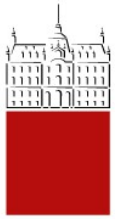
Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*



Uporaba NKA

15.marec 2014

Hitreje, višje, močneje

- Računalniki niso neskončno hitri
- Naivni pristopi pogosto dajo slabe (počasne) rešitve
- Ena najbolj ključnih nalog računalništva je iskanje hitrejših načinov reševanja
- S pomočjo NKA bomo razvili zelo hitro rešitev za iskanje vzorcev v nizih

Iskanje vzorcev

- Poiskati želimo podniz v dolgem nizu
- Npr. v genetiki iščemo ali se podano zaporedje pojavi v človeškem genomu

Primer

CGTCGACCGATCAGGATGAAAATTAGGGGCCCGAGAT
GATCAGGA

Naivno iskanje

- Hkrati primerjamo en znak v vzorcu in en znak v ciljnem nizu:
  - če se ujemata se v obeh premaknemo za en znak naprej,
  - če se ne ujemata se v vzorcu in ciljnem nizu pomaknemo nazaj za isto število znakov.

Manjši primer

Primer

AAAAAAAAAAAAAAAAAAAAAAAAAAAAAB

AAAAAAAAAB
AAAAAAAAAB
AAAAAAAAAB
.
.
.

Velikost vzorca: 10
Velikost ciljnega niza: 31
Koliko primerjav?

Primerjav:

$\approx 30 * 10 = 300$

Velik realni primer

Primer

Človeški genom: 3.2 milijarde baznih parov
Vzorec: milijon baznih parov

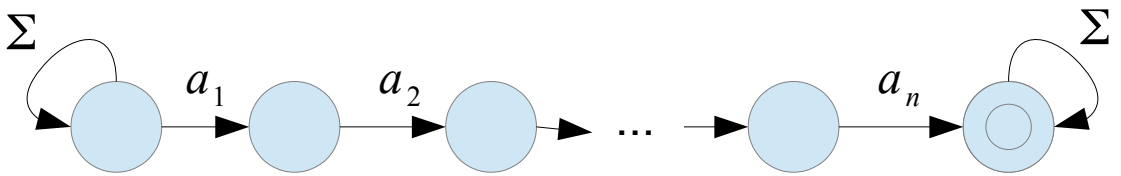
Št. primerjav:

$$\approx 10^6 * 3.2 * 10^9 = 3.2 * 10^{15}$$

Če ena primerjava traja 1ns, koliko časa bi trajalo tako iskanje?

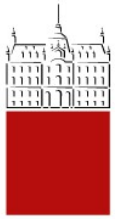
Vzorec zakodiran kot končni avtomat

$$w = a_1 a_2 a_3 \ldots a_n$$



Problem: simulacija NKA je zamudna, hitro razpoznavamo lahko zgolj z DKA

Univerza *v Ljubljani*
Fakulteta *za računalništvo*
in informatiko



Pretvorba NKA v DKA

15.marec 2014

Ekvivalenca DKA in NKA

- Pokazali bomo, da sta ta dva modela računanja povsem enakovredna:
  - Najprej bomo pokazali, da lahko vsak DKA pretvorimo v NKA
  - Nato pa še, da lahko vsak NKA pretvorimo v DKA

Pretvorba DKA v NKA

- Vsak DKA lahko zelo enostavno pretvorimo v NKA tako:
  - Vsa stanja ostanejo enaka, abeceda ostane enaka, začetno in končna stanja ostanejo enaka, malenkostno se spremeni zgolj funkcija prehodov.

Pretvorba NKA v NKA

- Pri simulaciji smo videli, da je avtomat po prebranem znaku v hkrati v več stanjih.
- Vsa ta stanja bomo združili in jih obravnavali kot eno stanje.
- Možna stanja so torej vse možne podmnožice množice stanj NKA-ja.
- Za vsa ta stanja (podmnožico) bomo poiskali v katero pomnožico pridemo po posameznih simbolih.

