3.2 Säännölliset kielet ja yhteydettömät kieliopit

Yhteydettömillä kieliopeilla voidaan siis kuvata joitakin ei-säännöllisiä kieliä (esimerkiksi kielet L_{match} ja L_{expr}). Osoitetaan, että myös kaikki säännölliset kielet voidaan kuvata yhteydettömillä kieliopeilla. Yhteydettömät kielet ovat siten säännollisten kielten aito yliluokka.

Yhteydetön kielioppi on *oikealle lineaarinen*, jos sen kaikki produktiot ovat muotoa $A \to aB$ tai $A \to \varepsilon$, ja *vasemmalle lineaarinen*, jos sen kaikki produktiot ovat muotoa $A \to Ba$ tai $A \to \varepsilon$.

Osoittautuu, että sekä vasemmalle että oikealle lineaarisilla kieliopeilla voidaan tuottaa täsmälleen säännölliset kielet, minkä takia näitä kielioppeja nimitetään myös yhteisesti *säännöllisiksi*. Todistetaan tässä väite vain oikealle lineaarisille kieliopeille.

Lause 3.1 Jokainen säännöllinen kieli voidaan tuottaa oikealle lineaarisella kieliopilla.

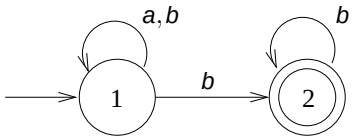
Todistus. Olkoon L aakkoston Σ säännöllinen kieli, ja olkoon $M = (Q, \Sigma, \delta, q_0, F)$ sen tunnistava (deterministinen tai epädeterministinen) äärellinen automaatti. Muodostetaan kielioppi G_M, jolla on $L(G_M) = L(M) = L$.

Kieliopin G_M pääteaakkosto on sama kuin M:n syöteaakkosto Σ, ja sen välikeaakkostoon otetaan yksi välike A_q kutakin M:n tilaa q kohden. Kieliopin lähtösymboli on A_{q_0}, ja sen produktiot vastaavat M:n siirtymiä:

(i) kutakin M:n lopputilaa $q \in F$ kohden kielioppiin otetaan produktio $A_q \to \varepsilon$;

(ii) kutakin M:n siirtymää $q \xrightarrow{a} q'$ (so. $q' \in \delta(q,a)$) kohden kielioppiin otetaan produktio $A_q \to aA_{q'}$.

Esimerkki. Automaatti:



Vastaava kielioppi:

$$\begin{aligned} A_1 &\to aA_1 \mid bA_1 \mid bA_2 \\ A_2 &\to \varepsilon \mid bA_2. \end{aligned}$$

Konstruktion oikeellisuuden tarkastamiseksi merkitään välikkeestä A_q tuotettavien päätejonojen joukkoa

$$L(A_q) = \{x \in \Sigma^* \mid A_q \underset{G_M}{\Rightarrow^*} x\}.$$

Induktiolla merkkijonon x pituuden suhteen voidaan osoittaa, että kaikilla q on

$$x \in L(A_q) \text{ joss } (q,x) \underset{M}{\vdash^*} (q_f, \varepsilon) \text{ jollakin } q_f \in F.$$

Erityisesti on siis

$$\begin{aligned} L(G_M) = L(A_{q_0}) &= \{x \in \Sigma^* \mid (q_0,x) \underset{M}{\vdash^*} (q_f,\varepsilon) \\ &\qquad \text{jollakin } q_f \in F\} \\ &= L(M) = L. \quad \square \end{aligned}$$

Lause 3.2 Jokainen oikealle lineaarisella kieliopilla tuotettava kieli on säännöllinen.

Todistus. Olkoon $G = (V, \Sigma, P, S)$ oikealle lineaarinen kielioppi. Muodostetaan kielen $L(G)$ tunnistava epädeterministinen äärellinen automaatti $M_G = (Q, \Sigma, \delta, q_S, F)$ seuraavasti:

M_G:n tilat vastaavat G:n välikkeitä:

$$Q = \{q_A \mid A \in V - \Sigma\}.$$

M_G:n alkutila on lähtösymbolia S vastaava tila q_S.

M_G:n syöteaakkosto on G:n pääteaakkosto Σ.

M_G:n siirtymäfunktio δ jäljittelee G:n produktioita siten, että kutakin produktiota $A \to aB$ kohden automaatissa on siirtymä $q_A \xrightarrow{a} q_B$ (so. $q_B \in \delta(q_A, a)$).

M_G:n lopputiloja ovat ne tilat, joita vastaaviin välikkeisiin liittyy G:ssä ε-produktio:

$$F = \{q_A \in Q \mid A \to \varepsilon \in P\}.$$

Konstruktion oikeellisuus voidaan jälleen tarkastaa induktiolla G:n tuottamien ja M_G:n hyväksymien merkkijonojen pituuden suhteen. □

3.4 Osittava jäsentäminen

Yksi (yleisessä muodossa tehoton!) tapa etsiä vasenta johtoa (jäsennyspuuta) annetun kieliopin G mukaiselle lauseelle x on aloittaa G:n lähtösymbolista ja generoida systemaattisesti kaikki mahdolliset vasemmat johdot (jäsennyspuut), samalla sovittaen muodostetun lausejohdoksen päätemerkkejä (puun lehtiä) x:n merkkeihin. Ei-yhteensopivuuden ilmetessä peruutetaan viimeksi tehty produktiovalinta ja kokeillaan järjestyksessä seuraavaa vaihtoehtoa.

