

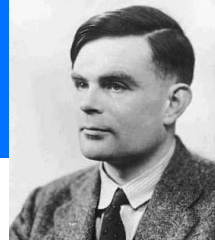
## 4. Laskennan universaaleja malleja

**Churchin-Turingin teesi** *Mikä tahansa mekaanisesti ratkeava ongelma voidaan ratkaista Turingin koneella*

Monet mekaanisen laskennan formalisoinnit ovat laskenta-voimaltaan ekvivalentteja Turingin koneiden kanssa:

- RAM-koneet, yksinkertaiset hajasaanti-tietokonemallit
- Ohjelmointikielät (1950-luku)
- Merkkijonomuunnossysteemit (Post, 1936 ja Markov, 1951)
- $\lambda$ -kalkyyli (Church, 1936)
- rekursiivisesti määritellyt funktiot (Gödel ja Kleene, 1936)

### 4.1 Turingin koneet



- Alan Turing (1912-1954) 1935-36
- Turingin kone on kuten äärellinen automaatti, jolla on käytössään (ääretön) työnauha
- Kone pystyy lukemaan ja kirjoittamaan työnauhan muistipaikkoihin
- Työnauhalla voidaan siirtyä molempiin suuntiin
- Koneen syöte annetaan työnauhalla, sen vasemmassa laidassa
- Koneella on sekä hyväksyvä, **accept**, että hylkäävä lopputila, **reject**
- Työnauhan loppuosa koostuu tyhjämerkeistä (`␣`), joka ei saa olla syöteaakkoston merkki

- Yhdessä siirtymäaskelella kone aina lukee merkin ja päättää tilansa sekä luetun merkin perusteella
  - uuden tilan,
  - nauhalle kirjoitettavan merkin
  - ja siirtymäsuunnan
- Jos kone pysähtyy hyväksyvässä lopputilassa, **accept**, syötemerkkijono kuuluu koneen tunnistamaan kieleen
- Jos kone pysähtyy hylkäävässä lopputilassa, **reject**, tai jos se ei pysähdy, niin syötemerkkijono ei kuulu kieleen

Formaalisti Turingin kone on seitsikko

$M = (Q, \Sigma, \Gamma, \delta, q_0, \text{accept}, \text{reject})$ , missä

- $Q$  on äärellinen **tilojen** joukko,
- $\Sigma$  on koneen **syöteaakkosto**,  $\_ \notin \Sigma$ ,
- $\Gamma$  on koneen **nauha-aakkosto**,  $\_ \in \Gamma$  ja  $\Sigma \subseteq \Gamma$ ,
- $q_0 \in Q$  on koneen **alkutila**,
- **accept**  $\in Q$  on **hyväksyvä lopputila**,
- **reject**  $\in Q$  on **hylkäävä lopputila** ja
- $\delta: Q' \times \Gamma \rightarrow Q \times \Gamma \times \{L, R\}$  on koneen **siirtymäfunktio**, missä

$$Q' = Q \setminus \{\text{accept}, \text{reject}\}$$

## Siirtymäfunktion arvo

$$\delta(q, a) = (q', b, \Delta)$$

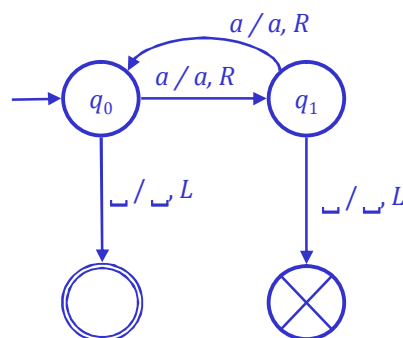
tarkoittaa, että

- ollessaan tilassa  $q$  ja lukiessaan nauhamerkin  $a$ ,
- kone siirtyy tilaan  $q'$ , kirjoittaa lukemaansa paikkaan merkin  $b$  ja siirtää nauhapäätä yhden paikan verran suuntaan  $\Delta \in \{L, R\}$

Siirtymäfunktion arvoilta vaaditaan, että

1. Nauhan vasemmanpuoleisimmasta muistipaikasta ei voida siirtyä vasemmalle. Jos tätä yritetään, pysyy lukupää paikallaan.
2. jos  $b = \_$ , niin  $a = \_$  ja  $\Delta = L$ .