Formalna pretvorba

NKA

$$M = (Q, \Sigma, \delta, q_{0,} F)$$

DKA

$$M = (2^Q, \Sigma, \delta', \{q_0\}, F')$$

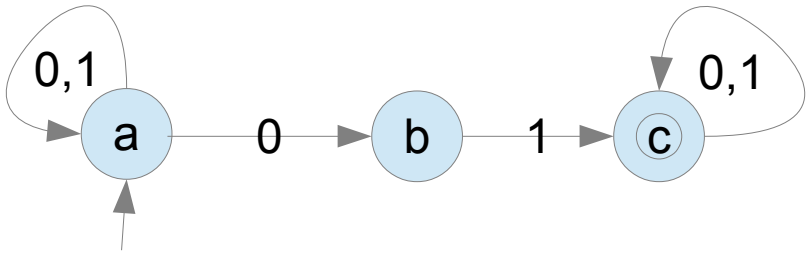
Za vsako podmnožico **P** izračunamo množico stanj v katero pridemo preko simbola **a**.

$$\delta'(P, a) = \bigcup_{q \in P} \delta(q, a)$$

Končna stanja so množice, ki vsebujejo vsaj eno stanje iz končne množice DKA.

$$F' = \{q \in 2^Q \mid q \cap F \neq \emptyset\}$$

Primer



Množica stanj: $\emptyset, \{a\}, \{b\}, \{c\}, \{a,b\}, \{a,c\}, \{b,c\}, \{a,b,c\}$

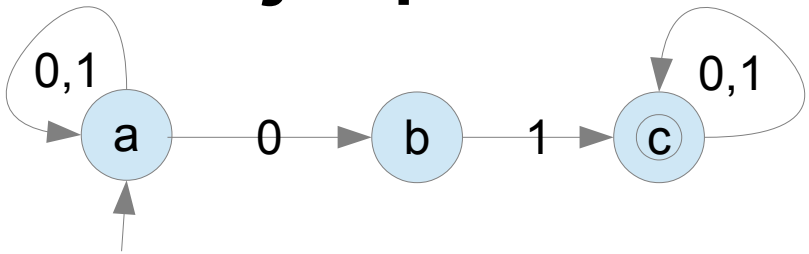
Končna stanja: $\{c\}, \{a,c\}, \{b,c\}, \{a,b,c\}$

Začetno stanje: $\{a\}$

Pretvorba s tabeliranjem podmnožic

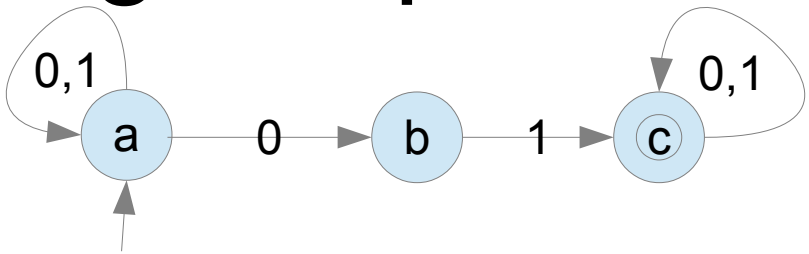
- Pri simulaciji smo videli, da je avtomat po prebranem znaku v hkrati v več stanjih.
- Vsa ta stanja bomo združili in jih obravnavali kot eno stanje.
- Možna stanja so torej vse možne podmnožice množice stanj NKA-ja.
- Za vsa ta stanja bomo tabelirali v katero pomnožico pridemo po posameznih simbolih.

