**Laskennan teorian opintopiiri**

# Tilavaativuus ja Savitchin lause

Jarno Leppänen

23.2.2014

**Tiivistelmä**

Työssä esitetään määritelmä Turingin koneen tilavaativuudelle. Lisäksi esitellään tilavaativuutta koskeva keskeinen tulos, Savitchin lause, joka osoittaa epädeterministisen ja deterministisen Turingin koneen välisen vastaavuuden tilavaativuuden kannalta.

## 1 Johdanto

Tila, tai käytetyn muistin määrä, on ajan ohella keskeinen resurssi, kun tarkastellaan laskennallisten ongelmien vaativuutta käytännössä. Algoritmin vaatimalle tilalle voidaan aikavaativuuden tavoin esittää formaali määritelmä Turingin koneen avulla. Tilavaativuudella on monia yhteisiä ominaisuuksia aikavaativuuden kanssa ja sen avulla voidaan jakaa laskennallisia ongelmia edelleen erilaisiin luokkiin perustuen ongelmien laskennalliseen vaativuuteen.

Työssä seurataan M. Sipserin kirjan *Introduction to the Theory of Computation*[1] esitystä tilavaativuudesta.

## 2 Turingin koneen tilavaativuus

**Määritelmä 1.** Olkoon $M$ kaikilla syötteillä pysähtyvä Turingin kone. Koneen $M$ ***tilavaativuus*** on funktio $f : \mathcal{N} \to \mathcal{N}$, missä $f(n)$ on suurin koneen $M$ käyttämien nauhapaikkojen lukumäärä $n$ pituisella syötteellä. Jos koneen $M$ tilavaativuus on $f(n)$, sanomme koneen toimivan tilassa $f(n)$.

Jos $M$ on epädeterministinen Turingin kone, jonka kaikki epädeterministiset haarat pysähtyvät kaikilla syötteillä, on tilavaativuus $f(n)$ suurin minkä tahansa haaran käyttämien nauhapaikkojen lukumäärä $n$ pituisella syötteellä.

**Määritelmä 2.** ***Tilavaativuusluokat*** **SPACE($f(n)$)** ja **NSPACE($f(n)$)** määritellään seuraavasti:

$$\begin{aligned}\text{SPACE}(f(n)) = \{L \mid\ & L \text{ on tilassa } \mathcal{O}(f(n)) \text{ toimivan deterministisen} \\ & \text{Turingin koneen tunnistama kieli } \} \\ \text{NSPACE}(f(n)) = \{L \mid\ & L \text{ on tilassa } \mathcal{O}(f(n)) \text{ toimivan epädeterministisen} \\ & \text{Turingin koneen tunnistama kieli } \}\end{aligned}$$

**Esimerkki 1.** *SAT*-, eli lauselogiikan toteutuvuusongelmassa kysytään, onko annetulle Boolen lausekkeen muuttujille löydettävissä totuusarvoja, joilla lauseke toteutuu. *SAT* kuuluu NP-täydellisten ongelmien luokkaan ja on yleisesti uskottu, ettei ongelmalle ole löydettävissä polynomisessa ajassa toimivaa algoritmia. Ongelmalle voidaan kuitenkin esittää seuraava yksinkertainen ratkaisu:

$M_1$ = "Syötteellä $\langle\phi\rangle$, missä $\phi$ on Boolen lauseke:

**1.** Kaikille lausekkeen $\phi$ muuttujien $x_1, \ldots, x_m$ arvoille:

**2.** Evaluoi $\phi$. Jos $\phi$ toteutuu, *hyväksy.*

**3.** Jos lauseketta $\phi$ ei hyväksytty millään arvoilla, *hylkää.*

Kone $M_1$ suoriutuu lineaarisessa tilassa, sillä algoritmin jokainen iteraatio voi uudelleenkäyttää nauhan samaa osaa. Koneen täytyy tallentaa kullakin iteraatioaskeleella ainoastaan lausekkeen sisältämien muuttujien totuusarvot, mikä voidaan tehdä tilassa $\mathcal{O}(m)$. Muuttujien lukumäärää rajoittaa syötteen pituus $n$, joten koneen $M_1$ tilavaativuus on $\mathcal{O}(n)$. Tila näyttääkin olevan aikaa tehokkaampi resurssi, sillä sitä voidaan käyttää uudelleen.

