Milenijumski problem - P protiv NP

Anđela Milovanović

Matematiki fakultet u Beogradu

Septembar 2024

Sadraj

Teorija Izračunljivosti

- Teorija je nastala tokom tridesetih godina prolog veka, u vreme kada klasini raunari nisu postojali.
- Pioniri ove nauke bili su Alonzo er, Alan Tjuring, Emil Post, Kurt Gedel i drugi.
- Prvi radovi na teoriji izraunljivosti bili su inspirisani uvenim Gedelovim dokazima teoreme o nekompletnosti.
- U ovom periodu definisan je koncept koji lii na dananje algoritme (efektivne procedure).
- Glavno pitanje kojim se ova teorija bavi: ta se sve moe uraditi pomou raunara?
- U nedostatku pravog kompjutera, za izvoenje dokaza i demonstraciju reenja problema esto je koriena Tjuringova maina, jednostavan ureaj koji je u stanju da izvri bilo koji algoritam, ba kao i dananji raunari.

Formalni jezici

Definicija

- Azbuka Σ je bilo koji neprazan skup simbola.
- Reč nad azbukom Σ je bilo koji konačan niz simbola azbuke Σ.
- Skup Σ^* je skup svih reči nad azbukom Σ.
- Jezik nad azbukom Σ je bilo koji podskup skupa Σ^*.

Primer

- Posmatrajmo azbuku $\Sigma = \{a, b, c\}$.
- Tada je $w = abba$ reč nad ovom azbukom.
- Primeri jezika nad azbukom Σ su:

$$L_1 = \varnothing \quad L_2 = \{a, ab, bc, ca\} \quad L_3 = \{\epsilon, a, aa, aaa, \dots\} \quad L_4 = \Sigma^*.$$

Hijerarhija Čomskog

Gramatika	Tip jezika	Automat
Tip 3	Regularni jezici	Konačni automat
Tip 2	Kontekstno slobodni jezici	Potisni Automat
Tip 1	Kontekstno-osetljivi jezici	Linearno ograničeni automat
Tip 0	Rekurzivno prebrojivi jezici	Tjuringova mašina

Chomsky-hierarchy.png

Sadraj

Tjuringova mašina (neformalno)

- Maina koristi beskonanu traku podeljenu u diskretne elije.
- Svaka elija moe da sadri neki od simbola iz alfabeta koji maina koristi (slova, brojevi).
- Maina ima glavu koja je u datom trenutku pozicionirana iznad neke elije.
- Maina ima svoje stanje, jedno iz predefinisanog seta moguih stanja.
- Glava moe da proita simbol koji se nalazi na traci neposredno ispod nje.
- Na bazi internog stanja, proitanog simbola i predefinisanih pravila uskladitenih u maini:
  - Glava maine u eliju ispod nje, upisuje neki drugi simbol ili ponovo upisuje isti.
  - Maina menja svoje interno stanje ili zadrava postojee. Jedno od njih zaustavlja rad maine.
  - Maina pomera glavu levo ili desno du beskonane trake na novu eliju.
- Tipina instrukcija: Ako je maina u stanju A i ako je proitan simbol 1, u eliju upisati simbol 0, promeniti stanje maine u B i pomeriti glavu u desnu stranu.
- U narednom slajdu dajemo formalnu definiciju Tjuringove mašine (Videti [?]).

Tjuringova mašina (formalno)

Definicija

Tjuringova mašina je uređena sedmorka: $M = \langle Q, \Gamma, b, \Sigma, \delta, q_0, F \rangle$ tako da važi:

- Q je konaan, neprazan skup stanja.
- Γ je konaan, neprazan skup simbola trake.
- $b \in \Gamma$ je prazan simbol (jedini koji može da se javi na traci beskonano mnogo puta).
- $\Sigma \subseteq \Gamma \setminus \{b\}$ je skup ulaznih simbola.
- $\delta : (Q \setminus F) \times \Gamma \nrightarrow Q \times \Gamma \times \{L, R\}$ je parcijalna funkcija tranzicije, gde je L pomeraj ulevo,a R pomeraj udesno. Ako δ nije definisana u trenutnom stanju i za trenutni simbol trake, onda se mašina zaustavlja (haltuje).
- $q_0 \in Q$ je poetno stanje.
- $F \subseteq Q$ je skup krajnjih stanja. Kažemo da mašina prihvata početnu nisku trake ako se zaustavi (haltuje) u nekom stanju iz F.

