Számítási modellek

3. előadás

Az algoritmikus fogalmának modelljei

Az 1930-as évektől egyre nagyobb igény mutatkozott az algoritmus matematikai modelljének megalkotására. Egymástól függetlenül több bíztató kísérlet is született:

- Kurt Gödel: rekurzív függvények
- Alonso Church: λ-kalkulus
- Alan Turing: Turing gép

Melyik az „igazi”, melyiket válasszuk?

Az 1930-as évek második felétől sorra születtek olyan tételek, melyek ezen modellek megegyező számítási erejét mondták ki.
A későbbiek során számos további számítási modellről sikerült bebizonyítani, hogy számítási erejük a Turing gépekkel ekvivalens.
Például:

- 0. típusú grammatika
- veremautomata 2 vagy több veremmel
- C, Java, stb.

A Church-Turing tézis

Valójában nem ismerünk olyan algoritmikus rendszert, amelyről tudnánk, hogy erősebb a Turing gépnél, és a legtöbb algoritmikus rendszerre bizonyított, hogy gyengébb, vagy ekvivalens.

Már a 30-as években megfogalmazásra került a következő:

Church-Turing tézis

Minden formalizálható probléma, ami megoldható algoritmussal, az megoldható Turing géppel is.

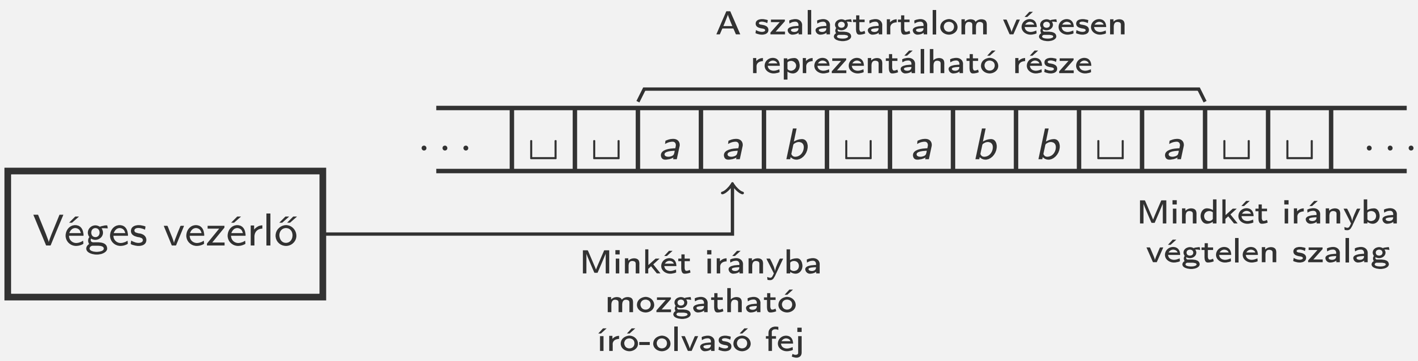
(illetve bármilyen, a Turing géppel azonos számítási erejű absztrakt modellben)

NEM TÉTEL!!!

A Church-Turing tézis nem bizonyítható, hiszen nem egy formális matematikai állítás, az algoritmus intuitív fogalmát használja.

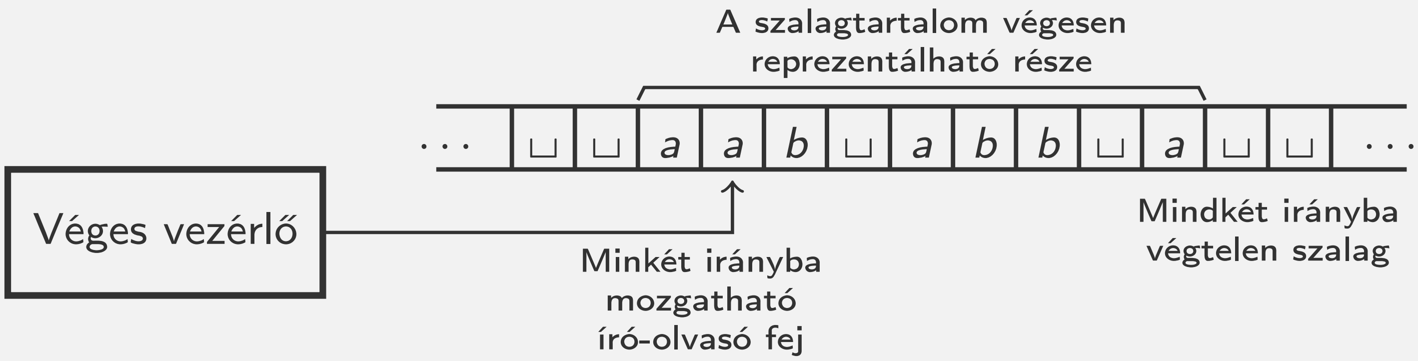
Ha elfogadjuk a tézis igazságát, a Turing gép (illetve bármely a Turing gépekkel ekvivalens modell) informálisan tekinthető az algoritmus matematikai modelljének.

