\mathbf{B^n 0^{m-n}}$

Primjer cjelobrojne funkcije: ÷ IV

- i ponovi se cijeli postupak dok **slučaj 2**:
  - micanjem u lijevo preko jedinica TS ne nađe 0 nego B
  - pomakne glavu u desno i umjesto nule nađe 1
    - znači m≤n (svih m nula s početka trake je postalo 1): $\mathbf{B^m 1^{m+1} 0^{n-m}}$
  - zamijeni preostale 0 i 1 s B i vrijednost: m÷n =0
    - $\mathbf{B^{m+n+1}}$

Primjer cjelobrojne funkcije: ÷ V izgradnja TS

- ts=(Q, Σ, Γ, δ, q_0, B, F)
- M=({q0,q1,q2,q3,q4,q5,q6},{0,1},{0,1,B},δ, q_0, B, ∅)
- δ:
  - δ(q0,0)=(q1,B,R) krajnje lijeva 0 ->B i pređe u q1
  - δ(q1,0)=(q1,0,R) prelazi preko nula i traži 1
  - δ(q1,1)=(q2,1,R) kad nađe 1 pređe u q2
  - δ(q2,1)=(q2,1,R) prelazi preko 1i traži krajnje lijevu 0
  - δ(q2,0)=(q3,1,L) kad nađe 0 0->1 i pređe u q3
  - δ(q3,0)=(q3,0,L) prelazi preko nula u lijevo
  - δ(q3,1)=(q3,1,L) prelazi preko 1 u lijevo
  - δ(q3,B)=(q0,B,R) kad nađe B pređe u q0 i počinje iz početka

Primjer cjelobrojne funkcije: ÷ VI izgradnja TS

- slučaj 1
  - $\delta(q2,B)=(q4,B,L)$
  - $\delta(q4,1)=(q4,B,L)$ 1-> B
  - $\delta(q4,0)=(q4,0,L)$ miče se u lijevo preko 0
  - $\delta(q4,B)=(q6,0,R)$ završi radom
- slučaj 2
  - $\delta(q0,1)=(q5,B,R)$ 1->B
  - $\delta(q5,0)=(q5,B,R)$ 0->B
  - $\delta(q5,1)=(q5,B,R)$ 1->B
  - $\delta(q5,B)=(q6,B,R)$ završi radom

Primjer cjelobrojne funkcije: ÷ VII (prihvaćanje niza)

- 2÷1
  - $q_0 0010 > B q_1 010 > B0 q_1 10 > B01 q_2 0 > B0 q_3 11 > B q_3 011 > q_3 B011 > B q_0 011 > BB q_1 11 > BB1 q_2 1 > BB11 q_2 > BB1 q_4 1 > BB q_4 1 > B q_4 > B0 q_6$
- 1÷2
  - $q_0 0100 > B q_1 100 > B1 q_2 00 > B q_3 110 > q_3 B110 > B q_0 110 > BB q_5 10 > BBB q_5 0 > BBBB q_5 > BBBBB q_6$

Složeni TS

- složene oznake ali lakša izrada TS
- definicija TS ostaje ista
- **višekomponentna oznaka stanja**
  - [q1, q2,...qn]
  - upravljački (upravlja radom TS) i radni dio (spremanje podataka)
  - uvodi se radna memorija!
- **višekomponentni znakovi trake**
  - više tragova ulazne trake

TS: Višekomponentna oznaka stanja

- komponenta oznake stanja qi: [q1, q2,...qn]
  - u komponente stanja se mogu pohraniti vrijednosti podataka, korisno za pomak znakova na traci u lijevo ili u desno,...
- upravljačke komponente (upravlja radom TS)
- radne komponente (spremanje podataka)
  - u komponentu se sprema znak sa trake
  - pomak znakova na traci u lijevo ili u desno,...

Primjer: TS s višekomponentnom oznakom stanja I

- složeno stanje [q, a] q- upravljačka, a-radna komp.
- u **a** se sprema znak sa trake
- **q** ima 2 vrijednosti q0 i q1:
  - u q0 se čita krajnje lijevi znak i sprema u a
  - u q1 se čita ostatak znakova na traci i uspoređuje sa znakom u a
- Q={[q0,B],[q0,0], [q0,1], [q1,B],[q1,0], [q1,1]}
  - početno stanje [q0,B]
- TS=({[q0,B],[q0,0], [q0,1], [q1,B],[q1,0], [q1,1]}, {0,1} B,{0,1,B},δ, [q0,B], {[q1,B]})

Primjer: TS s višekomponentnom oznakom stanja II

- Zadan je

TS=({[q0,B],[q0,0], [q0,1], [q1,B],[q1,0], [q1,1]}, B, {0,1},{0,1,B},δ, [q0,B], {[q1,B]})

- prihvaća nizove u kojima se krajnje lijevi znak razlikuje od svih ostalih znakova niza
  - krajnje lijevi znak se ne ponavlja u nizu
  -

Primjer: TS s višekomponentnom oznakom stanja III

- δ: - δ([q0,B],0)=([q1,0],0,R) na početku čitamo komponentu **a**
  - δ([q0,B],1)=([q1,1],1,R)
- ulazni znak se uspoređuje s **a:**
- ako su različiti čita slijedeći znak i pomiče se desno
  - δ([q1,0],1)=([q1,0],1,R)
  - δ([q1,1],0)=([q1,1],0,R)
- ako su jednaki funkcija prijelaza nije definirana
  - ne može radit dalje, ostaje u stanju [q1,0] ili [q1,1], ta stanja nisu prihvatljiva
  - niz se NE prihvaća
- pomak do praznog znaka B
  - δ([q1,0],B)=([q1,B],B,L)
  - δ([q1,1],B)=([q1,B],B,L)
  - završava u stanju [q1,B], prihvatljivo stanje, niz se prihvaća

Primjer: TS koji pomiče znakove 2 mjesta u desno I

- tri komponentna stanja: [q, r1, r2] (r1 i r2 pamte znak)
  - A1, A2 i A3 iz skupa Γ\{B,X}

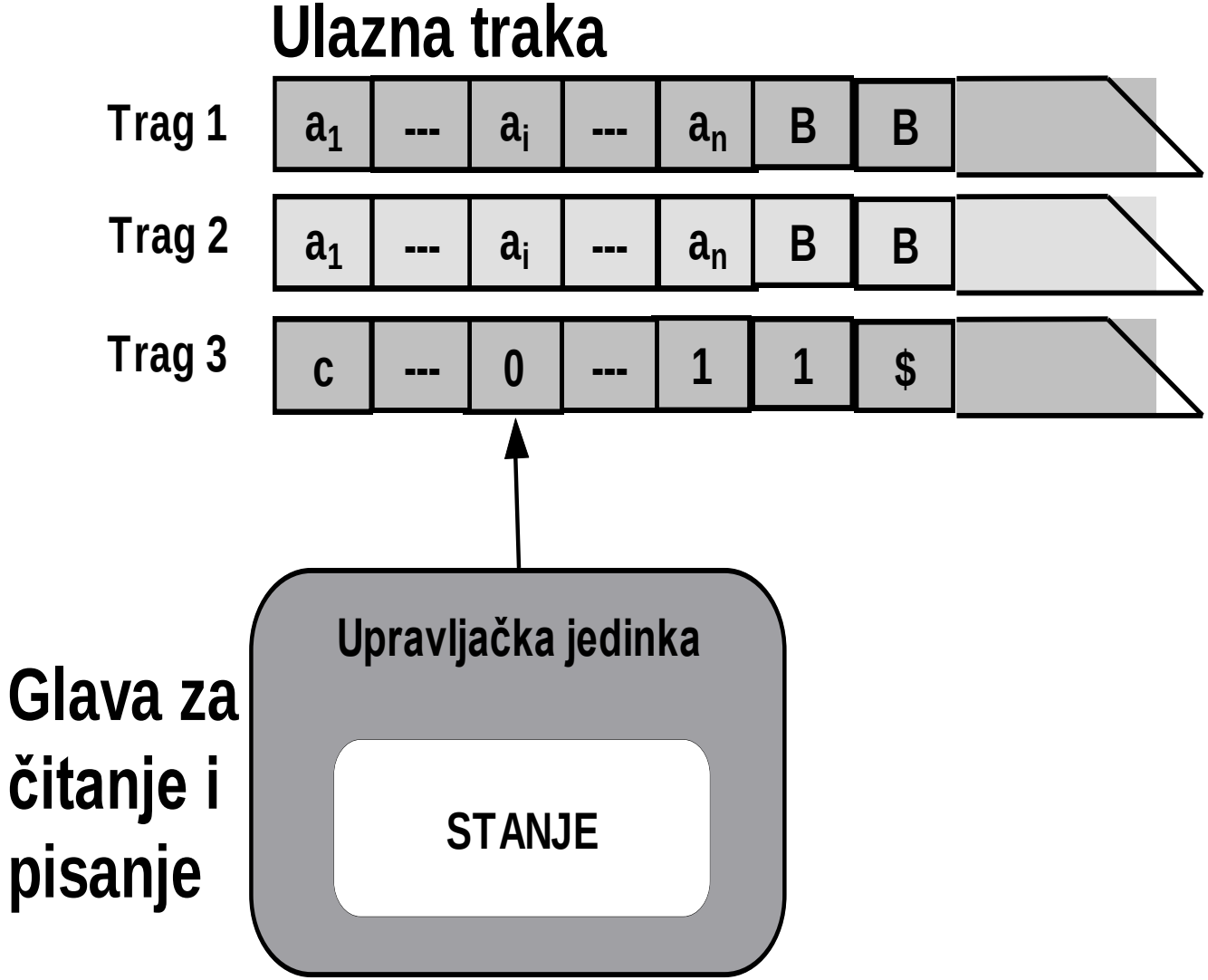
- δ([q1,B,B],A1)=([q1,B,A1],X,R) pročita A1 i zapiše X->A1
- δ([q1,B,A1],A2)=([q1,A1,A2],X,R) pročita A2 i zapiše X->A2
- δ([q1,A1,A2],A3)=([q1,A2,A3],A1,R) pročita A3 i zapiše A1->A3
- δ([q1,A1,A2],B)=([q1,A2,B],A1,R) pročita B i zapiše A1->B
- δ([q1,A1,B],B)=([q2,B,B],A1,L) završen pomak L i zapiše A1->B
- δ([q2,B,B],A)=([q2,B,B],A,L) pomak u lijevo do znaka X
  - A – svi znakovi iz A, kad pročita X prelazi u novo stanje i nastavlja s radom

