

Programiranje 1

2. predavanje — 1. dodatak

Saša Singer

`singer@math.hr`

`web.math.pmf.unizg.hr/~singer`

PMF – Matematički odsjek, Zagreb

Sadržaj predavanja

- “Matematički” model računala — Turingov stroj:
 - Ideja i važnost.
- Glavni dijelovi Turingovog stroja:
 - Traka.
 - Glava.
 - Stanja stroja.
 - Program.

“Matematički” model računala Turingov stroj

Sadržaj

- “Matematički” model računala — Turingov stroj:
 - Ideja i važnost.
- Glavni dijelovi Turingovog stroja:
 - Traka.
 - Glava.
 - Stanja stroja.
 - Program.

Uvod — modeli računala

Već smo ukratko opisali **von Neumannov model računala**.
Osnovna stvar u tom modelu:

- **podaci i instrukcije (algoritam)** spremljeni su u **istoj memoriji**.

Prednost: **efikasna** podloga za realizaciju računala **opće** namjene (izvršavanje raznih algoritama).

Dodatno, opisali smo i kako izgledaju osnovni dijelovi takvog računala — **memorija** i **procesor** (to je **osnova** za **pisanje algoritama** u nekom programskom jeziku).

Pored toga, zanimljivo je pogledati kako izgleda

- **matematički model** računala, tzv. **Turingov stroj**.

(Poslije se nećemo vraćati na to.)

Turingov stroj

Turingov stroj je

- matematički (apstraktni) stroj za izvršavanje algoritma.

Važnost u matematici — tzv. Churchova teza:

- sve što se uopće “može algoritamski izračunati” (bilo u matematici, bilo u praksi), može se realizirati, odnosno, izračunati i **Turingovim strojem**.

Drugim riječima:

- Turingov stroj je **univerzalni model** računala
- i “**nema jačeg**” stroja.

Opravdanje: ima i drugih “modela računanja”, ali je za sve njih (bar zasad) **dokazano** da su međusobno **ekvivalentni**.

Turingov stroj — malo povijesti

Turingov stroj je, kao ideja, stariji od von Neumannovoga.

- Nastao je krajem dvadesetih i početkom tridesetih godina prošlog stoljeća.

Autor je Alan Turing, logičar, koji nije bio samo teoretičar,

- već je projektirao i specijalna računala koja su Britanci u Bletchley Parku koristili za razbijanje šifri njemačkih strojeva za šifriranje (Enigma).

Zato Turinga smatraju “ocem računarstva”.

Jedna od najvećih nagrada u računarstvu je Turingova nagrada. Dodjeljuje ju ACM.

- Dobitnik 2005. g.: Peter Naur, autor jezika Algol 60.

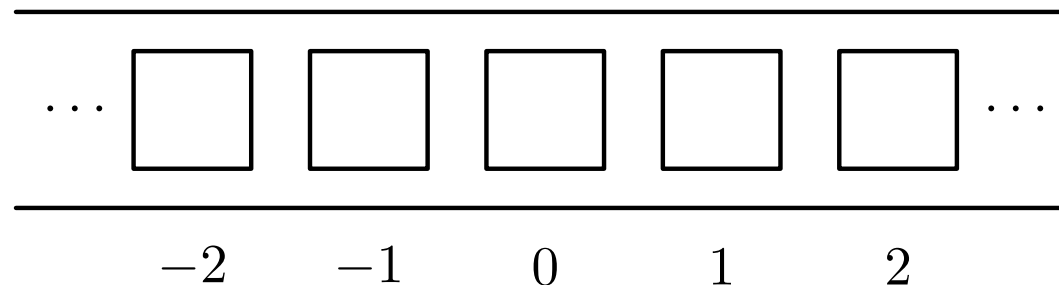
Turingov stroj — traka

Kako izgleda Turingov stroj?

- On se sastoji od nekoliko bitnih dijelova.

