Luku 3

Säännölliset kielet ja äärelliset automaatit

Tässä luvussa tutustumme yksinkertaisimpaan formaalien kielten luokkaan, säännöllisiin kieliin. Säännölliset kielet voidaan kuvata säännöllisten lausekkeiden avulla ja niitä vastaavat (päätös-)ongelmat voidaan ratkaista äärellisillä automaateilla.

Ongelma: Ohjelmointikielten muuttujat, vakiot jne. ovat tietynmuotoisia, ohjelmointikielen *syntaksin* ("oikeankirjoitusopin") mukaisia. Esim. 0.123 on luku, mutta 0.1.2.3 tai $z.33$ eivät ole. Mutta miten määritellä lukujen syntaksi? Entä miten voidaan tunnistaa täsmälleen oikeanmuotoiset merkkijonot? Osoittautuu, että oikeanmuotoiset merkkijonot voidaan kuvata säännöllisillä lausekkeilla, ja niiden tunnistus voidaan suorittaa äärellisellä automaatilla.

Tunnet ehkä UNIX:in käskyn `grep`, jolla voidaan etsiä tekstistä hahmoja. Esim.

`grep [A-Z][a-z]*[0-9] teksti` etsii tiedostosta `teksti` rivit, joilla on määrittelyn muotoisia sanoja: ensin suuri kirjain, sitten mielivaltainen määrä pieniä kirjaimia ja lopuksi numero. `grep` etsii nimenomaan säännöllisiä lausekkeita, ja nimi onkin lyhenne sanoista "Global search for Regular Expression and Print"

3.1 Säännölliset lausekkeet ja kielet

- Esim. Hyväksy merkkijonot, joissa esiintyy sana "kissa". Ts. merkkijonot ovat muotoa

  [0 tai useampia kirjaimia]$kissa$[0 tai useampia kirjaimia]

- Esim. Etsi tekstitiedostosta osoitteita, jotka ovat muotoa
  [$x$katu tai $x$tie][$numero$] [mahd. rapun kirjain]
  [mahd. asunnonnumero][postinumero][kunnan nimi]

- Kuinka esittää kompaktisti sallitut merkkijonot (ts. kuvata *kieli*, jonka tunnistusohjelma hyväksyy)?

- Määritellään avuksi kolme operaatiota kielten yhdistämiseen: Olkoot A ja B aakkoston Σ kieliä. Tällöin

  - A:n ja B:n *yhdiste* on kieli

    $$A \cup B = \{x \in \Sigma^* \mid x \in A \text{ tai } x \in B\}$$

  - A:n ja B:n *tulo* on kieli

    $$AB = \{xy \in \Sigma^* \mid x \in A,\ y \in B\}$$

  - A:n *potenssit* A^k, $k \geq 0$, määritellään iteratiivisesti:

    $$\begin{cases} A^0 &= \{\epsilon\}, \\ A^k &= AA^{k-1} \\ &= \{x_1 \ldots x_k \mid x_i \in A \quad \forall i = 1, \ldots, k\} \qquad (k \geq 1) \end{cases}$$

  - A:n *sulkeuma* on kieli

    $$\begin{aligned} A^* &= \bigcup_{k=0}^{\infty} A^k \\ &= \{x_1 \ldots x_k \mid k \geq 0,\ x_i \in A \quad \forall i = 1, \ldots, k\} \end{aligned}$$

- *Määritelmä:* Aakkoston Σ *säännölliset lausekkeet* (engl. regular expression) määritellään induktiivisesti säännöillä:
  (i) $\boldsymbol{\emptyset}$ ja $\boldsymbol{\epsilon}$ ovat Σ:n säännöllisiä lausekkeita;
  (ii) $\boldsymbol{a}$ on Σ:n säännöllinen lauseke kaikilla $a \in \Sigma$;
  (iii) jos r ja s ovat Σ:n säännöllisiä lausekkeita,
  niin $(r \cup s)$, (rs) ja r^* ovat Σ:n säännöllisiä lausekkeita;
  (iv) muita Σ:n säännöllisiä lausekkeita ei ole

- Kukin Σ:n säännöllinen lauseke r *kuvaa* kielen $L(r)$:
  (i) $L(\boldsymbol{\emptyset}) = \emptyset$;
  (ii) $L(\boldsymbol{\epsilon}) = \{\epsilon\}$;
  (iii) $L(\boldsymbol{a}) = \{a\}$ kaikilla $a \in \Sigma$;
  (iv) $L((r \cup s)) = L(r) \cup L(s)$;
  (v) $L((rs)) = L(r)L(s)$;
  (vi) $L(r^*) = (L(r))^*$

- Aakkoston $\{a, b\}$ säännöllisiä lausekkeita:

$$r_1 = \boldsymbol{((ab)b)}, \quad r_2 = \boldsymbol{(ab)^*},$$
$$r_3 = \boldsymbol{(ab^*)}, \quad r_4 = \boldsymbol{(a(b \cup (bb)))^*}.$$

  Lausekkeiden kuvaamat kielet:

$$\begin{aligned}
L(r_1) &= (\{a\}\{b\})\{b\} = \{ab\}\{b\} = \{abb\};\\
L(r_2) &= \{ab\}^* = \{\epsilon, ab, abab, ababab, \ldots\}\\
&= \{(ab)^i \mid i \geq 0\};\\
L(r_3) &= \{a\}(\{b\})^* = \{a, ab, abb, abbb, \ldots\}\\
&= \{ab^i \mid i \geq 0\};\\
L(r_4) &= (\{a\}\{b, bb\})^* = \{ab, abb\}^*\\
&= \{\epsilon, ab, abb, abab, ababb, \ldots\}\\
&= \{x \in \{a, b\}^* \mid \text{ kutakin } a\text{-kirjainta } x\text{:ssä}\\
&\quad \text{seuraa 1 tai 2 } b\text{-kirjainta } \}
\end{aligned}$$

- Sulkumerkkien vähentämissääntöjä:
  - Operaattoreiden prioriteetti:

$$\boldsymbol{*} \quad \succ \quad \cdot \quad \succ \quad \cup$$

- Yhdiste- ja tulo-operaatioiden assosiatiivisuus:

$$\begin{aligned} L(((r \cup s) \cup t)) &= L((r \cup (s \cup t))) \\ L(((rs)t)) &= L((r(st))) \end{aligned}$$

  $\Rightarrow$ peräkkäisiä yhdisteitä ja tuloja ei tarvitse suluttaa

- Käytetään tavallisia kirjasimia, mikäli sekaannuksen vaaraa merkkijonoihin ei ole.
  Yksinkertaisemmin siis:

$$r_1 = abb, \quad r_2 = (ab)^*, \quad r_3 = ab^*, \quad r_4 = (a(b \cup bb))^*$$

- *Määritelmä:* Kieli on *säännöllinen*, jos se voidaan kuvata säännöllisellä lausekkeella

- Esim. Olkoon aakkosto $\Sigma = \{a, b, c, ..., \}$. Hyväksytään merkkijonot, jotka ovat muotoa

$$l^* \text{kissa} l^*,$$

  missä l on lyhenne lausekkeelle $l = (\text{a} \cup \text{b} \cup ... \cup \text{ö})$ (ts. $l^* \in \Sigma^*$)

