T-79.148 **Syksy 2003**
Tietojenkäsittelyteorian perusteet
Harjoitus 12
Demonstraatiotehtävien ratkaisut

4. Tehtävässä halutaan todistaa seuraava ongelma ratkeamattomaksi:

   Hyväksyykö annettu Turingin kone M syötteen ε?

   Määritellään aluksi kieli $L = \{M \mid M \text{ pysähtyy syötteellä } \varepsilon\}$. Nyt L on rekursiivinen jos ja vain jos tehtävänannon ongelma on ratkeava. Seuraavaksi osoitetaan, että kieli $H = \{Mw \mid M \text{ pysähtyy syötteellä } w\}$ voidaan palauttaa rekursiivisesti (*recursively reduce*) kieleen L (merkitään $H \leq_m L$), joten L on vähintään yhtä vaikea kuin H. Koska H ei ole rekursiivinen (monisteen lause 6.9), ei L voi myöskään olla rekursiivinen.

   Rekursiivinen palautus määritellään seuraavasti: Olkoon $A \subseteq \Sigma^*$ ja $B \subseteq \Gamma^*$ kieliä. Nyt $A \leq_m B$ jos ja vain jos on olemassa rekursiivinen funktio $f : \Sigma^* \to \Gamma^*$ siten, että

   $$\forall w \in \Sigma^* : w \in A \Leftrightarrow f(w) \in B \ .$$

   Tässä tapauksessa halutaan löytää funktio f siten, että $f(Mw) \in L$ jos ja vain jos $Mw \in H$. Käytännössä tämä tarkoittaa sitä, että halutaan löytää systemaattinen tapa rakentaa Turingin kone M', joka pysähtyy tyhjällä syöttellä täsmälleen silloin, kun kone M pysähtyy syötteellä $w = w_1 w_2 \cdots w_n$.

   Onneksi tämä on helppo tehtävä. Kone M' kirjoittaa ensin nauhalle sanan w, siirtyy takaisin nauhan alkuun, ja alkaa simuloida konetta M tekemällä samat siirtymät kuin M:kin tekisi. Nyt M' pysähtyy jos ja vain jos M pysähtyy.

   Formaalisti f voidaan määritellä seuraavasti:

   $$f(\langle Q, \Sigma, \Gamma, \delta, q_0, q_{\text{acc}}, q_{\text{rej}} \rangle, w_1 w_2 \cdots w_n) = \langle Q', \Sigma, \Gamma, \delta', q_0', q_{\text{acc}}, q_{\text{rej}} \rangle,$$

   missä

   $$\begin{aligned} Q' &= Q \cup \{q_i' \mid 0 \leq i \leq n\} \\ \delta' &= \delta \cup \{\langle q_i', \varepsilon, q_{i+1}', w_{i+1}, R \rangle \mid 0 \leq i < n\} \\ &\quad \cup \{\langle q_n', x, q_n', x, L \rangle \mid x \in \Gamma \cup \{<\}\} \\ &\quad \cup \{\langle q_n', >, q_0, >, R \rangle\} \end{aligned}$$

   Koska koneeseen M lisätään vain äärellinen määrä uusia tiloja ja siirtymiä (n ei voi olla ääretön), on f selvästikin rekursiivinen funktio.

5. **Tehtävä**: Todista monisteen Lause 6.15:

   (i) Kieli $A \subseteq \Sigma^*$ on rekursiivinen, jos ja vain jos sen karakteristinen funktio

   $$\chi_A : \Sigma^* \to \{0, 1\}, \qquad \chi_A(x) = \begin{cases} 1, & \text{jos } x \in A; \\ 0, & \text{jos } x \notin A \end{cases}$$

   on rekursiivinen funktio.