Turing gépek – Informális kép



- a Turing gép (TG) az algoritmus egyik lehetséges modellje
- a TG egyetlen programot hajt végre (de bármely inputra!!!), azaz tekinthető egy célszámítógépnek.
- informálisan a gép részei a vezérlőegység (véges sok állapottal), egy mindkét irányba végtelen szalag, és egy mindkét irányba lépni képes író-olvasó fej
- kezdetben egy input szó van a szalagon (ε esetén üres), a fej ennek első betűjéről indul, majd a szabályai szerint működik. Ha eljut az elfogadó állapotába elfogadja, ha eljut az elutasító állapotába elutasítja az inputot. Van egy harmadik lehetőség is: nem jut el soha a fenti két állapotába, "végtelen ciklusba" kerül.

Turing gépek – Informális kép



- a gép alapértelmezetten determinisztikus, minden (nem megállási) konfigurációnak van és egyértelmű a rákövetkezője.
- végtelen szalag: potenciálisan végtelen tár
- egy $\mathcal{P}$ probléma példányait egy megfelelő ábécé felett elkódolva a probléma „igen”-példányai egy $L(\mathcal{P})$ formális nyelvet alkotnak. $L(\mathcal{P})$ (és így a probléma maga is) algoritmikusan eldönthető, ha van olyan mindig termináló Turing gép, mely pontosan $L(\mathcal{P})$ szavait fogadja el.
- a Church-Turing tézis értelmében informálisan úgy gondolhatjuk, hogy éppen a TG-pel eldönthető problémák (nyelvek) az algoritmikusan eldönthető eldöntési problémák.

Turing gépek

Definíció

A **Turing gép** (továbbiakban röviden TG) egy $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n \rangle$ rendezett hetes, ahol

- Q az állapotok véges, nemüres halmaza,
- $q_0, q_\mathrm{i}, q_\mathrm{n} \in Q$, q_0 a kezdő- q_i az elfogadó- és q_n az elutasító állapot,
- Σ és Γ ábécék, a bemenő jelek illetve a szalagszimbólumok ábécéje úgy, hogy $\Sigma \subseteq \Gamma$ és $\sqcup \in \Gamma \backslash \Sigma$.
- $\delta : (Q \backslash \{q_\mathrm{i}, q_\mathrm{n}\}) \times \Gamma \rightarrow Q \times \Gamma \times \{L, S, R\}$ az átmenet függvény. δ értelmezési tartománya $(Q \backslash \{q_\mathrm{i}, q_\mathrm{n}\}) \times \Gamma$.

$\{L, S, R\}$ elemeire úgy gondolhatunk mint a TG lépéseinek irányaira (balra lépés, helyben maradás, jobbra lépés).

Megjegyzés: Elég 2 irány, a helyben maradó lépések helyettesíthetők egy jobbra és egy balra lépéssel egy, csak erre az átmenetre használt új állapoton keresztül.

Turing gépek

Konfiguráció

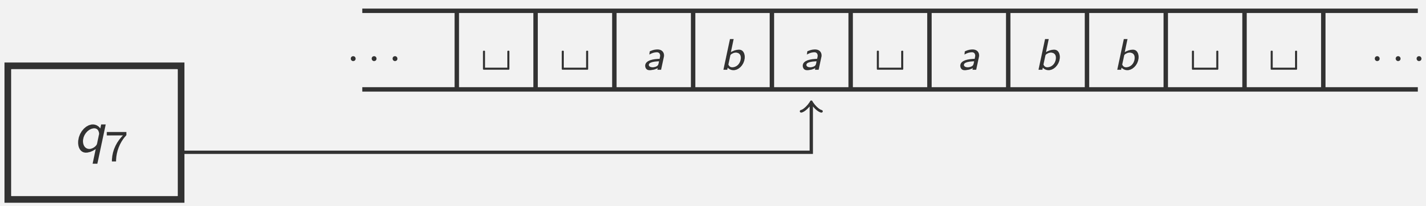
A TG működtetését a gép konfigurációival írhatjuk le.

Definíció

A TG **konfigurációja** egy uqv szó, ahol $q \in Q$ és $u, v \in \Gamma^*$, $v \neq \varepsilon$.

Az uqv konfiguráció egy tömör leírás a TG aktuális helyzetéről, mely a gép további működése szempontjából minden releváns információt tartalmaz: a szalag tartalma uv (uv előtt és után a szalagon már csak $\sqcup$ van), a gép a q állapotban van és az író-olvasó fej a v szó első betűjén áll.

Példa:



A fenti helyzetet az $abq_7a\sqcup abb$ konfigurációval írhatjuk le.

Turing gépek

Konfiguráció

Definíció

A gép egy $u \in \Sigma^*$-beli szóhoz tartozó **kezdőkonfigurációja** a $q_0 u \sqcup$ szó. (Vagyis $q_0 u$, ha $u \neq \varepsilon$ és $q_0 \sqcup$, ha $u = \varepsilon$).

Elfogadó konfigurációi azon konfigurációk, melyre $q = q_{\mathrm{i}}$.

Elutasító konfigurációi azon konfigurációk, melyre $q = q_{\mathrm{n}}$.

Az elfogadó és elutasító konfigurációk közös neve **megállási konfiguráció**.

Két konfigurációt **azonosnak tekintünk**, ha csak balra/jobbra hozzáírt $\sqcup$-ekben térnek el egymástól.

Például a $\sqcup abq_2 \sqcup$ és a $abq_2 \sqcup \sqcup$ konfigurációk azonosak.