Tällaista lauseenjäsennystapaa sanotaan *osittavaksi*, koska siinä tarkasteltu lause yritetään johtaa kieliopin lähtösymbolista osittamalla se valittujen produktioiden mukaisiin rakenneosiin ja yrittämällä näin, tarvittaessa toistuvasti edelleen osittamalla, sovittaa kieliopin tuottamaa rakennetta yhteen lauseen rakenteen kanssa.

Esim. Tarkastellaan kielioppia G:

$$\begin{aligned} E &\to T + E \mid T - E \mid T \\ T &\to a \mid (E). \end{aligned}$$

Lauseen $a - a$ osittava jäsennys G:n suhteen:

$$\begin{aligned}
E &\Rightarrow T + E \Rightarrow a + T && \text{[ristiriita; peruutetaan]} \\
&\qquad\quad \Rightarrow (E) + T && \text{[ristiriita; peruutetaan]} \\
&\Rightarrow T - E \Rightarrow a - E \Rightarrow a - T + E \Rightarrow a - a + E \\
&\qquad\quad && \text{[ristiriita; peruutetaan]} \\
&\Rightarrow T - E \Rightarrow a - E \Rightarrow a - T + E \Rightarrow a - (E) + E \\
&\qquad\quad && \text{[ristiriita; peruutetaan]} \\
&\qquad\quad \Rightarrow a - T - E \Rightarrow a - a - E \\
&\qquad\quad && \text{[ristiriita; peruutetaan]} \\
&\qquad\quad \Rightarrow a - T - E \Rightarrow a - (E) - E \\
&\qquad\quad && \text{[ristiriita; peruutetaan]} \\
&\qquad\quad \Rightarrow a - T \Rightarrow a - a && \text{[OK!]}
\end{aligned}$$

Em. osittava jäsennystekniikka saadaan huomattavasti tehokkaammaksi, jos kieliopilla on sellainen ominaisuus, että jäsennyksen joka vaiheessa määrää tavoitteena olevan lauseen *seuraava merkki* yksikäsitteisesti sen, mikä lavennettavana olevaan välikkeeseen liittyvä produktio on valittava. Kielioppia, jolla on tämä ominaisuus, sanotaan *LL(1)-tyyppiseksi*.

Muokataan G:stä välikkeen E produktiot "tekijöimällä" ekvivalentti kielioppi G':

$$\begin{aligned} E &\rightarrow TE' \\ E' &\rightarrow +E \mid -E \mid \varepsilon \\ T &\rightarrow a \mid (E). \end{aligned}$$

Esimerkiksi lauseen $a-a$ jäsentäminen G':n suhteen (kulloisenkin produktiovalinnan määräävä syötemerkki on tässä merkitty vastaavan johtonuolen päälle):

$$E \underset{\text{lm}}{\Rightarrow} TE' \overset{a}{\underset{\text{lm}}{\Rightarrow}} aE' \overset{-}{\underset{\text{lm}}{\Rightarrow}} a-E \underset{\text{lm}}{\Rightarrow} a-TE' \overset{a}{\underset{\text{lm}}{\Rightarrow}} a-aE' \overset{\varepsilon}{\underset{\text{lm}}{\Rightarrow}} a-a.$$

LL(1)-tyyppiselle kieliopille on helppo kirjoittaa jäsennysohjelma suoraan rekursiivisina proseduureina. Esimerkiksi kieliopin G' pohjalta voidaan muodostaa seuraava C-kielinen funktiokokoelma, joka syötejonon jäsennyksen yhteydessä tulostaa sen tuottavan vasemman johdon produktiot järjestyksessä.

```
#include <stdio.h>

int next;
void E(void); void Eprime(void); void T(void);

void E(void)
{
  printf("E  -> TE'\n");
  T(); Eprime();
}
```



```
void Eprime(void)
{
  if (next == '+') {
    printf("E' -> +E\n");
    next = getchar();
    E();
  }
  else if (next == '-') {
    printf("E' -> -E\n");
    next = getchar();
    E();
  }
  else
    printf("E' -> \n");
}
```



```
void T(void)
{
  if (next == 'a') {
    printf("T  -> a\n");
    next = getchar();
  }
  else if (next == '(') {
    printf("T  -> (E)\n");
    next = getchar();
    E();
    if (next != ')')
      ERROR(") expected.");
    next = getchar();
  }
  else ERROR("T cannot start with this.");
}
```

```
void ERROR(char *msg)
{
  printf("%s\n",msg); exit(1);
}

int main(void)
{
  next = getchar();
  E();
  exit(0);
}
```


Esimerkiksi syötejonoa `a-(a+a)` käsitellessään ohjelma tulostaa seuraavat rivit:

```
E → TE′
T → a
E′ → -E
E → TE′
T → (E)
E → TE′
T → a
E′ → +E
E → TE′
T → a
E′ →
E′ →
```

Tulostus vastaa vasenta johtoa:

$$
\begin{aligned}
E &\Rightarrow TE' \Rightarrow aE' \Rightarrow a-E \Rightarrow a-TE' \\
&\Rightarrow a-(E)E' \Rightarrow a-(TE')E' \\
&\Rightarrow a-(aE')E' \Rightarrow a-(a+E)E' \\
&\Rightarrow a-(a+TE')E' \Rightarrow a-(a+aE')E' \\
&\Rightarrow a-(a+a)E' \Rightarrow a-(a+a).
\end{aligned}
$$