   (ii) Kieli $A \subseteq \Sigma^*$ on rekursiivisesti numeroituva, jos ja vain jos on $A = \emptyset$ tai on olemassa rekursiivinen funktio $g : \{0, 1\}^* \to \Sigma^*$, jolla

   $$A = \{g(x) \mid x \in \{0, 1\}^*\}.$$

   Vastaus: Määritellään aluksi viisi yksinkertaista apukonetta:

- **1** kirjoittaa nauhalle merkin 1, siirtää lukupäätä askeleen oikealle ja pysähtyy.
- **0** kirjoittaa nauhalle merkin 0, siirtää lukupäätä askeleen oikealle ja pysähtyy ja pysähtyy.
- C tyhjentää koneen syötenauhan, palauttaa lukupään nauhan alkuun ja pysähtyy.
- NEXT lukee nauhalla olevan merkkijonon $x \in \Sigma^*$ ja korvaa sen leksikografisesti seuraavalla merkkijonolla (tehtävän 9.2 tapaan).
- $Cmp^{i,j}$ vertailee moninauhaisen Turingin koneen nauhoja i ja j keskenään ja hyväksyy syötteen mikäli niiden sisältö on sama.

Koneet ovat yksinkertaisia, joten niitä ei esitetä tässä.

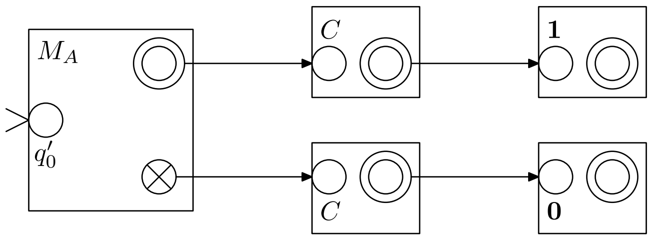
(i) [$\Rightarrow$] Olkoon kieli $A \subseteq \Sigma^*$ rekursiivinen. Tällöin on olemassa Turingin kone

$$M_A = \langle Q, \Sigma, \Gamma, \delta, q_0, q_{\text{acc}}, q_{\text{rej}} \rangle$$

siten, että:

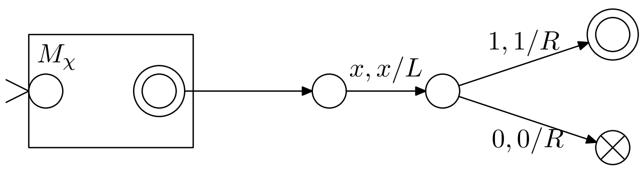
$$\begin{aligned} \forall w \in \Sigma^* : w \in L \Leftrightarrow (q_0, w) \vdash^*_{M_A} (q_{\text{acc}}, \alpha) \quad \text{ja} \\ w \notin L \Leftrightarrow (q_0, w) \vdash^*_{M_A} (q_{\text{rej}}, \alpha) \end{aligned}$$

Muodostetaan Turingin kone M yhdistämällä kone M_A koneisiin **1**, **0** ja C seuraavaan tapaan:



Mikäli $w \in L$, niin kone M_A päätyy hyväksyvään tilaan, minkä jälkeen M tyhjentää nauhan ja kirjoittaa nauhalle luvun 1. Muussa tapauksessa nauhalle kirjoitetaan 0. Koska A on rekursiivinen, M_A pysähtyy aina, joten myös M pysähtyy aina, joten M laskee funktion $\chi(w) = \begin{cases} 1, w \in A \\ 0, w \notin A \end{cases}$, joka on kielen A karakteristinen funktio.

[$\Leftarrow$] Seuraavaksi oletetaan, että funktio $\chi(w)$ on rekursiivinen. Tällöin on olemassa Turingin kone M_χ, joka laskee sen arvon. Määritelmän mukaan M_χ pysähtyy aina, ja laskennan tulos kirjoitetaan nauhalle lukupään vasemmalle puolelle. Muodostetaan Turingin kone M seuraavaan tapaan:



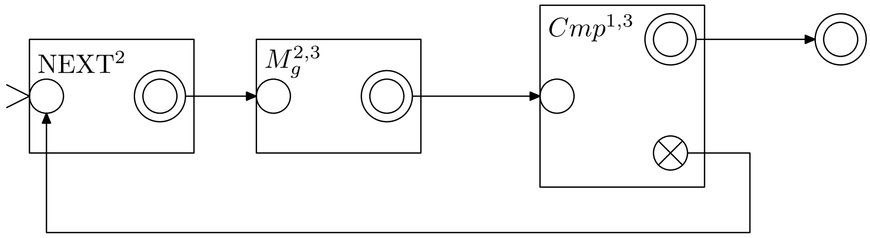
Nyt kone M hyväksyy sanan w aina, kun $\chi(w) = 1$, ja hylkää sen, kun $\chi(w) = 0$. Näin ollen M tunnistaa kielen A totaalisesti, joten A on rekursiivinen.