Turing gépek

Konfigurácóátmenet

Jelölje C_M egy M TG-hez tartozó lehetséges konfigurációk halmazát. Az M Turing gép $\vdash \subseteq C_M \times C_M$ **konfigurációátmenet relációját** az alábbiak szerint definiáljuk. (közvetlen v. egylépéses konfigurációátmenet)

Definíció

Legyen $uqav$ egy konfiguráció, ahol $a \in \Gamma$, $u, v \in \Gamma^*$.

- Ha $\delta(q, a) = (r, b, R)$, akkor $uqav \vdash ubrv'$, ahol $v' = v$, ha $v \neq \varepsilon$, különben $v' = \sqcup$,
- ha $\delta(q, a) = (r, b, S)$, akkor $uqav \vdash urbv$,
- ha $\delta(q, a) = (r, b, L)$, akkor $uqav \vdash u'rcbv$, ahol $c \in \Gamma$ és $u'c = u$, ha $u \neq \varepsilon$, különben $u' = u$ és $c = \sqcup$.

Példa: Tegyük fel, hogy $\delta(q_2, a) = (q_5, b, L)$ és $\delta(q_5, c) = (q_1, \sqcup, R)$. Legyen továbbá $C_1 = bcq_2a\sqcup b$, $C_2 = bq_5cb\sqcup b$, $C_3 = b\sqcup q_1b\sqcup b$. Ekkor $C_1 \vdash C_2$ és $C_2 \vdash C_3$.

Turing gépek

Konfigurációátmenet, felismert nyelv

Az $\vdash^* \subseteq C_M \times C_M$ többlépéses konfigurációátmenet reláció a $\vdash$ reláció reflexív, tranzitív lezártja.

Példa: (folytatás) Legyen C_1, C_2, C_3 ugyanaz, mint a fenti példában. Mivel $C_1 \vdash C_2$ és $C_2 \vdash C_3$ is teljesült, ezért $C_1 \vdash^* C_1$, $C_1 \vdash^* C_2$, $C_1 \vdash^* C_3$ is fennállnak.

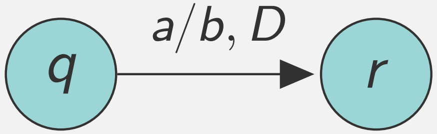
Definíció

Az M TG által **felismert nyelv**
$L(M) = \left\{u \in \Sigma^* \mid q_0 u \sqcup \vdash^* x q_i y \text{ valamely } x, y \in \Gamma^*, y \neq \varepsilon \text{ -ra}\right\}$.

Figyeljük meg, hogy $L(M)$ csak Σ feletti szavakat tartalmaz.

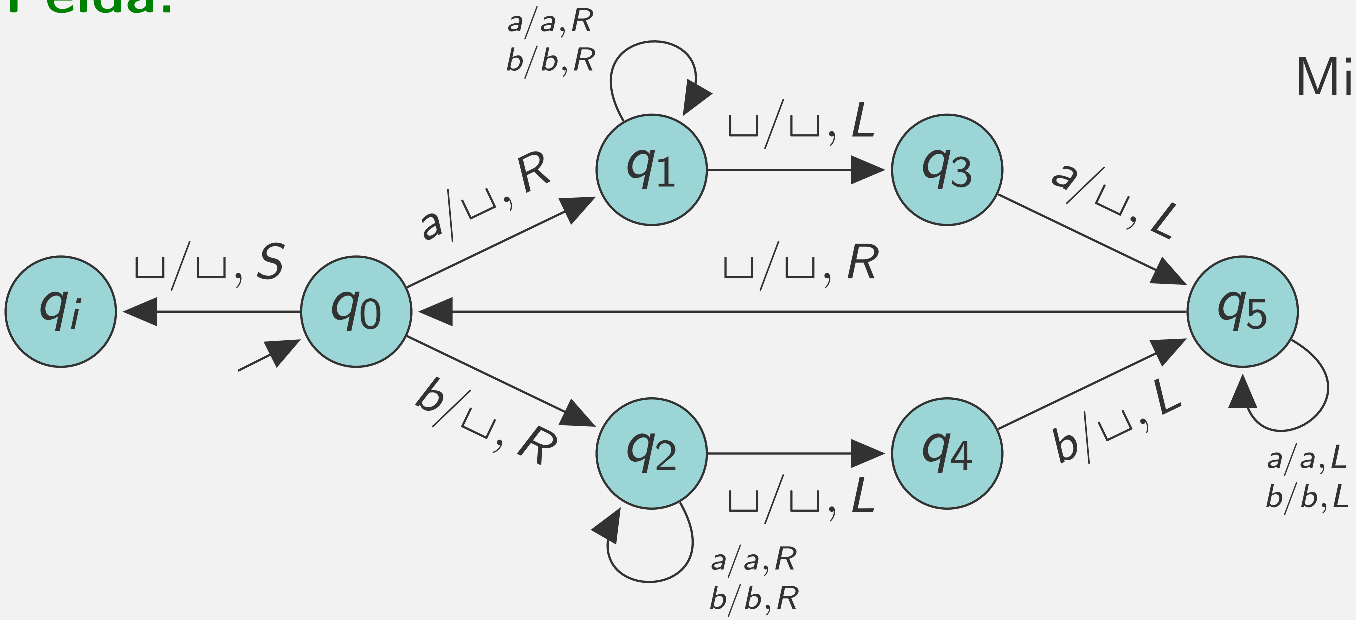
Turing gépek – Példa

Az **átmenetdiagram**:



megfelel a $\delta(q, a) = (r, b, D)$ átmenetnek $(q, r \in Q, a, b \in \Gamma, D \in \{L, S, R\})$

Példa:



Minden más átmenet q_n-be megy

$L(M) = \{ww^{-1} \mid w \in \{a, b\}^*\}$. Példa konfigurációátmenetek sorozatára az aba input esetén:

$q_0 aba \vdash q_1 ba \vdash b q_1 a \vdash ba q_1 \sqcup \vdash b q_3 a \vdash q_5 b \vdash q_5 \sqcup b \vdash q_0 b \vdash q_2 \sqcup \vdash q_4 \sqcup \vdash q_n \sqcup$.

