Laskennan vaativuus **Ratkaisuehdotuksia**
Harjoitus 1, 20.3.2007

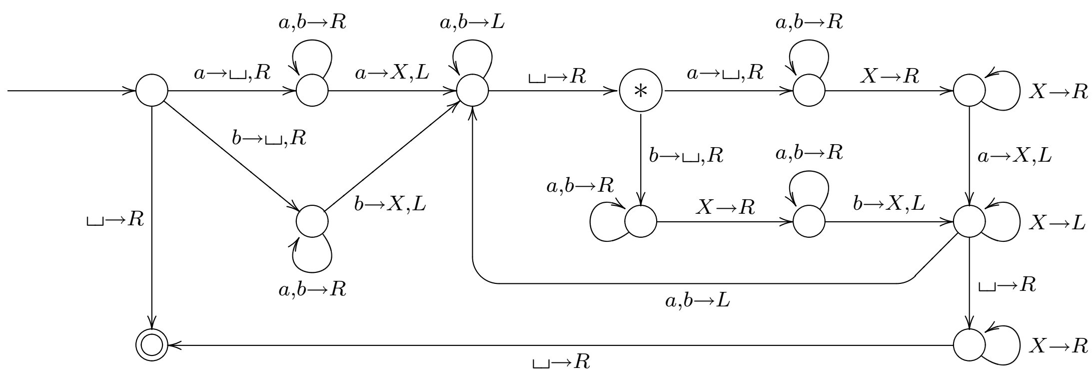
1. (a) Tosi: On olemassa sellaiset c ja n_0, joilla jokaisella $n \geq n_0$ pätee $2n \leq c \cdot n$. Tarkoitukseen soveltuvat arvot ovat nimittäin $c = 2$ ja $n_0 = 0$.
   (b) Epätosi: Vakion c valinnasta riippumatta $n^2 > cn$, kun $n > c$.
   (c) Epätosi: $n > \log^k n$ kaikilla $k > 1$ ja $n \geq 1$.
   (d) Tosi: $n \log n \leq n^2 = \mathcal{O}(n^2)$ kaikilla n
   (e) Tosi: $3^n = 2^{\log_2 3 \cdot n} = 2^{\mathcal{O}(n)}$
   (f) Tosi

2. Jokaista a:ta kohden etsitään c, minkä jälkeen palataan seuraavaan a:han. Tämä toistetaan $i \cdot (i + j)$ kertaa. Vastaavasti toimitaan b:n ja d:n kanssa, jolloin nauhaa kuljetaan edestakaisin $j \cdot (i + j)$ kertaa. Yhteensä aikaa siis kuluu

$$2 \cdot i \cdot (i + j) + 2 \cdot j \cdot (i + j) = 2(i + j)^2.$$

   Aikavaativuus on siten $\mathcal{O}(n^2)$ $(n \in \mathbb{N})$.

3. Tehtävän kuvausta vastaava epädeterministinen Turingin kone on esitetty Laskennan mallien luentomateriaalissa. Siirtymät hylkäävään tilaan on jätetty esittämättä:



Olkoon koneen saama syöte $x_0 x_1 \ldots x_{n-1}$. Aluksi päätetään epädeterministisesti nauhan kohta k, mistä kone aloittaa testaamaan, päteekö $x_0 \ldots x_{k-1} = x_k \ldots x_{n-1}$. Kohdan k valinnan jälkeen tiedetään, että $x_0 = x_k$. Jo tutkitut merkit alussa korvataan tyhjällä merkillä ja lopussa merkillä X. Kone etenee vasemmalle kunnes se löytää tyhjän ja sen oikeanpuoleisen merkin x_i, joka korvataan merkillä ␣. Seuraavaksi edetään oikealle, kunnes löytyy X:n oikealla puolella oleva merkki x_{i+k}. Jos $x_i \neq x_{i+k}$, niin syöte hylätään. Muuten syötteen läpikäynti jatkuu taas vasemmalle edettäessä ␣:n oikeanpuoleisesta merkistä. Jos kaikki syötteen merkit on korvattu, kone päätyy hyväksyvään tilaan. Kone hyväksyy myös tyhjän syötteen.

Selvästi syötteen hyväksymisen edellytyksenä on, että syötteen pituus n on parillinen ja $k = n/2$. Kaikissa muissa tapauksissa kone päätyy hylkäävään tilaan. Kieleen $\{ww | w \in \{a, b\}^*\}$ kuulumattomien syötteiden tunnistaminen on nopeampaa kuin kieleen kuuluvien, sillä suoritus ei tällöin pääse milloinkaan etenemään loppuun asti, vaan osa merkeistä jää aina korvaamatta. Aikavaatimusten selvittämiseksi riittää siis tutkia vain hyväksyttäviä syötteitä.

