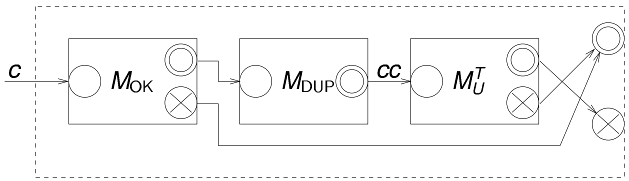
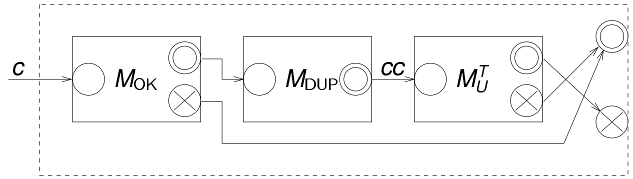
Lause 6.7 Kieli U ei ole rekursiivinen.

Todistus. Oletetaan, että kielellä U olisi totaalinen tunnistajakone M_U^T. Tällöin voitaisiin Lemman 6.5 kielelle D muodostaa totaalinen tunnistajakone M_D seuraavasti.

Olkoon M_{OK} totaalinen Turingin kone, joka testaa, onko syötteenä annettu merkkijono kelvollinen Turingin koneen koodi, ja olkoon M_{DUP} totaalinen Turingin kone, joka muuntaa syötejonon c muotoon cc. Kone M_D muodostetaan koneista M_U^T, M_{OK} ja M_{DUP} yhdistämällä seuraavan kaavion esittämällä tavalla:





Selvästi kone M_D on totaalinen, jos kone M_U^T on, ja

$$\begin{aligned} c \in L(M_D) &\Leftrightarrow c \notin L(M_{\mathrm{OK}}) \text{ tai } cc \notin L(M_U^T) \\ &\Leftrightarrow c \notin L(M_c) \\ &\Leftrightarrow c \in D. \end{aligned}$$

Mutta lemman 6.5 mukaan kieli D ei ole rekursiivinen; ristiriita.

$\square$.

Seuraus 6.8 Kieli

$$\tilde{U} = \{c_M w \mid w \notin L(M)\}$$

ei ole rekursiivisesti numeroituva.

Todistus. Kieli $\tilde{U}$ on oleellisesti sama kuin universaalikielen U komplementti $\bar{U}$; tarkasti ottaen on $\bar{U} = \tilde{U} \cup \mathrm{ERR}$, missä ERR on helposti tunnistettava rekursiivinen kieli

$$\begin{aligned} \mathrm{ERR} \;=\; & \{x \in \{0,1\}^* \mid x \text{ ei sisällä alkuosanaan} \\ & \text{kelvollista Turingin koneen koodia}\}. \end{aligned}$$

Jos siis kieli $\tilde{U}$ olisi rekursiivisesti numeroituva, olisi samoin myös kieli $\bar{U}$. Koska kieli U on rekursiivisesti numeroituva, seuraisi tästä, että U on peräti rekursiivinen. Mutta tämä on vastoin edellisen lauseen tulosta, mistä päätellään, että kieli $\tilde{U}$ ei voi olla rekursiivisesti numeroituva. $\square$

6.5 Turingin koneiden pysähtymisongelma

Lause 6.9 Kieli

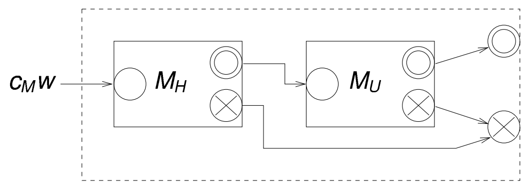
$$H = \{c_M w \mid M \text{ pysähtyy syötteellä } w\}$$

on rekursiivisesti numeroituva, mutta ei rekursiivinen.

Todistus. Todetaan ensin, että kieli H on rekursiivisesti numeroituva. Lauseen 6.6 todistuksessa esitetystä universaalikoneesta M_U on helppo muokata kone, joka syötteellä $c_M w$ simuloi koneen M laskentaa syötteellä w ja pysähtyy hyväksyvään lopputilaan, jos ja vain jos simuloitu laskenta ylipäätään pysähtyy.