(ii) Halutaan todistaa väite:

> Kieli $A \subseteq \Sigma^*$ on rekursiivisesti numeroituva ($A \in RE$), jos ja vain jos $A = \emptyset$ tai on olemassa rekursiivinen funktio $g : \{0,1\}^* \to \Sigma^*$, jolla
>
> $$A = \{g(x) \mid x \in \{0,1\}^*\}$$

Mikäli $A = \emptyset$, niin triviaalisti $A \in RE$ ja $g(x) = 0$ vastaava rekursiivinen funktio.

Jos annetut ehdot täyttävä funktio g on olemassa, niin on olemassa Turingin kone M_g, joka laskee g:n. Tästä voidaan triviaalisti muodostaa kaksinauhainen kone $M_g^{1,2}$, joka laskee g:n mutta tallettaa tuloksen toiselle nauhalle ensimmäisen asemasta. Muodostetaan 3-nauhainen Turingin kone M_A seuraavasti:



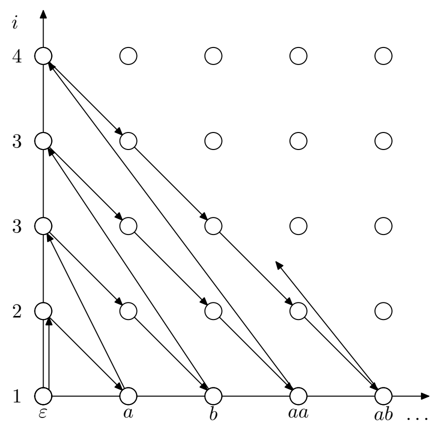
Kone saa syötteensä ensimmäisellä nauhalla, ja sitä ei enää missään vaiheessa muuteta. Varsinaisessa laskentasilmukassa kone M_A korvaa 2. nauhalla olevan bittijonon x leksikografisesti seuraavalla bittijonolla y, minkä jälkeen lasketaan $g(y)$ ja talletetaan tulos 3. nauhalle. Lopuksi vertaillaan 1. ja 3. nauhojen sisältö keskenään. Mikäli nauhojen sisältö on sama, sana hyväksytään, muuten palataan alkuun.

[$\Leftarrow$] Tarkastellaan sanaa $w \in A$. Oletetaan, että annetut ehdot täyttävä rekursiivinen funktio g on olemassa. Tällöin $w = g(x)$ jollain $x = x_1 x_2 \cdots x_n$, missä n on äärellinen. Koska millä tahansa äärellisen mittaisella merkkijonolla on vain äärellinen määrä edeltäjiä leksikografisessa järjestyksessä, generoi NEXT lopulta x:n, jolloin $M_g^{2,3}$ laskee kolmannelle nauhalle sanan w, ja kone hyväksyy sanan. Näin ollen M_A tunnistaa kielen A, joten $A \in RE$.

[$\Rightarrow$] Seuraavaksi oletetaan, että $A \in RE - \{\emptyset\}$. Tällöin on olemassa Turingin kone M_A, joka tunnistaa A:n. Määritellään apukone $M_{A,i}$, joka simuloi koneen M_A toimintaa i:n askeleen ajan. Kone $M_{A,i}$ hyväksyy syötteen x mikäli M_A hyväksyy sen käyttäen korkeintaan i askelta, muuten se hylkää sen. Huomataan, että kone $M_{A,i}$ on totaalinen, eli se pysähtyy aina.