TS: Višekomponentni znakovi trake

- $A_j=[a_1,a_2,..,a_j]$ je složeni znak trake s n komponenti a_i
- ako je broj komponenti konačan, i ako je broj vrijednosti koje komponente mogu zauzeti konačan onda su konačni i kardinalni brojevi skupa složenih znakova trake i kardinalni brojevi skupa složenih ulaznih znakova što zadovoljava definiciju TS

- traka je podijeljena na zasebne tragove ulazne trake
  - trag 1, trag 2, .. , trag k
  - broj tragova = broju komponenata

TS: Višekomponentni znakovi trake II



Primjer TS: Prim broj I

- TS s 3 traga trake ispituje je li binarni broj prim broj
- ulaz je prvi trag sadrži binarni broj: 10111
  - binarni broj je ograđen graničnicima **#10111$**
- preostala dva traga su pomoćna
  - koriste se za ispitivanje binarnog broja
- složene oznake s 3 komponente
  - [#,B,B], [$,B,B], [0,B,B], [1,B,B]
  - prazna ćelija [B,B,B]

Primjer TS: Prim broj II

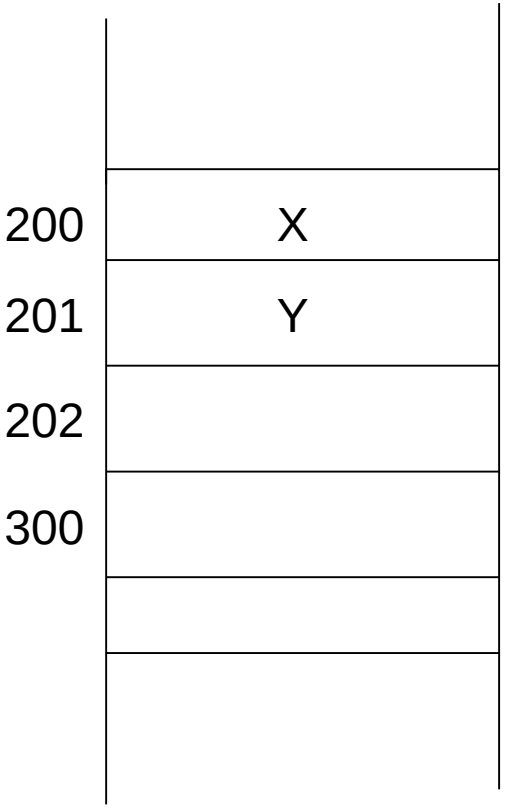
- na tragu 1 je ulazni binarni broj
- na pomoćnim tragovima su
  - trag 2: 2. binarni broj (djelitelj)
  - trag 3: prepisan ulazni binarni broj
- RAD : TS dijeli broj s traga 1 s brojem s traga 2 i zapisuje na trag 3
  - dijeljenje je realizirano s uzastopnim (petlja) oduzimanjem dvaju brojeva: **trag 3 - trag 2 -> trag 3**
  - moguća su 3 ishoda oduzimanja: ostatak ≠0, =0, jednaki brojevi

AOR: Primjer dijeljenja s oduzimanjem

- izračunaj y=X/2

for (i=0; x>1; i++)

x=x-2;



Primjer TS: Prim broj III

- ostatak ≠0 i brojevi na tragu 1 i tragu 2 su različiti: trag2=trag2+1, trag1 -> trag 3
  - postupak dijeljenja se ponovi s djeliteljem koji je povećan za 1
- ostatak =0 onda TS stane
  - brojevi su djeljivi, i ulazni broj nije prim broj
- ostatak ≠0 i jednaki brojevi trag1 = trag 2 onda TS stane
  - ostatak dijeljenja nije jednak 0 za brojeve veće od jedan i manje od ulaznog broja => ulazni broj je prim broj

Primjer TS: Prim broj IV
broj 23 podijeliti s 5

	korak		
trag	1	3	5
1 (23)	#10111$	#10111$	#10111$
2 (5)	BB101BB	BB101BB	BB101BB
3	23: B10111B	13:BB1101B	3:BB11BB

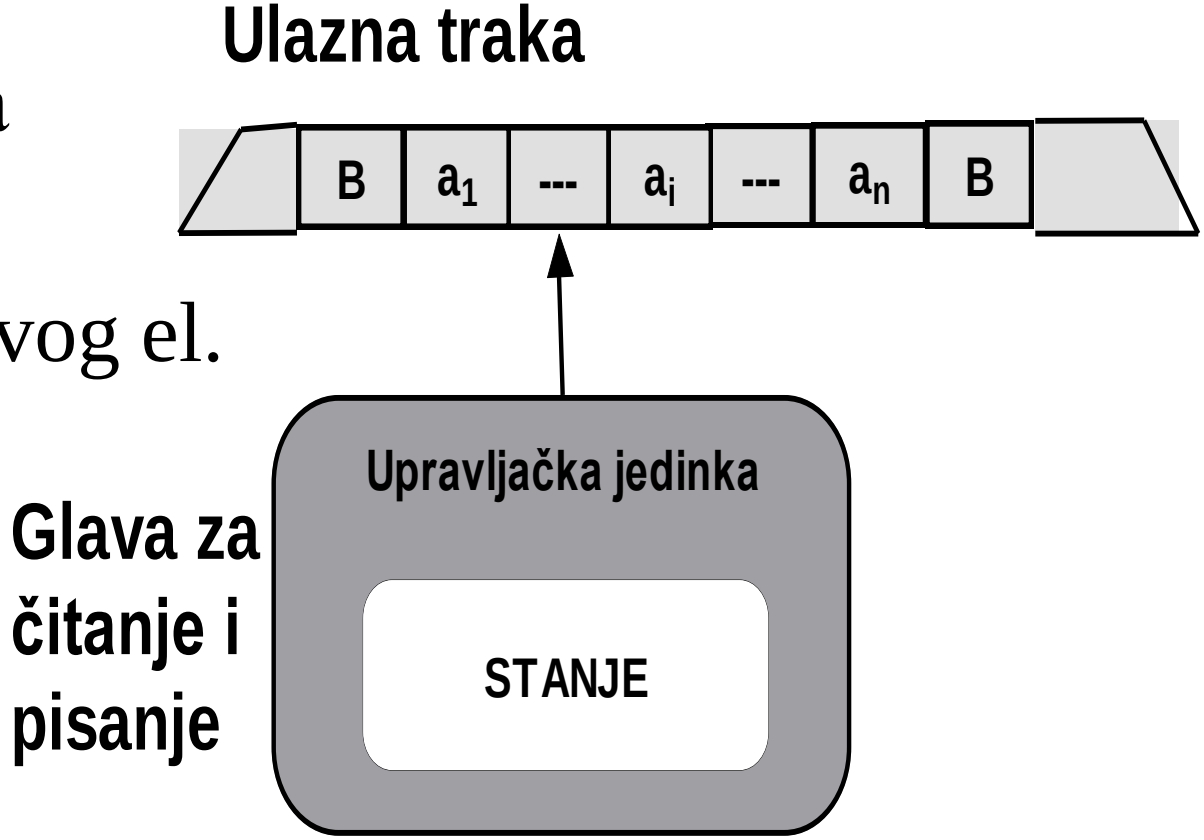
- 2. korak 23-5=18, 3. korak 18-5=13, 4. korak 13-5= 8, 5. korak 8-5=3
- na 5. koraku: ostatak ≠0 i brojevi na tragu 1 i tragu 2 su različiti: trag2=trag2+1, trag1 -> trag 3
  - postupak dijeljenja se ponovi s djeliteljem koji je povećan za 1
  - znači upiše se 6 u tragu 2, i ponovi dijeljenje

Prošireni modeli TS

- složeni problemi zahtijevaju korištenje proširenih modela TS:
  - TS s **dvostranom beskonačnom trakom**
  - TS s **višestrukim trakama**
    - beskonačnim
  - **Nedeterministički** TS
  - TS s **višedimenzionalnim ulaznim poljem**
  - TS s **više glava za čitanje i pisanje**
  - **Neizravni** TS

TS s dvostranom beskonačnom trakom I

- TS s dvostranom beskonačnom trakom istovjetan je osnovnom TS
- traka beskonačna
  - na obje strane
- nema krajnje lijevog el.