Osoitetaan sitten, että kieli H ei ole rekursiivinen. Oletetaan nimittäin, että olisi $H = L(M_H)$ jollakin totaalisella Turingin koneella M_H. Oletetaan lisäksi, että kone M_H pysähtyessään jättää nauhalle alkuperäisen syötteensä, mahdollisesti tyhjämerkeillä jatkettuna. Olkoon M_U lauseen 6.6 todistuksessa konstruoitu universaalikone.

Kielelle U voitaisiin nyt muodostaa totaalinen tunnistaja yhdistämällä koneet M_H ja M_U seuraavasti:



Lauseen 6.7 mukaan tällaista kielen U tunnistajakonetta ei kuitenkaan voi olla olemassa. Saatu ristiriita osoittaa, että H ei voi olla rekursiivinen. □

Seuraus 6.10 Kieli

$$\tilde{H} = \{c_M w \mid M \text{ ei pysähdy syötteellä } x\}$$

ei ole rekursiivisesti numeroituva. □

6.7 Ricen lause

Ricen lauseen mukaan *kaikki* Turingin koneiden tunnistamia kieliä, t. niiden laskemia I/O-kuvauksia koskevat epätriviaalit kysymykset ovat ratkeamattomia.

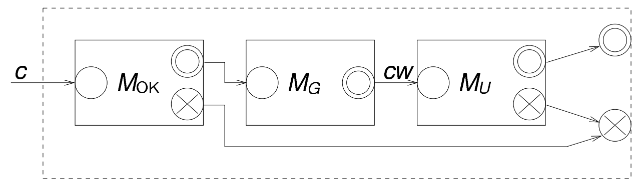
Johdantona lauseen todistukseen tarkastellaan ensin yhtä sen erikoistapausta, Turingin koneiden tunnistamien kielten *epätyhjyysongelmaa*: "Hyväksyykö annettu Turingin kone yhtään syötemerkkijonoa?" Ongelman esitys formaalina kielenä on

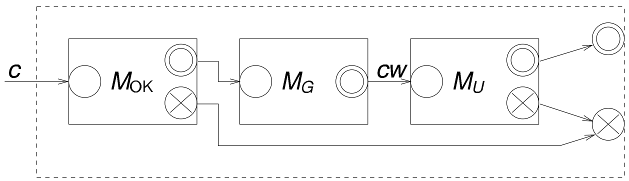
$$\mathrm{NE} = \{c \in \{0,1\}^* \mid L(M_c) \neq \emptyset\}.$$

Lause 6.11 Kieli NE on rekursiivisesti numeroituva, mutta ei rekursiivinen.

Todistus. Todetaan ensin, että kieli NE on rekursiivisesti numeroituva muodostamalla sille tunnistajakone M_{NE}. Kone M_{NE} on helpointa suunnitella epädeterministisenä.

Olkoon M_{OK} Turingin kone, joka testaa onko annettu syöte kelvollinen Turingin koneen koodi, ja olkoon M_G epädeterministinen Turingin kone, joka kirjoittaa nauhalla jo olevan merkkijonon perään mielivaltaisen binäärijonon w. Kone M_{NE} voidaan muodostaa yhdistämällä koneet M_{OK}, M_G ja universaalikone M_U seuraavasti:



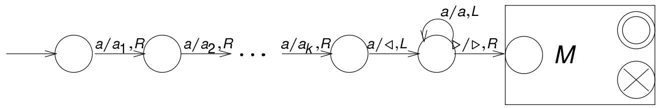


Selvästi on:

$$
\begin{array}{ll}
 & c \in L(M_{\mathrm{NE}}) \\
\Leftrightarrow & c \text{ on kelvollinen Tk-koodi ja } \exists\, w \text{ s.e. } cw \in U \\
\Leftrightarrow & c \text{ on kelvollinen Tk-koodi ja } \exists\, w \text{ s.e. } w \in L(M_c) \\
\Leftrightarrow & L(M_c) \neq \emptyset.
\end{array}
$$

Osoitetaan, ettei kieli NE ole rekursiivinen. Oletetaan, että kielellä NE olisi totaalinen tunnistajakone M^T_{NE}, ja muodostetaan sitä käyttäen totaalinen tunnistajakone M^T_U kielelle U. (Ristiriita.)