Funktio g rakennetaan koneen $M_{A,i}$ avulla. Kaikki syötteet x ja rajat i koodataan funktion $c(x, y) = 0^x 1 0^y$ avulla bittijonoiksi pareittain[1], ja $g(c(x, y)) = x$, mikäli $M_{A,y}$ hyväksyy sanan x. Funktiota g määriteltäessä valitaan jokin kiinnitetty sana $x_0 \in A$, joka palautetaan aina kun $M_{A,y}$ ei pysähdy tai jos g:n syöte on väärää muotoa.

Konstruktion perusajatuksena on käydä läpi kaikki mahdolliset M_A:n syötteet lomitettuina: ensin ajetaan M_A:ta yhden askeleen verran ensimmäisellä syötteellä, sitten kaksi askelta ensimmäisellä syötteellä, yksi askel toisella syötteellä, jne. Alla olevassa kuvassa on tämä havainnollistettu aakkostolle $\Sigma = \{a, b\}$ (huomaa samankaltaisuus tehtävän 2.5 kanssa):



[1] Tässä samaistetaan Σ^*:n sanat luonnollisten lukujen kanssa tavanomaiseen tapaan käyttäen leksikografista järjestystä.

Esitetään vielä lopuksi todistus formaalimmin: Määritellään funktio $g' : \{0,1\}^* \to \{0,1\}^*$ seuraavasti:

$$g'(w) = \begin{cases} x, & w = 0^x 10^y \text{ ja } M_{A,y}(x) \text{ pysähtyy} \\ x_0, & \text{muulloin }, \end{cases}$$

missä $x_0 \in A$. Lopuksi määritellään funktio $g(x) = d(g'(x))$, missä d on funktio, joka kuvaa bittijonon 0^x joukon Σ^* x:nteen alkioon leksikografisessa järjestyksessä. Funktion $g'(x)$ arvo voidaan laskea äärellisessä ajassa, koska $M_{A,y}(x)$ ei voi jäädä ikuiseen silmukkaan. Näin ollen g' on rekursiivinen, joten myös g:kin on.

Tässä vaiheessa on huomattava, että vaikka g on aina olemassa, niin sitä ei ole välttämättä mahdollista löytää, sillä sopivan alkion $x_0 \in A$ etsiminen on yleisessä tapauksessa ratkeamaton ongelma.

6. **Tehtävä**: Osoita, että yhteysherkät kielet voidaan tunnistaa lineaarisesti rajoitetuilla automaateilla. (Käytä hyväksesi sitä, että kieliopin produktioita sovellettaessa lausejohdoksen pituus ei voi koskaan lyhentyä, paitsi tyhjän merkkijonon muodostamassa erikoistapauksessa.) Päättele edellisen perusteella, että kaikki yhteysherkät kielet ovat rekursiivisia.

   Vastaus: Yhteysherkän kieliopin määritelmän mukaan produktiot ovat sellaisia, että yksittäisessä säännössä oikea puoli on aina vähintään yhtä pitkä kuin vasen.

   Yhteysherkkä kieli voidaan tunnistaa lineaarisesti rajoitetulla automaatilla, joka epädeterminisesti siirtyy johonkin kohtaan syötettä ja soveltaa jotain sääntöä oikealta vasemmalle. Koska merkkijono tällöin lyhenee, lisätilaa ei tarvita. Toisaalta, jos väliin jäisi tyhjiä merkkejä, voidaan implementoida suoraviivainen tiivistys. Jos nauhalle jää pelkkä kieliopin aloitussymboli, siirrytään hyväksyvään lopputilaan.