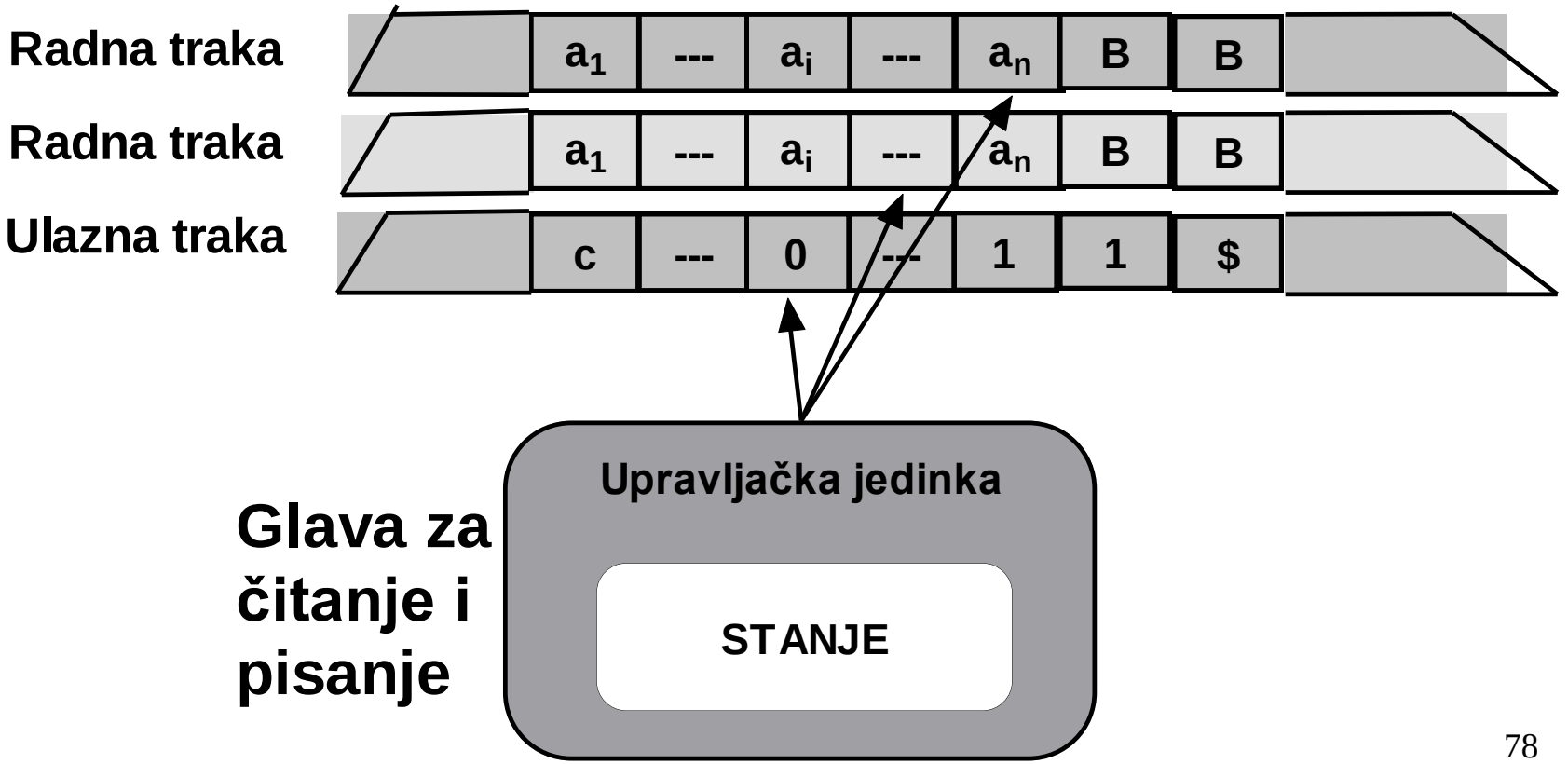


TS s dvostranom beskonačnom trakom II

- relacija promjena konfiguiracije $\succ$ je jednaka kao kod osnovnog TS osim za 2 slučaja
- ako se od zadnjeg znaka pomakne lijevo glava se postavi na praznu ćeliju
  - $\delta(q,X)=(p,Y,L)$ onda je $qX\alpha \succ_M pBY\alpha$
  - kod osnovnog TS $\delta(q,X)=(p,B,R)$ $qX\alpha \succ Bp\alpha$
- kod dvostranog TS vrijedi
  - $\delta(q,X)=(p, B,R)$ onda je $qX\alpha \succ_M p\alpha$
  - nema oznake prazne ćelije lijevo od novog stanja

TS s višestrukim trakama I

- višestruke beskonačne trake na obje strane
  - jedna traka je ulazna ostale su radne
  - za svaku traku jedna nezavisna glava



Rad TS s višestrukim trakama

- k glava za čitanje i pisanje i k dvostrano beskonačnih traka
- upravljačka jedinka TS donosi odluku na temelju dviju grupa parametara:
  - stanja upravljačke jedinke
  - k pročitanih znakova sa k traka
- jednim prijelazom TS
  - promijeni stanje
  - zapiše k znakova na k traka
  - pomakne k glava u lijevo ili desno nezavisno jednu od druge

$$\delta : Q \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R\}^k$$

$$\delta(q_i,a_1,a_2,..,a_k)=(q_j,b_1,b_2,..,b_k,L,R,..,L)$$

- na jednoj traci je ulazni niz koji ispitujemo ostalih k-1 su radne trake

Primjer TS: s 3 trake I

Položaj glave 1		X	...	...	...	
Sadržaj trake 1	A1	A2	...	...	...	Am
Položaj glave 2			...	X	...	
Sadržaj trake 2	B1	B2	...	...	...	Bm
Položaj glave 3	X		...	...	...	
Sadržaj trake 3	C1	C2	...	...	...	Cm

- istovjetni osnovni TS se gradi tako:
  - svaka traka ima 2k tragova (2 traga za svaku traku)
  - na prvom tragu je položaj glave X a na drugom sadržaj
  - svako stanje ima k+2 komponenti
    - u prvom je pravo stanje, u drugom je brojilo za oznake položaje glave, a ostalih k pohranjuje k znakova sa k traka

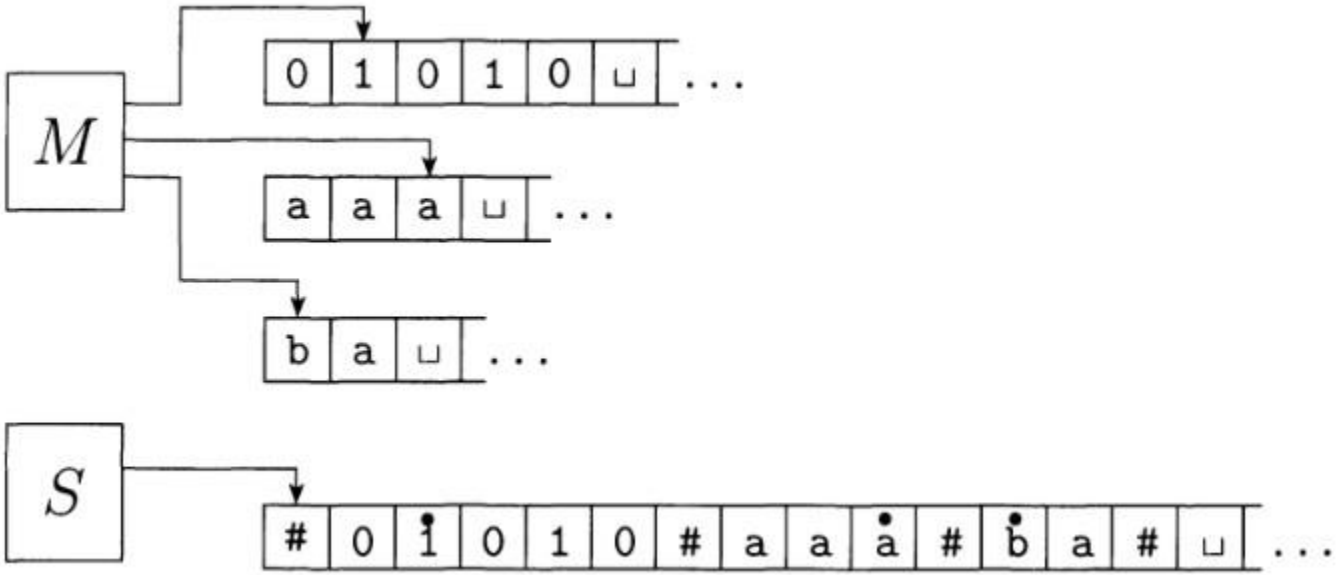
Primjer TS: s 3 trake II

- jedan prijelaz TS izvodi se u 2 koraka:
- glava se miče od **krajnje lijeve do krajnje desne** oznake položaja glave
  - kad naiđe na oznaku u k-tom tragu pročita znak u odgovarajuću k-tu komponentu
  - brojilo provjeri jesu li obiđene sve oznake položaja glava
    - kad se nađe oznaka položaja glave brojilo se smanji za 1
    - kad je brojilo 0 TS ima sve potrebne podatke za donošenje odluke: ili prijelaz, ili prihvaćanje
    - ako postoji prijelaz TS počne raditi u suprotnome smjeru

Primjer TS: s 3 trake III

- glava se miče od **krajnje desne do krajnje lijeve** oznake položaja glave (u suprotnom smjeru)
  - kad naiđe na oznaku u k-tom tragu promjeni sadržaj ćelije u odgovarajućem tragu
  - pomakne se na novu oznaku položaja
  - brojilo provjeri jesu li obiđene sve oznake položaja glava
    - kad se nađe oznaka položaja glave brojilo se smanji za 1
    - kad je brojilo 0 TS ima sve potrebne podatke za donošenje odluke: ili prijelaz, ili prihvaćanje
    - ako nastavlja rad počinje prijelaz u suprotnome smjeru