Turing gépek időigénye

Definíció

Az M Σ inputábécéjű TG **futási ideje** az $u \in \Sigma^*$ szóra a konfigurációátmenetek száma az u-hoz tartozó kezdőkonfigurációból egy megállási konfigurációba.

Vagyis a működés során megtett lépések száma. Ha a gép nem áll meg egy szóra, akkor a futási ideje erre a szóra ∞.

Példa: Az előző példában a TG-ünk az *aba* inputra 10 lépésben jutott megállási konfigurációba, így *aba*-ra a futási idő 10.

Definíció

Legyen $f : \mathbb{N} \to \mathbb{N}$ egy függvény. Azt mondjuk, hogy az M Σ inputábécéjű TG **$f(n)$ időkorlátos** (vagy M $f(n)$ időigényű), ha minden $u \in \Sigma^*$ szóra M futási ideje $\leqslant f(|u|)$.

Példa: Meggondolható, hogy az előző példában látott TG $O(n)$ iterációt végez iterációnként $O(n)$ lépéssel, így $O(n^2)$ időkorlátos.

Rekurzíve felsorolható és rekurzív nyelvek

Definíció

Egy $L \subseteq \Sigma^*$ nyelv **Turing-felismerhető**, ha $L = L(M)$ valamely M TG-re.

Egy $L \subseteq \Sigma^*$ nyelv **eldönthető**, ha létezik olyan M TG, mely minden bemeneten megállási konfigurációba jut és $L(M) = L$.

A Turing-felismerhető nyelveket **rekurzívan felsorolható**nak (vagy **parciálisan rekurzívnak**, vagy **félig eldönthetőnek**) az eldönthető nyelveket pedig **rekurzív**nak is nevezik.

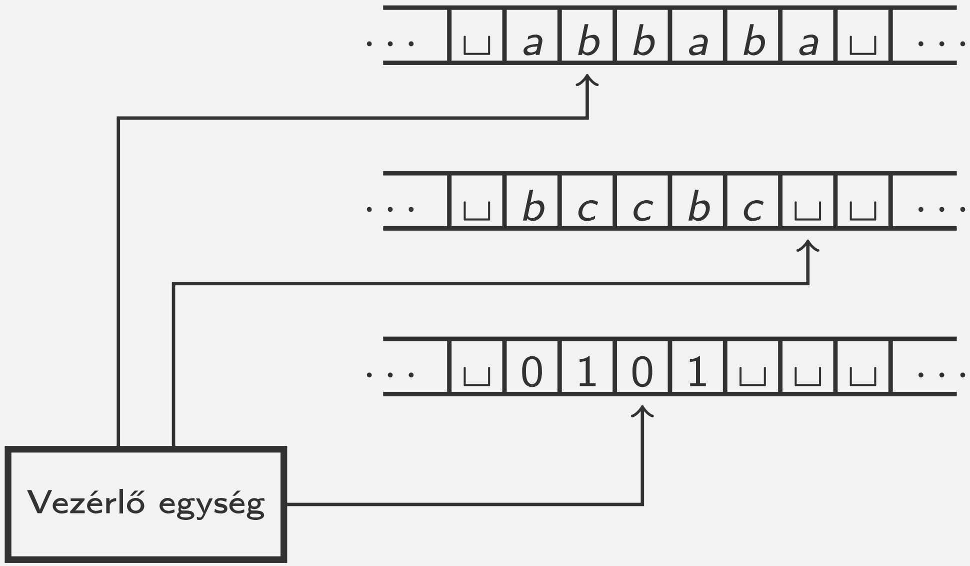
Definíció:

$\mathrm{RE} = \{L \mid L \text{ Turing-felismerhető}\}$,

$\mathrm{R} = \{L \mid L \text{ eldönthető}\}$.

Nyilván $\mathrm{R} \subseteq \mathrm{RE}$.

Többszalagos Turing gép – Informális kép



- Véges vezérlő egység, $k(\geqslant 1)$ darab kétirányba végtelen szalag, minden szalaghoz egy-egy saját író-olvasó fej.
- Egy ütem: Minden szalagról a fejek által mutatott betűk egyszerre történő beolvasása, átírása és a fejek léptetése egyszerre, de egymástól független irányokba.
- Az egyszalagos géppel analóg elfogadás fogalom.
- Az egyszalagos géppel analóg időigény fogalom (1 lépés $=$ 1 ütem).

k-szalagos Turing gép

Definíció

Adott egy $k \geqslant 1$ egész szám. A ***k*-szalagos Turing gép** egy olyan $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n \rangle$ rendezett hetes, ahol

- Q az állapotok véges, nemüres halmaza,
- $q_0, q_i, q_n \in Q$, q_0 a kezdő- q_i az elfogadó- és q_n az elutasító állapot,
- Σ és Γ ábécék, a bemenő jelek illetve a szalagszimbólumok ábécéje úgy, hogy $\Sigma \subseteq \Gamma$ és $\sqcup \in \Gamma \backslash \Sigma$,
- $\delta : (Q \backslash \{q_i, q_n\}) \times \Gamma^k \to Q \times \Gamma^k \times \{L, S, R\}^k$ az átmenet függvény.