- Esim. 2: Olkoon $\Sigma = \{A, B, ..., , a, b, ..., , 0, 1, 2, ..., 9\}$. Osoite on muotoa

$$(Ll^*)(\text{katu } \cup \text{tie }) dd^* (l \cup \epsilon)(dd^* \cup \epsilon)\ ddddd\ Ll^*,$$

  missä d on lyhenne lausekkeelle

$$d = (0 \cup 1 \cup ... \cup 9)$$

  ja L on lyhenne lausekkeelle

$$l = (\text{A} \cup \text{B} \cup ... \cup \text{Ö}).$$

- Esim. 3 C-kielen etumerkittömät liukuluvut (float, double, long double) määritellään seuraavasti:

  - (kokonaisosa).(desimaaliosa) (e tai E) [+ tai −] (eksponetti) [suffiksi]
  - kokonaisosa ja desimaaliosa koostuvat digiteistä
  - joko kokonaisosa tai desimaaliosa voi puuttua (mutta eivät molemmat)

  - joko (i) desimaalipiste tai (ii) (e tai E) ja eksponentti voivat puuttua (mutta eivät molemmat)
  - suffiksi: F tai f: float, L tai l: long double, muuten double

- Etumerkittömät liukuluvut tunnistava kieli voidaan määritellä säännöllisellä lausekkeella (ilman suffikseja):
  number $= (d^+.d^* \cup .d^+)(\epsilon \cup ((e \cup E)(+ \cup - \cup \epsilon)d^+)) \cup$
  $d^+(e \cup E)(+ \cup - \cup \epsilon)d^+$

  Vai? number $= d^*(.dd^*) \cup$
  $d^*(.dd^* \cup \epsilon)(\mathrm{E} \cup \mathrm{e})(+ \cup - \cup \epsilon)dd^* \cup$
  $dd^*(\mathrm{E} \cup \mathrm{e})(+ \cup - \cup \epsilon)dd^*$

- Kieleen kuuluvat esim. seuraavat merkkijonot: 12., .12, 1.2, 1.2E3, 1.2e3, 1.2E-3,1E2, 1e23

3.1.1 Säännöllisten lausekkeiden sieventäminen

- Säännöllisillä kielillä on yleensä useita vaihtoehtoisia kuvauksia, esim.:

$$\begin{aligned} \Sigma^* &= L((a \cup b)^*) \\ &= L((a^*b^*)^*) \\ &= L(a^*b^* \cup (a \cup b)^*ba(a \cup b)^*). \end{aligned}$$

- *Määritelmä:* Säännölliset lausekkeet r ja s ovat *ekvivalentit*, merk. $r = s$, jos $L(r) = L(s)$

- Lisäksi merkitään $r \subseteq s$ tarkoittamaan $L(r) \subseteq L(s)$

- Lausekkeen sieventäminen = "yksinkertaisimman" ekvivalentin lausekkeen määrittäminen

- Huom: Usein merkitään $r^+ = rr^* = r^*r$

3.1.2 Sievennyssääntöjä

$$
\begin{aligned}
r \cup r &= r \text{ (mutta } rr \neq r, \text{ kun } r \neq \emptyset, \epsilon) \\
r \cup (s \cup t) &= (r \cup s) \cup t \\
r(st) &= (rs)t \\
r \cup s &= s \cup r \\
r(s \cup t) &= rs \cup rt \\
(r \cup s)t &= rt \cup st \\
\emptyset^* &= \epsilon \\
\emptyset r &= \emptyset \text{ (mutta } \emptyset \cup r = r) \\
\epsilon r &= r \text{ (mutta } \epsilon \cup r \neq r, \text{ kun } r \neq \epsilon) \\
r^* &= r^* r \cup \epsilon = r^+ \cup \epsilon \\
r^* &= (r \cup \epsilon)^* \\
(r^*)^* &= r^*
\end{aligned}
$$

- Lisäksi pätee päättelysääntö:

  Jos $r = rs \cup t$, niin $r = ts^*$, kun $\epsilon \notin L(s)$

- $L(r) = L(s) \Leftrightarrow L(r) \subseteq L(s) \wedge L(s) \subseteq L(r)$ eli $r = s \Leftrightarrow r \subseteq s \wedge s \subseteq r$

- Esim. yllä:

  1. $(a \cup b) \subseteq (a^*b^*) \Rightarrow (a \cup b)^* \subseteq (a^*b^*)^*$
  2. $((a^*b^*)^*) \subseteq a^*b^* \cup (a \cup b)^*ba(a \cup b)^*$:
     - jos muotoa a^*b^*, niin selvä
     - muuten sisältää osajonon ba
  3. $a^*b^* \cup (a \cup b)^*ba(a \cup b)^*) \subseteq (a \cup b)^*$, sillä $(a \cup b)^*$ kuvaa kaikki Σ:n merkkijonot

- Voidaan todistaa seuraavat sulkeumaominaisuudet: Jos L ja M ovat säännöllisiä kieliä, myös

  1. $L \cap M$ (kielten leikkaus)
  2. $\overline{L} = \Sigma^* \setminus L$ (kielen komplementti)
  3. $L^R = \{w^R | w \in L\}$ (kielen käänteiskieli, jossa sanat on kirjoitettu takaperin)

  ovat säännöllisiä

Harjoituksia säännöllisistä lausekkeista

1. UNIX:in egrep-komennolla (extended grep) voi etsiä tekstistä hahmoja, jotka on määritelty säännöllisinä lausekkeina. egrepin perussyntaksi on seuraava: `egrep <lauseke> <tiedosto>`, missä lauseke voi olla

   - hakasuluissa lista merkkejä, esim. $[abcd]$: mikä tahansa merkeistä a, b, c, d
   - $(lauseke)(lauseke)$: kahden lausekkeen katenaatio
   - $(lauseke1)|(lauseke2)$: joko lauseke1 tai lauseke2
   - $(lauseke)*$: lauseke toistuu 0 kertaa tai useampia kertoja (sulkeuma)
   - $\backslash b$: tyhjä merkki sanan reunassa $\backslash B$: tyhjä merkki sanan keskellä

   Huom! Lauseke kannattaa laittaa hipsuihin ('lauseke'). Lisää tietoa komennolla man egrep.

   Testaa egrep-komentoa! Voit käyttää syötetiedostona mitätahansa (C-)ohjelmakoodia, esim. kotisivun tiedostoa esim.c.

   Millaisia hahmoja etsivät seuraavat komennot:
   egrep '[1]*'
   egrep '[1][0]'
   egrep '[1]|[0]'

2. Millaisella egrep-komennolla löydät seuraavat rivit:
   a) Rivit, joilla on numeroita
   b) Rivit, joilla esiintyy sana $while$ tai for
   c) Rivit, joilla esiintyy numero 10
   d) Rivit, joilla esiintyy kokonaislukuja. (Huom! Komentosi ei siis saa hyväksyä desimaalilukuja.)

3. Tarkastellaan seuraavia aakkoston $\Sigma = \{a, b\}$ kieliä. Anna kustakin kielestä kaksi merkkijonoa, jotka kuuluvat kieleen, ja kaksi, jotka eivät kuulu kieleen!
   a) a^*b^*
   b) $a(ba)^*b$
   c) $a^* \cup b^*$
   d) $(aaa)^*$
   e) $(\epsilon \cup a)b$
   f) $\Sigma^*a\Sigma^*a\Sigma^*a\Sigma^*$

4. Mitä merkkijonoja kuuluu seuraavaan lausekkeen kuvaamaan kieleen?
   $(c \cup h \cup m \cup r)at((c \cup t)a \cup (s \cup t)o)ught(m \cup l \cup tw \cup r)ice$

5. Kuinka määrittelet säännöllisen lausekkeen, jolla löydät tietoa lumimyrskystä? Huomaa, että ilmaisu voi esiintyä myös taipuneena (sanavartalo ei onneksi taivu) tai yhdistettynä, esim. ”lumi- ja ukkosmyrskyvaroitus”.