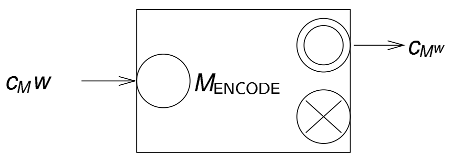
Konstruktio perustuu syötteiden koodaamiseen Turingin koneiden “ohjelmavakioiksi”. Olkoon M mielivaltainen Turingin kone, jonka toimintaa syötteellä $w = a_1 a_2 \ldots a_k$ halutaan tutkia. Merkitään M^w:llä konetta, joka aina korvaa “todellisen” syötteensä merkkijonolla w ja toimii sitten kuten M:



Koneen M^w toiminta ei siis riipu lainkaan sen todellisesta syötteestä, vaan se joko hyväksyy tai hylkää kaikki merkkijonot, sen mukaan miten M suhtautuu w:hen:

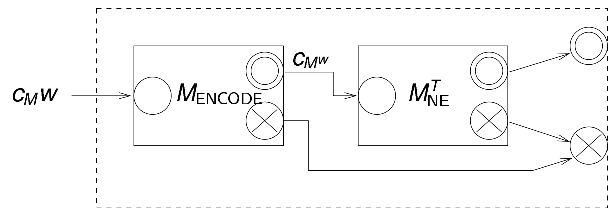
$$L(M^w) = \begin{cases} \{0,1\}^*, & \text{jos } w \in L(M); \\ \emptyset, & \text{jos } w \notin L(M). \end{cases}$$

Olkoon sitten M_{ENCODE} Turingin kone, joka saa syötteenään mielivaltaisen Turingin koneen M koodista c_M ja binäärijonosta w muodostuvan jonon $c_M w$ ja jättää tuloksenaan nauhalle edellä kuvatun koneen M^w koodin c_{M^w}:



(Jos syöte ei ole muotoa cw, missä c on kelvollinen Turingin koneen koodi, kone M_{ENCODE} päätyy hylkäävään lopputilaan.) Kone M_{ENCODE} operoi siis Turingin koneiden *koodeilla*. Annetun koneen M koodiin se lisää siirtymäviisikoita (“konekäskyjä”) ja muuttaa tilojen numerointia siten, että koodi tulee koneen M sijaan esittämään konetta M^w.

Universaalikielelle U voitaisiin nyt koneet M_{ENCODE} ja hypoteettinen M^T_{NE} seuraavalla tavalla yhdistämällä muodostaa totaalinen tunnistajakone M^T_U:



Kone M^T_U on totaalinen, jos M^T_{NE} on, ja $L(M^T_U) = U$, koska:

$$c_M w \in L(M^T_U) \Leftrightarrow c_{M^w} \in L(M^T_{\mathrm{NE}}) = \mathrm{NE} \Leftrightarrow L(M^w) \neq \emptyset \Leftrightarrow w \in L(M).$$

Mutta kieli U ei ole rekursiivinen, joten tällainen totaalinen tunnistajakone M^T_U ei ole mahdollinen. Saadusta ristiriidasta päätellään, että myöskään kielellä NE ei voi olla totaalista tunnistajaa M^T_{NE}. $\square$

Ricen lause

Turingin koneiden *semanttinen ominaisuus* $\mathcal{S}$ on mikä tahansa kokoelma rekursiivisesti numeroituvia aakkoston $\{0,1\}$ kieliä; koneella *M on ominaisuus* $\mathcal{S}$, jos $L(M) \in \mathcal{S}$. *Triviaalit ominaisuudet* ovat $\mathcal{S} = \emptyset$ (ominaisuus, jota ei ole millään koneella) ja $\mathcal{S} = RE$ (ominaisuus, joka on kaikilla koneilla).

Ominaisuus $\mathcal{S}$ on *ratkeava*, jos joukko

$$\mathrm{codes}(\mathcal{S}) = \{c \mid L(M_c) \in \mathcal{S}\}$$

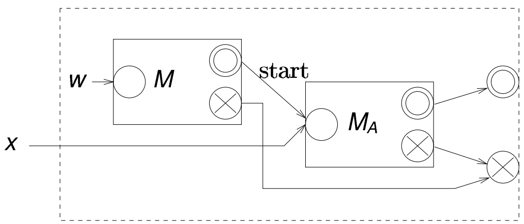
on rekursiivinen. Toisin sanoen: ominaisuus on ratkeava, jos annetusta Turingin koneen koodista voidaan algoritmisesti päätellä, onko koneella kysytty semanttinen ominaisuus.

Lause 6.12 [Rice 1953] Kaikki Turingin koneiden epätriviaalit semanttiset ominaisuudet ovat ratkeamattomia.