Kielen  $\{a^{2k} \mid k \geq 0\}$  tunnistava Turingin kone:



Turingin koneen tilannetta merkitään  $u q a v$ ,  
missä

- $q \in Q$ ,
- $a \in \Gamma \cup \{\epsilon\}$
- $u, v \in \Gamma^*$

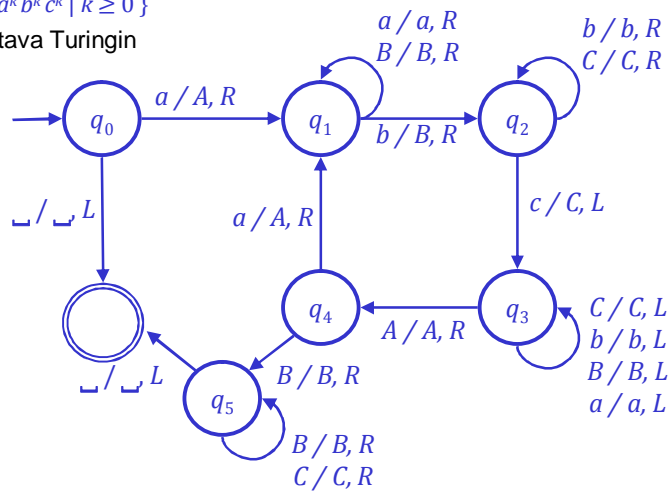
Intuitiivisesti

- kone on tilassa  $q$ ,
- nauhapään kohdalla on merkki  $a$ ,
- nauhan sisältö alusta nauhapään vasemmalle puolelle on  $u$  ja
- nauhapään oikealta puolelta käytetyn osan loppuun  $v$

- Koneen alkutilanne syötteellä  $w$  on  $q_0 w$
- Merkintä  $u a q_i b v \succ_M u q_j a c v$  tarkoittaa, että tilanne  $u a q_i b v$  johtaa suoraan tilanteeseen  $u q_j a c v$
- Se määritellään seuraavasti
  - jos  $\delta(q_i, b) = (q_j, c, L)$ , niin  $u a q_i b v \succ_M u q_j a c v$ ,
  - jos  $\delta(q_i, b) = (q_j, c, R)$ , niin  $u a q_i b v \succ_M u a c q_j v$ ,
  - Kun nauhapää on työnauhan vasemmassa laidassa, on tilanne  $q_i b v$  ja
    - siirtymä  $\delta(q_i, b) = (q_j, c, L)$  johtaa suoraan tilanteeseen  $q_j c v$
    - siirtymä  $\delta(q_i, b) = (q_j, c, R)$  johtaa suoraan tilanteeseen  $c q_j v$
  - Kun nauhapää on työnauhan käytetyn osan oikeassa laidassa, on tilanne  $u a q_i \_ ja$ 
    - siirtymä  $\delta(q_i, \_) = (q_j, c, L)$  johtaa suoraan tilanteeseen  $u q_j a c$
    - siirtymä  $\delta(q_i, \_) = (q_j, c, R)$  johtaa suoraan tilanteeseen  $u a c q_j \_$
    - vain edellisessä tilanteessa voi olla  $c = \_$

- Siirtymäfunktion arvo on määrittelemätön lopputiloilla **accept** ja **reject**, joten
  - tilanteet joissa esiintyy **accept** tai **reject** eivät johda mihinkään muuhun tilanteeseen, vaan niissä kone **pysähtyy**
- Turingin kone  $M$  hyväksyy syötteen  $w$  jos on olemassa koneen tilanteiden jono  $C_1, C_2, \dots, C_k$  s.e.
  1.  $C_1$  on koneen  $M$  alkutilanne syötteellä  $w$ ,
  2. Kukin  $C_i$  johtaa suoraan tilanteeseen  $C_{i+1}$  ja
  3.  $C_k$  on hyväksyvä tilanne.
- Koneen  $M$  tunnistama kieli  $L(M)$  on kaikkien sen hyväksymien merkkijonojen joukko

Kielen  $\{a^k b^k c^k \mid k \geq 0\}$   
tunnistava Turingin  
kone:



## 4.1.1 Turingin koneiden laajennuksia

Seuraavat Turingin koneiden laajennukset eivät muuta koneilla tunnistettavien kielten luokkaa

### Moniuraiset koneet

- Koneen nauha koostuu  $k$ :sta rinnakkaisesta urasta
- Koneella on edelleen vain yksi nauhapää
- Kone lukee ja kirjoittaa kullekin uralle jokaisessa laskenta-askellessa
- Jokaisen uran sisältö "topataan" yhtä pitkäksi erityisen tyhjämärkin (#) avulla
- Syöte annetaan uran 1 vasemmassa laidassa

