**NP-polni problemi**

**3.1. Nedeterminizem**

Opozorili smo, da nedeterministični Turingov stroj (NTS) ni model nobenega realističnega računskega stroja, ampak je le miselni pripomoček za preprostejši opis algoritmov, ki vključujejo iskanje in pregledovanje.

Definirali smo *problem izpolnljivosti izjavnih izrazov v konjunktivni normalni obliki (KNO)* (angl. *SAT*) in *problem izpolnljivosti izjavnih izrazov v k-KNO* (angl. *k-SAT*).

Skicirali smo determinističen algoritem za problem SAT in pokazali, da ima časovno zahtevnost Ω(2|*I*|√), kjer je |*I*| dolžina zapisa danega izjavnega izraza *I*. Ta algoritem torej ni polinomski.

Nato smo skicirali še nedeterminističen algoritem za problem SAT in pokazali, da ima časovno zahtevnost *O*(|*I*|). Ta algoritem ima torej polinomsko časovno zahtevnost.

Pokazali smo, da lahko NTS *S* simuliramo z DTS v času *O*(*cTS*(*n*)), kjer je *c*>1 neka konstanta (odvisna od stroja *S*).

**3.2. Razreda P in NP**

Fiksirajmo abecedo Σ, kjer je |Σ|≥2.

***Definicija****.*  **P** je razred vseh jezikov nad Σ, ki jih lahko razpoznavamo z DTS v polinomskem času.

***Definicija****.*  **NP** je razred vseh jezikov nad Σ, ki jih lahko razpoznavamo z NTS v polinomskem času.

*13.11.2013*

Navedli smo še nekaj zgledov problemov iz **NP**: *Delitev, Natančno pokritje, Kromatično število, Klika, Hamiltonov cikel (HC)*.

Pokazali smo, da je razred **P** zaprt za unijo, presek in komplement jezikov.

Pokazali smo, da je razred **NP** zaprt za unijo in presek jezikov. Za komplemente problemov SAT, Delitev, Natančno pokritje, Kromatično število, Klika, HC ne vemo, ali so v **NP**.

**3.3. Polinomska prevedljivost problemov**

***Definicija*.** Jezik *J*1⊆Σ∗ je *prevedljiv* na jezik *J*2⊆Σ∗ *v polinomskem času*, če obstaja preslikava *f*:Σ∗→Σ∗, za katero velja:

1. ∀*w*∈Σ∗: (*w*∈*J*1 ⟺ *f*(*w*)∈*J*2,

2. *f* je izračunljiva z DTS v polinomskem času.

V tem primeru pišemo: *J*1→*P*  *J*2.

***Trditev*.** Naj bo *J*1→*P*  *J*2. Potem velja:

1. *J*2∈**P**  ⟹ *J*1∈**P**

2. *J*2∈**NP**  ⟹ *J*1∈**NP**

3. *J*1∉**P**  ⟹ *J*2∉**P**

4. *J*1∉**NP**  ⟹ *J*2∈**NP**

*20.11.2013*

Pokazali smo, da je relacijaprevedljivosti jezikov v polinomskem času tranzitivna.

Jezik *J*⊆Σ∗ je *netrivialen*, če *J*≠∅,Σ∗. Pokazali smo, da lahko vsak jezik iz **P** prevedemo na katerikoli netrivialen jezik v polinomskem času.

**3.4. Cook-Levinov izrek**

***Definicija.*** Jezik *J*⊆Σ∗ je **NP**-*poln*, če *J*∈**NP** in je vsak jezik iz **NP** prevedljiv na *J* v polinomskem času.

*Oznaka:* **NPC**={*J*∈Σ∗; *J* **NP**-poln}

*Trditev.* Naj bo *J*∈**NPC**. Potem velja:

*J*∈**P** ⟺ **P**=**NP**.

***Cook-Levinov izrek.*** *SAT*∈**NPC**.

***Lema.*** *Naj bo J1∈****NPC****, J2∈****NP*** *in J*1 *prevedljiv na J*2 *v polinomskem času. Potem je tudi J2∈****NPC****.*

*27.11.2013*

***Trditev.*** *3-SAT∈****NPC****.*

Definirali smo še naslednje probleme: *Vozliščno pokritje (VP), Antiklika* in *Trirazsežno prirejanje (3DM).*

***Trditev.*** *VP∈****NPC****.*

*4.12.2013*

***Izrek.*** *HC∈****NPC****.*

***Izrek.*** *3DM∈****NPC****.*

*11.12.2013*

***Trditev.*** *Delitev∈****NPC****.*

Omenili smo tri tehnike dokazovanja NP-polnosti: *Načrtovanje komponent*, *lokalno zamenjavo* in *zožitev na NP-poln podproblem*. Kot zgled zadnje smo dokazali:

***Trditev.*** *Nahrbtnik∈****NPC****.*

*18.12.2013*

Definirali smo ***psevdopolinomske algoritme*** in s pomočjo dinamičnega programiranja konstruirali psevdopolinomski algoritem za problem Nahrbtnika. Definirali smo še ***krepko NP-polne probleme*** in pokazali, da iz obstoja psevdopolinomskega algoritma za katerega od krepko NP-polnih problemov sledi **P**=**NP**. Kot primer NP-polnega problema, ki ni krepko NP-poln (razen v primeru **P**=**NP**), smo navedli problem Nahrbtnika, kot primer krepko NP-polnega problema pa problem Potujočega trgovca (PT) in tudi vse NP-polne probleme, pri katerih je po absolutni vrednosti največje nastopajoče celo število omejeno s polinomsko funkcijo dolžine zapisa podatkov.

Definirali smo še ***NP-težke probleme*** v najširšem smislu.