6. Mitä merkkijonoja kuuluu kieleen $L(\emptyset^*)$? Entä $L(\epsilon^*)$?

7. Etsi lyhin merkkijono, joka kuuluu seuraavan lausekkeen kuvaamaan kieleen!

   a) $a^*(b \cup abb)b^*b$
   b) $a^*b^*b(a \cup (ab)^*)^*b^*$
   c) $(a \cup ab)(a^* \cup ab)^*b$

8. Muodosta seuraavia kieliä vastaavat säännölliset lausekkeet:
   a) $\{w \in \{a,b\}^* | w$:n kolmanneksi viimeinen merkki on $a\}$
   b) $\{w \in \{a,b\}^* | w$ sisältää joko merkkijonon ab tai $ba\}$
   c) $\{w \in \{a,b\}^* | w$ sisältää merkkijonon aba mutta ei merkkijonoa $bab\}$

9. Muodosta seuraavia kieliä vastaavat säännölliset lausekkeet:
   a) $\{w \in \{a,b\}^* | w$ sisältää parillisen määrän merkkiä $a\}$
   b) $\{w \in \{a,b\}^* | w$:n pituus on pariton$\}$
   c) $\{w \in \{a,b\}^* | w$:n sisältämien b-merkkien lukumäärä on kolmella jaollinen$\}$

10. Esitä yksikertaisemmassa muodossa seuraavat lausekkeet! (s.e. ne yhä generoivat saman kielen!)
    a) $(0 \cup 1 \cup 01 \cup 11)^*$
    b) $(0^* \cup 10^*)^*$
    c) $1^*(011^*)^* \cup 1^*(011^*)^*0$

3.2 Äärelliset automaatit

- Ongelma: Kahviautomaatti, joka ei anna vaihtorahaa, hyväksyy vain 50 sentin ja yhden euron kolikoita ja minimimaksu on 2 euroa. Millaisia syötejonoja kahviautomaatti hyväksyy?

- Kelvollisia syötejonoja ovat esim. seuraavat (yksikkönä snt):
  50 + 50 + 50 + 50
  100 + 100
  50 + 100 +100
  100 + 50 + 50 + 100

- Ts. kahviautomaatti hyväksyy syötejonot, jotka ovat muotoa

  ```
  1 euro +  1 euro +
  [0 tai useampia 50 sentin tai 1 euron kolikoita]
  ```

  tai

  ```
  1 euro +  50 senttiä +
  [1 tai useampia 50 sentin tai 1 euron kolikoita]
  ```

  tai

  ```
  50 senttiä + 1 euro +
  [1 tai useampia 50 sentin tai 1 euron kolikoita]
  ```

  tai

  ```
  50 senttiä +  50 senttiä + 1 euro +
  [0 tai useampia 50 sentin tai 1 euron kolikoita]
  ```

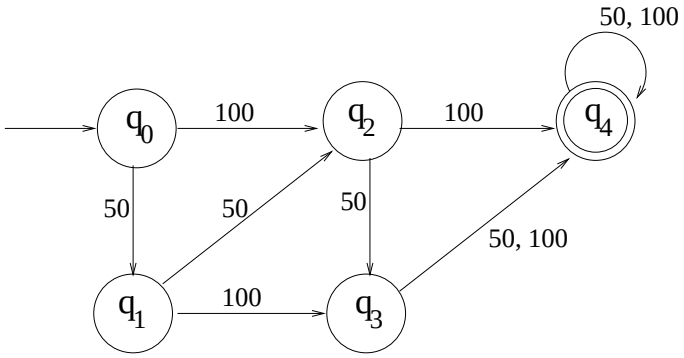
  tai

  ```
  50 senttiä + 50 senttiä + 50 senttiä +
  [1 tai useampia 50 sentin tai 1 euron kolikoita]
  ```

- Kahviautomaatin toiminta voidaan kuvata *äärellisenä automaattina*

- Automaatin syötteitä ovat 50 sentin ja 1 euron kolikot ja automaatti hyväksyy ”syötejonon”, jos siihen sisältyvien rahojen summa on vähintään 2 euroa

- Automaatti voidaan esittää *tilasiirtymäkaaviona*



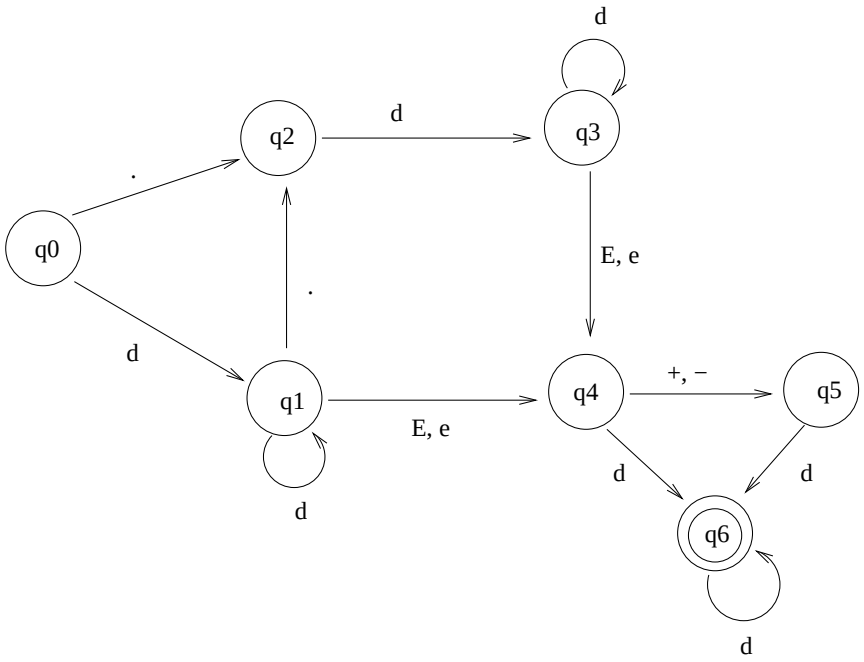
3.2.1 Äärellisen automaatin esitys

- tilasiirtymäkaavio

- tilasiirtymätaulukko

	50 snt	1 euro
$\rightarrow q_0$	q_1	q_2
q_1	q_2	q_3
q_2	q_3	q_4
q_3	q_4	q_4
$\leftarrow q_4$	q_4	q_4

- Esimerkki: C-kielen mukaisen etumerkittömän liukuluvun tunnistava automaatti:



– Tilasiirtymätaulukkona:

		d	.	E, e	+, −
→	q_0	q_1	q_2		
	q_1	q_1	q_3	q_4	
	q_2	q_3			
←	q_3	q_3		q_4	
	q_4	q_6			q_5
	q_5	q_6			
←	q_6	q_6			

Tässä d $= \{0, 1, \ldots, 9\}$. Taulukon puuttuvat kohdat vastaavat virhetilaa "Error", joka käytännössä jätetään merkitsemättä selkeyden takia

