

581336 Laskennan teoria (kevät 2006)

Harjoitus 11, ratkaisuja

- (a) Ongelma voidaan selvästi ratkaista esim. kokeilemalla kaikkia mahdollisia verkon solmujen läpikäyntijärjestyksiä (joita on $n!$ kappaletta), tarkistamalla mitkä niistä ovat laillisia (ts. kaikki tarvittavat kaaret ovat olemassa) ja laskemalla kutakin laillista läpikäyntijärjestyksestä vastaavien kaarten kokonaispaino. Ongelma on siis ratkeava, ja näin ollen myös osittain ratkeava.

Ratkeavuus polynomisessa ajassa on mielenkiintoisempi kysymys. Väitämme, että ongelma ratkeaa polynomisessa ajassa, jos ja vain jos $P = NP$.

Tehtävässä esittyy *etsintäongelmaa* vastaava *päätösongelma*

Annettu: suuntaamaton verkko $G = (V, E)$, jokaiselle kaarelle $e \in E$ paino $w(e)$, luonnollinen luku C

Kysymys: onko verkossa jokin kaartensa kokonaispainolta korkeintaan C oleva Hamiltonin kehä (s.o. polku, joka käy verkon jokaisessa solmussa tasan kerran ja palaa lähtöpisteeseensä)

on kurssilta tuttu tuttu kauppamatkustajan ongelma (TSP), joka tiedetään NP-täydelliseksi. Tämä päätösongelma siis ratkeaa polynomisessa ajassa, jos ja vain jos $P = NP$.

Huom. Tehtävän oleellinen kohta on se, mitä edellä sanottiin vastaavasta päätösongelmasta. Alla esitetty yhteys päätös- ja etsintäongelman välillä kuuluu kyllä kurssin alueeseen, mutta on vähemmän keskeinen.

Selvästi jos etsintäongelma voidaan ratkaista polynomisessa ajassa, niin myös päätösongelma voidaan. Tällöin siis olisi $P = NP$.

Olkoon toisaalta $P = NP$. Tällöin siis kysymys, onko painotetussa verkossa (V, E, w) kokonaispainoltaan korkeintaan C oleva Hamiltonin kehä, mitä jatkossa merkitään $(V, E, w, C) \in \text{TSP}$, voidaan ratkaista polynomisessa ajassa. Tehtävän etsintäongelmalle saadaan seuraava algoritmi:

```
% Etsi binäärihaulla lyhin kauppamatkustajan reitin pituus C:
Aseta  $C_- := 0$  ja  $C_+ = \sum_{e \in E} w(e)$ .
Jos  $(V, E, w, C_+) \notin \text{TSP}$  niin palauta "Verkossa ei ole Hamiltonin kehää".
Toista kunnes  $C_- = C_+$ :
     $C := \frac{1}{2} \lfloor C_- + C_+ \rfloor$ 
    Jos  $(V, E, w, C) \in \text{TSP}$  niin  $C_+ := C$  muuten  $C_- := C + 1$ .
 $C := C_+$ 
% Etsi Hamiltonin kehä, jonka paino on tasan C:
Aseta  $E' := E$ .
Toista kaikilla  $e \in E$ :
    Jos  $(V, E' - \{e\}, w, C) \in \text{TSP}$  niin  $E' := E' - \{e\}$ .
Palauta joukon  $E'$  kaarista koostuva polku.
```

Ensimmäisessä vaiheessa binäärihaku tuottaa pienimmän C , jolla $(V, E, w, C) \in \text{TSP}$. Tämä C on siis pienin kokonaispaino Hamiltonin kehälle painotetussa verkossa (V, E, w) .

Toisen vaiheen aikana pidetään yllä invarianttia, että painotetussa verkossa (V, E', w) on painoltaan C oleva Hamiltonin kehä. Siis lopputilanteessakin painotetussa verkossa (V, E', w) on ainakin yksi tällainen Hamiltonin kehä. Olkoon sillä olevien kaarten joukko S . Siis erityisesti $S \subseteq E'$ kaikkien iteraatiovaiheiden aikana. Jos $e \notin S$, niin myös $S \subseteq E' - \{e\}$, jolloin $(V, E' - \{e\}, w, C) \in \text{TSP}$, ja kaari e tulee poistetuksi verkosta E' . Siis lopuksi joukossa E' on joukon S kaaret eikä mitään muuta, joten algoritmi tuottaa halutun minimipainoisen Hamiltonin kehän.