TS s 3 trake u osnovni TS



- s • je označen el. s kojim se trenutno radi
- sadržaj traka je razdvojen s #

Nedeterministički TS

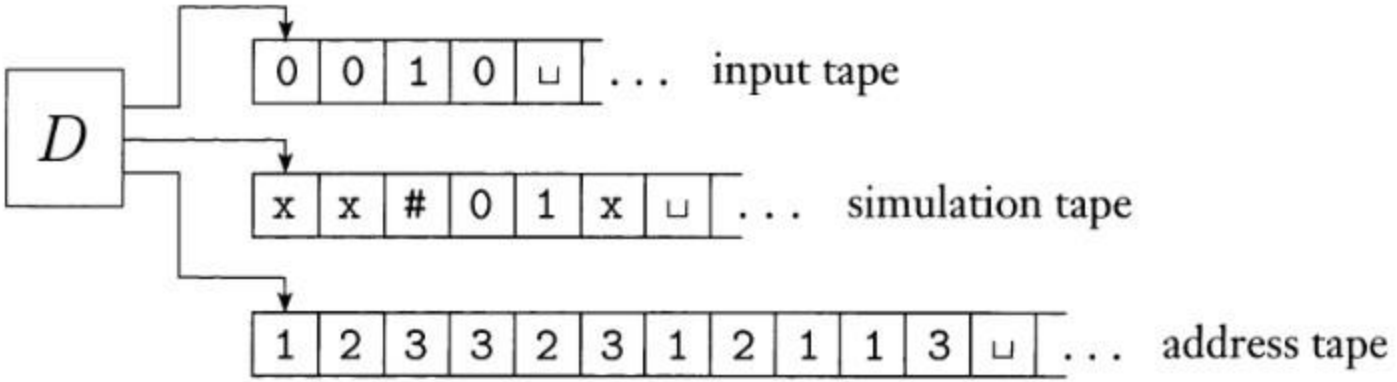
- funkcija δ nije jednoznačna

$$\delta : Q \times \Gamma \rightarrow \wp(Q \times \Gamma \times \{L, R\})$$

$\delta(q,X)=\{(p_1,Z_1,D_1),(p_2,Z_2,D_2), .., (p_k,Z_k,D_k)\}$

  - prijeđe u skup stanja
  - u stanju p_i, zapiše znak Z_i i pomakne u smjeru D_i
- nedeterminizam: gradi se stablo koje se grana na svakom koraku
  - ako barem jedna slijed (grana) završi u prihvatljivom stanju niz se prihvaća
- nedeterministički TS je istovjetan determinističkom TS

Nedeterministički TS II



- Nederministički TS simuliramo determinističkim TS-om s 3 trake
- 3. traka sadrži adresu čvora u kojem se nalazimo
  - sadrži sva grananja
  - adresa 231: polazimo iz korijena u sljedeću razinu i to s lijeva 2 dijete pa 3 dijete pa 1 dijete

Nedeterministički TS III

- postupak
  - na početku je na 1. traci ulazni niz w, a ostale dvije trake su prazne
  - sadržaj 1. trake se kopira na 2. traku
  - na 2. traci se simulira rad jedne grane ndt.TS-a
    - radi prijelaze u skladu s prijelazima ndt. TS-a
    - prije svakog prijelaza provjeri dozvoljena grananja iz 3. trake
    - ukoliko dođeš u prihvatljivo stanje prihvati
    - ukoliko nema više grananja prijeđi na sljedeći korak
  - na 3. traku zapiši adrese (povećaj za 1) slijedeće grane i ponovi prijašnji korak

Decider

- nedeterministički TS koji za svaki ulazni niz stane u svakoj od grana

TS s višedimenzionalnim ulaznim poljem I

- umjesto trake se koristi k-dimenzionalno polje ćelija

B	B	B	a1	B	B	B
B	B	a2	a3	a4	a5	B
a6	a7	a8	a9	B	a10	B
B	a11	a12	a13	B	a14	a15
B	B	a16	a17	B	B	B

- k-dim polje se sprema na traku kao niz:**BBBa1BBB*BBa2a3 a4a5B*a6a7a8a9Ba10B*Ba11a12a13Ba14a15*BBa16a17BBB**
- glavu je moguće pomaknuti na 2k različita načina
  - oko svake k osi u oba smjera

Primjer: TS 2D ulazno polje > TS s jednom trakom s dva traga I

- jedan trag označava položaj glave, a stanje 2D TS se spremi u komponentu stanja
- moguća su 4 pomaka glave
  - horizontalno unutar pravokutnika
  - vertikalno unutar pravokutnika
  - horizontalno izvan pravokutnika
  - horizontalno izvan pravokutnika

TS s više glava za čitanje i pisanje

- jedna traka i k glava
- glave se nezavisno miču u lijevo ili u desno
  - TS donosi odluku na temelju stanja i k pročitanih znakova
- TS s k glava istovjetan je TS-u s jednom glavom
  - jedna traka ima k+1 tragova
  - prvi trag ima sadržaj
  - a k tragova označava položaje k glava

Neizravni TS

- koristi se za dokazivanje prostorne složenosti
  - prostorna složenost prihvaćanja jezika
  - prostorna složenost računanja cjelobrojnih funkcija
- jedna ulazna (samo se čita) i više radnih traka
  - niz na ulaznoj traci je označen graničnicima, glava ne može otići izvan graničnika
- neizravni TS je istovjetan TS-u s višestrukim trakama

Istovjetnosti TS

- TS s dvostranom beskonačnom trakom istovjetan je osnovnom TS
- TS s višestrukim trakama istovjetan je osnovnom TS
- nedeterministički TS je istovjetan determinističkom TS
- TS s dvodimenzionalnim poljem istovjetan je TS s jednodimenzionalnom trakom
- TS s k glava istovjetan je TS-u s jednom glavom
- neizravni TS je istovjetan TS-u s višestrukim trakama

Pojednostavljeni model TS

- Pojednostavljeni TS su istovjetni osnovnom TS

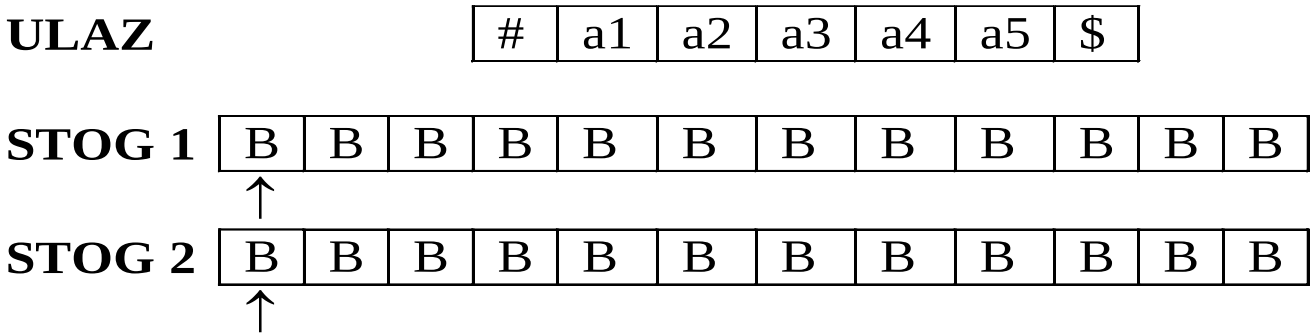
- **Stogovni stroj**
- **Stroj s brojilima**
- TS s **ograničenim** brojem stanja i znakova trake
- **Univerzalni** TS

Stogovni stroj I

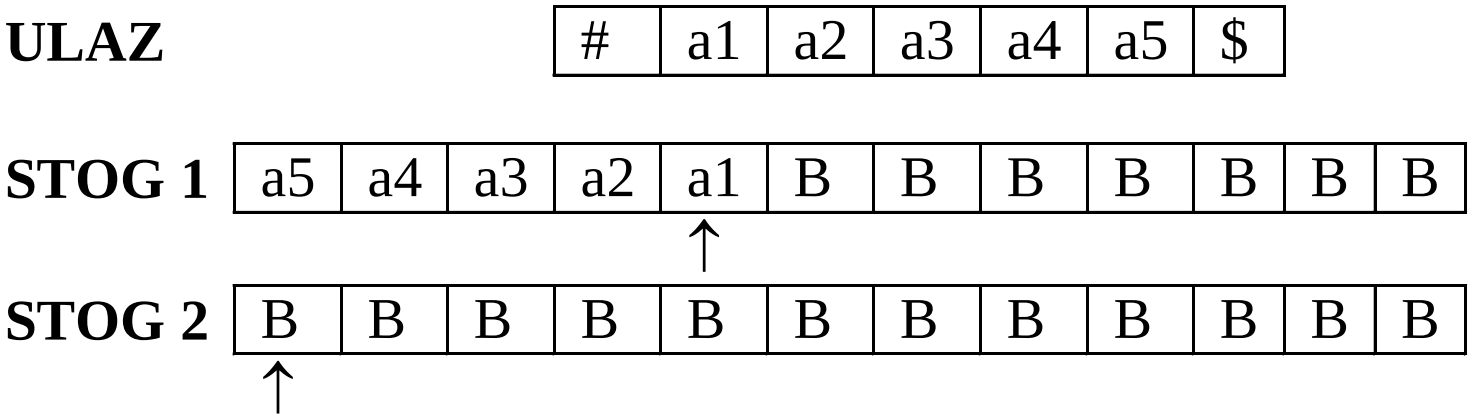
- deterministički TS s jednom ulaznom trakom i više stogova
  - ulaznu traku je moguće samo čitati
  - stogovi su realizirani kao posebne radne trake s pojednostavljenom funkcijom prijelaza
    - ako se glava pomakne u lijevo u ćeliju se obavezno upiše oznaka prazne ćelije
    - sve ćelije stoga desno od glave za pisanje su prazne
- TS s jednom trakom moguće je simulirati stogovnim strojem s dva stoga