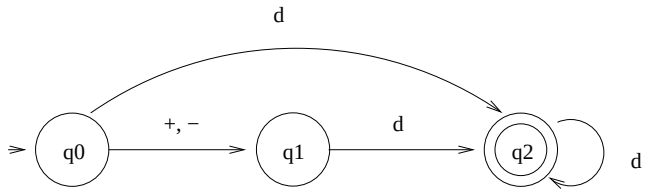
– Ohjelmana:

```
int IsDigit(char c); /* palauttaa 1, jos c on numero, 0 muuten */

int q = 0
char c while ( (c = fgetc(stdin))!=EOF )
{
   switch ( q )
   {
```

```
        case 0: if (IsDigit(c) ) q = 1;
                else if (c=='.') q = 2 else q = 99;
                break;
        case 1: if (IsDigit(c) ) q = 1;
                else if (c=='.') q=3;
                else if (c=='e' || c=='E') q=4; else q=99;
                break;
        case 2: if (IsDigit(c)) q=3; else q=99;
                break;
        case 3: if (IsDigit(c)) q=3;
                else if (c=='e' || c=='E') q = 4 else q = 99;
                break;
        case 4: if (IsDigit(c)) q=6;
                else if (c=='+' || c=='-') q = 5 else q = 99;
                break;
        case 5: if (IsDigit(c)) q=6; else q = 99;
                break;
        case 6: if (IsDigit(c)) q=6; else q = 99;
                break;
        case 99: break;
    }
}
if ( q == 3 || q == 6 ) printf("luku OK!");
else printf("Virheellinen luku");
```

- Äärellisen automaatin pohjalta laadittuun ohjelmaan voidaan liittää myös semanttisia toimintoja
- Esim. Etumerkillisen kokonaisluvun tunnistaminen



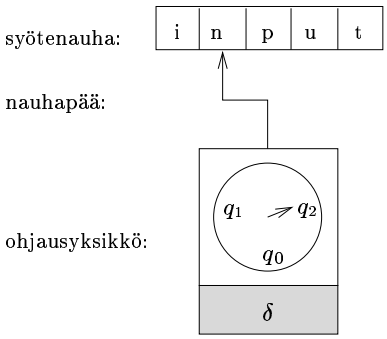
- Vastaava ohjelma, joka lisäksi evaluoi luvun arvon:

```
int IsDigit(char c); /* palauttaa 1, jos c on numero, 0 muuten */
```

```
int q = 0
char c int sign = 1; int val = 0; while ( (c = fgetc(stdin))!=EOF )
{
   switch ( q )
   {
      case 0: if (c=='+' || c=='-') {
                q=1;
                if (c=='-') sign = -1;
                }
                else if (IsDigit(c) {
                q=2;
                val = c -' 0';
                }
                break;
      case 1: if (IsDigit(c) {
                q=2;
                val = c -' 0';
                }
                else q=99;
                break;
      case 2: if (IsDigit(c)) {
                q=2;
                val = 10 * val + (c -' 0');
                }
                else q=99;
                break;
      case 99: break;
   }
}
if ( q == 2) printf("luvun arvo on %d", sgn * val);
else printf("Virheellinen luku");
```

3.2.2 Äärellisen automaatin formaali määrittely

- Äärellinen automaatti M
  - äärellistilainen *ohjausyksikkö*, jonka toimintaa säätelee automaatin *siirtymäfunktio* δ
  - merkkipaikkoihin jaettu *syötenauha*



  - *nauhapää*, joka kullakin hetkellä osoittaa yhtä syötenauhan merkkiä

- Automaatin "toiminta":
  - Automaatti käynnistetään erityisessä *alkutilassa* q_0, siten että tarkasteltava syöte on kirjoitettuna syötenauhalle ja nauhapää osoittaa sen ensimmäistä merkkiä
  - Yhdessä toiminta-askelessa automaatti lukee nauhapään kohdalla olevan syötemerkin, päättää ohjausyksikön tilan ja luetun merkin perusteella siirtymäfunktion mukaisesti ohjausyksikön uudesta tilasta, ja siirtää nauhapäätä yhden merkin eteenpäin
  - Automaatti pysähtyy, kun viimeinen syötemerkki on käsitelty. Jos ohjausyksikön tila tällöin kuuluu erityiseen *(hyväksyvien) lopputilojen* joukkoon, automaatti *hyväksyy* syötteen, muuten *hylkää* sen
  - Automaatin *tunnistama kieli* on sen hyväksymien merkkijonojen joukko

- *Määritelmä: Äärellinen automaatti* (engl. finite automaton) on viisikko

$$M = (Q, \Sigma, \delta, q_0, F),$$

  missä
  - Q on automaatin *tilojen* äärellinen joukko;
  - Σ on automaatin *syöteaakkosto*;
  - $\delta : Q \times \Sigma \to Q$ on automaatin *siirtymäfunktio*;
  - $q_0 \in Q$ on automaatin *alkutila*;
  - $F \subseteq Q$ on automaatin *(hyväksyvien) lopputilojen* joukko.

- Esim. reaalilukuautomaatin formaali esitys:

$$\begin{aligned} M \;=\; & (\{q_0, \ldots, q_6, error\}, \{\texttt{0,1,...,9,.,E,e,+,-}\}, \\ & \delta, q_0, \{q_3, q_6\}), \end{aligned}$$

missä δ on kuten aiemmin taulukossa; esim.

$$\delta(q_0, \texttt{0}) = \delta(q_0, \texttt{1}) = \cdots = \delta(q_0, \texttt{9}) = q_1,$$

$$\delta(q_0, .) = q_2, \quad \delta(q_0, E) = error, \quad \delta(q_1, \texttt{E}) = q_4 \quad \text{jne.}$$

- Automaatin *tilanne* on pari $(q, w) \in Q \times \Sigma^*$
  - automaatin *alkutilanne syötteellä* x on pari (q_0, x)
  - q on nykyinen tila ja w on syötemerkkijonon käsittelemätön osa
- Tilanne (q, w) *johtaa suoraan* tilanteeseen (q', w'), merk.

  $$(q, w) \underset{M}{\vdash} (q', w'),$$

  jos on $w = aw'$ $(a \in \Sigma)$ ja $q' = \delta(q, a)$.
  Tilanne (q', w') on tilanteen (q, w) *välitön seuraaja*
- Tilanne (q, w) *johtaa tilanteeseen* (q', w') eli tilanne (q', w') on tilanteen (q, w) *seuraaja*, merk.

  $$(q, w) \underset{M}{\vdash^*} (q', w'),$$

  jos on olemassa välitilannejono (q_0, w_0), (q_1, w_1), $\ldots$, (q_n, w_n), $n \geq 0$, siten että

  $$(q, w) = (q_0, w_0) \underset{M}{\vdash} (q_1, w_1) \underset{M}{\vdash} \cdots \underset{M}{\vdash} (q_n, w_n) = (q', w').$$

  Erikoistapaus: $n = 0$, $(q, w) \underset{M}{\vdash^*} (q, w)$ millä tahansa tilanteella (q, w)
- Automaatti M *hyväksyy* merkkijonon $x \in \Sigma^*$, jos on voimassa

  $$(q_0, x) \underset{M}{\vdash^*} (q_f, \epsilon) \qquad \text{jollakin } q_f \in F;$$

  muuten M *hylkää* x:n. Ts. automaatti hyväksyy x:n, jos sen alkutilanne syötteellä x johtaa johonkin hyväksyvään lopputilanteeseen
- *Määritelmä:* Automaatin M *tunnistama kieli*

  $$L(M) = \{x \in \Sigma^* \mid (q_0, x) \underset{M}{\vdash^*} (q_f, \epsilon) \quad \text{jollakin } q_f \in F\}$$