**Esimerkki 2.** Tarkastellaan ongelmaa, joka kysyy, hyväksyykö annettu epädeterministinen automaatti kaikki mahdolliset automaatin aakkostosta muodostetut merkkijonot. Olkoon kieli $ALL_{\text{NFA}}$ tällaisten automaattien kuvausten joukko:

$$ALL = \{\langle A\rangle \mid A \text{ on } \mathsf{NFA} \text{ ja } L(A) = \Sigma^*\}.$$

Kielen $ALL_{\text{NFA}}$ ei tiedetä olevan luokassa NP tai coNP. Kielen komplementille $\overline{ALL_{\text{NFA}}}$ voidaan kuitenkin esittää epädeterministinen lineaarisessa tilassa toimiva algoritmi. Algoritmin ideana on käyttää epädetermismiä arvaamaan merkkijono, jonka automaatti

hylkää ja pitää kirjaa automaatin konfiguraatiosta lineaarisessa tilassa.

$N =$ "Syötteellä $\langle M \rangle$, missä $M$ on NFA:

**1.** Merkitse automaatin $M$ aloitustila.

**2.** Toista $2^q$ kertaa, missä $q$ on automaatin $M$ sisältämien tilojen lukumäärä:

**3.** Valitse epädeterministisesti syötemerkki ja simuloi automaatin $M$ toimintaa päivittämällä seuraavat mahdolliset tilat.

**4.** *Hyväksy*, jos vaiheissa 2 ja 3 löytyi jokin merkkijono, jonka $M$ hylkää, eli jollakin iteraatioaskeleella yksikään merkatuista tiloista ei ole hyväksyvä tila. *Hylkää* muutoin.

Simuloidulla automaatilla voi olla korkeintaan $2^q$ merkattujen tilojen kombinaatiota. Jos automaatti hylkää jonkin merkkijonon, merkkijonon on oltava tätä lyhyempi, sillä tätä pidemmissä merkkijonoissa tilojen kombinaation on toistuttava ja toistuvien kombinaatioiden väliä vastaava merkkijonon osa voidaan aina poistaa. Näin ollen $N$ päättää kielen $\overline{ALL_{\text{NFA}}}$.

Epädeterministisen algoritmin täytyy pitää kirjaa ainoastaan merkatuista tiloista sekä toistolauseen laskurista, joten algoritmi toimii tilassa $\mathcal{O}(n)$.

## 3 Savitchin lause

Epädeterministisen Turinin koneen simuloiminen deterministisella Turingin koneella näyttää vaativan eksponentiaalisen määrän aikaan epädeterministiseen koneeseen verrattuna. Tilavaativuuden suhteen näin ei kuitenkaan ole, vaan epädeterministisen koneen simulointi onnistuu yllättäen vain neliöllisessä tilassa deterministisellä Turingin koneella.

Varhaisimpiin tilavaativuutta koskeviin teoreettisiin tuloksiin kuuluva Savitchin lause osoittaa, että tilassa $f(n)$ toimivaa epädeterminstistä Turingin konetta voidaan simuloida tilassa $f^2(n)$ toimivalla deterministisellä Turingin koneella. Lauseen perusteella esimerkiksi edellinen, varsin teoreettiselta tuntunut algoritmi voidaan muuttaa deterministiseksi, neliöllisessä tilassa toimivaksi algoritmiksi.

**Savitchin lause.** Kaikille funktioille $f : \mathcal{N} \to \mathcal{R}^+$, missä $f(n) \geq n$, pätee

$$\text{NSPACE}(f(n)) \subseteq \text{SPACE}(f^2(n)).$$

## Viitteet

[1] Michael Sipser. *Introduction to the Theory of Computation.* Cengage Learning, third edition, 2013.