Todistus. Olkoon $\mathcal{S}$ mielivaltainen epätriviaali semanttinen ominaisuus. Voidaan olettaa, että $\emptyset \notin \mathcal{S}$: toisin sanoen, että tyhjän joukon tunnistavilla Turingin koneilla ei ole tarkasteltavaa ominaisuutta. Jos nimittäin $\emptyset \in \mathcal{S}$, voidaan ensin osoittaa, että ominaisuus $\bar{\mathcal{S}} = \mathrm{RE} - \mathcal{S}$ on ratkeamaton, ja päätellä edelleen tästä että myös ominaisuus $\mathcal{S}$ on ratkeamaton. (Koska $\mathrm{codes}(\bar{\mathcal{S}}) = \overline{\mathrm{codes}(\mathcal{S})}$.)

Koska $\mathcal{S}$ on epätriviaali, on olemassa jokin Turingin kone M_A, jolla on ominaisuus $\mathcal{S}$ — jolla siis $L(M_A) \neq \emptyset \in \mathcal{S}$.

Olkoon nyt M_{ENCODE} Turingin kone, joka muodostaa syötteenä annetusta merkkijonosta $c_M w$ seuraavanlaisen Turingin koneen M^w koodin.



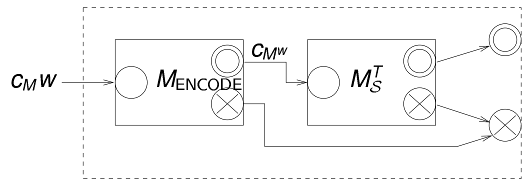
Jos syöte ei ole vaadittua muotoa, M_{ENCODE} päätyy hylkäävään lopputilaan.

Syötteellä x kone M^w toimii ensin kuten M syötteellä w. Jos M hyväksyy w:n, M^w toimii kuten kone M_A syötteellä x. Jos M hylkää w:n, myös M^w hylkää x:n. Kone M^w tunnistaa siis kielen

$$L(M^w) = \begin{cases} L(M_A), & \text{jos } w \in L(M); \\ \emptyset, & \text{jos } w \notin L(M). \end{cases}$$

Koska oletuksen mukaan $L(M_A) \in \mathcal{S}$ ja $\emptyset \notin \mathcal{S}$, on koneella M^w ominaisuus $\mathcal{S}$, jos ja vain jos $w \in L(M)$.

Oletetaan sitten, että ominaisuus $\mathcal{S}$ olisi ratkeava, so. että kielellä $\mathrm{codes}(\mathcal{S})$ olisi totaalinen tunnistajakone $M_{\mathcal{S}}^T$. Tällöin saataisiin edellisen todistuksen tapaan totaalinen tunnistajakone kielelle U yhdistämällä koneet M_{ENCODE} ja $M_{\mathcal{S}}^T$ seuraavasti:



Selvästi kone M_U^T on totaalinen, jos $M_{\mathcal{S}}^T$ on, ja

$$c_M w \in L(M_U^T) \Leftrightarrow c_{M^w} \in L(M_{\mathcal{S}}^T) = \mathrm{codes}(\mathcal{S}) \Leftrightarrow L(M^w) \in \mathcal{S} \Leftrightarrow w \in L(M).$$

Koska kieli U ei ole rekursiivinen, tämä on mahdotonta, mistä päätellään, ettei ominaisuus $\mathcal{S}$ voi olla ratkeava. □

6.8 Muita ratkeamattomuustuloksia

Lause 6.13 (Predikaattikalkyylin ratkeamattomuus; Church/Turing 1936)

Ei ole olemassa algoritmia, joka ratkaisisi, onko annettu ensimmäisen kertaluvun predikaattikalkyylin kaava ϕ validi ("loogisesti tosi", todistuva predikaattikalkyylin aksioomista).

□

Lause 6.14 ("Hilbertin 10. ongelma"; Matijasevitsh/Davis/Robinson/Putnam 1953–70)

Ei ole olemassa algoritmia, joka ratkaisisi, onko annetulla kokonaislukukertoimisella polynomilla $P(x_1, \dots, x_n)$ kokonaislukunollakohtia (so. jonoja $(m_1, \dots, m_n) \in \mathbf{Z}^n$, joilla $P(m_1, \dots, m_n) = 0$). Ongelma on ratkematon jo, kun $n = 15$ tai $\deg(P) = 4$. □

Eräiden kielioppiongelmien ratkeavuus, kun annettuna on kieliopit G ja G' Chomskyn hierarkian tietyltä tasolta i ja merkkijono w. Taulukossa $R \sim$ "ratkeava", $E \sim$ "ei ratkeava", $T \sim$ "aina totta".