- Esim. merkkijonon "`0.25E2`" käsittely edellä esitetyllä liukulukuautomaatilla:

  $$\begin{array}{lllll} (q_0, \texttt{0.25E2}) & \vdash & (q_1, \texttt{.25E2}) & \vdash & (q_3, \texttt{25E2}) \\ & \vdash & (q_3, \texttt{5E2}) & \vdash & (q_3, \texttt{E2}) \\ & \vdash & (q_4, \texttt{2}) & \vdash & (q_6, \epsilon). \end{array}$$

  Koska $q_6 \in F = \{q_3, q_6\}$, on siis $\texttt{0.25E2} \in L(M)$

3.3 Automaattien minimointi

- Kaksi automaattia, jotka tunnistavat täsmälleen saman kielen ovat keskenään *ekvivalentteja*
- Äärellinen automaatti on *minimaalinen* jos se on tilamäärältään pienin ekvivalenttien automaattien joukossa
- Automaatti, jossa on enemmän tiloja kuin ekvivalentissa minimaalisessa automaatissa on *redundantti*
- Automaatteja muodostava algoritmit eivät aina tuota minimaalista automaattia
- On helpompi nähdä mikä on minimaalisen automaatin tunnistama kieli kuin redundantin automaatin tunnistama kieli
- On turha tallettaa ylimääräisiä tiloja
- Minimaalisen automaatin käsittely on tehokkaampaa kuin redundantin automaatin

3.3.1 Apukäsitteitä

- Määritellään M:lle laajennettu siirtymäfunktio δ^*, joka voi saada parametrinaan merkkijonon:
  jos $q \in Q$, $x \in \Sigma^*$, niin

$$\delta^*(q, x) = \text{ se } q' \in Q, \text{ jolla } (q, x) \vdash_M^* (q', \epsilon)$$

- Tilojen ekvivalenssi: M:n tilat q ja q' ovat *ekvivalentit*, merk.

$$q \equiv q',$$

  jos kaikilla $x \in \Sigma^*$ on

$$\delta^*(q, x) \in F \quad \text{jos ja vain jos} \quad \delta^*(q', x) \in F$$

  (ts. jos automaatti q:sta ja q':sta lähtien hyväksyy täsmälleen samat merkkijonot)

- *k-ekvivalenssi*: tilat q ja q' ovat k-ekvivalentit, merk.

$$q \stackrel{k}{\equiv} q',$$

jos kaikilla $x \in \Sigma^*$, $|x| \leq k$, on

$$\delta^*(q, x) \in F \quad \text{jos ja vain jos} \quad \delta^*(q', x) \in F$$

(ts. jos mikään enintään k:n pituinen merkkijono ei pysty erottamaan tiloja toisistaan)

- Selvästi pätee:

$$\begin{array}{lllll} \text{(i)} & q \stackrel{0}{\equiv} q', & \text{joss} & \text{sekä } q \text{ että } q' \text{ ovat lopputiloja} \\ & & & \text{tai kumpikaan ei ole; ja} \\ \text{(ii)} & q \equiv q', & \text{joss} & q \stackrel{k}{\equiv} q' \text{ kaikilla } k = 0, 1, 2, \ldots \end{array}$$

- Minimoinnin idea: syötteenä annetun automaatin tilojen k-ekvivalenssiluokkia ositetaan $(k+1)$-ekvivalenssiluokiksi kunnes saavutetaan täysi ekvivalenssi

3.3.2 Äärellisen automaatin minimointi -algoritmi

- Syöte: Äärellinen automaatti $M = (Q, \Sigma, \delta, q_0, F)$.

1. (Turhien tilojen poisto) Poista M:stä kaikki tilat, joita ei voida saavuttaa tilasta q_0 millään syötemerkkijonolla.
2. (0-ekvivalenssi) Osita M:n jäljelle jääneet tilat kahteen luokkaan: ei-lopputiloihin ja lopputiloihin.
3. (k-ekvivalenssi $\rightarrow$ $(k+1)$-ekvivalenssi)

```
while not(tilasiirtymäfkt yhteensopiva luokkajaon kanssa) {
    jaa luokan sisällä eri tavalla käyttäytyvät tilat eri luokkiin;
}
return M̂ = (Q̂, Σ, δ̂, q̂0, F̂),
```

missä

- $\hat{Q}$=M:n tilaluokat
- $\hat{\delta}$=luokkien välinen siirtymäfunktio
- $\hat{q}_0$=M:n alkutilan luokka

  - $\widehat{F}$=M:n lopputilojen luokat

- Lopputulos
  - M:n kanssa ekvivalentti äärellinen automaatti $\widehat{M}$, jossa on minimimäärätiloja
  - $\widehat{M}$ on tilojen nimeämistä vaille yksikäsitteinen

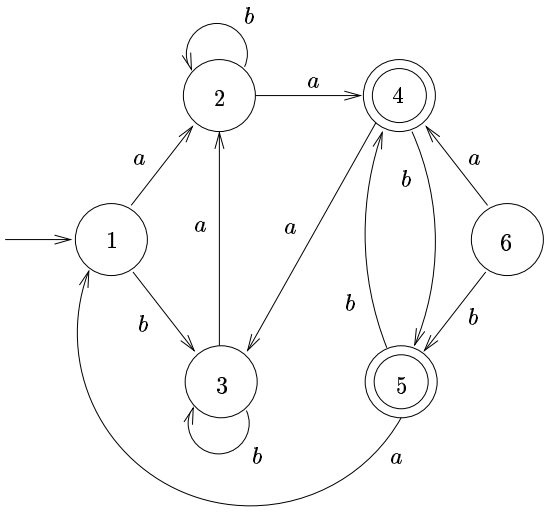
- Huom: Tiloja on alun perin äärellinen määrä ja joka askeleessa nro 3 (paitsi viimeisessä) ositetaan vähintään yksi tilaluokka, joten algoritmi päättyy aina

- Todistus siitä, että algoritmin tuottama automaatti $\widehat{M}$ on minimiautomaatti ja yksikäsitteinen sivuutetaan (Orposen moniste s. 17–18)

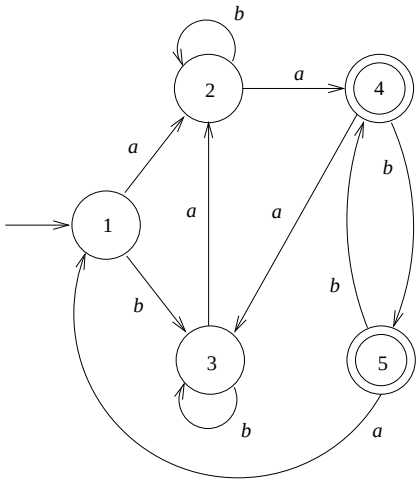
Esimerkki: Olkoon $M = (Q, \Sigma, \delta, q_0, F)$, missä