1. Traka

Turingov stroj ima dvostrano **beskonačnu traku** koja sadrži **polja** numerirana cijelim brojevima:



Svako polje (“kvadratić” na slici) može sadržavati **jedan** znak iz nekog **skupa znakova** kojeg stroj “prepoznaje”, tj. zna “pročitati” i “napisati” znakove iz tog skupa.

Turingov stroj — traka (nastavak)

Jednostavnosti radi, uzmimo da se taj skup znakova sastoji iz samo 3 simbola: '0', '1' i '□' (“praznina”).

- Prva 2 znaka su binarne znamenke i služe za zapis “korisnih” informacija na traci, a praznina služi kao oznaka kraja zapisa podataka.
- Uočiti: praznina osigurava konačnost zapisa na traci, tj. konačnost “ulaznog” niza binarnih znamenki!

Mogli smo uzeti i veći skup znakova, ali manji ne smijemo!

Razlog: kôdiranje ulaza mora biti razumno kratko (sažeto), da stroj ne bi trošio previše vremena na čitanje i pisanje.

Ilustracija na sljedećem primjeru.

Primjer kôdiranja

Primjer. Kako u alfabetu od dva znaka '0', '1', kôdiramo nenegativne cijele brojeve, tj. brojeve iz skupa $\mathbb{N} \cup \{0\}$?

Ako je $n \in \mathbb{N}$, onda broj n ima pozicioni prikaz u bazi $b = 2$:

$$n = a_k \cdot 2^k + \dots + a_1 \cdot 2 + a_0, \quad a_i \in \{0, 1\}, \quad a_k > 0,$$

gdje su a_i binarne znamenke. Njih kôdiramo redom, koristeći: $0 \mapsto '0'$ i $1 \mapsto '1'$.

Prikaz broja n na traci je naprosto niz znamenki:

$$a_k, a_{k-1}, \dots, a_1, a_0.$$

Uz to, moramo se dogovoriti da zapis broja $n = 0$ ima oblik $a_0 = 0$, a duljina je jedna znamenka (kao i u običnom životu).

Primjer kôdiranja (nastavak)

Pogledajmo **koliko** nam je **znakova** z **potrebno** za kôdiranje broja n , za $n > 0$. Treba nam **točno** $z = k + 1$ znak, samo treba izraziti k u funkciji od n .

Zbog $a_k > 0$, vrijedi $n \geq 2^k$, a zbog $a_i \in \{0, 1\}$, mora biti i $n < 2^{k+1}$. Iz

$$2^k \leq n < 2^{k+1},$$

logaritmiranjem slijedi $k \leq \log_2 n < k + 1$, pa je $k = \lfloor \log_2 n \rfloor$.

Dakle, za kôdiranje nam je potrebno

$$z = \begin{cases} \lfloor \log_2 n \rfloor + 1, & n > 0, \\ 1, & n = 0, \end{cases}$$

znakova, odnosno, znamenki. ■

Primjer kôdiranja (nastavak)

Primjer. Što se događa ako uzmemo **manji** alfabet, koji se sastoji **samo** od znakova '0' i '□', uz dogovor da praznina opet služi za oznaku **kraja ulaza**?

Tada se svaki $n \in \mathbb{N}$ može zapisati kao **niz** od n znakova '0'.

● Međutim, duljina zapisa je sad **linearna** u n !

Javlja se još jedan **problem** — kako zapisati broj 0.

● Očiti zapis '0' **nije** moguće koristiti, jer smo ga “potrošili” na zapis broja 1.

Moguće rješenje: n kôdiramo **nizom** od $n + 1$ znakova '0'.