- Moniuraista Turingin konetta on helppo simuloida standardimallisella
- Riittää ottaa koneen nauha-aakkostoksi sellainen, jonka merkeillä voidaan esittää (tarvittaessa) kaikki  $k$ -uraisen koneen mahdolliset  $k$  päällekkäistä merkkiä
- Esim.:

$$\delta(q, (a_1, \dots, a_k)) \rightarrow \hat{\delta} \left( q, \begin{bmatrix} a_1 \\ \vdots \\ a_k \end{bmatrix} \right)$$

- Tekemällä edellisen kaltainen aakkoston laajennus systemaattisesti ja muuntamalla syötteen kaikki merkit

$$a \rightarrow \begin{bmatrix} a \\ \# \\ \vdots \\ \# \end{bmatrix}$$

uralle 1, voidaan moniuraista Turingin konetta simuloida standardimallisella

**Lause** *Jos formaali kieli voidaan tunnistaa  $k$ -uraisella Turingin koneella, niin se voidaan tunnistaa myös standardimallisella Turingin koneella*

## Moninauhaiset koneet

- Nyt Turingin koneella voi olla  $k$  työnauhaa, joilla kullakin on oma nauhapäänsä
- Kone lukee ja kirjoittaa kullekin nauhalle jokaisessa laskenta-askellessa
- Nauhapäät siirtyvät toisistaan riippumatta
- Syöte annetaan nauhan 1 vasemmassa laidassa
- Moninauhaista Turingin konetta on helppo simuloida moniuraisella
- Kutakin nauhaa vastaten otetaan kaksi uraa, joista toinen vastaa nauhaa ja sen sisältöä, toisella uralla on merkki nauhapään sijaintikohdassa — merkkiä siirretään simuloiden nauhapään siirtymistä moninauhaisessa koneessa

- Lukemalla kaikki urat vasemmalta oikealle saadaan selville kaikkien nauhapäiden kohdilla olevat merkit
- Nyt tiedetään nauhoille kirjoitettavat merkit ja nauhapäiden siirtosuunnat
- Nämä voidaan toteuttaa/simuloida oikealta vasemmalle suuntautuvan pyyhkäisyn yhteydessä

**Lause 3.13** *Jos formaali kieli voidaan tunnistaa  $k$ -nauhaisella Turingin koneella, niin se voidaan tunnistaa myös standardimallisella Turingin koneella*

## Epädeterministiset koneet

- Sallimalla Turingin koneen siirtymäfunktiolle "ennustuskyky" voidaan määritellä epädeterministinen Turingin kone
- Sinänsä epärealistinen mekaanisen laskennan malli, mutta kelpaa ongelmien formaaliin kuvaamiseen ja ratkeavuuden osoittamiseen
- Vaikkakin epädeterministiset Turingin koneet ovat kielten tunnistuskyvyltään ekvivalenteja determinististen kanssa, niin silti ne ovat keskeisessä asemassa vaativuusteoriassa

Formaalisti epädeterministinen Turingin kone on seitsikko

$M = (Q, \Sigma, \Gamma, \delta, q_0, \text{accept}, \text{reject})$ , missä

$$\delta: Q' \times \Gamma \rightarrow \mathcal{P}(Q \times \Gamma \times \{L, R\}),$$

jossa  $Q' = Q \setminus \{\text{accept}, \text{reject}\}$

Siirtymäfunktion arvo

$$\delta(q, a) = \{(q_1, b_1, \Delta_1), \dots, (q_k, b_k, \Delta_k)\}$$

tarkoittaa, että

- ollessaan tilassa  $q$  ja lukiessaan merkin  $a$
- kone voi valita mieleisensä (tehtävänsä kannalta parhaan) kolmikön  $(q_i, b_i, \Delta_i)$

• Ei-negatiivisten *yhdistettyjen* lukujen tunnistaminen:

$$n = pq? \quad (p, q > 1)$$

- Luku, joka ei ole yhdistetty on **alkuluku**
- Kaikki tunnetut deterministiset yhdistettyjen lukujen testit joutuvat pahimmassa tapauksessa käymään läpi suuren joukon potentiaalisia tekijöitä (salauksen perusta)
- Epädeterministisellä Turingin koneella yhdistettyjen lukujen "tunnistaminen" kuitenkin on helppoa
- Epädeterministinen Turingin kone ei anna algoritmia, vaan on vain laskennallinen kuvaus yhdistetyille luvuille

