TKT20005 Laskennan mallit Viikko6

Tehtävä 1 Pysähtymisongelma.

Tässä tehtävässä pyritään ymmärtämään pysähtymisongelman määritelmä.

Mitä voit päätellä seuraavien väittämien totuusarvoista kurssilla tähän mennessä esitettyjen määritelmien ja tulosten perusteella? (Vaihtoehdot ovat siis ”tosi”, ”epätosi” ja ”annetut tiedot eivät riitä asian ratkaisemiseen”.) Perustele.

(1) On olemassa algoritmi, joka ratkaisee mille tahansa ohjelmalle P, millä syötteillä P pysähtyy ja millä ei.

epätosi

(sivu 225)
Jos olisi algoritmi, joka kaikille ohjelmille ratkaisee, millä syötteillä P pysähtyy ja millä ei, siitä tulisi ratkaisija Universaalikielen pysähtymisongelmaan.

(2) On olemassa sellainen ohjelma P, että mikään algoritmi ei pysty ratkaisemaan, millä syötteillä P pysähtyy ja millä ei.

tosi

On olemassa tietty ohjelma P, jonka pysähtymissyötteiden joukko ei ole ratkeava; Universaali simulaattori U: Ohjelma P, Syötteellä y: tulkitse y koodina $\langle M, w\rangle$ ja simuloi M:ää syötteellä w; jos M pysähtyy, myös P pysähtyy.

Tällöin P:n pysähtymisjoukko on H joka ei ole ratkeava. Eli jonkin P:n kohdalla ei ole algoritmia, joka kertoisi kaikille syötteille pysähtyykö P

(3) Kaikille ohjelmille P pätee, että mikään algoritmi ei pysty ratkaisemaan, millä syötteillä P pysähtyy ja millä ei.

epätosi

Väite on liian kattava.
Vastatapaus: ohjelma, joka pysähtyy kaikilla syötteillä (pysähtymisjoukko Σ^*, siis ratkeava), tai ohjelma, joka ei pysähdy millään syötteellä (pysähtymisjoukko $\emptyset$, eli taas ratkeava)

Tehtävä 2 Turing-ratkeamattomuus.

Tässä tehtävässä harjoitellaan, kuinka jokin kieli todistetaan ratkeamattomaksi.
Tarkastellaan kieltä

$$A = \{ \langle M, w, q \rangle \mid M \text{ on Turingin kone, joka syötteellä } w \text{ menee ainakin kerran tilaan } q \}.$$

Viime viikon tehtävässä 3 todistimme, että tämä kieli on Turing-tunnistettava.
Todista, että kieli A ei ole ratkeava.

Tehdään vastaoletus että A on ratkeava. Olkoon R Turingin kone joka päättää A

Käyttämällä apuna konetta R hyväksymisongelma

$$A_{\text{TM}} = \{\langle M, w\rangle \mid M \text{ hyväksyy syötteen } w\}$$

voitaisiin ratkaista Turingin koneell, joka syötteellä y toimii:

(1) Jos y ei ole muotoa $\langle M, w\rangle$, hylkää

(2) Poimi M:n koodista sen hyväksyvä tila q_{acc} ja muodosta $\langle M, w, q_{\text{acc}}\rangle$

(3) Simuloi konetta R syötteellä $\langle M, w, q_{\text{acc}}\rangle$

- Jos R hyväksyi, niin hyväksy
- Jos R hylkäsi, niin hylkää

M hyväksyy w täsmälleen silloin, kun sen laskennassa syötteellä w vieraillaan tilassa q_{acc} ainakin kerran; eli täsmälleen silloin kun $\langle M, w, q_{\text{acc}}\rangle \in A$. Koska R päättää A:n, yllä oleva kone päättää A_{TM}.
Mutta A_{TM} ei ole ratkeava mistä seuraa ristiriita. Siis A ei ole ratkeava. ■

Tehtävä 3 Vaativuusluokka P

Tässä tehtävässä harjoitellaan, kuinka ongelma määritellään formaalina kielenä ja kuinka todistetaan, että jokin ongelma kuuluu vaativuusluokkaan P.
Tarkastellaan seuraavaa ongelmaa:

Annettu: Lista kokonaislukuja L ja kokonaisluku k.
Kysymys: Esiintyykö k listassa L?

