

## 582206 Laskennan mallit (syksy 2007)

Harjoitus 12 (4.–7.12.)

### Perustehtävät

Ratkaise seuraavat tehtävät ennen laskuharjoitustilaisuutta. Jos niissä on jotain epäselvää, valmistaudu kysymään asiasta laskuharjoituksissa.

1. Mitä voit päätellä seuraavien väittämien totuusarvoista olettaen tunnetuksi pysähtymisongelman ratkeamattomuuden sekä asiaan liittyvät perusmääritelmät? (Vaihtoehdot ovat siis ”tosi”, ”epätosi” ja ”annetut tiedot eivät riitä asian ratkaisemiseen”).
  - (a) On olemassa algoritmi, joka ratkaisee mille tahansa ohjelmalle  $P$ , millä syötteillä  $P$  pysähtyy ja millä ei.
  - (b) On olemassa sellainen ohjelma  $P$ , että mikään algoritmi ei pysty ratkaisemaan, millä syötteillä  $P$  pysähtyy ja millä ei.
  - (c) Kaikille ohjelmille  $P$  pätee, että mikään algoritmi ei pysty ratkaisemaan, millä syötteillä  $P$  pysähtyy ja millä ei.

(Jatkokysymys: jos jonkin edeltävän väitteen totuus ei ollut ratkaistavissa pelkästään pysähtymisongelman ratkeamattomuuden perusteella, ratkaise se käyttäen muita kurssilla esitettyjä tuloksia.)

### Yhteistehtävät

Lue seuraavat tehtävät huolellisesti ja kertaa tarvittavat käsitteet kurssikirjasta. Valmistaudu osallistumaan tehtävien ratkaisemiseen laskuharjoitustilaisuudessa yhteisvoimin. (Näitä tehtäviä siis *ei* ole tarkoitus ratkaista itsenäisesti etukäteen.)

2. Määritellään Turingin koneen hyväksymisongelman  $A_{TM}$  kanssa analogisesti äärellisen automaatin hyväksymisongelma

$$A_{DFA} = \{ \langle A, w \rangle \mid A \text{ on DFA joka hyväksyy merkkijonon } w \}.$$

Tässä DFA ajatellaan koodatuksi esim. samaan tapaan kuin Turingin koneet kielessä  $A_{TM}$ . Selvästi  $A_{DFA}$  on ratkeava, mutta onko se säännöllinen? Toisin sanoen onko olemassa ”universaali DFA”, joka osaa simuloida mitä tahansa DFA:ta? Perustele.

3. Tehtävänä on laatia ohjelmantarkastustyökalu, johon voi ladata tarkastettavaksi minkä tahansa Java-kielisen ohjelman. Käyttäjä valitsee haluamansa mielivaltaisen 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 laatia? Entä jos tehtävää muutetaan niin, että käyttäjä ei valitse syötettä, vaan tehtävänä on päättää, tulisiko koodirivi suoritetuksi edes *jollain* syötteellä? Perustele.

4. Oletetaan, että Churchin-Turingin teesi ei pidäkään paikkaansa ja joku valmistaa koneen, jolla voi ratkaista tehokkaasti Turingin koneiden pysähtymisongelman. Sanotaan tätä konetta *pysähtymisoraakkeli*. Pysähtymisoraakkeliä käyttäen muodostetaan *superalgoritmeja*, jotka ovat muuten ”perinteisiä” algoritmeja, mutta voivat normaalien laskenta-askelien lisäksi halutessaan tehdä kysymyksiä pysähtymisoraakkelille.

Superalgoritmia käyttämällä on tietysti triviaalia ratkaista Turingin koneen pysähtymisongelma. Mutta onko olemassa superalgoritmi, joka ratkaisee *superalgoitmien* pysähtymisongelman? Perustele.