- tilojen joukko $Q = \{1, 2, 3, 4, 5, 6\}$,
- syöteaakkosto $\Sigma = \{a, b\}$,
- alkutila $q_0 = \{1\}$,
- lopputilojen joukko $F = \{4, 5\}$ ja
- siirtymäfunktio δ:

		a	b
$\rightarrow$	1	2	3
	2	4	2
	3	2	3
$\leftarrow$	4	3	5
$\leftarrow$	5	1	4
	6	4	5



Askel 1: Turhien tilojen poisto



Askel 2: 0-ekvivalenssi

- Osita M:n jäljelle jääneet tilat kahteen luokkaan: lopputiloihin ja muihin tiloihin

			a	b
I:	$\rightarrow$	1	2, I	3, I
		2	4, II	2, I
		3	2, I	3, I
II:	$\leftarrow$	4	3, I	5, II
	$\leftarrow$	5	1, I	4, II

- Nyt osituksesta voidaan muodostaa automaatti, jossa
  - kutakin luokkaa vastaa yksi tila ja
  - kustakin tilasta on kaikki erilliset siirtymät, jotka luokkaan kuuluvilla tiloilla on

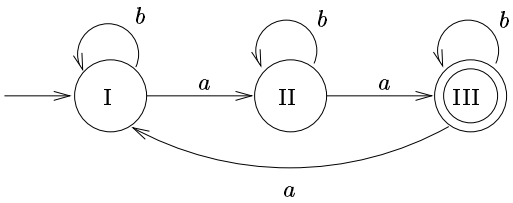
- Tila on *epädeterministinen*, jos siitä voidaan siirtyä jollain merkillä useampaan kuin yhteen tilaan
- Esimerkissä tila I on epädeterministinen, koska merkillä a voidaan siirtyä tilaan I tai tilaan II

Askel 3: k-ekvivalenssi $\Rightarrow$ $(k+1)$-ekvivalenssi

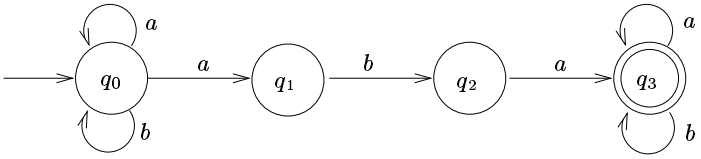
- Jos $\widehat{M}$:ssä ei ole epädeterministisiä tiloja, niin algoritmi päättyy ja palauttaa $\widehat{M}$:n
- Muutoin hienonna kutakin $\widehat{M}$:n epädeterminististä tilaa vastaavan luokan ositusta edelleen:
  - Jaa sen sisällä alkuperäiset tilat eri luokkiin s.e. kustakin luokasta on vain samanlaisia siirtymiä
  - Suorita askel 3 uudestaan
- Esimerkissämme jaetaan tila I kahtia
- Tämän jälkeen ei ole enää epädeterminisiä tiloja ja algoritmi päättyy

Lopputulos:

			a	b
I:	$\rightarrow$	1	2, II	3, I
		3	2, II	3, I
II:		2	4, III	2, II
III:	$\leftarrow$	4	3, I	5, III
	$\leftarrow$	5	1, I	4, III



3.4 Epädeterministiset äärelliset automaatit



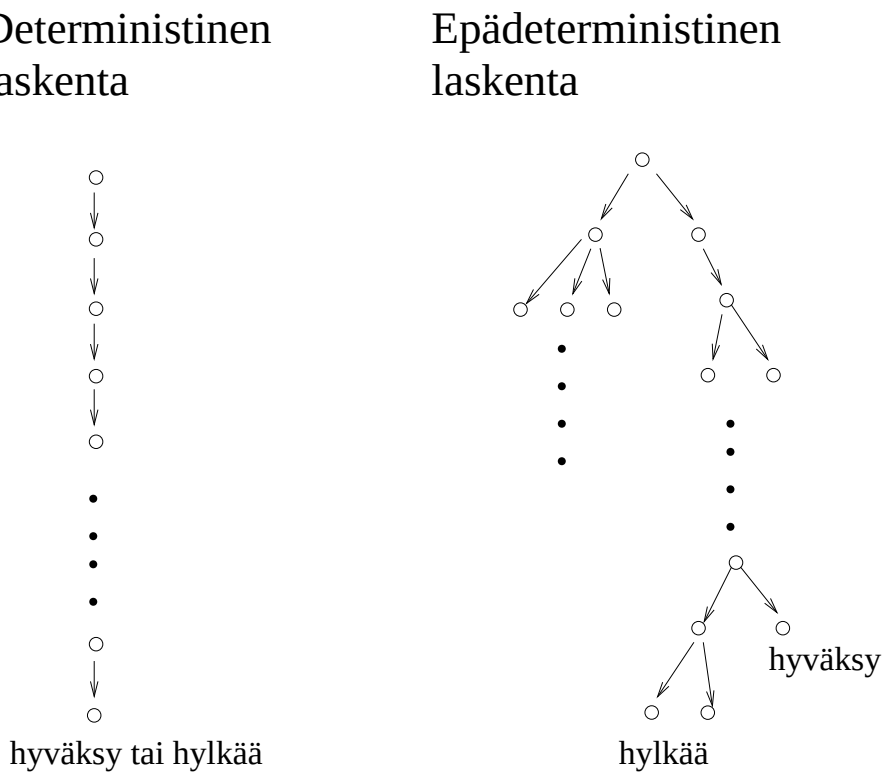
- Esimerkiksi alimerkkijonon *aba* etsiminen aakkoston $\{a, b\}$ merkkijonoista on luontevasti kuvattavissa epädeterministisellä automaatilla. Epädeterminismi siis helpottaa automaatin konstruointia
- Epädeterminististen automaattien avulla luodaan yhteys determinististen äärellisten automaattien ja *säännöllisten kielten* välille
  - deterministiset ja epädeterministiset automaatit tunnistavat täsmälleen samat kielet
  - epädeterministiset automaatit tunnistavat täsmälleen säännölliset kielet
  - $\Rightarrow$ Siis deterministiset automaatit tunnistavat täsmälleen säännölliset kielet
- Epädeterministisellä automaatilla siirtymäfunktio liittää vanhan tilan ja syötemerkin pariin (q, x) *joukon* mahdollisia seuraavia tiloja
- Epädeterministinen automaatti hyväksyy merkkijonon jos jokin mahdollisten tilojen jono johtaa lopputilaan. Jos yhtään tällaista jonoa ei ole, niin epädeterministinen automaatti hylkää syötemerkkijonon
- Esim. kuvan automaatti hyväksyy syötejonon *abbaba*, koska se voidaan käsitellä seuraavasti:

$$(q_0, abbaba) \vdash (q_0, bbaba) \vdash (q_0, baba)$$
$$\vdash (q_0, aba) \vdash (q_1, ba) \vdash (q_2, a) \vdash (q_3, \epsilon)$$

- Toisaalta voidaan myös päätyä hylkäävään tilaan

$$(q_0, abbaba) \vdash (q_0, bbaba) \vdash (q_0, baba)$$
$$\vdash (q_0, aba) \vdash (q_0, ba) \vdash (q_0, a) \vdash (q_0, \epsilon)$$

- Epädeterministinen automaatti voidaan ajatella suorittavan kaikki johdot rinnakkain



- *Määritelmä:* Epädeterministinen äärellinen automaatti (engl. nondeterministic finite automaton) on viisikko $M = (Q, \Sigma, \delta, q_0, F)$, missä
  - Q on äärellinen tilojen joukko,
  - Σ on syöteaakkosto,
  - $\delta : Q \times \Sigma \to \mathcal{P}(Q)$ on (joukkoarvoinen) siirtymäfunktio,
  - $q_0 \in Q$ on alkutila ja
  - $F \subseteq Q$ lopputilojen joukko.