   Tarkastellaan esimerkiksi yhteysherkkää kielioppia:

$$\begin{aligned} S &\to aA \mid bB \\ aA &\to abB \mid ab \\ bB &\to baA \mid ba \end{aligned}$$

   Tässä kielen tunnistava lineaarisesti rajoitettu automaatti muokkaisi syötenauhaansa seuraavaan tapaan (syötteellä $abab$):

$$\begin{array}{|c|c|c|c|c|c|}\hline > & \underline{\text{a}} & \text{b} & \text{a} & \text{b} & < \\ \hline\end{array} \vdash^* \begin{array}{|c|c|c|c|c|c|}\hline > & \text{a} & \text{b} & \underline{\text{a}} & \text{A} & < \\ \hline\end{array}$$

$$\vdash^* \begin{array}{|c|c|c|c|c|}\hline > & \text{a} & \underline{\text{b}} & \text{B} & < \\ \hline\end{array}$$

$$\vdash^* \begin{array}{|c|c|c|c|}\hline > & \underline{\text{a}} & \text{A} & < \\ \hline\end{array}$$

$$\vdash^* \begin{array}{|c|c|c|}\hline > & \underline{\text{S}} & < \\ \hline\end{array}$$

   Yllä kuvattu kone ei kuitenkaan ole totaalinen, sillä on mahdollista, että sen laskenta ei koskaan pysähdy. Esimerkiksi, jos käsiteltävä kielioppi on:

$$\begin{aligned} S &\to \varepsilon \\ ab &\to ba \\ ba &\to ab, \end{aligned}$$

   niin kaikki laskennat syötteellä, jossa on ainakin yksi ab tai ba pari jäävät ikuiseen silmukkaan.

   Asian voi korjata huomaamalla, että koska koneen nauhan pituus ei voi kasvaa, on koneella on vain rajallinen määrä mahdollisia tilanteita. Näiden tilanteiden lukumäärä on suuruusluokkaa $q \times n \times |\Gamma|^n$, missä q on tilojen lukumäärä, n syötteen pituus ja $|\Gamma|$ aakkoston koko.

Edellä mainitulla tavalla rakennettu kone saadaan totaaliseksi liittämällä siihen laskuri, joka laskee suoritettujen askelten määrän. Helpoiten tämä käy siten, että tehdään koneesta kaksiurainen, jonka toiselle uralle kirjoitetaan askelten määrä binäärilukuna, jota kasvatetaan aina kunkin alkuperäisen koneen laskenta-askeleen yhteydessä yhdellä. Kun laskuri ylittää raja-arvon, sana hylätään, sillä tällöin kone on joutunut silmukkaan.

Lopuksi täytyy tarkistaa, että laskuri voidaan toteuttaa rikkomatta vaatimusta lineaarisesta tilankäytöstä. Luvun $q \times n \times |\Gamma|^n$ esittämiseen tarvitaan $k = \log_2(q) + \log_2(n) + n\log(|\Gamma|)$ bittiä, joka kasvaa lineaarisesti n:n kasvaessa. Vaikka tässä $k > n$, niin laskuri saadaan puristettua käytössä olevaan tilaan esittämällä se sopivassa kannassa.

7. **Tehtävä**: Osoita, että jokainen rajoittamattomalla kieliopilla tuotettava kieli voidaan tuottaa kieliopilla, jossa produktioiden vasemmalla puolella ei esiinny päätemerkkejä.

   Vastaus: Kieliopin G kanssa saman kielen tuottavan tehtävänannon mukaisen kieliopin G' voi muodostaa systemaattisesti siten, että jokaista kielessä esiintyvää päätemerkkiä a kohden otetaan uusi välike A_a ja tälle sääntö $A_a \to a$. Tällä uudella välikkeellä korvataan kaikki päätemerkit a, jotka esiintyvät G:n säännöissä.

   Seuraavaksi sama formaalimmin: Olkoon $G = (V, \Sigma, P, S)$ yleinen kielioppi. Muodostetaan kielioppi $G' = (V', \Sigma, P', S')$, missä

$$V' = V \cup \{A_a \mid a \in \Sigma\}$$

   Kukin kieliopin G produktio $r = x_1 \cdots x_n \to x_{n+1} \cdots x_{n+m}$, missä $x_i \in V$ muutetaan muotoon:

$$c(r) = y_1 \cdots y_n \to y_{n+1} \cdots y_{n+m}$$

   missä

$$y_i = \begin{cases} x_i, & x_i \in V - \Sigma \\ A_{x_i}, & x_i \in \Sigma \ . \end{cases}$$