δ az egész $(Q \backslash \{q_i, q_n\}) \times \Gamma^k$-n értelmezett függvény.

k-szalagos Turing gépek konfigurációi

Definíció

k-szalagos TG **konfigurációja** egy $(q, u_1, v_1, \ldots, u_k, v_k)$ szó, ahol $q \in Q$ és $u_i, v_i \in \Gamma^*$, $v_i \neq \varepsilon$ $(1 \leqslant i \leqslant k)$.

Ez azt reprezentálja, hogy

- az aktuális állapot q és
- az i. szalag tartalma $u_i v_i$ $(1 \leqslant i \leqslant k)$ és
- az i. fej v_i első betűjén áll $(1 \leqslant i \leqslant k)$.

Definíció

Az u szóhoz tartozó **kezdőkonfiguráció**: $u_i = \varepsilon$ $(1 \leqslant i \leqslant k)$, $v_1 = u\sqcup$, és $v_i = \sqcup$ $(2 \leqslant i \leqslant k)$.

Azaz, az input szó az első szalagon van, ennek az első betűjéről indul az első szalag feje. A többi szalag kezdetben üres.

k-szalagos Turing gépek megállási konfigurációi

Definíció

A $(q, u_1, v_1, \ldots, u_k, v_k)$ konfiguráció, ahol $q \in Q$ és $u_i, v_i \in \Gamma^*$, $v_i \neq \varepsilon$ $(1 \leqslant i \leqslant k)$,

- **elfogadó konfiguráció**, ha $q = q_i$,
- **elutasító konfiguráció**, ha $q = q_n$,
- **megállási konfiguráció**, ha $q = q_i$ vagy $q = q_n$.

k-szalagos TG-ek egylépéses konfigurációátmenete

Definíció

Egy $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n \rangle$ k-szalagos Turing gép $\vdash \subseteq C_M \times C_M$ **egylépéses konfigurációátmenet** relációját az alábbiak szerint definiáljuk.

Legyen A $C = (q, u_1, a_1 v_1, \ldots, u_k, a_k v_k)$ egy konfiguráció, ahol $a_i \in \Gamma,\ u_i, v_i \in \Gamma^*$ $(1 \leqslant i \leqslant k)$. Legyen továbbá $\delta(q, a_1, \ldots, a_k) = (r, b_1, \ldots, b_k, D_1, \ldots, D_k)$, ahol $q, r \in Q$, $b_i \in \Gamma, D_i \in \{L, S, R\}$ $(1 \leqslant i \leqslant k)$. Ekkor $C \vdash (r, u_1', v_1', \ldots, u_k', v_k')$, ahol minden $1 \leqslant i \leqslant k$-ra

- ha $D_i = R$, akkor $u_i' = u_i b_i$ és $v_i' = v_i$, ha $v_i \neq \varepsilon$, különben $v_i' = \sqcup$,
- ha $D_i = S$, akkor $u_i' = u_i$ és $v_i' = b_i v_i$,
- ha $D_i = L$, akkor $u_i = u_i' c$ $(c \in \Gamma)$ és $v_i' = c b_i v_i$ ha $u_i \neq \varepsilon$, különben $u_i' = \varepsilon$ és $v_i' = \sqcup b_i v_i$.

k-szalagos TG-ek többlépéses konfigurációátmenete

Tehát egy szalagjára vetítve a többszalagos TG pont úgy működik, mint az egyszalagos TG.

Példa:

Legyen $k{=}2$ és $\delta(q, a_1, a_2) = (r, b_1, b_2, R, S)$ a TG egy átmenete. Ekkor $(q, u_1, a_1v_1, u_2, a_2v_2) \vdash (r, u_1b_1, v_1', u_2, b_2v_2)$, ahol $v_1' = v_1$, ha $v_1 \neq \varepsilon$, különben $v_1' = \sqcup$.

Vegyük észre, hogy a fejek nem kell, hogy egyazon irányba lépjenek.

A k-szalagos TG-ek **többlépéses konfigurációátmenet** relációját ugyanúgy definiáljuk, mint az egyszalagos esetben. Jelölés: $\vdash^*$.

k-szalagos TG – felismert nyelv, időigény

Definíció

$M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n \rangle$ k-szalagos TG által felismert nyelv:
$L(M) = \big\{ u \in \Sigma^* \,\big|\, (q_0, \varepsilon, u\sqcup, \varepsilon, \sqcup, \ldots, \varepsilon, \sqcup) \vdash^* (q_i, x_1, y_1, \ldots x_k, y_k),$
valamely $x_1, y_1 \ldots, x_k, y_k \in \Gamma^*, y_1, \ldots,, y_k \neq \varepsilon$ -ra$\big\}$.