Oletuksen $P = NP$ vallitessa testi $(V, E, w, C) \in \text{TSP}$ voidaan toteuttaa polynomisessa ajassa. Algoritmin ensimmäinen vaihe tekee $O(\log \sum_e w_e)$ iteraatiota, mikä on syötteen

pitouden suhteen lineaarinen määrä (yhden painon $w(e)$ koodaaminen vie noin $\log w(e)$ bittiä). Toisessa vaiheessa tehdään kaarten lukumäärän suhteen lineaarinen määrä iteraatioita. Algoritmin aikavaativuus on siis tällöin polynominen.

- (b) Koska luku 3 saadaan positiivisten kokonaislukujen summana vain muutamalla eri tavalla ($3 = 2 + 1 = 1 + 1 + 1$), ongelma voidaan ratkaista seuraavalla algoritmilla:

```

r := 0; s := 0; t := 0;
for i := 1 to n do
  if a_i = 1 then r := r + 1
  else if a_i = 2 then s := s + 1
  else if a_i = 3 then t := t + 1;
if (t > 0) or ((r > 0) and (s > 0)) or (r ≥ 3)
  then return "kyllä"
  else return "ei".

```

Algoritmi toimii lineaarisessa ajassa, joten ongelma on polynomisessa ajassa ratkeava. Eri-tyisesti se siis on myös ratkeava ja osittain ratkeava.

- (c) Kysymyksessä on Turingin koneen M ei-triviaali semanttinen ominaisuus, joten ongelma on ratkeamaton.

Itse asiassa ongelma ei ole edes osittain ratkeava. Esitetään ongelma formaalina kielenä

$$S = \{ w \in \{0, 1\}^n \mid \text{kielen } L(M_w) \text{ voi tunnistaa äärellisellä automaatilla} \}.$$

Muodostetaan rekursiivinen palautus $f: \tilde{H} \leq_m S$ pysähtymisongelman (hieman modifioidusta) komplementista

$$\tilde{H} = \{ w111x \mid w \text{ validi koodi, } M_w \text{ ei pysähdy syötteellä } x \}.$$

Tiedämme, että \tilde{H} ei ole rekursiivisesti lueteltava (luennot, s. 106), joten tämän palautuksen olemassaolosta seuraa, että S ei ole rekursiivisesti lueteltava.

Palautusta varten valitaan ensin jokin rekursiivinen kieli B , jota ei kuitenkaan voi tunnistaa äärellisellä automaatilla. (esim. $B = \{0^n 1^n \mid n \in \mathbf{N}\}$, ks. Ohjelmoinnin ja laskennan perusmallit). Olkoon u koodi jollekin Turingin koneelle, joka pysähtyy kaikilla syötteillä ja tunnistaa kielen B (siis esim. $L(M_u) = \{0^n 1^n \mid n \in \mathbf{N}\}$). Jos v ei ole muotoa $w111x$, missä w on validi koodi, niin asetetaan palautusfunktion arvoksi $f(v) = u$. Muuten $f(v)$ on koodi seuraavalle Turingin koneelle:

- i. Simuloi konetta M_w syötteellä x . (Tässä ei siis vielä lueta mitään syötettä; vrt. Ricen lauseen todistus.)
- ii. Jos M_w pysähtyi, niin lue syöte; olkoon syötemerkkijono z . Simuloi konetta M_u syötteellä z ja hyväksy tai hylkää sen mukaan, mitä M_u tekisi.

Jos $v = w111x$, missä w on validi koodi mutta M_w ei pysähdy syötteellä x , niin edellä kuvattu kone $M_{f(v)}$ jää ikuisen silmukkaan vaiheessa (i) syötteestä riippumatta. Tällöin siis $M_{f(v)} = \emptyset$, joka voidaan tunnistaa äärellisellä automaatilla. Muuten $M_{f(v)} = M_u = B$, jota ei voida tunnistaa äärellisellä automaatilla. Siis $f(v) \in S$, jos ja vain jos $v \in \tilde{H}$. Koska f on selvästi laskettavissa, se on haluttu palautus $\tilde{H} \leq_m S$.