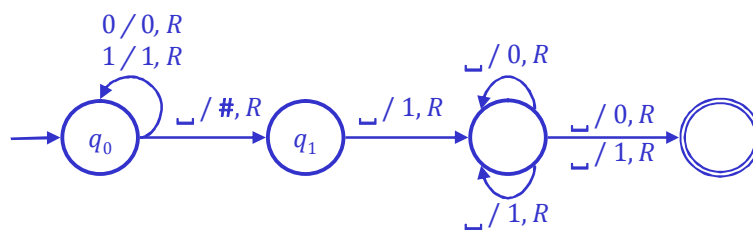
### Tarvittavia Turingin koneita

- **CHECK-MULT** tunnistaa kielen  
 $\{ n\#p\#q \mid n, p, q \in \{0, 1\}^*, n = pq \}$
- **GEN-INT** tuottaa mv. kokonaisluvun ( $>1$ ) binäärimuodossa nauhan loppuun
- **GO-START** siirtää nauhapään osoittamaan nauhan ensimmäistä merkkiä

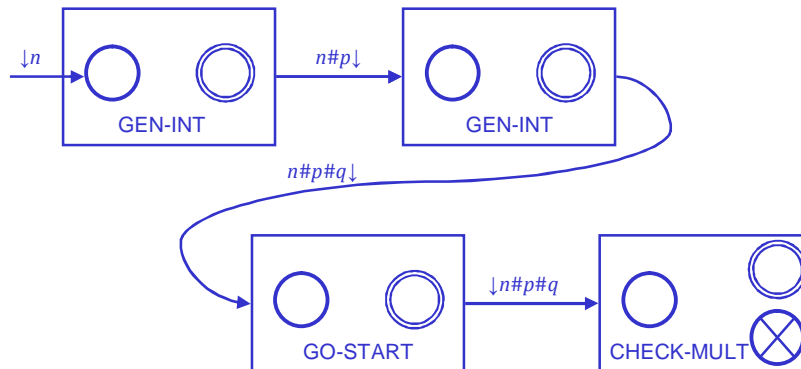
Jotta voidaan muodostaa kone

- **TEST-COMPOSITE**, joka tunnistaa kielen  
 $\{ n \in \{0, 1\}^* \mid n \text{ on yhdistetty luku} \}$

### Turingin kone GEN-INT



## Turingin kone TEST-COMPOSITE



## Alkulukutestaus

- Vuonna 2002 intialaiset Agarwal, Kayal & Saxena onnistuivat kehittämään ensimmäisen "tehokkaan" ( $n^{12}$ ) deterministisen alkuluvun tunnistavan algoritmin
- Vuosisatoja tutkittu ongelma (Eratosteneen seula, noin vuodelta 240 ennen ajanlaskun alkua)
- AKS-algoritmi lainaa tekniikoita satunnaisesti toimivilta alkulukutestauksen algoritmeilta, jotka edelleenkin lienevät käyttökelpoisempia menetelmiä
- Koska yhdistettyjen lukujen tunnistaminen on käänteinen ongelma, niin sekin siis ratkeaa polynomisessa ajassa deterministisesti
- Sen sijaan tehokasta tekijöihin jakoa ei tunneta (sen kuitenkin uskotaan olevan mahdollista)



**Lause 3.16** *Jos formaali kieli voidaan tunnistaa epädeterministisellä Turingin koneella, niin se voidaan tunnistaa myös standardimallisella deterministisellä Turingin koneella*

- Epädeterminististä konetta voidaan simuloida kolmenauhaisella Turingin koneella, joka läpikäy systemaattisesti epädeterministisen koneen mahdollisia laskentoja
- Nauha 1 säilyttää syötteen
- Nauha 2 simuloi epädeterministisen koneen työnauhaa
- Nauhalla 3 pidetään kirjaa mahdollisten laskentojen etenemisestä



- Epädeterministisen koneen mahdollisten laskentapolkujen muodostamaa puuta on käytävä läpi leveysuuntaisesti, ei syvyysuuntaisesti
- Olkoon  $b$  Turingin koneen tilojen seuraajavaihtoehtojen suurin määrä
- Kukin puun solmu voidaan indeksoida aakkoston  $\{1, 2, \dots, b\}$  merkkijonolla
- Esimerkiksi 231 on solmu, joka on juuren toisen lapsen kolmannen lapsen ensimmäinen lapsi
- Kaikki merkkijonot eivät vastaa koneen sallittuja laskentoja (puun solmuja), ne hylätään
- Merkkijonojen kuvaamien laskentavaihtoehtojen läpikäynti kanonisessa järjestyksessä käy puun läpi leveysuunnassa