Stogovni stroj II

- na početku rada na ulaznu traku se zapiše ulazni niz
- oba stoga su prazna

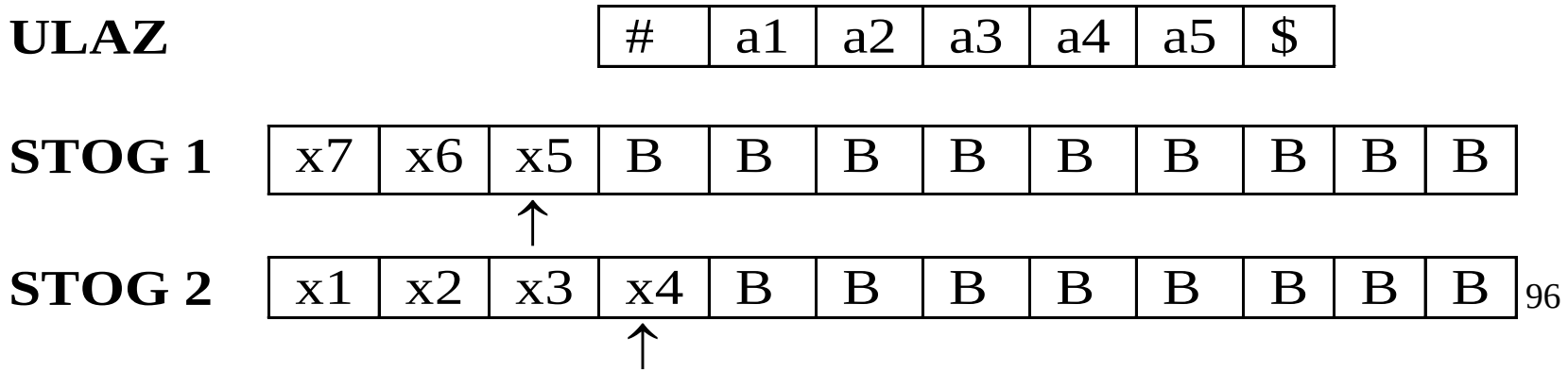


- ulazni niz se prepiše na stog (sve ćelije desno od glave su prazne)



Stogovni stroj III

- pomakne li se glava stroja u lijevo smanji se broj znakova na stogu 2 a poveća na stogu 1
- kad na stogu 1 dođe do kraja (prazan) na donjem stogu su spremljeni znakovi od krajnje lijeve ćelije do glave
  - donji se i dalje može povećavati a gornji ostaje prazan
- na slici korak za konfiguraciju x1x2x3x4px5x6x7
  - ako glava ide u desno smanji se stog 1 a poveća stog 2
  - ako glava ide u lijevo smanji se stog 2 a poveća stog 1



Stroj s brojilima I

- pojednostavi se stogovni stroj
  - umjesto 2 stoga 4 brojila
- skup znakova stoga sadrži:
  - oznaku prazne ćelije B
  - oznaku dna stoga X
- glavu nije moguće pomaknuti lijevo od oznake dna stoga X
- ako se glava pomakne u desno za **i** ćelija u brojilo se upiše **i**
- micanjem glave povećava se ili smanjuje vrijednost brojila

Stroj s brojilima II

- kad glava čita X (dno stoga) u brojilu mora biti 0
- TS s jednom trakom moguće je simulirati strojem s četiri brojila
  - TS s jednom trakom simulira se stogovnim strojem s dva stoga
  - rad jednog stoga simulira se s dva brojila
- količinu brojila moguće je smanjiti na 2
  - pa je TS s jednom trakom moguće simulirati strojem s dva brojila
  - rad četiri brojila simulira s pomoću dva brojila

TS s ograničenim brojem stanja i znakova trake

- istodobno se ograniči broj stanja, broj traka i broj znakova trake
  - s time se ograniči broj različitih TS koje je na taj način moguće izgradit
  - ne prihvaća isti skup jezika kao osnovni TS
- ograniči se broj stanja na 3 a broj znakova trake i traka nije ograničen
  - onda je za prihvaćanje bilo kojeg rekurzivno prebrojivog jezika moguće imati jednu traku i 3 stanja (1 prihvatljivo i 2 neprihvatljiva)
- ograniči se broj znakova {0,1,B} i traka a broj stanja nije ograničen

Univerzalni TS

- univerzalni TS M0 može simulirati bilo koji TS s jednom trakom
- ima 3 trake
  - prva: funkcije prijelaza TS i ulazni niz w <M,w> (kodiran s oznakam # i • itd.)
  - druga: traka za rad
  - treća: stanje u kojem se TS nalazi
- prepiše ulazni niz w na drugu traku
- simulira prijelaze TS s druge trake uzme stanje s treće i zapisuje prijelaz u stanje na treću traku
  - ako nema prijelaza za znak s 2 i stanje s 3 se rad TS zaustavlja
  - ako je stanje na trećoj traci prihvatljivo niz se prihvaća

Generiranje jezika TSom I

- TS s višestrukim trakama
  - jedna traka je izlazna: kad se znak zapiše na izlaznu traku više ga nije moguće mijenjati ili brisati
  - glava se na izlaznoj traci miče isključivo u desno
  - izlazna traka koristi se za generiranje jezika G(M)
  - izlazni nizovi nad abecedom Σ su podijeljeni s #
- jezik kojeg TS generira G(M) (nije isti kao L(M) kojeg TS prihvaća) je skup nizova $w \in \Sigma^*$

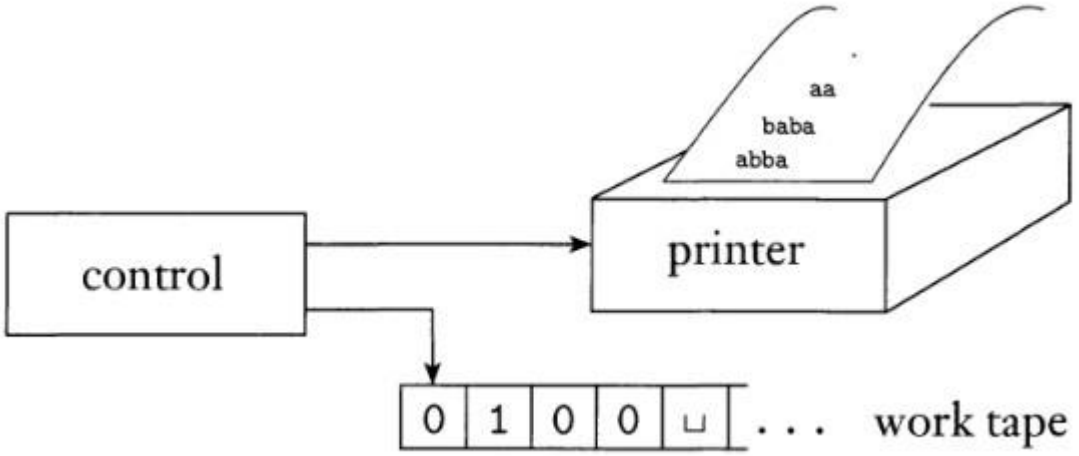
Generiranje jezika TSom II

- ako TS stane je G(M) konačan
- ako TS ne stane je G(M) beskonačan
- isti niz moguće je generirati više puta

- TS **generira** klasu rekurzivno prebrojivih jezika
  - jezik L je rekurzivno prebrojiv ako i samo ako postoji TS M1 koji generira jezik G(M1)=L

Sipser

- Turing recognizable languages- rekurzivno prebrojivi



- Turing decidable languages- rekurzivni, odlučivi

Prihvaćanje jezika generiranog TSom I

- za bilo koji TS M2 koji prihvaća jezik L(M2) moguće je izgraditi TS M1 koji generira jezik G(M1)=L(M2)
  - TS M1 može biti jednostavan pa generira rekurzivni jezik G(M1)=L(M2)
  - TS M1 može biti složen pa generira rekurzivno prebrojiv jezik G(M1)=L(M2)

Prihvaćanje rekurzivnog jezika generiranog TSom II

- TS M1 može biti jednostavan pa generira rekurzivni jezik G(M1)=L(M2)
  - određenim redom M1 ispisuje sve nizove w1,w2,w3,.. iz Σ* na radnu traku
  - zatim se simulira rad M2
    - ako se niz prihvati prepiše ga se na izlaznu traku
  - budući da je G(M1)=L(M2) rekurzivan jezik simulacijom prihvaćanja TS M2 uvijek stane s radom (nema beskonačnih petlji)

Prihvaćanje rekurzivno prebrojivog jezika generiranog TSom III

- TS M1 može biti složen pa generira rekurzivno prebrojiv jezik G(M1)=L(M2)
  - moguće je da postoji niz w_j koji nije u jeziku L(M2) i za koji TS M2 prilikom simulacije prihvaćanja nikad ne stane
    - to znači da se nikada ne pređe u ispitivanje prihvaćanja nizova w_{j+1}, w_{j+2},...
    - ako je među tim nizovima w_{j+k} niz koji se nalazi u L(M2) i nije na izlaznoj traci a M1 bi ga trebao generirati
    - rješenje: M2 ne smije imati neograničen broj prijelaza (broj prijelaza se ograniči)

LITERATURA

- S. Srbljić: *Jezični procesori I + II*, Element, Zagreb, 2002.
- B.A. Trahtenbrot Što su algoritmi : *Algoritmi i računski automati*. Moderna matematika, Školska knjiga Zagreb 1978.
- J.E. Hopcroft, J.D. Ullman: *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley, USA,1979.
- Michael Sipser, *Introduction to the Theory of Computation, second edition*, Course Technology, MIT, 2005.