2. Ongelmat voidaan esittää seuraavina formaaleina kielinä:

$$\begin{aligned}
L_{(a)} &= \{ w \in \{0, 1\}^* \mid \text{jos } |x| \leq 5 \text{ niin } x \in L(M_w) \} \\
L_{(b)} &= \{ w \in \{0, 1\}^* \mid \text{on olemassa } x \in L(M_w), \text{ jolla } |x| > 5 \} \\
L_{(c)} &= \{ w \in \{0, 1\}^* \mid \text{jos } x \in L(M_w) \text{ niin } |x| \leq 5 \}.
\end{aligned}$$

Kaikissa kohdissa (a)–(c) kysymys on muotoa ”Päteekö $L(M) \in S$ ”, missä (a)-kohdassa

$$S = \{ A \in \text{RE} \mid \text{jos } |x| \leq 5 \text{ niin } x \in A \},$$

(b)-kohdassa

$$S = \{ A \in \text{RE} \mid \text{on olemassa } x \in L(M_w), \text{ jolla } |x| > 5 \}$$

ja (c)-kohdassa

$$S = \{ A \in \text{RE} \mid \text{jos } x \in L(M_w) \text{ niin } |x| \leq 5 \}.$$

(Siis RE on rekursiivisesti lueteltavien kielten luokka.) Kysymys on siis semanttisista ominaisuuksista. Koska jokaisessa tapauksessa on olemassa rekursiivisesti lueteltavat kielet A ja B , joilla $A \in S$ ja $B \notin S$, ominaisuudet ovat ei-triviaaleja.

Siis Ricen lauseen nojalla kohtien (a), (b) ja (c) ongelmat ovat kaikki ratkeamattomia.

Kohdan (a) ongelma on osittain ratkeava. Osittainen ratkaisualgoritmi saadaan simuloimalla järjestyksessä koneen M laskentaa kullakin korkeintaan 5-merkkisellä merkkijonolla, joita on äärellinen määrä. Jos kaikki simuloitut laskennat hyväksyvät, niin hyväksytään. Jos jokin simuloitu laskenta hylkää, niin hylätään. (Ja jos jokin simuloitu laskenta jää ikuisen silmukkaan, myös simulaatio jää, joten tämä ei osoita ongelmaa ratkeavaksi.)

Kohdan (b) ongelma on samoin osittain ratkeava. Osittainen ratkaisualgoritmi valitsee epädeterministisesti jonkin yli 5 merkin merkkijonon ja simuloi koneen M laskentaa tällä syötteellä. Jos M hyväksyy, niin hyväksytään; jos M hylkää, niin hylätään.

Selvästi jos on olemassa yli 5 merkin jono, jonka M hyväksyy, tämä simulaatio hyväksyy sopivilla epädeterministisillä valinnoilla; muuten simulaation hylkää aina. Siis kyseessä on epädeterministinen osittainen ratkaisualgoritmi ongelmalle, ja tunnetusti epädeterministinen osittainen ratkaisualgoritmi voidaan muuntaa deterministiseksi osittaiseksi ratkaisualgoritmiksi.

Kohdan (c) ongelma ei ole edes osittain ratkeava. Kyseessä on kohdan (b) ongelman komplementti. Tunnetusti jos sekä ongelma että sen komplementti ovat osittain ratkeavia, kumpikin on myös ratkeava. Koska kohdan (b) ongelma on osittain ratkeava mutta ei ratkeava, sen komplementti ei voi olla osittain ratkeava.

3. (a) Olkoot $A \subseteq \Sigma^*$ ja $B \subseteq \Gamma^*$ ovat kieliä. Kuten muistetaan, merkintä $A \leq_m^p B$ tarkoittaa, että on olemassa polynomisessa ajassa laskettava funktio $f: \Sigma^* \rightarrow \Gamma^*$, jolla

$$f(x) \in B \Leftrightarrow x \in A.$$

Kysymys siis on, päteekö tämä erikoistapauksessa $B = A$, kun kielestä A ei tehdä mitään oletuksia.