   Nyt voidaan määritellä sääntöjoukko P' seuraavasti:

$$P' = \{c(r) \mid r \in P\} \cup \{A_a \to a \mid a \in \Sigma\} \ .$$

   Tarkastellaan uudelleen edellisessä tehtävässä esiintynyttä kielioppia:

$$\begin{aligned} S &\to aA \mid bB \\ aA &\to abB \mid ab \\ bB &\to baA \mid ba \end{aligned}$$

   Konstruktion tuloksena syntyy kielioppi:

$$\begin{aligned} S &\to A_aA \mid A_bB \\ A_aA &\to A_aA_bB \mid A_aA_b \\ A_bB &\to A_bA_aA \mid A_bA_a \\ A_a &\to a \\ A_b &\to b \end{aligned}$$

8. **Tehtävä**: Osoita, että jokainen yhteysherkkä kielioppi voidaan saattaa normaalimuotoon, jossa produktiot ovat muotoa $S \to \varepsilon$ tai $\alpha A \beta \to \alpha\omega\beta$, missä A on kieliopin välike ja $\omega \neq \varepsilon$. (S on tässä kieliopin lähtösymboli.)

   Vastaus: Normaalimuotoon saattamisessa on kolme vaihetta:

   i) Poistetaan alkusymboli S sääntöjen oikealta puolelta.

   ii) Poistetaan kaikki päätemerkit sääntöjen vasemmalta puolelta.

iii) Muutetaan väärää muotoa olevat säännöt oikeanlaisiksi.

Vaiheet määritellään seuraavasti:

i) Mikäli S esiintyy jonkin säännön oikealla puolella, niin lisätään kielioppiin uusi alkusymboli S' ja sääntö $S' \to S$. (Yhteysherkkien kielioppien määritelmän mukaan tällöin ei sääntöjoukossa voi olla sääntöä $S \to \varepsilon$, joten tätä tapausta ei tarvitse käsitellä.)

ii) Päätemerkkien poistaminen sääntöjen vasemmalta puolelta tapahtuu käyttäen tehtävän 5. vastauksessa esitettyä konstruktiota.

iii) Jokaista väärää muotoa olevaa sääntöä:

$$X_1 \cdots X_n \to Y_1 \cdots Y_m \ ,$$

missä $m \geq n$, kohden lisätään kielioppiin $n-1$ uutta välikettä $(Z_1, \ldots, Z_{n-1})$, ja sääntö korvataan joukolla sääntöjä:

$$\begin{aligned}
X_{n-1}X_n &\to Z_1X_n \\
Z_1X_n &\to Z_1Y_n \cdots Y_m \\
X_{n-2}Z_1 &\to Z_2Z_1 \\
Z_2Z_1 &\to Z_2Y_{n-1} \\
&\vdots \\
Z_{n-1}Z_{n-2} &\to Z_{n-1}Y_2 \\
Z_{n-1}Y_2 &\to Y_1Y_2
\end{aligned}$$

Tarkastellaan, miten iii)-kohta käsittelee sääntöä:

$$ABBA \to BAABA$$

Koska $n = 4$, tarvitaan kolme uutta välikettä: Z_1, Z_2 ja Z_3. Sääntöjoukoksi tulee:

$$\begin{aligned}
BA &\to Z_1A \\
Z_1A &\to Z_1BA \\
BZ_1 &\to Z_2Z_1 \\
Z_2Z_1 &\to Z_2A \\
AZ_2 &\to Z_3Z_2 \\
Z_3Z_2 &\to Z_3A \\
Z_3A &\to BA \ .
\end{aligned}$$

Tällöin vastaavaksi johdoksi tulee:

$$\begin{aligned}
AB\underline{BA} &\Rightarrow AB\underline{Z_1A} \Rightarrow A\underline{BZ_1}BA \Rightarrow A\underline{Z_2Z_1}BA \Rightarrow \underline{AZ_2}ABA \\
&\Rightarrow \underline{Z_3Z_2}ABA \Rightarrow \underline{Z_3A}ABA \Rightarrow BAABA \ .
\end{aligned}$$