Ongelma: onko	Taso i: 3	2	1	0
$w \in L(G)$?	R	R	R	E
$L(G) = \emptyset$?	R	R	E	E
$L(G) = \Sigma^*$?	R	E	E	E
$L(G) = L(G')$?	R	E	E	E
$L(G) \subseteq L(G')$?	R	E	E	E
$L(G) \cap L(G') = \emptyset$?	R	E	E	E
$L(G)$ säännollinen?	T	E	E	E
$L(G) \cap L(G')$ tyyppiä i?	T	E	T	T
$\overline{L(G)}$ tyyppiä i?	T	E	T	E

5. RAJOITTAMATTOMAT KIELIOPIT

Määritelmä 5.1 *Rajoittamaton kielioppi* t. *yleinen merkkijonomuunnossysteemi* on nelikko

$$G = (V, \Sigma, P, S),$$

missä

- V on kieliopin aakkosto;
- $\Sigma \subseteq V$ on kieliopin *päätemerkkien* joukko; $N = V - \Sigma$ on *välikemerkkien* t. *-symbolien* joukko;
- $P \subseteq V^+ \times V^*$ on kieliopin *sääntöjen* t. *produktioiden* joukko ($V^+ = V^* - \{\varepsilon\}$);
- $S \in N$ on kieliopin *lähtösymboli.*

Produktiota $(\omega, \omega') \in P$ merkitään tavallisesti $\omega \to \omega'$.

Merkkijono $\gamma \in V^*$ *tuottaa* t. *johtaa suoraan* merkkijonon $\gamma' \in V^*$ kieliopissa G, merkitään

$$\gamma \underset{G}{\Rightarrow} \gamma'$$

jos voidaan kirjoittaa $\gamma = \alpha\omega\beta$, $\gamma' = \alpha\omega'\beta$ ($\alpha, \beta, \omega' \in V^*$, $\omega \in V^+$), ja kieliopissa on produktio $\omega \to \omega'$.

Jos kielioppi G on yhteydestä selvä, merkitään $\gamma \Rightarrow \gamma'$.

Merkkijono $\gamma \in V^*$ *tuottaa* t. *johtaa* merkkijonon $\gamma' \in V^*$ kieliopissa G, merkitään

$$\gamma \underset{G}{\Rightarrow}^* \gamma'$$

jos on olemassa jono V:n merkkijonoja $\gamma_0, \gamma_1, \dots, \gamma_n$ ($n \geq 0$), siten että

$$\gamma = \gamma_0 \underset{G}{\Rightarrow} \gamma_1 \underset{G}{\Rightarrow} \dots \underset{G}{\Rightarrow} \gamma_n = \gamma'.$$

Jälleen, jos G on yhteydestä selvä, merkitään $\gamma \Rightarrow^* \gamma'$.

Merkkijono $\gamma \in V^*$ on kieliopin G *lausejohdos*, jos on $S \underset{G}{\Rightarrow}^* \gamma$.

Pelkästään päätemerkeistä koostuva G:n lausejohdos $x \in \Sigma^*$ on G:n *lause*.

Kieliopin G *tuottama* t. *kuvaama kieli* $L(G)$ koostuu G:n lauseista, s.o.:

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

Esimerkki. Rajoittamaton kielioppi ei-yhteydettömälle kielelle $\{a^k b^k c^k \mid k \geq 0\}$.