Kun valitaan funktioksi f identiteettifunktio $f(x) = x$, pätee selvästi $f(x) \in A$, jos ja vain jos $x \in A$, mikä on ylläoleva ehto tapauksessa $B = A$. Koska identiteettifuntio ilmeisesti voidaan laskea polynomisessa ajassa, tämä osoittaa, että $A \leq_m^p A$ kaikilla A .

(b) SAT ja HC on luennoilla todettu NP-täydellisiksi ongelmiksi. Siis

- SAT \in NP ja HC \in NP ja
- kaikilla $A \in$ NP pätee $A \leq_m^p$ SAT ja $A \leq_m^p$ HC.

Erityisesti tästä seuraa HC \leq_m^p SAT ja SAT \leq_m^p HC.

Ongelma PATH voidaan ratkaista polynomisessa ajassa esim. leveysuuntaisella etsinnällä (Tietorakenteet), joten PATH \in P. Koska P \subseteq NP, tästä seuraa PATH \in NP. Ongelmien SAT ja HC NP-täydellisyyden nojalla siis PATH \leq_m^p SAT ja PATH \leq_m^p HC.

Koska PATH \in P, mille tahansa ongelmalle A ehdosta $A \leq_m^p$ PATH seuraa $A \in$ P. Erityisesti SAT \leq_m^p PATH jos ja vain jos SAT \in P. Koska SAT on NP-täydellinen, SAT \in P on yhtäpitävää sen kanssa, että P = NP.

Siis SAT \leq_m^p PATH jos ja vain jos P = NP; on avoin ongelma onko näin. Samoin HC \leq_m^p PATH jos ja vain jos P = NP.

4. Todistetaan osumajoukko-ongelma HS NP-täydelliseksi osoittamalla

- (a) $HS \in NP$ ja
- (b) $VC \leq_m^p HS$.

Koska VC tunnetaan NP -täydelliseksi, väite seuraa.

Osumajoukko-ongelma voidaan ratkaista seuraavalla epädeterministisellä algoritmilla, joka selvästi toimii polynomisessa ajassa:

- (a) Alusta $Y := \emptyset$.
- (b) Toista arvoilla $j := 1, \dots, n$:
asetä epädeterministisesti joko $Y := Y \cup \{x_j\}$ tai $Y := Y$.
- (c) Jos $|Y| > k$, niin hylkää.
- (d) Toista arvoilla $i := 1, \dots, m$:
jos $Y \cap A_i = \emptyset$ niin hylkää.
- (e) Hyväksy.

Siis $HS \in NP$.

Palautuksen perusajatuksena on huomata, että VC on oikeastaan HS :n erikoistapaus, jossa $|A_i| = 2$ kaikilla i . Tarkemmin voidaan muodostaa seuraava palautus, joka kuvaa solmupeiteongelman tapauksen $((V, E), k)$ osumajoukko-ongelman tapaukseksi $(X, (A_1, \dots, A_m), k')$. Palautus tehdään seuraavasti:

- (a) Olkoon $V = \{v_1, \dots, v_n\}$. Asetetaan myös $X = \{v_1, \dots, v_n\}$.
- (b) Olkoon $m = |E|$ ja $E = \{(u_i, w_i) \mid i = 1, \dots, m\}$. Asetetaan $A_i = \{u_i, w_i\}$ kun $i = 1, \dots, m$.
- (c) Asetetaan $k' = k$.

Tämä palautus voidaan selvästi laskea polynomisessa ajassa. Mikä tahansa verkon (V, E) solmupeite on myös näin syntyvän joukkokokoelman osumajoukko. Erityisesti siis millä tahansa k verkossa on kokoa k oleva solmupeite jos ja vain jos joukkokokoelmalla on kokoa k oleva osumajoukko, eli $((V, E), k) \in VC$ jos ja vain jos $(X, (A_1, \dots, A_m), k') \in HS$. Siis $VC \leq_m^p HS$.