1. Esitä yllä annettu ongelma formaalina kielenä.
2. Todista, että määrittelemäsi formaali kieli kuuluu luokkaan P.

(1) Esitä yllä annettu ongelma formaalina kielenä.

Käytetään aakkostoa $\Sigma = \{0, 1, \#, -\}$.
Määritellään kokonaislukujen koodijoukko binäärisenä

$$E \;=\; \{\,0\,\} \;\cup\; \{\,s\,1x \mid s \in \{\varepsilon, -\},\; x \in \{0,1\}^*\,\}$$

eli luvun koodi on joko 0 tai {valinnainen miinus + 1 + mielivaltainen bittijono}
Silloin päätös ongelmaa esiintyykö k listassa L? esittävä kieli on (## erottaa ”listan”ja ”kohteen”)

$$L \;=\; \{\,u_1\#u_2\#\cdots\#u_m\#\#v \mid m \geq 0,\; \forall i :\; u_i \in E,\; v \in E,\; \exists i :\; u_i = v\,\}$$

(2) Todista, että määrittelemäsi formaali kieli kuuluu luokkaan P.

$$\text{B} = \{\, \langle L, k\rangle \mid k \text{ esiintyy listassa } L \,\} = \{\, u_1\#\cdots\#u_m\#\#v \mid \exists i :\ u_i = v \,\}$$

Väite: B $\in$ **P**

Laajennetaan nauha-aakkostoa työsymboleilla r ja t joilla merkitään jo verratut bitit. Syöte x, pituus n, on muodossa $u_1\#\cdots\#u_m\#\#v$

1. Muototarkistus.
   Kulje vasemmalta oikealle ja tarkista muoto $u_1\#\cdots\#u_m\#\#v$ ja että jokainen lohko on koodista $E = \{0\} \cup 1\{0,1\}^*$. Jos ei, hylkää
2. *Vertaa jokaista u_i:tä v:hen merkitsemällä.* Käy listalohkot $u_1, \ldots, u_m$ vasemmalta oikealle:
   (a) Palauta (tarvittaessa) v ennalleen: korvaa kaikki r, t v-osassa takaisin $0, 1$:iksi
   (b) Aloita u_i:n alusta. Toista:
   - Etsi u_i:stä vasemmalta lukien ensimmäinen merkitsemätön symboli. Jos sellaista ei ole (eli tuli vastaan '#'), tarkasta että myös v:ssä ei ole merkitsemättömiä $0/1$-symboleja. Jos ei ole, hyväksy
   - Lue löytynyt bitti $b \in \{0,1\}$; siirry v:n alkuun (skannaa '##') ja etsi v:stä ensimmäinen merkitsemätön bitti. Jos bitti on b, merkitse molemmat (kirjoita $0 \mapsto r$ tai $1 \mapsto t$ sekä u_i:ssä että v:ssä) ja palaa u_i:n alkuun jatkamaan
3. Jos kaikki lohkot käsiteltiin ilman hyväksyntää, hylkää.

Aika-analyysi. Kukin merkintä/vasta-merkinnän palautus voi vaatia pään siirtelyä korkeintaan $O(n)$ solun yli (edestakaisin listan ja v:n välillä). Yhtä lohkoa kohden merkitään enintään sen pituus + $|v|$ symboleja, ja koko syötteen eri lohkoissa on yhteensä $O(n)$ symboleja. Siten kokonaisaika on $O(n) \cdot O(n) = O(n^2)$. Polynominen aika $\Rightarrow$ B $\in$ **P**

Tehtävä 4 Lisäharjoitusta ratkeamattomuustodistuksiin.