Tjuringova mašina

Turing_Machine.jpg

(a) Tjuringova mašina - ilustraciona maketa

Turing Machine - graph.png

(b) Tjuringova mašina - definicija preko grafa

Tjuringova mašina - ekvivalencija sa modernim računarima

- Tjuringova mašina je u stanju da reši svaki problem koji je moguće rešiti korišćenjem klasičnih kompjutera.
- Tjuringova mašina formalizuje koncept algoritma kao i složenosti algoritma (broj koraka koji je potreban Tjuringovoj mašini da prihvati nisku).
- Pojam Turing-Complete se vezuje za programske jezike koji imaju istu ekspresivnost kao Tjuringove mašine (C, C++, Java, Python i dr.).
- Čak je i LaTeX Turing-Complete. U teoriji, svaki C program je moguće napisati i u LaTeX-u.

Sadraj

Problemi odluivosti i funkcijski problemi

- Postoje dve vrste problema koje Tjuringova maina moe da reava:
  - Problemi odluivosti: U pitanju su problemi na koje se moe odgovoriti samo sa da ili ne.
  - Funkcijski problemi: U ovim problemima Tjuringova maina treba da generie izlaz koji predstavlja rezultat primene odreene funkcije na ulazne podatke.
- Praktino svi znaajni kompjuterski problemi mogu se formulisati na oba naina: i kao problem odluivosti, i kao funkcijski problem.
- Primer problema odluivosti:
  - Da li broj 240.996.255.422.547 ima prost faktor manji od 15.000.000?
  - Da li postoji zatvorena putanja koja obilazi sve srpske gradove sa vie od 5.000 stanovnika a kraa je od 2.000 kilometara?
- Funkcionalna, uoptenija verzija prethodna dva problema (koja je obino tea za reavanje):
  - Nai proste inioce broja 240.996.255.422.547.
  - Kolika je duina najkrae zatvorene putanje koja obilazi sve evropske glavne gradove?
- **Nadalje emo se baviti iskljuivo problemima odluivosti**.

Klasa složenosti $\mathcal{P}$

- Nadalje problem odlučivosti zapravo postaje problem pripadanja niske jeziku. Jezici čije niske uvek prihavata neka Tjuringova mašina se nazivaju rekurzivno prebrojivi.
- Zapravo, u nastakvu reč problem treba poistovetiti sa rečju jezik, te ćemo sad dati definiciju $\mathcal{P}$ klase jezika.

Definicija

Ako za determinističku Tjuringovu mašinu M postoji polinom $p(n)$ tako da za svaku nisku dužine n, mašina M ne napravi više od $p(n)$ koraka, kažemo da je M **polinomska deterministička Tjuringova mašina**.
Klasa $\mathcal{P}$ je skup svih jezika za koje postoji polinomska deterministička Tjuringova mašina koja ih prihvata.

- Drugim rečima, $\mathcal{P}$ je klasa **problema** za koje postoji algoritam polinomske složenosti koji se može izvršiti na klasičnom računaru.

Klasa složenosti $\mathcal{NP}$

Definicija

Ako za nedeterminističku Tjuringovu mašinu M postoji polinom $p(n)$ tako da za svaku nisku dužine n, mašina M ne napravi više od $p(n)$ koraka, kažemo da je M **polinomska nedeterministička Tjuringova mašina**.
Klasa $\mathcal{NP}$ je skup svih jezika za koje postoji polinomska nedeterministička Tjuringova mašina koja ih prihvata.