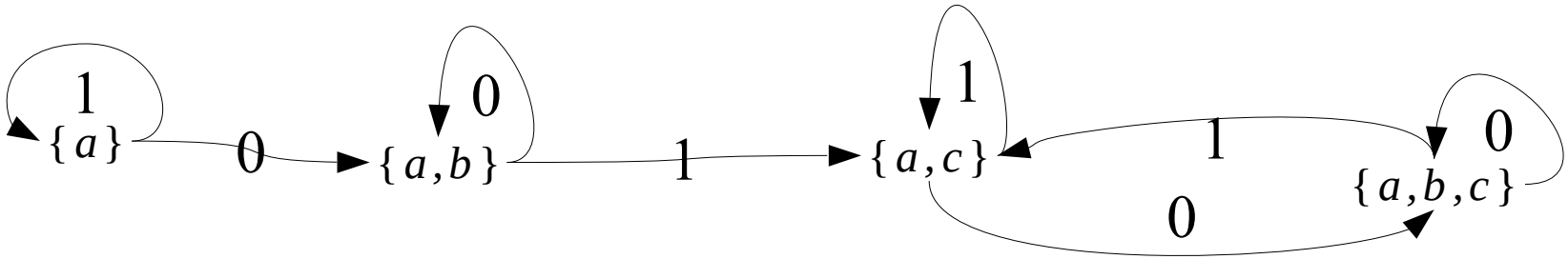
Funkcija prehodov

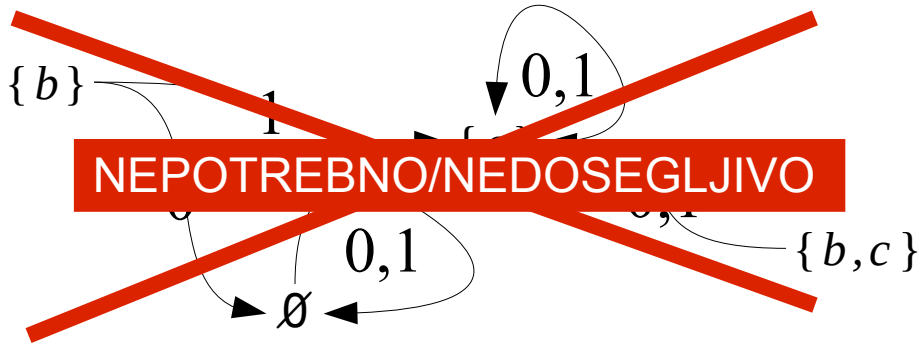


Podmožica	0	1
$\emptyset$	$\emptyset$	$\emptyset$
$\{a\}$	$\{a,b\}$	$\{a\}$
$\{b\}$	$\emptyset$	$\{c\}$
$\{c\}$	$\{c\}$	$\{c\}$
$\{a,b\}$	$\{a,b\}$	$\{a,c\}$
$\{a,c\}$	$\{a,b,c\}$	$\{a,c\}$
$\{b,c\}$	$\{c\}$	$\{c\}$
$\{a,b,c\}$	$\{a,b,c\}$	$\{a,c\}$

Diagram prehodov

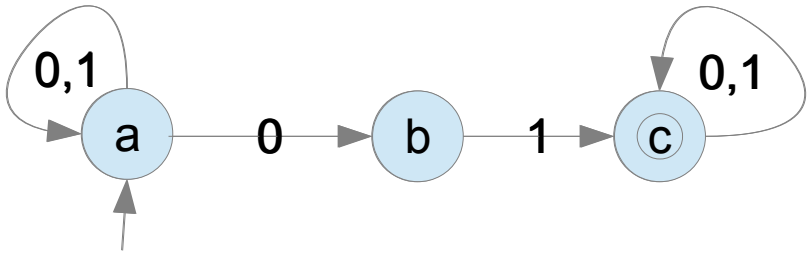




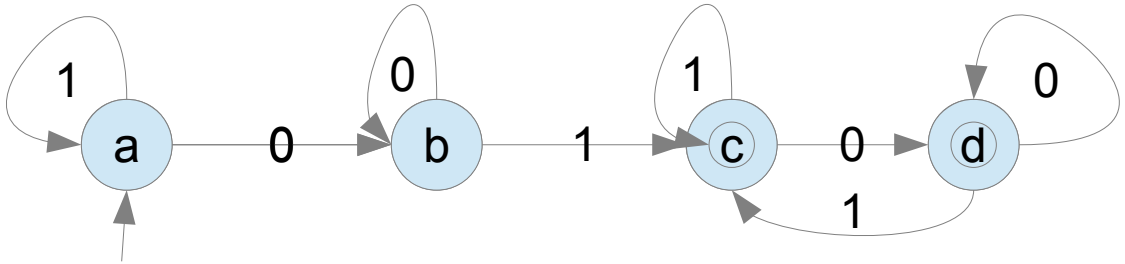


Končni rezultat

NKA:



DKA:



Pretvorba z razširjanjem stanj

- Videli smo, da so številna stanja nedosegljiva iz začetnega stanja.
- Zato je nekoristno, če izračunamo prehode za vse mogoče podmnožice, ampak samo za dosegljive.
- Začnemo v začetnem stanju, in za vse simbole izračunamo v katere množice nas pripeljejo.
- Postopek ponavljamo, dokler ni več množic, ki jih še nismo obdelali.

Primer