### Toimintaidea:

- Kopioi syöte nauhalta 1 nauhalle 2
- Simuloi nauhan 3 kertomaa, (kanonisessa) järjestyksessä seuraavaa epädeterministisen koneen toimintavaihtoehtoa tehden työnauhalle kohdistuvat päivitykset nauhalle 2
- Huom.: siirtymävaihtoehtoja (seuraajatilanteita) on äärellinen määrä
- Systemaattinen epädeterministisen koneen mahdollisten laskentojen läpikäynti johtaa hyväksyvään tilaan vain jos alkuperäisellä koneella on syöteen hyväksyvä laskenta
- Jos hyväksyvää laskentaa ei ole, niin muodostettu kone ei pysähdy

## 4.2 Kieliopeista

- Kontekstittomissa kieliopeissa sallitaan vain välikkeiden korvaaminen merkkijonolla
- **Rajoittamattomissa kieliopeissa** i. yleisissä muunnossysteemeissä sallitaan minkä tahansa epätyhjän (välikkeiden ja päätteiden muodostaman) merkkijonon korvaaminen toisella (mahdollisesti tyhjällä merkkijonolla)

Rajoittamaton kielioppi on nelikko  $G = (V, \Sigma, R, S)$ , missä

- $V$  on kielioopin *muuttujien* eli välikemerkkien joukko,
- $\Sigma$  on kielioopin *päätemerkkien* joukko,
- $\Gamma = V \cup \Sigma$  on  $G$ :n aakkosto,
- $R \subseteq \Gamma^* \times \Gamma^*$  on kielioopin sääntöjen joukko ja
- $S \in V$  on kielioopin lähtösymboli

$(w, w') \in R$  merk.  $w \rightarrow w'$

- Olk.
  - $G = (V, \Sigma, R, S)$ ,
  - merkkijonot  $v \in \Gamma^+$  ja  $u, w, x \in \Gamma^*$  ja
    - $v \rightarrow x$  produktio  $R$ :ssä
  - $uvw$  tuottaa suoraan merkkijonon  $uxw$  kieliopissa  $G$ ,  
 $uvw \succ_G uxw$
  - Merkkijono  $v$  tuottaa merkkijonon  $w$  kieliopissa  $G$ ,  
 $v \ggg_G w$ ,

jos on olemassa jono  $v_1, v_2, \dots, v_k \in \Gamma^*$  ( $k \geq 0$ ) s.e.

$$v \succ_G v_1 \succ_G v_2 \succ_G \dots \succ_G v_k \succ_G w$$
  - $k=0$ :  $v \ggg_G v$  millä tahansa  $v \in \Gamma^*$

- $u \in \Gamma^*$  on  $G$ :n lausejohdos, jos  $S \ggg_G u$
- Pelkistä päätemerkeistä koostuva lausejohdos  $w \in \Gamma^*$  on  $G$ :n lause
- $G$ :n tuottama kieli koostuu lauseista  
 $L(G) = \{ w \in \Gamma^* \mid S \ggg_G w \}$

Kieli  $\{ a^k b^k c^k \mid k \geq 0 \}$  ei ole kontekstiton; se voidaan tuottaa rajoittamattomalla kieliopilla, joka

1. Tuottaa välikeyonon  $L(ABC)^k$  (tai  $\varepsilon:n$ )
2. Järjestää välitteet aakkosjärjestykseen  $\Leftrightarrow LA^k B^k C^k$
3. Muuttaa välitteet päätemerkeiksi

$S \rightarrow LT \mid \varepsilon$   
 $T \rightarrow ABCT \mid ABC$

$BA \rightarrow AB$   
 $CB \rightarrow BC$   
 $CA \rightarrow AC$

$LA \rightarrow a$   
 $aA \rightarrow aa$   
 $aB \rightarrow ab$   
 $bB \rightarrow bb$   
 $bC \rightarrow bc$   
 $cC \rightarrow cc$

Esimerkiksi lause *abbcc* voidaan johtaa seuraavasti

$S \Rightarrow LT \Rightarrow LABCT \Rightarrow LABCABC$   
 $\Rightarrow LABACBC \Rightarrow LAABCBC$   
 $\Rightarrow LAABBCC \Rightarrow aABBCC$   
 $\Rightarrow aaBBCC \Rightarrow aabBCC$   
 $\Rightarrow aabbCC \Rightarrow aabbcC$   
 $\Rightarrow aabbcc$

**Lause** Jos formaali kieli  $L$  voidaan tuottaa rajoittamattomalla kieliopilla, niin se voidaan tunnistaa Turingin koneella.