Tavoitteena on laatia ohjelmantarkastustyökalu, johon voi ladata tarkastettavaksi minkä tahansa Python-ohjelman. Käyttäjä valitsee haluamansa syötteen annettavaksi tarkastettavalle ohjelmalle. Lisäksi käyttäjä valitsee ohjelmasta jonkin koodirivin. Tarkastustyökalun tehtävänä on päättää, suorittaisiko tarkastettava ohjelma valitun koodirivin ainakin kerran, kun ohjelmalle annetaan valittu syöte.

Onko tällainen tarkastustyökalu mahdollista toteuttaa niin, että se varmasti toimii oikein millä tahansa ohjelmalla, syötteellä ja ohjelmasta valitulla koodirivillä? Perustele.

Olkoon Rivi kieli mikä sisältää kolmikot $\langle P, x, \ell \rangle$, kun Python-ohjelma P syötteellä x suorittaa koodirivin ℓ ainakin kerran.

Väite
Kieli Rivi ei ole ratkeava; siis pyydettyä työkalua, joka aina pysähtyy ja vastaa oikein kaikilla syötteillä, ei ole olemassa

Perustelu (hyväksymisongelmasta A_{TM})
Tiedetään että hyväksymisongelma

$$A_{\mathsf{TM}} = \{\langle M, w \rangle \mid M \text{ hyväksyy syötteen } w\}$$

on ratkeamaton.

Näytetään että jos Rivi olisi ratkeava, myös A_{TM} olisi, mikä on mahdotonta

Annetulle parille $\langle M, w \rangle$ rakennetaan Python-ohjelma $P_{M,w}$, joka tekee:

```
simuloi M:ää syötteellä w;
M hyväksyy: suorita rivi L ja pysähdy;
M hylkää tai ei pysähdy: älä koskaan suorita riviä L  (esim. jää silmukkaan)
```

Tällöin

$$\langle M, w \rangle \in A_{\mathsf{TM}} \iff P_{M,w} \text{ suorittaa rivin } L \text{ syötteellä } x$$

(riippumatta vakiovalinnasta x, esim. $x = \varepsilon$)

Jos meillä olisi päättävä työkalu C ongelmaan Rivi, voisimme ratkaista A_{TM}:n algoritmilla: syötteellä $\langle M, w \rangle$ muodosta $\langle P_{M,w}, x, L \rangle$ ja palauta C:n vastaus. Tämä tekisi A_{TM}:stä ratkeavan eli siis ristiriita.

Siksi Rivi ei ole ratkeava, eikä pyydettyä yleispätevää tarkastustyökalua voi toteuttaa. ■

Tehtävä 5 Kieli, joka ei ole Turing-tunnistettava.

Todista, että kieli

$$B = \{ \langle M, w, q \rangle \mid M \text{ on Turingin kone, joka syötteellä } w \textit{ ei mene} \text{ kertaakaan tilaan } q \}.$$

ei ole Turing-tunnistettava.
(*Vihje:* Tarkastele kielen B komplementtia ja hyödynnä luentomonisteen lausetta 4.7 sekä tehtävän 2 ja viime viikon tehtävän 3 tulosta, että kieli

$$A = \{ \langle M, w, q \rangle \mid M \text{ on Turingin kone, joka syötteellä } w \text{ menee ainakin kerran tilaan } q \}$$

on Turing-tunnistettava, mutta ei ratkeava.)

Olkoon

$$A = \{\langle M, w, q\rangle \mid M \text{ syötteellä } w \text{ käy ainakin kerran tilassa } q\}$$

ja

$$B = \{\langle M, w, q\rangle \mid M \text{ syötteellä } w \textit{ ei} \text{ käy kertaakaan tilassa } q\}$$

Selvästi $B = \overline{A}$ (komplementti hyvinmuodostettujen kolmikkojen joukossa)

Viime viikolla todistettiin, että A on Turing-tunnistettava, mutta ei ratkeava. Oletetaan, että B olisi Turing-tunnistettava. Tällöin sekä A että $\overline{A}$ olisivat tunnistettavia ja Lauseen 4.7 mukaan A olisi ratkeava. Tämä on ristiriidassa edellä mainitun tuloksen kanssa.

Siis B *ei* ole Turing-tunnistettava. □