Formalni jezici i jezični procesori I

TURINGOV STROJ

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

Stroj s četiri brojila I

- k različitih znakova stoga: $Z_0, Z_1, Z_2,..,Z_{k-1}$
- na stogu su zapisani znakovi: $Zi_1, Zi_2,..,Zi_m$
- $i_1, i_2,..,i_m$ indeksi znakova su cjelobrojne vrijednosti
- na vrhu stoga je Zi_m koji ima vrijednost i_m
- vrijednost znakova stoga je cijeli broj po bazi k
  - $j= i_m+k\ i_{m-1}+k^2\ i_{m-2}+k^3\ i_{m-3}+..+k^{m-1}\ i_1$
- osnovne operacije:
  - stavljanje znaka na vrh stoga
  - uzimanje znaka s vrha stoga
  - određivanje znaka na vrhu stoga

Stroj s četiri brojila II

- stavljanje znaka Zr na vrh stoga
- na stogu su: Zi_1, Zi_2,..,Zi_m,$Zi_{r,,}$
  - a njihova vrijednost je jk+r
  - s dva brojila A i B se računa vrijednost jk+r
    - na početku A: j i B: 0
  - jk se izračuna:
    - A: j-1 B:k i smanjuje se dok u A: 0 a u B: jk,
    - na zadnjem koraku se u B doda r B: jk+r

Stroj s četiri brojila III

- uzimanje znaka Zi_m s vrha stoga
- na stogu su: Zi_1, Zi_2,..,Zi_m i njihova vrijednost je j
  - nakon uzimanja Zi_m s vrha preostala vrijednost je
  - cijeli dio |j/k|
  - j/k se izračuna:
    - početak A: j B:0
    - A: j-k B: +1 (B se poveća za 1)
    - A: 0 a u cijeli dio B: j/k

Stroj s dva brojila

- rad četiri brojila simulira s pomoću dva brojila
- i, j, k, l vrijednost 4 brojila u jedno brojilo se može spremiti vrijednost $n=2^i3^j5^k7^l$
  - 2,3,5,7 su prim brojevi => n jednoznačno određen
- osnovne operacije:
  - povećanje/smanjivanje brojila za 1: i, j, k, l se povećaju za 1
    - vrijednost brojila se množi s 2,3,5,ili 7
  - određivanje da li je vrijednost brojila = 0
    - brojilo se dijeli s 2,3,5,7
    - s kojim brojem nije djeljivo odgovarajuće brojilo =0 ???

Generiranje jezika kanonskim slijedom

- kraći nizovi su ispred duljih nizova
- redosljed nizova jednake duljine odredi se pomoću numeričke vrijednosti
  - baza za izračun je kardinalni broj skupa ulaznih znakova
  - binarna abeceda $\Sigma=\{0,1\}$ baza je 2 a kanonski slijed je : ε, 0,1,00,01,10,11,000,001, ..111, 0000,..
- rekurzivne jezike moguće je generirati kanonskim slijedom

Istovjetnost rekurzivnih jezika i jezika generiranih kanonskim slijedom

- jezik L(M2) je rekurzivan (znači da ga je moguće generirati jednostavnim TSom)
  - izlazni nizovi se generiraju na izlaznoj traci istim redoslijedom kao i na radnoj traci
  - ako se generiraju kanonskim slijedom na radnu trsku idu kanonskim slijedom i na izlaznu traku
- Jezik G(M1) koji je moguće generirati kanonskim slijedom je rekurzivan jezik

Formalni jezici i jezični procesori I
2017./2018.

SVOJSTVA REKURZIVNIH I REKURZIVNO PREBROJIVIH JEZIKA

dr. sc. Sanda Martinčić - Ipšić

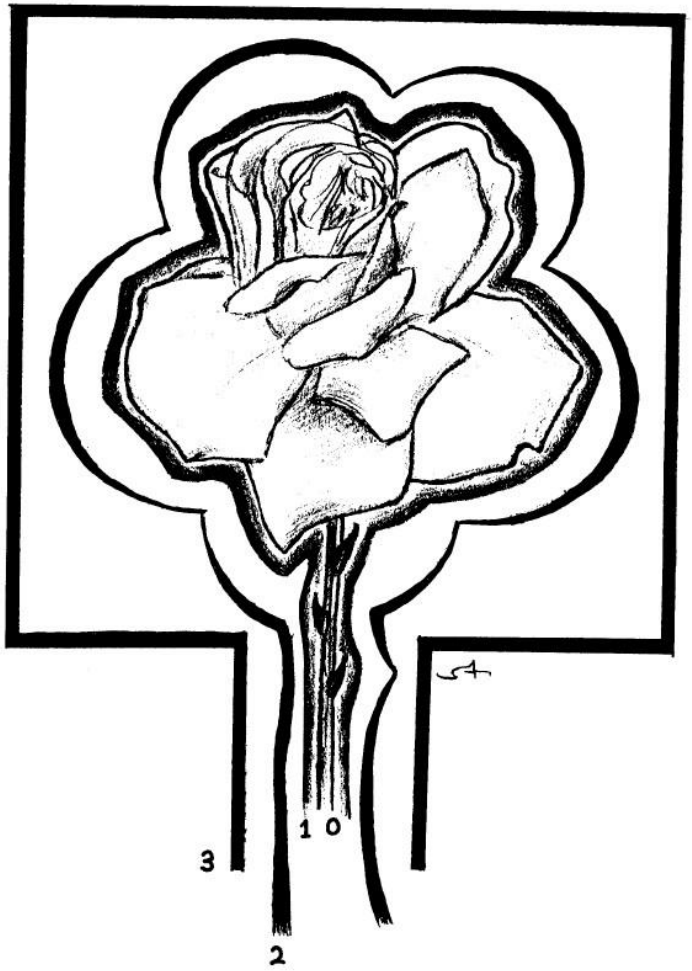
smarti@inf.uniri.hr

Gramatika neograničenih produkcija (GNP) I

- produkcije **regularne gramatike** su ograničene
  - **lijevo linearne ili desno linearne**
- produkcije **kontekstno neovisne gramatike** imaju ograničene oblike
  - **na lijevoj strani jedan znak**
- produkcije **kontekstno ovisne gramatike**
  - **na lijevoj strani manje ili jednako znakova desnoj**
- gramatika **neograničenih produkcija** (gramatika tipa 0) **nema ograničenja**, produkcije oblika
  - $\alpha \rightarrow \beta$, **α i β su nizovi završnih i nezavršnih nizova gramatike,** α ne smije biti prazan

Ograničenje gramatika

- type0: gramatike neograničenih produkcija
- type1: kontekstno ovisne gramatike
- type2: kontekstno neovisne gramatike
- type3: regularne gramatike



2.29 *The silhouette of a rose, approximated by Type 3 to Type 0 grammars*

Gramatika neograničenih produkcija II

- gramatika neograničenih produkcija G=(V, T, P, S)
  - za produkciju $\alpha \rightarrow \beta$ je definirana relacija =>: $\gamma\alpha\delta \rightarrow \gamma\beta\delta$
  - =>* je refleksivno tranzitivno okruženje relacije =>
  - gramatika G generira jezik L(G)
    - $L(G)=\{w \mid w \in T^* \text{ i } S=>^*w\}$

- GNP generira klasu rekurzivno prebrojivih jezika

Primjer: gramatike neograničenih produkcija I

- GNP G=({A,B,C,D,E},{a}, P, S) generira jezik L(G)

  1) $S \rightarrow ACaB$
  2) $Ca \rightarrow aaC$
  3) $CB \rightarrow DB$
  4) $CB \rightarrow E$
  5) $aD \rightarrow Da$
  6) $AD \rightarrow AC$
  7) $aE \rightarrow Ea$
  8) $AE \rightarrow \varepsilon$

- $L(G)=\{a^i \mid i=2^k\ k$ je cijeli broji $k>0\}$
- primjeri generiranja niza aa i aaaa

Primjer: gramatike neograničenih produkcija II

- **generiranje niza aa**

S => ACaB	primjenom S→ ACaB
=> AaaCB	primjenom Ca → aaC
=> AaaE	primjenom CB → E
=> AaEa	primjenom aE → Ea
=> AEaa	primjenom aE → Ea
=> aa	primjenom AE → ε

Primjer: gramatike neograničenih produkcija III

- **generiranje niza aaaa**

S => ACaB	primjenom S→ ACaB
=> AaaCB	primjenom Ca → aaC
=> AaaDB	primjenom CB → DB
=> AaDaB	primjenom aD → Da
=> ADaaB	primjenom aD → Da
=> ACaaB	primjenom AD → AC
=> AaaCaB	primjenom Ca → aaC

Primjer: gramatike neograničenih produkcija IV

=> AaaaaCB	primjenom $Ca \rightarrow aaC$
=> AaaaaE	primjenom $CB \rightarrow E$
=> AaaaEa	primjenom $aE \rightarrow Ea$
=> AaaEaa	primjenom $aE \rightarrow Ea$
=> AaEaaa	primjenom $aE \rightarrow Ea$
=> AEaaaa	primjenom $AE \rightarrow \varepsilon$
=> aaaa	

Primjer: gramatike neograničenih produkcija V

- nezavršni znakovi A i B su graničnici
- znak C pomakom u desno udvostručuje broj završnih znakova a
- znak D pomakom u lijevo omogućava ponavljanje udvostručavanja znaka a
- znak E pomakom u lijevo zaustavlja daljnje udvostručavanje
- ..