Azaz, csakúgy mint az egyszalagos esetben, azon inputábécé feletti szavak halmaza, melyekkel (az első szalagján) a TG-et indítva az az elfogadó, q_i állapotában áll le.

- A k-szalagos TG-ek által **felismerhető** illetve **eldönthető** nyelvek fogalma szintén analóg az egyszalagos esettel.
- Egy k-szalagos Turing gép **futási ideje** egy u szóra a hozzá tartozó kezdőkonfigurációból egy megállási konfigurációba megtett lépések száma.
- Az **időigény** definíciója megegyezik az egyszalagos esetnél tárgyalttal.

k-szalagos Turing gép – átmenetdiagram

A k-szalagos TG-ek **átmenetdiagramja** egy csúcs- és élcímkézett irányított gráf, melyre

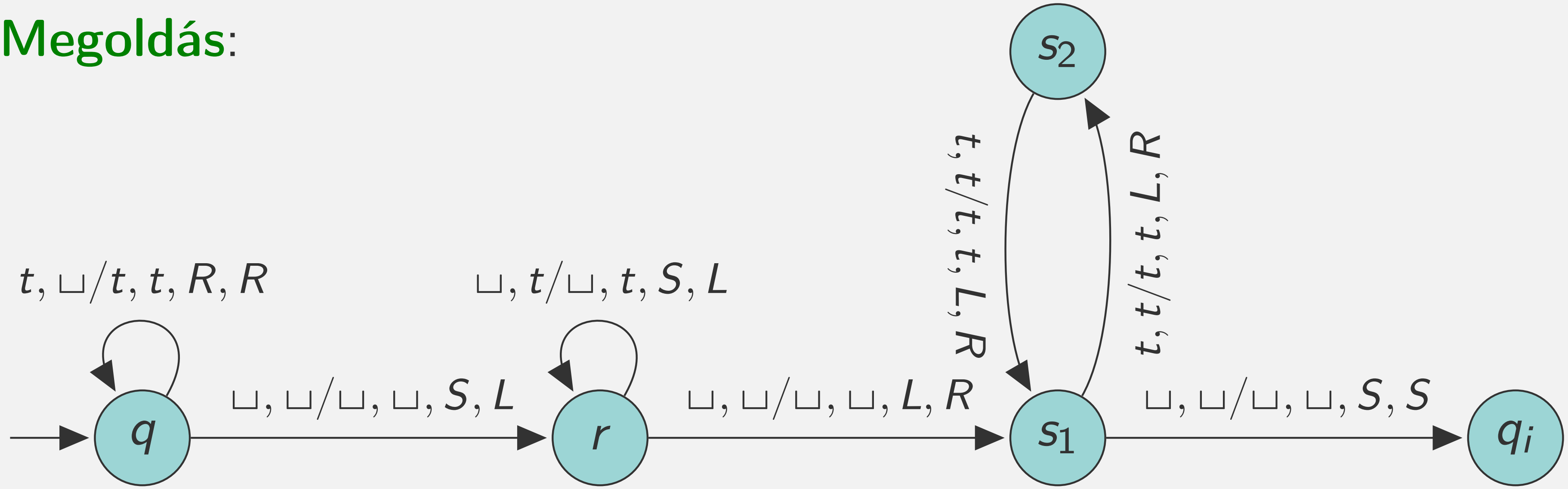
$$q \xrightarrow{a_1, \ldots, a_k / b_1, \ldots, b_k, D_1, \ldots, D_k} r$$

$$\iff \delta(q, a_1, \ldots, a_k) = (r, b_1, \ldots, b_k, D_1, \ldots, D_k)$$
$$(q, r \in Q, a_1, \ldots, a_k, b_1, \ldots, b_k \in \Gamma, D_1, \ldots, D_k \in \{L, S, R\})$$

Feladat: Készítsünk egy M kétszalagos Turing gépet, melyre $L(M) = \{ww^{-1} \mid w \in \{a, b\}^*\}$!

k-szalagos Turing gép – példa

Megoldás:



$t \in \{a, b\}$ tetszőleges. A többi átmenet q_n-be megy.

Például $(q, \varepsilon, abba, \varepsilon, \sqcup) \vdash (q, a, bba, a, \sqcup) \vdash (q, ab, ba, ab, \sqcup) \vdash (q, abb, a, abb, \sqcup) \vdash (q, abba, \sqcup, abba, \sqcup) \vdash (r, abba, \sqcup, abb, a) \vdash (r, abba, \sqcup, ab, ba) \vdash (r, abba, \sqcup, a, bba) \vdash (r, abba, \sqcup, \varepsilon, abba) \vdash (r, abba, \sqcup, \varepsilon, \sqcup abba) \vdash (s_1, abb, a, \varepsilon, abba) \vdash (s_2, ab, ba, a, bba) \vdash (s_1, a, bba, ab, ba) \vdash (s_2, \varepsilon, abba, abb, a) \vdash (s_1, \varepsilon, \sqcup abba, abba, \sqcup) \vdash (q_i, \varepsilon, \sqcup abba, abba, \sqcup)$

Mennyi a TG időigénye? Ez egy $O(n)$ időkorlátos TG, mivel egy n hosszú inputra legfeljebb $3n + 3$ lépést tesz.