**Todistus.** Olk.  $G = (V, \Sigma, R, S)$  kielen  $L$  tuottava rajoittamaton kielioppi. Muodostetaan kaksinauhainen epädeterministinen Turingin kone  $M_G$  kielen  $L$  tunnistamiseen.

$M_G$  säilyttää nauhalla 1 syötejonoa. Nauhalla 2 puolestaan on jokin  $G$ :n lausejohdos, jonka kone pyrkii muuntamaan syötejonoksi.

Lähtötilanteessa nauhalla 2 on kieliopin lähtösymboli  $S$ .

Koneen  $M_G$  laskenta toistaa seuraavia vaiheita



1. nauhan 2 nauhapää siirretään epädeterministisesti johonkin kohtaan nauhalla;
2. valitaan epädeterministisesti jokin  $G$ :n produktio ja yritetään soveltaa sitä valittuun nauhankohtaan
3. jos nauhalla olevat merkit sopivat yhteen produktion vasemman puolen kanssa,  $M_G$  korvaa ne nauhalla 2 produktion oikean puolen merkeillä
4. verrataan nauhojen 1 ja 2 sisältämiä merkkijonoja toisiinsa;
  - a) jos ne ovat samat, niin Turingin kone siirtyy hyväksyvään lopputilaan ja pysähtyy,
  - b) muuten palataan kohtaan 1

□



**Lause** Jos formaali kieli  $L$  voidaan tunnistaa Turingin koneella, niin se voidaan tuottaa rajoittamattomalla kielioilla.

**Todistus.** Olk.  $M = (Q, \Sigma, \Gamma, \delta, q_0, \text{accept}, \text{reject})$  kielen  $L$  tunnistava standardimallinen Turingin kone.

Muodostetaan kielen  $L$  tuottava rajoittamaton kelioppi  $G_M$ .

Kieliopin välikkeiksi otetaan kaikkia  $M$ :n tiloja  $q \in Q$  edustavat symbolit. Koneen tilanne  $u q av$  esitetään merkkijonona  $[uqav]$ .

$M$ :n siirtymäfunktion perusteella  $G_M$ :ään muodostetaan produktiot s.e.

$$[uqav] \gg_{G_M} [u'q'a'v'] \Leftrightarrow u q av \gg_M u' q' a' v'$$

Tällöin

$$x \in L(M) \Leftrightarrow [q_0x] \gg_{G_M} [u\text{accept}v], u, v \in \Sigma^*$$

$G_M$ :ssä on kolmenlaisia produktioita:

1. ne, joilla lähtösymbolista voidaan tuottaa mikä tahansa merkkijono  $x[q_0x]$ ,  $x \in \Sigma^*$  ja  $[, ], q_0 \in V$

$$\begin{aligned} S &\rightarrow T[q_0] \\ T &\rightarrow \varepsilon \\ T &\rightarrow aTA_a \\ A_a[q_0] &\rightarrow [q_0A_a \\ A_a b &\rightarrow bA_a \\ A_a] &\rightarrow a] \end{aligned}$$

## 2. ne, jotka simuloivat Turingin koneen siirtymäfunktiota

$$\begin{array}{ll}
 \delta(q, a) = (q', b, R) & qa \rightarrow bq' \\
 \delta(q, a) = (q', b, L) & cqa \rightarrow q'cb \\
 \delta(q, \sqcup) = (q', b, R) & q] \rightarrow bq'] \\
 \delta(q, \sqcup) = (q', b, L) & cq] \rightarrow q'cb] \\
 \delta(q, \sqcup) = (q', \sqcup, L) & cq] \rightarrow q'c]
 \end{array}$$

## 3. ne, joilla muotoa $[uacceptv]$ oleva merkkijono muutetaan tyhjäksi

$$\begin{array}{l}
 \mathbf{accept} \rightarrow E_L E_R \\
 aE_L \rightarrow E_L \\
 [E_L \rightarrow \varepsilon \\
 E_R a \rightarrow E_R \\
 E_R ] \rightarrow \varepsilon
 \end{array}$$

Nyt kieleen  $L(M)$  kuuluva merkkijono  $x$  voidaan tuottaa seuraavasti

$$S \ggg_1 x[q_0x] \ggg_2 x[uacceptv] \ggg_3 x$$

□