Ensimmäinen kierros nauhalla vie ajan $2k$. Kun kone on nauhan kohdassa i ja tilassa (*), se merkitse nauhalla olevan merkin, etenee oikealle loput $k-i-1$ syöteaakkoston merkkiä, $i-1$ X-merkkiä ja lopulta korvaa viimeisen X:n oikealla puolella olevan merkin. Seuraavaksi kone palaa vasemmalle $i-1$ X-merkkiä, $k-i$ syöteaakkoston merkkiä ja lopuksi päätyy takaisin tilaan (*) kulkemalla yhden merkin oikealle. Kone kulkee jokaisella kierroksella siis $2k$ askelta nauhalla. Esitetyn kaltainen toiminta suoritetaan $k-2$ kertaa. Lopussa siirrytään vasemmalle k askelta, mutta vastaan ei tulekaan syöteaakkoston merkkejä vaan $\sqcup$ kertomassa, että syötemerkkijonon alkuosa on kokonaan läpikäyty. Seuraavaksi kuljetaan oikealle k askelta, jotta huomataan että myöskään syötteen lopussa ei ole enää merkkejä. Koneen aikavaatimukseksi syötteen hyväksyvässä suorituksessa saadaan siis

$$\mathcal{O}(2k+(k-2)\cdot 2k+2k)=\mathcal{O}(2k^2)=\mathcal{O}(2(\frac{n}{2})^2)=\mathcal{O}(n^2).$$

4. **Kertolaskumenetelmä:**

   Oletetaan että koneelle annettu syöte on binäärimuodossa. Olkoon syötteen pituus n. Arvotaan kaksi lukua $a=a_0a_1\ldots a_{k-1}$ ja $b=b_0b_1\ldots b_{l-1}$, jossa $k+l\le n+1$. Suurempia lukuja ei tarvitse tutkia. Muodostetaan moninauhainen epädeterministinen Turingin kone, joka toimii seuraavasti.

   1. Muodosta binääriluku a epädeterministisesti nauhalle t_a. $\mathcal{O}(n)$
   2. Muodosta binääriluku b epädeterministisesti nauhalle t_b. $\mathcal{O}(n)$
   3. Tarkista, että nauhojen t_a ja t_b luvut ovat sallittuja (nauha ei ole tyhjä ja merkitsevin bitti ei ole nolla paitsi jos se on nauhan ainoa bitti). $\mathcal{O}(n)$
   4. Kerro luvut a ja b keskenään ja tallenna tulo nauhalle t_{ab}.
   5. Vertaa nauhan t_{ab} sisältöä syötteeseen. $\mathcal{O}(n)$
   6. Hyväksy tai hylkää vertailun perusteella.

   Kone toimii nyt muuten lineaarisessa ajassa, mutta valittu kertolaskun toteutus vaikuttaisi määräävän ajankäytön. Koulusta tuttu kertolaskumenetelmä onnistuu ajassa $\mathcal{O}(n^2)$. Nopein tunnettu menetelmä kertoa lukuja keskenään on Schönhage-Strassen -algoritmi, joka toimii ajassa $\mathcal{O}(n\log n\cdot\log\log n)$. Jos käytetään yksinkertaista kertolaskua, aikavaatimukseksi moninauhaiselle koneelle saadaan $\mathcal{O}(n^2)$. Suoraviivaisesti yksinauhaiseksi muunnettu kone kuluttaisi siis aikaa $\mathcal{O}(n^4)$.

   Jakolaskumenetelmä:

   (Pienillä syötteillä saatetaan joutua turvautumaan erikoistapauksiin.)

   1. Muodosta binääriluku $a\le n/2$ epädeterministisesti nauhalle t_a. $\mathcal{O}(n)$
   2. Lisää syötteenä annettuun lukuun 1 merkitsevimmäksi bitiksi (etumerkkibitiksi)
   3. Vähennä lukua a syötteestä kunnes syötteeksi tulee 0 tai etumerkkibitti muuttuu nollaksi.
   4. Jos syöte on nolla, hyväksy syöte.

   Vähennyslasku on toteutettavissa ajassa $O(n^2)$. Jos syöte on yhdistetty luku, sitä sovelletaan b kertaa, missä b on syöteluvun pienin jakaja (a on suurin). Ajankäytöltään pahin tapaus saadaan, kun syötteenä ei ole yhdistettyä lukua, vaan kone mm. vähentää luvulla kaksi $\lceil n/2\rceil$ kertaa. Aikavaatimukseksi tulee siis $\mathcal{O}(n^3)$.