- Kuvan automaatin siirtymäfunktio

	a	b
$\rightarrow q_0$	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\{q_2\}$
q_2	$\{q_3\}$	$\emptyset$
$\leftarrow q_3$	$\{q_3\}$	$\{q_3\}$

- Nyt virhetilanne on helposti ilmaistavissa tyhjän seuraajatilajoukon avulla (esim. $\delta(q, a) = \emptyset$ merkitsee sitä, että a aiheuttaa virheen tilassa q_1)
- (q, w) *voi* johtaa suoraan tilanteeseen (q', w'), $(q, w) \underset{M}{\vdash} (q', w')$, jos $w = aw'$ ja $q' \in \delta(q, a)$. Tilanne (q', w') on (q, w):n *mahdollinen* välitön seuraaja
- Muutoin määritelmät epädeterministisille automaateille ovat samat kuin aiemmin

- Deterministiset automaatit ovat epädeterminististen erikoistapaus $\Rightarrow$ kaikki edellisillä tunnistettavat kielet ovat tunnistettavissa myös jälkimmäisillä

- Mutta myös kääntäen: *deterministiset ja epädeterministiset äärelliset automaatit ovat yhtä vahvoja*

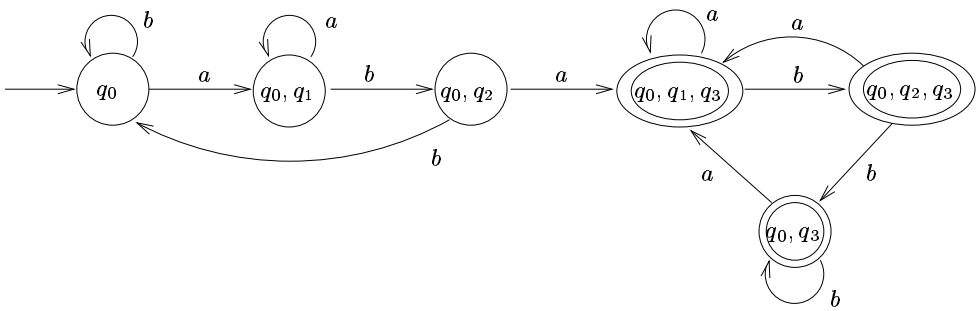
3.4.1 Automaatin determinisointi

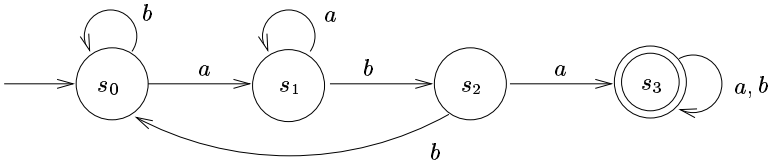
- Muodostetaan epädeterminististä automaattia M vastaava deterministinen automaatti $\widehat{M}$:

  1. Muodosta $\widehat{M}$:n tilat $S \subseteq \mathcal{P}(Q)$
     - ts. kaikki M:n tilojen joukot $\mathcal{P}(Q)$
     - Merk. $\mathcal{P}(Q) = \{\emptyset, s_1, s_2, ..., s_m\}$, missä tyhjä joukko vastaa virhetilaa ja $m = 2^n - 1$
  2. Lisää $\widehat{M}$:n tilojen välille siirtymät:
     - $s_i \xrightarrow{a} s_j$, missä $s_j = \bigcup\{q' | \delta(q, a) = q', q \in s_i\}$
     - ts. tilajoukkojen s_i ja s_j välille siirtymä $s_i \xrightarrow{a} s_j$, jos niiden osajoukkojen välillä oli siirtymä: $\forall q' \in s_j \; \exists q \in s_i$ siten että $q \xrightarrow{a} q'$.
     - s_i:n seuraajajoukkoon merkillä a kuuluvat siis kaikki ne tilat q', jotka voidaan saavuttaa s_i:n tiloista q merkillä a
     - Huom! Voisimme valita seuraajat myös seuraavalla ehdolla: $\exists q \in s_i \; \exists q' \in s_j$ siten että $q \xrightarrow{a} q'$. Nyt tulee kuitenkin paljon turhia siirtymiä, joista on riesaa minimointivaiheessa!

3. Alkutilaksi $\{q_0\}$ (alkuperäisestä alkutilasta muodostettu joukko)
4. Lopputiloiksi kaikki alkuperäisen lopputilan q_f sisältävät tilajoukot s_i, $q_f \in s_i$
5. Karsi turhat tilat, joita ei voi saavuttaa alkutilasta
6. Minimoi automaatti
   - jaa loppu- ja muihin tiloihin
   - hienonna luokkajakoa, kunnes yhdenmukainen siirtymäfkt:n kanssa

- Esimerkki:





- *Lause:* Olk. $A = L(M)$ jonkin epädeterministisen äärellisen automaatin M tunnistama kieli. Tällöin on olemassa deterministinen automaatti $\widehat{M}$, jolla $L(\widehat{M}) = A$.
  *Todistus: Olk. $A = L(M)$, $M = (Q, \Sigma, \delta, q_0, F)$. Laaditaan determ. automaatti $\widehat{M} = (\hat{Q}, \Sigma, \hat{\delta}, \hat{q_0}, \hat{F})$, joka simuloi M:n toimintaa kaikissa sen kullakin hetkellä mahdollisissa tiloissa rinnakkain. Automaatin $\widehat{M}$ tilat vastaavat M:n tilojen joukkoja

$$\begin{aligned} \hat{Q} &= \mathcal{P}(Q), \\ \hat{q_0} &= \{q_0\}, \\ \hat{F} &= \{S \subseteq Q \mid S \text{ sisältää jonkin } q_f \in F\}, \\ \hat{\delta}(S, a) &= \bigcup_{q \in S} \delta(q, a). \end{aligned}$$

Tarkastetaan, että $L(\widehat{M}) = L(M)$. Kielten ekvivalenssi seuraa, kun todistetaan kaikilla $x \in \Sigma^*$ ja $q \in Q$:

$$(q_0, x) \underset{M}{\vdash}^* (q, \epsilon) \Leftrightarrow (\{q_0\}, x) \underset{\widehat{M}}{\vdash}^* (S, \epsilon) \text{ ja } q \in S.$$

Todistus induktiolla merkkijonon x pituuden suhteen:

1. $|x| = 0$: $(q_0, \epsilon) \underset{M}{\vdash}^* (q, \epsilon) \Leftrightarrow q = q_0$.
   Samoin $(\{q_0\}, \epsilon) \underset{\widehat{M}}{\vdash}^* (S, \epsilon) \Leftrightarrow S = \{q_0\}$.
2. *Induktio-oletus*: väite pätee kun $|x| \leq k$.
3. $|x| = k + 1$: tällöin $x = ya$ jollakin y, $|y| = k$, jolle väite pätee induktio-oletuksen perusteella. Nyt