Zaključak. **Nije** dobro koristiti alfabet sa samo 2 znaka, jer je duljina zapisa **linearna**, a ne više **logaritamska** u n , što može drastično utjecati na složenost algoritama. ■

Turingov stroj — glava

2. Glava

Traka Turingovog stroja ima **glavu** koja može napraviti sljedeće operacije s trakom:

- **pročitati jedan** znak s trake — s polja koje se nalazi “ispod” glave,
- **napisati jedan** znak na traku — u polje “ispod” glave,
- **pomaknuti se**, relativno obzirom na trenutnu poziciju,
 - za **jedno** polje **nadesno** (pomak $+1$), ili
 - za **jedno** polje **nalijevo** (pomak -1).

Iz dosad rečenog, očito je da **traka** služi kao **memorija**, a **glava** je **ulazno–izlazna** jedinica i (donekle) **kontrolni** mehanizam Turingovog stroja.

Turingov stroj — stanja ili programski dio

3. Stanja ili “programski” dio

Programski dio Turingovog stroja sastoji se od **konačnog broja** tzv. **stanja** u kojima se stroj može nalaziti. U **svakom** trenutku

- stroj se nalazi u **točno jednom** od mogućih stanja.

Sva moguća **stanja** podijeljena su u **3** “vrste”:

- “**regularna**” stanja (“međustanja” ili radna stanja): zovemo ih q_1, \dots, q_s ,
- “**početno**” stanje: zovemo ga q_0 ,
- “**završno**” stanje: zovemo ga q_f .

Na **početku** rada stroj se nalazi u **početnom** stanju q_0 .

Turingov stroj — završna stanja

Stroj **završava** rad u trenutku kad stigne u **završno** stanje q_f .

Katkad se uzima da stroj ima **više** završnih stanja — koja odgovaraju **raznim** mogućim **rezultatima** algoritma.

- Na primjer, ako algoritam daje odgovor **da/ne** na neko pitanje, onda stroj može imati **2** završna stanja:
 - q_y — ako je odgovor **da** i
 - q_n — ako je odgovor **ne**.

Međutim, to nije jako bitno. **Zapisivanjem** odgovora na **traku** uvijek možemo postići da stroj ima **jedno** završno stanje.

Turingov stroj — programski modul

4. Program ili programski modul

Taj modul stvarno **upravlja** strojem i “vodi ga” kroz korake pojedinog **algoritma**.

Kako radi programski modul? Njegovo “ponašanje” ovisi o:

- trenutnom **stanju** stroja i
- znaku koji **glava** učitava s **trake**.

Tada stroj:

- **prijeđe** u neko drugo **stanje**,
- **napiše** neki **znak** na **traku** i
- **pomiče glavu** jedno mjesto **udesno** ili **ulijevo**.

Dakle, različita **stanja** stroja su neka vrsta “**programske**” **memorije** stroja.

Turingov stroj — programski modul (precizno)

Precizniji opis kako radi programski modul.

- Uzmimo da je stroj u nekom stanju q , koje nije završno, $q \neq q_f$, i da je glava u tom trenutku učitala “simbol” s trake. Na osnovu para

(q, simbol)

programski modul odlučuje sljedeće 3 stvari:

- koje je sljedeće stanje q' u koje prelazi stroj,
- koji će znak napisati u polje ispod glave (taj znak zovemo “novi simbol”),
- pomiče li se traka jedno polje nadesno, ili jedno polje nalijevo (pomak za $+1$ ili -1).

Turingov stroj — programski modul (precizno)

- Dakle, jedan **programski korak** je veza

$$(q, \text{simbol}) \longrightarrow (q', \text{novi simbol}, \text{pomak}).$$

Ako je ova veza **funkcija**, tj. svakom paru (q, simbol) pridruži **točno** jednu trojku $(q', \text{novi simbol}, \text{pomak})$, onda je stroj **deterministički**. Ali to ne mora biti tako!

- Kad stroj dođe u **završno** stanje q_f , onda se računanje **prekida** (završava).
- Na **početku** stroj je u stanju q_0 , a glava se nalazi nad poljem s brojem 1.

Ovime smo, zapravo, napravili sve elemente za **formalnu definiciju Turingovog stroja**, što je sasvim dovoljno za opis.