5. **Yhdiste**:
Olkoot A ja B luokkaan $\mathbf{P}$ kuuluvia kieliä. On siis olemassa polynomisessa ajassa kielen A hyväksyvä Turingin kone M_A ja vastaava kone M_B kielelle B. Muodostetaan kielen $A \cup B$ tunnistava kone M. Ensin M muodostaa syötteestä x kopion polynomisessa ajassa $g(|x|)$ nauhan loppuun. Alkuperäisen syötteen ja kopion väliin sijoitetaan erityinen merkki #, joka ei sisälly syöteaakkostoon. Koska kielelle A on olemassa polynimisessa ajassa toimiva kone M_A, niin on olemassa polynomisessa ajassa toimiva kone M'_A, joka toimii muuten kuin M_A, mutta se suhtautuu merkkiin # kuin nauhan alkuun. Sovitaan että koneen M'_A suoritusaikaa edustaa polynomifunktio f_A. Kone M_B toimii puolestaan ajassa f_B.

   Kone M tutkii ensin, hyväksyykö M'_A kopioidun syötteen. Jos M'_A päätyy hyväksyvään tilaan, myös M hyväksyy syötteen. Jos M'_A päätyy hylkäävään tilaan, poistetaan syötteen kopio nauhan lopusta polynomisessa ajassa $g(|x|)$ ja tutkitaan hyväksyykö M_B syötteen. Tällöin kone M hyväksyy syötteen vain jos M_B päätyy hyväksyvään tilaan. Selvästi kone M hyväksyy tai hylkää syötteen x ajassa $\mathcal{O}(f_A(|x|) + f_B(|x|) + 2g(|x|)) \in \mathbf{TIME}(\mathbf{P})$ ja kieli $A \cup B \in \mathbf{P}$.

   Komplementti:
Olkoon M_A kielen A polynomisessa ajassa tunnistava kone. Komplementtikielen $\overline{A}$ tunnistava kone $M_{\overline{A}}$ voidaan muodostaa yksinkertaisesti tutkimalla, mihin tilaan M_A päätyy. Jos M_A hyväksyy syötteen, $M_{\overline{A}}$ hylkää sen. Jos M_A puolestaan päätyy hylkäävään tilaan, $M_{\overline{A}}$ hyväksyy syötteen. Kone $M_{\overline{A}}$ toimii selvästi polynomisessa ajassa. Siksi myös $\overline{A} \in \mathbf{P}$.

   Peräkkäinen yhdistäminen:
Oletetaan jälleen, että kielet A ja B kuuluvat luokkaan $\mathbf{P}$ ja niiden tunnistamiseen ovat olemassa Turingin koneet M_A ja M_B. Kielen AB tunnistavan koneen M tulee tunnistaa, onko syötteenä annettu merkkijono x muotoa $x = ab$, jossa $a \in A$ ja $b \in B$. Merkkijonosta $x_0 x_1 \ldots x_{n-1}$ voidaan muodostaa $n+1$ merkkijonoparia tyhjät alut ja loput mukaan luettuna.

   On olemassa polynomisessa ajassa toimivat koneet M'_A ja M'_B, jotka toimivat kuten M_A ja M_B, mutta kohtelevat merkkiä # nauhan alun tavoin. Olkoon näiden koneiden aikavaativuus f_A ja f_B. Nyt jokaisessa vaiheessa haluttu osa syötteestä kopioidaan nauhan loppuun vertailua varten. Kopiointiin ja tyhjentämiseen kuluu polynominen aika $g(n)$. Aloitetaan tutkimalla hyväksyykö $M'_A(\epsilon)$. Jos M'_A päätyy hyväksyvään tilaan, tutkitaan $M'_B(x_0 \ldots x_{n-1})$. Jos M'_B ei hyväksy syötettään, M hylkää syötteen ja suoritusta jatketaan tutkimalla $M'_A(x_0)$ ja $M'_B(x_1 \ldots x_{n-1})$. Tätä jatketaan kunnes molemmat koneet hyväksyvät syötteensä, jolloin M hyväksyy syötteen, tai merkkijonoparit loppuvat, jolloin M hylkää syötteen. Aikaa kuluu nyt $\mathcal{O}((n+1)(f_A(n) + f_B(n) + 4g(n))) \in \mathbf{TIME}(\mathbf{P})$ ja $AB \in \mathbf{P}$.