k-szalagos TG szimulálása egyszalagossal

Definíció

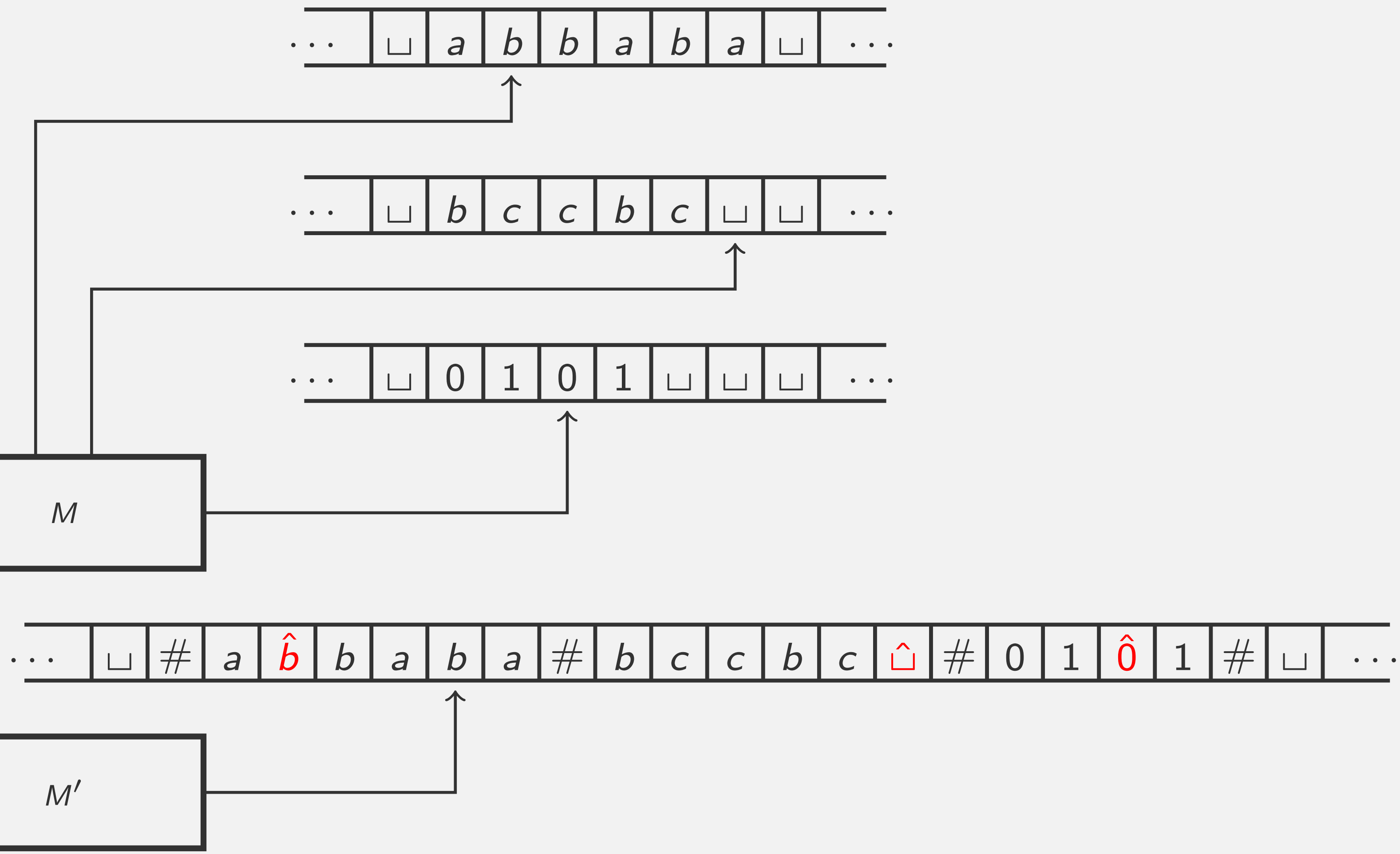
Két TG **ekvivalens**, ha ugyanazt a nyelvet ismerik fel.

Tétel

Minden M k-szalagos Turing géphez megadható egy vele ekvivalens M' egyszalagos Turing gép. Továbbá, ha M legalább lineáris időigényű $f(n)$ időkorlátos gép (azaz $f(n) = \Omega(n)$), akkor M' $O(f(n)^2)$ időkorlátos.

Többszalagos TG szimulálása egyszalagossal

A szimuláció alapötlete:



A szalagok tartalmát #-ekkel elválasztva eltárolhatjuk M' egyetlen szalagján, a fejek helyzetét ˆ-al megjelölve.

Turing-gép egy irányban végtelen szalaggal

Egyirányban végtelen szalagos TG: Bal oldalon zárt a TG szalagja, a fej nem tud "leesni", ha a legbaloldalibb cellán balra lépés az utsítás, akkor a fej helyben marad.

Tétel

Minden egyszalagos M TG-hez van vele ekvivalens egyirányban végtelen szalagos M'' TG.

A bizonyítás ötlete, hogy először egy kétszalagos egyirányban végtelen szalagos géppel szimulálunk egy egyszalagos kétirányban végtelen szalagost (az első szalagon tárolva a kezdőpozíciótól jobbra levő tartalmat, a másodikon pedig a balra lévő tartalom tükörképét). Majd a kétszalagos egyirányban végtelen szalagos gépet szimuláljuk egy ugyanilyen egyszalagossal.

Megjegyzés: Nyilvánvalóan a másik irány is igaz.