- Kako su sve determinističke mašine podskup nedeterminističkih, to je

$$\mathcal{P} \subseteq \mathcal{NP}.$$

$\mathcal{P}$ vs. $\mathcal{NP}$

P vs NP.png

$\mathcal{NP}$-complete klasa problema

- $\mathcal{NP}$-complete nećemo definisati formalno.
- Problemi iz skupa $\mathcal{NP}$ mogu se redukovati (transformisati) jedan u drugi.
- Od znaaja su transformacije koje se mogu izvesti u polinomskom vremenu.
- Na taj nain problem A moe da se svede na ekvivalentan problem B. Ako smo u stanju da reimo problem B, to reenje moe da se brzo adaptira i iskoristi i za reavanje problema A.
- Za problem kaemo da je $\mathcal{NP}$-kompletan ($\mathcal{NP}$-Complete):
  - ako je u pitanju problem odluivosti (rezultat se svodi na da ili ne);
  - ako se reenje problema moe proveriti brzo, u polinomskom vremenu (sudoku, rubikova kocka)
  - ako problem moe da se iskoristi za reavanje bilo kog drugog problema iz skupa $\mathcal{NP}$.
- Do danas je definisano preko 1.000 problema koji spadaju u $\mathcal{NP}$-kompletan skup. Svi $\mathcal{NP}$ problemi svode se na neki od njih.
- $\mathcal{NP}$-kompletni problemi su kao domine: dovoljno je reiti samo jedan $\mathcal{NP}$-kompletan problem u polinomskom vremenu pa da svi $\mathcal{NP}$ problemi budu takoe reeni u polinomskom vremenu.

$\mathcal{NP}$-hard klasa problema

- Ni ovu klasu ne definišemo formalno.
- $\mathcal{NP}$-teki problemi u najveem broju sluajeva predstavljaju komplementarnu, funkcionalnu verziju problema odluivosti koji spadaju u $\mathcal{NP}$ ili $\mathcal{NP}$-kompletne probleme. Kaemo da ovi problemi nastaju uoptavanjem (generalizacijom tj. redukcijom) $\mathcal{NP}$ problema u polinomskom vremenu.
- $\mathcal{NP}$-teki problemi mogu biti i odluivog i funkcionalnog tipa.
- Kaemo da su ovi problemi teki za reavanje koliko i najtei problemi iz $\mathcal{NP}$ skupa. ak je i okvirna procena reenja teka ili nemogua.
- Za razliku od $\mathcal{NP}$ problema ije reenje mora biti proverljivo u polinomskom vremenu, $\mathcal{NP}$-teki problemi nemaju to ogranienje.
- U $\mathcal{NP}$-teku grupu problema spadaju i neodluivi problemi, tj. problemi za koje ne postoji algoritamsko reenje.
- Mnogi $\mathcal{NP}$-kompletni problemi imaju svoju $\mathcal{NP}$-teku verziju (problem najvee klike, problem sume podskupa, izraunavanje hromatskog broj grafa). Primere ćemo videti u nastavku.

Sadraj

Primeri nekih $\mathcal{P}$ problema

- Traženje elementa u nizu - $O(n)$.

Primeri nekih $\mathcal{P}$ problema

- Traženje elementa u nizu - $O(n)$.
- Traženje elementa u sortiranom nizu (binarna pretraga) - $O(\log n)$.

Primeri nekih $\mathcal{P}$ problema

- Traženje elementa u nizu - $O(n)$.
- Traženje elementa u sortiranom nizu (binarna pretraga) - $O(\log n)$.
- Provera da li je broj prost - $O(\sqrt{n})$.

Primeri nekih $\mathcal{P}$ problema

- Traženje elementa u nizu - $O(n)$.
- Traženje elementa u sortiranom nizu (binarna pretraga) - $O(\log n)$.
- Provera da li je broj prost - $O(\sqrt{n})$.
- Traženje medijane u nizu - $O(n \log n)$.