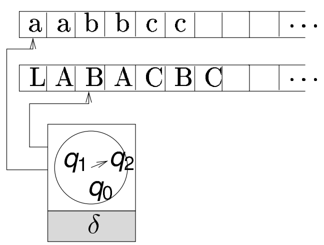
$$
\begin{array}{rcl@{\qquad\qquad}rcl}
S & \rightarrow & LT \mid \varepsilon & & & \\
T & \rightarrow & ABCT \mid ABC & aA & \rightarrow & aa \\
BA & \rightarrow & AB & aB & \rightarrow & ab \\
CB & \rightarrow & BC & bB & \rightarrow & bb \\
CA & \rightarrow & AC & bC & \rightarrow & bc \\
LA & \rightarrow & a & cC & \rightarrow & cc.
\end{array}
$$

Esimerkiksi lauseen *aabbcc* johto:

$$
\begin{array}{cllll}
\underline{S} & \Rightarrow L\underline{T} \Rightarrow LABC\underline{T} & \Rightarrow LABC\underline{CA}BC & \Rightarrow LA\underline{BA}CBC \\
 & \Rightarrow LAAB\underline{CB}C & \Rightarrow \underline{LA}ABBCC & \Rightarrow \underline{aA}BBCC \\
 & \Rightarrow a\underline{aB}BCC & \Rightarrow aa\underline{bB}CC & \Rightarrow aabb\underline{bC}C \\
 & \Rightarrow aabb\underline{cC} & \Rightarrow aabbcc. &
\end{array}
$$

Lause 5.1 Jos formaali kieli L voidaan tuottaa rajoittamattomalla kieliopilla, se voidaan tunnistaa Turingin koneella.

Todistus. Olkoon $G = (V, \Sigma, P, S)$ kielen L tuottava rajoittamaton kielioppi. Muodostetaan kielen L tunnistava kaksinauhainen epädeterministinen Turingin kone M_G seuraavasti:



Nauhalla 1 kone säilyttää kopiota syötejonosta. Nauhalla 2 on kullakin hetkellä jokin G:n lausejohdos, jota kone pyrkii muuntamaan syötejonon muotoiseksi. Toimintansa aluksi M_G kirjoittaa kakkosnauhalle kieliopin lähtösymbolin S.

Koneen M_G laskenta koostuu vaiheista. Kussakin vaiheessa kone:

(i) vie kakkosnauhan nauhapään epädeterministisesti johonkin kohtaan nauhalla;

(ii) valitsee epädeterministisesti jonkin G:n produktion, jota yrittää soveltaa valittuun nauhankohtaan (produktiot on koodattu M_G:n siirtymäfunktioon;

(iii) jos produktion vasen puoli sopii yhteen nauhalla olevien merkkien kanssa, M_G korvaa ao. merkit produktion oikean puolen merkeillä;

(iv) vaiheen lopuksi M_G vertaa ykkös- ja kakkosnauhan merkkijonoja toisiinsa: jos jonot ovat samat, kone siirtyy hyväksyvään lopputilaan ja pysähtyy, muuten aloittaa uuden vaiheen (kohta (i)). $\square$

Lause 5.2 Jos formaali kieli L voidaan tunnistaa Turingin koneella, se voidaan tuottaa rajoittamattomalla kieliopilla.

Todistus. Olkoon $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{acc}, q_{rej})$ kielen L tunnistava standardimallinen Turingin kone. Muodostetaan kielen L tuottava rajoittamaton kielioppi G_M seuraavasti.

Idea: Kieliopin G_M välikkeiksi otetaan (muiden muassa) kaikkia M:n tiloja $q \in Q$ edustavat symbolit. Koneen M tilanne $(q, u\underline{a}v)$ esitetään merkkijonona $[uqav]$. M:n siirtymäfunktion perusteella G_M:ään muodostetaan produktiot, joiden ansiosta

$$[uqav] \underset{G_M}{\Rightarrow} [u'q'a'v'] \quad \text{joss} \quad (q, uav) \underset{M}{\vdash} (q', u'a'v').$$

Siten M hyväksyy syötteen x, jos ja vain jos

$$[q_0x] \underset{G_M}{\Rightarrow}^* [uq_{acc}v]$$

joillakin $u, v \in \Sigma^*$.

Kaikkiaan kielioppiin G_M tulee kolme ryhmää produktioita:

1. Produktiot, joilla lähtösymbolista S voidaan tuottaa mikä tahansa merkkijono muotoa $x[q_0x]$, missä $x \in \Sigma^*$ ja [, q_0 ja] ovat G_M:n välikkeitä.
2. Produktiot, joilla merkkijonosta $[q_0x]$ voidaan tuottaa merkkijono $[uq_{acc}v]$, jos ja vain jos M hyväksyy x:n.
3. Produktiot, joilla muotoa $[uq_{acc}v]$ oleva merkkijono muutetaan tyhjäksi merkkijonoksi.