Konstrukcija TS za jezik zadan gramatikom neograničenih produkcija I

- Ako GNP G=(V, T, P, S) generira jezik L(G) onda je L(G) rekurzivno prebrojiv jezik
  - jezik L(G) je rekurzivno prebrojiv ako postoji TS M koji ga prihvaća L(M)=L(G)
- gradi se nedeterministički TS M s dvije trake koji simulira rad gramatike G:
  - na prvu traku se zapiše ulazni niz w
  - na drugu traku početni nezavršni znak gramatike
  - na drugu traku se tijekom rada zapisuju nizovi α koje generira gramatike
  - TS usporedi α i w na dvije trake (ako su isti prijeđe u prihvatljivo stanje i prihvati)

Konstrukcija TS za jezik zadan gramatikom neograničenih produkcija II

- TS nedeterministički izabere mjesto **i** u nizu **α** s druge trake $1 \leq i \leq |\alpha|$
  - nedeterminizam uzrokuje više mogućih simulacija za sve vrijednost i na tom mjestu u produkciji
- TS nedeterministički izabere produkciju $\beta \rightarrow \gamma$ iz G
  - ako je više produkcija $\beta \rightarrow \gamma$ postupak se izvede za sve (nedeterminizam)
  - ako je β na mjestu i onda se β zamijeni s γ
  - niz generiran na drugoj usporedi se s w na prvoj traci
    - ako su jednaki TS zaustavi rad i prihvati niz w
    - ako nisu ponovi od prve točke

Konstrukcija gramatike za jezik zadan TS I

- ako TS M prihvaća rekurzivno prebrojiv jezik L(M) onda postoji gramatika G neograničenih produkcija koja generira jezik L(G)=L(M)
- gradi se gramatika G=(V, T, P, S) koja simulira TS M=(Q, Σ, Γ, δ, q_0, B, F):
  - gramatika G generira međunizove koji su konfiguracija TS
  - nezavršni znak q u međunizu predstavlja oznaku stanja TS
  - početna konfiguracija je međuniz a nezavršnim znakovima oblika [ai,ai] gdje je ai ulazni znak iz Σ
  - prva komponenta nezavršnog znaka [ai,ai] čuva znakove a1a2a3,..an dok druga predstavlja znakove trake

Konstrukcija gramatike za jezik zadan TS II

- na temelju znakova prve komponente i nezavršnog znaka gramatika G generira niz završnih znakova a1a2a3..an ako i samo ako TS prihvaća taj isti niz znakova
- početna konfiguracija: q0[a1,a1] [a2,a2].. [an,an]
- na kraju je m oznaka praznih ćelija: $[\varepsilon, B]^m$
  - tijekom simulacije nezavršni znakovi q0[a1,a1] [a2,a2].. [an,an] $[\varepsilon, B]^m$
- tijekom simulacije nezavršni znakovi poprimaju oblik[b,X] ulazni znak $b \in \Sigma$ a X znak trake $X \in \Gamma$

Konstrukcija gramatike za jezik zadan TS III

- konfiguracija $X_1X_2..X_{r-1}qX_rX_{r+1}..X_s$ predstavlja se

$[a_1,X_1]\ [a_2,X_2]..\ [a_{r-1},X_{r-1}]\ q\ [a_r,X_r]..\ [a_s,X_s][a_{s+1},X_{s+1}]..\ [a_{n+m}X_{n+m}]$

$X_{s+1}..X_{n+m}$ su oznake praznih ćelija

- ako je nezavršni znak q u generiranom međunizu prihvatljivo stanje TS prihvaća w i prihvaća niz a1a2a3..an

Konstrukcija gramatike za jezik zadan TS IV

- produkcije se dijele u 4 skupine
  - produkcije koje generiraju početnu konfiguraciju
  - produkcije koje dodaju potreban broj praznih ćelija
  - produkcije koje simuliraju rad TS-a
  - produkcije prihvatljivih stanja

Svojstva rekurzivnih i rekurzivno prebrojivih jezika

- zatvorenost rekurzivnih jezika
  - unija i komplement
- zatvorenost rekurzivnih prebrojivih jezika
  - unija
- izračunljivost
  - Church-Turingova hipoteza
- odlučivost
  - univerzalni TS

Zatvorenost

- zatvorenost rekurzivnih i rekurzivno prebrojivih jezike se dokazuje TSom
  - TS mora uvijek stati, za bilo koji ulazni niz
- zatvorenost rekurzivnih jezika
  - unija i komplement
- zatvorenost rekurzivno prebrojivih jezika
  - unija

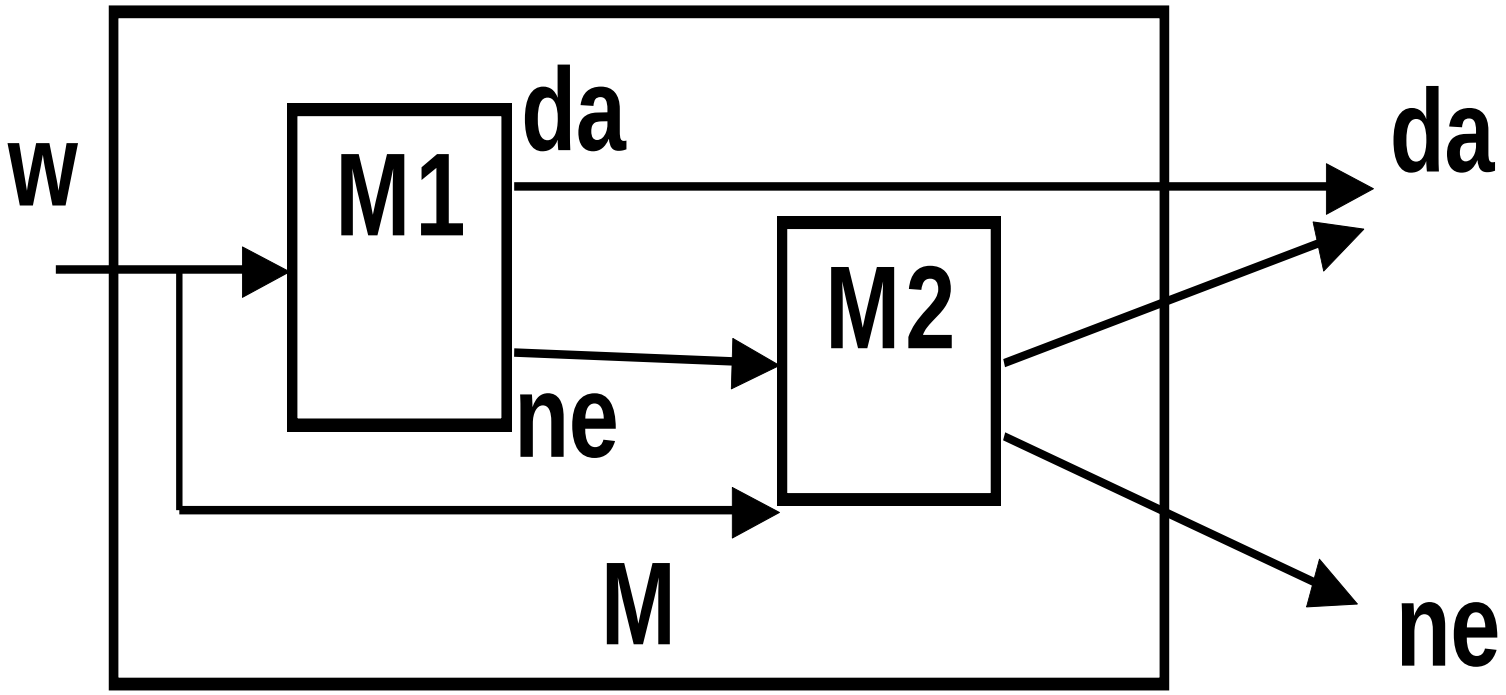
Zatvorenost II

- pokazano je svojstvo komplementa rekurzivno prebrojivog jezika
  - ako je komplement rekurzivno prebrojivog jezika L rekurzivno prebrojiv onda su jezik L i njegov komplement rekurzivni
  - ako komplement rekurzivno prebrojivog jezika L nije rekurzivno prebrojiv onda jezik L sigurno nije rekurzivan a moguće nije ni rekurzivno prebrojiv

Zatvorenost - UNIJA

- unija **rekurzivnih jezika** je rekurzivni jezik
  - konstrukcija TS-a M koji prihvaća uniju rekurzivnih jezika
  - radi se **slijedna** simulacija M, M1 i M2
  - M1 i M2 prihvaćaju L1 i L2 i uvijek stanu
  - ako jedan prihvati niz: niz se prihvaća, inače se ne prihvaća

