

Algoritmi in podatkovne strukture 1

Visokošolski strokovni študij Računalništvo in informatika



**Analiza
algoritmov**



Analiza algoritmov



*Kolikokrat je
potrebno obrniti
pogonsko ročico,
da dobimo
rezultat?*

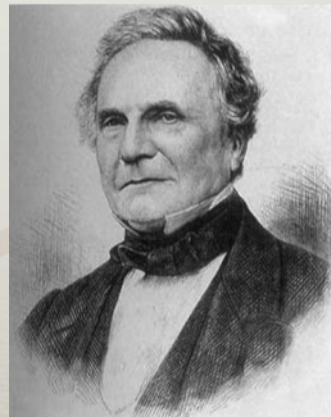


Analiza algoritmov

- Temeljno področje algoritmike
 - proučuje porabo virov algoritmov

*As soon as an Analytic Engine exists,
it will necessarily guide the future course of the science.
Whenever any result is sought by its aid, the question will arise - By what
course of calculation can these results be arrived at
by the machine in the **shortest time**?*

Charles Babbage



1791 – 1871

Viri algoritma

- **Katere vire** potrebuje algoritem za svoje izvajanje?
- **Viri:**
 - **čas:** realni čas, št. korakov, št. operacij, št. dostopov do pomnilnika
 - **prostor:** poraba pomnilnika, diska
 - **energija:** poraba električne energije
 - **komunikacija:** pasovna širina, št. paketov



Viri algoritma

- **Koliko vira** potrebuje algoritem za svoje izvajanje?
 - koliko realnega časa
 - koliko operacij
 - koliko pomnilnika
 - koliko električne energije



Zahtevnost algoritma

**Katere in koliko virov
potrebuje algoritem za svoje izvajanje
v nekem *modelu računanja*?**

Zahtevnost algoritma

- Od česa je odvisna zahtevnost algoritma?
 - od algoritma oz. implementacije algoritma
 - od modela računanja oz. računalnika
 - od naloge, ki jo podamo algoritmu
 - **velikost** naloge
 - **podatki** v nalogi

Zahtevnost algoritma

- Odvisnost zahtevnosti od **velikosti naloge**
 - množenje: $2*3$ vs $1234*5678$
 - urejanje: $2\ 1\ 3$ vs $3\ 1\ 4\ 2\ 5\ 9\ 6\ 0\ 7\ 8$
- Zapis v obliki funkcije
 - velikost naloge označimo z n
 - časovna zahtevnost $T(n)$
 - primerjave: $T(n) = n-1$
 - zamenjave: $T(n) \leq n$
 - globina rekurzije: $T(n) = \log n$
 - prostorska zahtevnost $S(n)$
 - št. pomnilniških celic: $S(n) = 2n + 3$

Zahtevnost algoritma

- Odvisnost zahtevnosti od **podatkov v nalogi**
 - množenje: $1234 * 1000$ vs $1234 * 5678$
 - urejanje: $0\ 1\ 2\ 3\ 4\ 5\ 6\ 7\ 8\ 9$ vs $3\ 1\ 4\ 2\ 5\ 9\ 6\ 0\ 7\ 8$
 - opazujemo zahtevnost pri vseh možnih nalogah enake velikosti n
- Govorimo o zahtevnosti
 - v najboljšem primeru (*best case*): $T_{BEST}(n)$
 - **v najslabšem primeru (*worst case*): $T_{WORST}(n)$**
 - v povprečju (*average*): $T_{AVG}(n)$
 - v poljubnem primeru: $T(n)$