Primeri nekih $\mathcal{P}$ problema

- Traženje elementa u nizu - $O(n)$.
- Traženje elementa u sortiranom nizu (binarna pretraga) - $O(\log n)$.
- Provera da li je broj prost - $O(\sqrt{n})$.
- Traženje medijane u nizu - $O(n \log n)$.
- Množenje kvadratnih matrica - $O(n^3)$

Problem faktorizacije broja

- Fundamentalna teorema teorije brojeva kaže da se svaki broj može na jedistven način napisati kao proizvod prostih brojeva (faktorizovati).
- Na primer, $108 = 2^2 \cdot 3^3$.
- Za sada nije poznat nijedan algoritam koji ovaj problem rešava u polinomskom vremenu.
- Takođe nije poznato da li je ovaj problem $\mathcal{NP}$-kompletan, te se njegovim rešavanjem u polinomskom vremenu ne bi pokazalo da važi $\mathcal{P} = \mathcal{NP}$. Pretpostavlja se da ovaj problem pripada posebnoj klasi $\mathcal{NP}$-intermediate.
- Za sada, najbolji poznati algoritam ima složenost $O\left((1+\varepsilon)^b\right)$ gde je ε neka vrlo mala konstanta a b broj bitova koji se faktorizuje.
- Na ovom algoritmu se zasniva ogroman deo internet kriptografije (*RSA*-algoritam). Zasad nisu pronađene mane ovog algoritma.

SAT problem (Boolean satisfiability problem)

- **SAT** je prvi problem za koji je dokazano da je $\mathcal{NP}$-kompletan. U svom revoucionarnom radu iz 1971. godine **Stiven Kuk** i **Leonid Levin** su dokazali da se svaki $\mathcal{NP}$ problem može svesti na SAT u polinomskom vremenu (Pogledati [**?**]).
- Sat problem se odnosi na logičke izraze u kojima može figurisati proizvoljan broj *true/false* promenljivih i logičkih operatora. Na primer:

$$((x_1 \lor \neg x_2) \Rightarrow (x_3 \land \neg x_4)) \land (\neg(x_1 \land x_3) \lor x_2)$$

- **Odlučiva verzija problema:** Da li se svakoj promenjljivoj može dodeliti vrednost tako da logički izraz na kraju ima vrednost *true*? Algoritam grube sile ima složenost $O(2^n)$ gde je n broj promenjljivih u izrazu.
- Postoji i uporšćena verzija problema koja posmatra samo izraze u *KNF* čiji konjunkti imaju tri disjunkta. Ova verzija se zove **3SAT** i takođe je $\mathcal{NP}$-kompletna. Primer izraza koje posmatra 3SAT:

$$(x_1 \lor \neg x_2 \lor x_3) \land (\neg x_1 \lor x_2 \lor x_4) \land (x_3 \lor \neg x_4 \lor x_5)$$

Problem trgovačkog putnika (Travelling salesman)

Formulacija problema: Naći najkraći zatvoreni put koji kroz svako zadato mesto prolazi samo jednom.

- U svojoj odlučivoj verziji (Da li postoji zatvoreni put dužine manje od k?), problem je $\mathcal{NP}$-kompletan.
- Originalni problem je $\mathcal{NP}$-težak!.
- Najbolji poznat algoritam za rešavanje $\mathcal{NP}$-teškog problema je složenosti $O(n^2 2^n)$. Naivni algoritam grube sile (proveravanje svih permutacija) ima složenost $O(n!)$.
- Problem se sreće svuda: u planiranju, logistici, konstrukciji mikroprocesora i mnogim problemima optimizacije. Grana primenjene matematike koja se bavi pitanjima sličnim ovom problemu se naziva operaciona istraživanja.

Travelling Salesman - Wolfram.png

Sadraj

Implikacije $\mathcal{P}$ vs $\mathcal{NP}$ problema

- Postoje dva mogua dokaza za P=NP, konstruktivan i nekonstruktivan
- U sluaju konstruktivnog dokaza svet bi bio potpuno drugaije mesto.
- U sluaju nekonstruktivnog dokaza (verovatniji scenario) sve bi se odvijalo po starom
- Veina matematiara danas veruje da je P razliito od NP.
- **Donald Knuth:** My main point, however, is that I don't believe that the equality P = NP will turn out to be helpful even if it is proved, because such a proof will almost surely be nonconstructive.

`DonaldKnuth.jpg`

Slika: Enter Caption

Literatura

- J. E. Hopcroft, R. Motwani, and J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, 3rd Edition, Addison-Wesley, 2006.
- S. Cook, *The Complexity of Theorem-Proving Procedures*, Proceedings of the Third Annual ACM Symposium on Theory of Computing (STOC), 1971, pp. 151158.