Zatvorenost – UNIJA II

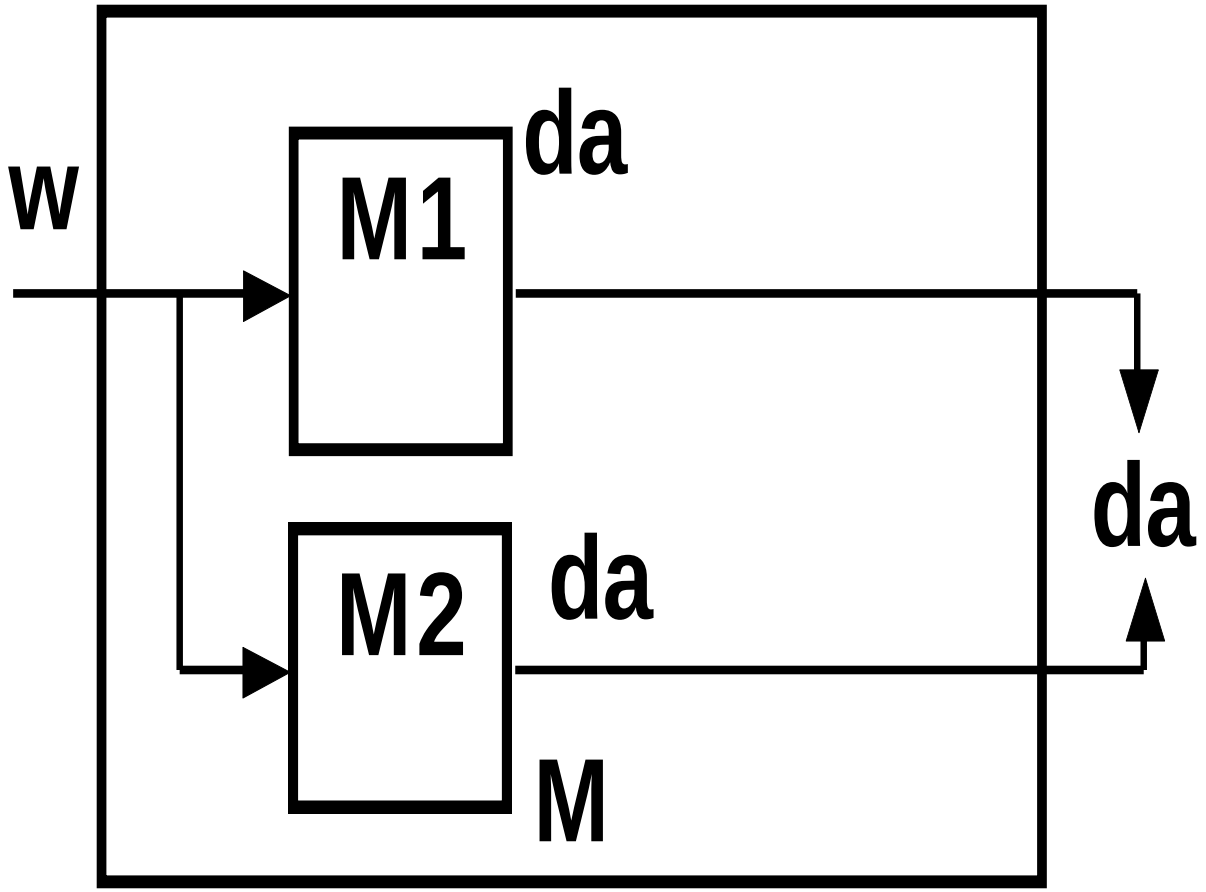


Zatvorenost – UNIJA III

- unija **rekurzivno prebrojivih jezika** je rekurzivno prebrojiv jezik
  - konstrukcija TS-a M koji prihvaća uniju rekurzivnih jezika
  - radi se **paralelna** simulacija M1 i M2
  - M1 i M2 prihvaćaju L1 i L2 i uvijek stanu
  - ako jedan prihvati niz: niz se prihvaća, inače se ne prihvaća

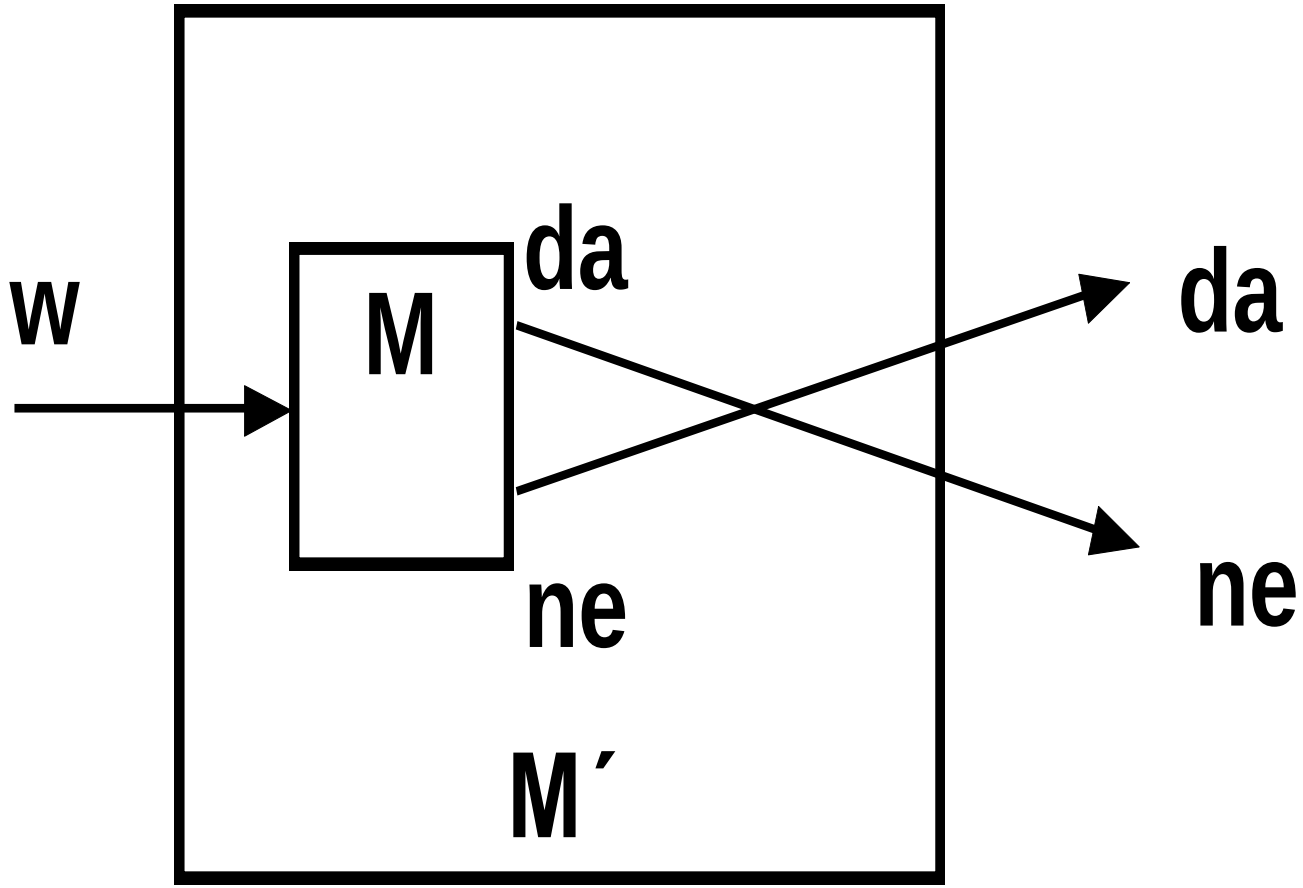
Zatvorenost – UNIJA IV

- paralelna simulacija M1 i M2



Zatvorenost – KOMPLEMENT I

- komplement rekurzivnog jezika je rekurzivan jezik



Algoritam

- skup jednostavnih naredbi za izvršenje zadatka (procedura, recept,..)
- točna definicija tek u 20.st.
- matematičar Hilbert 1900. identificirao 23 problema
  - 10. problem se odnosi na algoritme
  - *Find an algorithm to determine whether a given polynomial with integer coefficients has an integer solution*
  - najprije definirati algoritam a zatim dokazati da ne postoji rješenje

Algoritam II

- definiranje algoritma 1936.
  - **A. Church**: λ-calculus za definiranje algoritma
  - **A. Turing**: Turing Machines za definiranje algoritma
- 1970. dokaz da ne postoji rješenje 10 problema
  - **Matiyasevich's** theorem Every recursively enumerable set is Diophantine

Izračunljivost

- intuitivana (“meka”) definicija:
  - problem je izračunljiv ako postoji automat koji postupkom korak po korak (mehaničkim) rješava zadani problem
  - nema ograničenja:
    - broja koraka
    - veličine spremnika
    - postupak se ne mora zaustaviti
  - potrebno samo raščlaniti rješavanje na slijed postepenih koraka

Church - Turingova hipoteza

- izračunljive funkcije se poistovjećuju s klasom parcijalno rekurzivnih funkcija
  - točnost hipoteze se ne dokazuje jer nema formalne definicije
  - pokazuje se prikladnost i razumnost hipoteze
    - parcijalno rekurzivne funkcije su izračunljive jer je za njih moguće izgraditi TS
    - TS funkcije računa “mehaničkim” putem: korak po korak
    - TS ne mora stati za svaki ulazni niz

Odlučivost

- rekurzivni jezici su odlučivi
  - jer ih prihvaćaju TS koji **uvijek stanu** i **odluče** o prihvaćanju ili neprihvaćanju niza
- rekurzivno prebrojivi jezici nisu odlučivi
  - ne postoji TS koji uvijek stane
  - ako niz nije u jeziku TS neće stati
  - ako **ne stane** ne možemo odrediti da li ga prihvaćamo ili ne

Odlučivost i izračunljivost

- rekurzivni jezici su **izračunljivi i odlučivi**
- rekurzivno prebrojivi jezici su **izračunljivi** ali **nisu odlučivi**
  - univerzalni jezik je primjer
  - str. 163.

LITERATURA

- S. Srbljić: *Jezični procesori I + II*, Element, Zagreb, 2002.
- J.E. Hopcroft, J.D. Ullman: *Introduction to Automata Theory, Languages and Computation*, Addison-Wesley, USA,1979.
- A.V. Aho, R. Sethi, J.D. Ullman: *Compilers Principles, Techniques and Tools*, 1987.
- Michael Sipser, *Introduction to the Theory of Computation, second edition*, Course Technology, MIT, 2005.

Formalni jezici i jezični procesori I

SVOJSTVA REKURZIVNIH I REKURZIVNO PREBROJIVIH JEZIKA

prof. dr. sc. Sanda Martinčić - Ipšić

smarti@inf.uniri.hr