$$T_{BEST}(n) \leq T(n) \leq T_{WORST}(n)$$

Zahtevnost algoritma

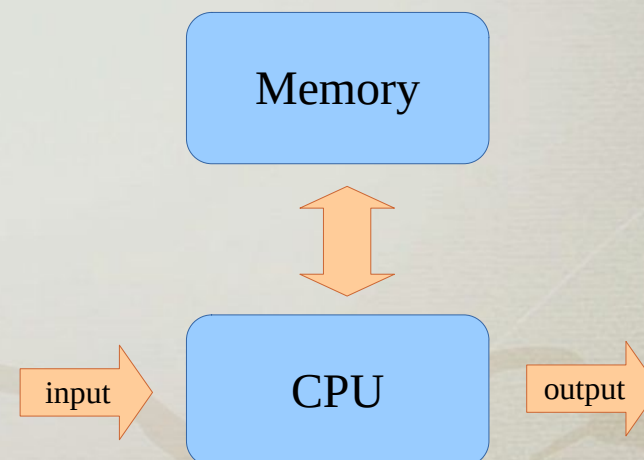
- Zakaj najpogosteje uporabljamo zahtevnost v **najslabšem primeru**?
 - podaja največjo možno porabo vira za izvedbo algoritma na katerikoli nalogi
 - za veliko algoritmov je najslabši primer zelo pogost
 - npr. iskanje elementa, ko elementa ni v seznamu
 - zahtevnost v povprečju je pogosto (asimptotično) enaka zahtevnosti v najslabšem primeru.
 - zahtevnost v povprečju je pogosto težko analizirati



1903 – 1957

Model računanja

- Von Neumannov model
 - računalniška arhitektura
 - CPU
 - aritmetično logična enota, kontrolna enota
 - registri (ukazni register, programski števec)
 - pomnilnik
 - vsebuje **podatke** in **ukaze**
 - Von Neumannovo ozko grlo
 - branje ukazov in podatkov





Model računanja

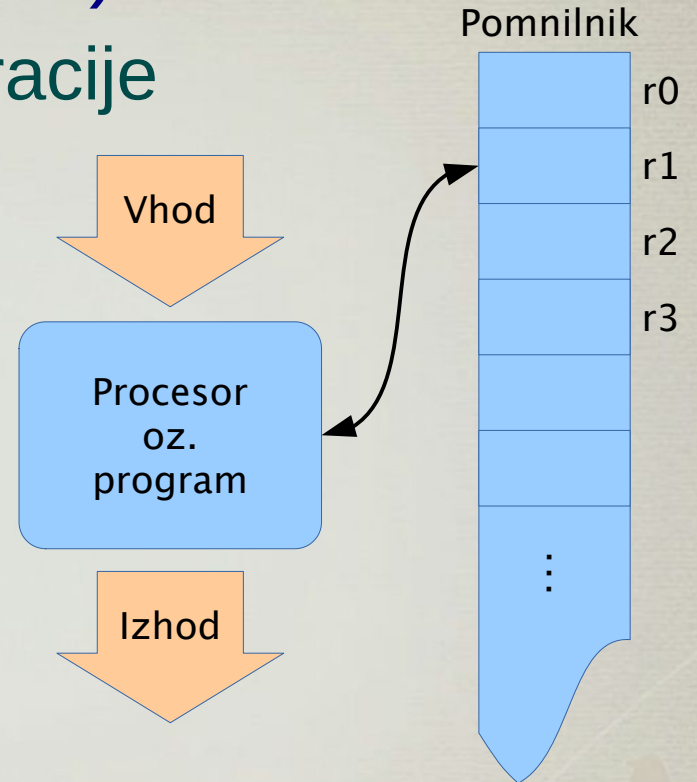
- Model računanja (*model of computation*)
 - množica dovoljenih **operacij**
 - realnost operacij
 - kompleksnost operacij
 - vsaka operacija ima neko **ceno**
 - cena ene izvedbe
 - cene so lahko različne
 - **enostavnost** in **realnost** modela
 - uporabnost





Model računanja

- RAM (*Random Access Machine*)
 - zaporedno izvaja običajne operacije
 - program je zapečen v procesor
 - ocena zahtevnosti
 - (solidna) ocena časa
 - (dobra) ocena prostora
 - RAM kot ciljni stroj
 - Algoritme pišemo v višjem programskem jeziku
- RAM pa si predstavljamo kot ciljni stroj.





Model računanja

- Veliko vrst modelov
 - avtomati, Turingovi stroji,
 - stroji s števcem, kazalcem,
 - RAM, PRAM, RASP,
 - programski jeziki, MMIX,
 - programi brez zank
 - bitni izračun (logična vezja)
 - odločitveno drevo
 - itd.