Kieleen $L(M)$ kuuluvan merkkijonon x tuottaminen tapahtuu tällöin seuraavasti:

$$S \overset{(1)}{\Rightarrow^*} x[q_0x] \overset{(2)}{\Rightarrow^*} x[uq_{acc}v] \overset{(3)}{\Rightarrow^*} x.$$

Määritellään siis $G = (V, \Sigma, P, S)$, missä

$$V = \Gamma \cup Q \cup \{S, T, [,], E_L, E_R\} \cup \{A_a \mid a \in \Sigma\},$$

ja produktiot P muodostuvat seuraavista kolmesta ryhmästä:

1. Alkutilanteen tuottaminen:

$$\begin{array}{lcll}
S & \to & T[q_0] & \\
T & \to & \varepsilon & \\
T & \to & aTA_a & (a \in \Sigma) \\
A_a[q_0 & \to & [q_0A_a & (a \in \Sigma) \\
A_ab & \to & bA_a & (a, b \in \Sigma) \\
A_a] & \to & a] & (a \in \Sigma)
\end{array}$$

2. M:n siirtymien simulointi ($a, b \in \Gamma$, $c \in \Gamma \cup \{[\}$) :

Siirtymät:	*Produktiot:*
$\delta(q, a) = (q', b, R)$	$qa \to bq'$
$\delta(q, a) = (q', b, L)$	$cqa \to q'cb$
$\delta(q, \rhd) = (q', \rhd, R)$	$q[\to [q'$
$\delta(q, \lhd) = (q', b, R)$	$q] \to bq']$
$\delta(q, \lhd) = (q', b, L)$	$cq] \to q'cb]$
$\delta(q, \lhd) = (q', \lhd, L)$	$cq] \to q'c]$

3. Lopputilanteen siivous:

$$\begin{array}{lcll} q_{\mathrm{acc}} & \to & E_L E_R & \\ q_{\mathrm{acc}}[& \to & E_R & \\ aE_L & \to & E_L & (a \in \Gamma) \\ [E_L & \to & \varepsilon & \\ E_R a & \to & E_R & (a \in \Gamma) \\ E_R] & \to & \varepsilon & \end{array}$$

□

Yhteysherkät kieliopit

Rajoittamaton kielioppi on *yhteysherkkä*, jos sen kaikki produktiot ovat muotoa $\omega \to \omega'$, missä $|\omega'| \geq |\omega|$, tai mahdollisesti $S \to \varepsilon$, missä S on lähtösymboli.

Lisäksi vaaditaan, että jos kieliopissa on produktio $S \to \varepsilon$, niin lähtösymboli S ei esiinny minkään produktion oikealla puolella.

Formaali kieli L on *yhteysherkkä*, jos se voidaan tuottaa jollakin yhteysherkällä kieliopilla.

Normaalimuoto: Jokainen yhteysherkkä kieli voidaan tuottaa kieliopilla, jonka produktiot ovat muotoa $S \to \varepsilon$ ja $\alpha A \beta \to \alpha\omega\beta$, missä A on välike ja $\omega \neq \varepsilon$. (Säännön $A \to \omega$ sovellus "kontekstissa" α_β.)

Lause 5.3 Formaali kieli L on yhteysherkkä, jos ja vain jos se voidaan tunnistaa epädeterministisellä Turingin koneella, joka ei tarvitse enempää työtilaa kuin syötejonon pituuden verran — siis koneella, jolla ei ole muotoa $\delta(q, \triangleleft) = (q', b, \Delta)$ olevia siirtymiä, missä $b \neq$ '$\triangleleft$'. □

Lauseen 5.3 koneita sanotaan *lineaarisesti rajoitetuiksi automaateiksi*.

Avoin ongelma ("LBA ?= DLBA"): onko epädeterminismi lauseessa 5.3 välttämätöntä?

Chomskyn hierarkia

Kielioppien, niillä tuotettavien kielten ja vastaavien tunnistusautomaattien ryhmittely:

Luokka 0: rajoittamattomat kieliopit / rekursiivisesti numeroituvat kielet / Turingin koneet.

Luokka 1: yhteysherkät kieliopit / yhteysherkät kielet / lineaarisesti rajoitetut automaatit.

Luokka 2: yhteydettömät kieliopit / yhteydettömät kielet / pinoautomaatit.

Luokka 3: oikealle ja vasemmalle lineaariset (säännölliset) kieliopit / säännölliset kielet / äärelliset automaatit.