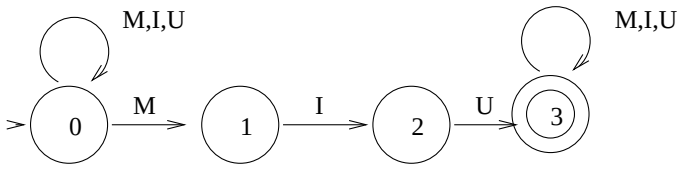
$$\begin{aligned}
& (q_0, x) = (q_0, ya) \underset{M}{\vdash}^* (q, \epsilon) \\
\Leftrightarrow\ & \exists q' \in Q \text{ s.e. } (q_0, ya) \underset{M}{\vdash}^* (q', a) \text{ ja } (q', a) \underset{M}{\vdash} (q, \epsilon) \\
\Leftrightarrow\ & \exists q' \in Q \text{ s.e. } (q_0, y) \underset{M}{\vdash}^* (q', \epsilon) \text{ ja } (q', a) \underset{M}{\vdash} (q, \epsilon) \\
\Leftrightarrow\ & \exists q' \in Q \text{ s.e. } (\{q_0\}, y) \underset{\widehat{M}}{\vdash}^* (S', \epsilon) \text{ ja } q' \in S' \text{ ja } q \in \delta(q', a) \\
\Leftrightarrow\ & (\{q_0\}, y) \underset{\widehat{M}}{\vdash}^* (S', \epsilon) \text{ ja } \exists q' \in S' \text{ s.e. } q \in \delta(q', a) \\
\Leftrightarrow\ & (\{q_0\}, y) \underset{\widehat{M}}{\vdash}^* (S', \epsilon) \text{ ja } q \in \bigcup_{q' \in S'} \delta(q', a) = \hat{\delta}(S', a) \\
\Leftrightarrow\ & (\{q_0\}, ya) \underset{\widehat{M}}{\vdash}^* (S', a) \text{ ja } q \in \hat{\delta}(S', a) = S \\
\Leftrightarrow\ & (\{q_0\}, ya) \underset{\widehat{M}}{\vdash}^* (S', a) \text{ ja } (S', a) \underset{\widehat{M}}{\vdash} (S, \epsilon) \text{ ja } q \in S \\
\Leftrightarrow\ & (\{q_0\}, x) = (\{q_0\}, ya) \underset{\widehat{M}}{\vdash}^* (S, \epsilon) \text{ ja } q \in S.
\end{aligned}$$

□

3.4.2 Hahmontunnistus epädeterministisellä automaatilla

Epädeterministisillä automaateilla voidaan kätevästi kuvata hahmontunnistusongelmia. Ohjelmatoteutusta verten automaatti on kuitenkin determinisoitava. Edellä huomasimme, että pahimmassa tapauksessa vastaavan deterministisen automaatin tilamäärä on eksponentiaalinen. Hahmontunnistusongelmissa determinisointialgoritmi tuottaa kuitenkin automaatin, jossa on *yhtä monta tilaa* kuin alkuperäisessä epädeterministisessä automaatissa, ja joka on samalla *minimiautomaatti!*

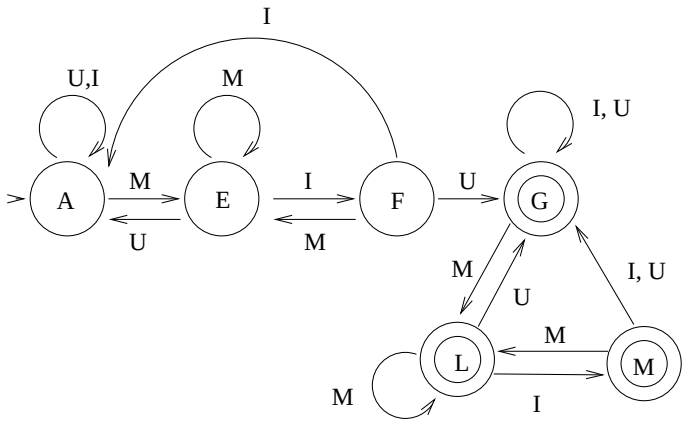
Esim. Olkoon aakkosto $\{M, I, U\}$. Esiintyykö hahmo MIU merkkijonossa? Vastaava epädeterministinen automaatti:



Kuva 3.1: Epädeterministinen MIU-automaatti.

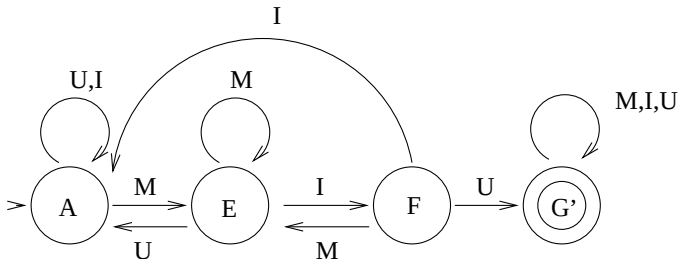
Muodostetaan vastaava deterministinen automaatti. Taulukossa on esitetty kaikki tilat, myös ne joita ei voi saavuttaa alkutilasta. Käytännössä voit oikaista, kun lähdet liikkeelle alkutilasta ja lasket ensin sen seuraajat, sitten näiden tilojen seuraajat, jne. Nyt mukaan tulevat vain ne tilat, jotka voit oikeasti saavutata alkutilasta.

		M	I	U	
		$\emptyset$	$\emptyset$	$\emptyset$	$\emptyset$
$\rightarrow$	**A**	$\{0\}$	$\{0,1\}$=E	$\{0\}$=A	$\{0\}$=A
	B	$\{1\}$	$\emptyset$	$\{2\}$=C	$\emptyset$
	C	$\{2\}$	$\emptyset$	$\emptyset$	$\{3\}$=D
$\leftarrow$ D	$\{3\}$	$\{3\}$=D	$\{3\}$=D	$\{3\}$=D	
	E	$\{0,1\}$	$\{0,1\}$=E	$\{0,2\}$=F	$\{0\}$=A
	F	$\{0,2\}$	$\{0,3\}$=E	$\{0\}$=A	$\{0,3\}$=G
$\leftarrow$	**G**	$\{0,3\}$	$\{0,1,3\}$=L	$\{0,3\}$=G	$\{0,3\}$=G
	H	$\{1,2\}$	$\emptyset$	$\{2\}$=C	$\{3\} = D$
$\leftarrow$	I	$\{1,3\}$	$\{3\}$=D	$\{2,3\}$=J	$\{3\}$=D
$\leftarrow$	J	$\{2,3\}$	$\{3\}$=D	$\{3\}$=D	$\{3\}$=D
	K	$\{0,1,2\}$	$\{0,1\}$=E	$\{0,2\}$=F	$\{0,3\}$=G
$\leftarrow$	**L**	$\{0,1,3\}$	$\{0,1,3\}$=L	$\{0,2,3\}$=M	$\{0,3\}$=G
$\leftarrow$	**M**	$\{0,2,3\}$	$\{0,1,3\}$=L	$\{0,3\}$=G	$\{0,3\}$=G
$\leftarrow$	N	$\{1,2,3\}$	$\{3\}$=D	$\{2,3\}$=J	$\{3\}$=D
$\leftarrow$	O	$\{0,1,2,3\}$	$\{0,1,3\}$=L	$\{0,2,3\}$=M	$\{0,3\}$=G



Kuva 3.2: Deterministinen MIU-automaatti.

Lopuksi yhdistetään lopputilat (minimointialgoritmin mukaan).



Kuva 3.3: Lopputulos: minimaalinen deterministinen MIU-automaatti.