Nemdeterminisztikus Turing gép

Definíció

Az (egyszalagos) **nemdeterminisztikus Turing gép** (NTG) csak átmenetfüggvényében különbözik a determinisztikustól.
$\delta : (Q \setminus \{q_{\mathrm{i}}, q_{\mathrm{n}}\}) \times \Gamma \to \mathcal{P}(Q \times \Gamma \times \{L, S, R\})$.

Azaz míg a **determinisztikus** esetben a δ átmenetfüggvény minden egyes $(Q \setminus \{q_{\mathrm{i}}, q_{\mathrm{n}}\}) \times \Gamma$-beli párhoz **pontosan egy**, addig egy **nemdeterminisztikus** TG **akárhány** (pl. 0,1,5,100) darab $Q \times \Gamma \times \{L, S, R\}$-beli rendezett hármast rendelhet hozzá.

Nemdeterminisztikus Turing-gép

egylépéses konfigurációátmenet

A konfiguráció fogalma azonos, jelölje most is C_M az M gép lehetséges konfigurációinak halmazát.A $\vdash \subseteq C_M \times C_M$ **egylépéses konfigurációátmenet** relációt a következőképpen definiáljuk.

Definíció

Legyen $uqav$ egy konfiguráció, ahol $a \in \Gamma,\ u, v \in \Gamma^*$.

- Ha $(r, b, R) \in \delta(q, a)$, akkor $uqav \vdash ubrv'$, ahol $v' = v$, ha $v \neq \varepsilon$, különben $v' = \sqcup$,
- ha $(r, b, S) \in \delta(q, a)$, akkor $uqav \vdash urbv$,
- ha $(r, b, L) \in \delta(q, a)$, akkor $uqav \vdash u'rcbv$, ahol $c \in \Gamma$ és $u'c = u$, ha $u \neq \varepsilon$, különben $u' = u$ és $c = \sqcup$.

Míg a **determinisztikus** esetben minden nem-megállási C konfigurációhoz **pontosan egy** C' konfiguráció létezett, melyre $C \vdash C'$, addig a **nemdeterminisztikus** esetben **több** ilyen is létezhet (de véges sok, hiszen $|Q \times \Gamma \times \{L, S, R\}|$ véges).

Nemdeterminisztikus Turing-gép

többlépéses konfigurációátmenet, felismert nyelv

Többlépéses konfigurációátmenet: $\vdash$ reflexív tranzitív lezártja, jelölése $\vdash^*$.

Legyen C egy konfiguráció. Míg a **determinisztikus** esetben **legfeljebb egy** C' megállási konfiguráció létezhetett, melyre $C \vdash^* C'$, addig a **nemdeterminisztikus** esetben **több** ilyen is létezhet.

Definíció

Az M NTG által felismert nyelv
$L(M) = \left\{u \in \Sigma^* \,\middle|\, q_0 u \sqcup \vdash^* x q_i y \text{ valamely } x, y \in \Gamma^*, y \neq \varepsilon \text{ -ra}\right\}$.

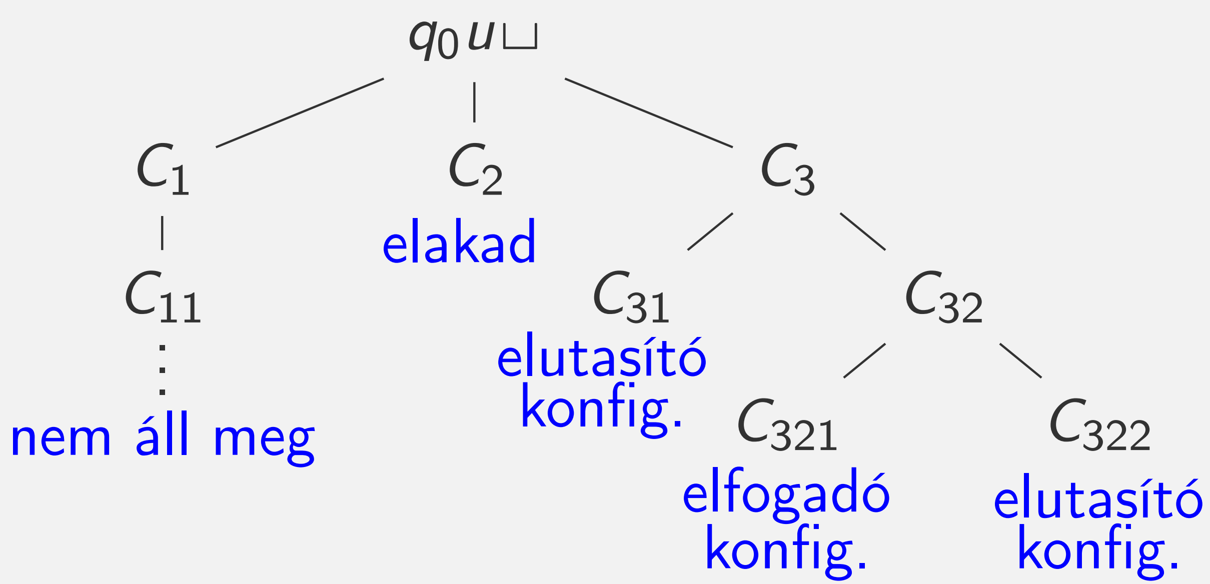
$\vdash^*$ fogalmának módosulása miatt a felismert nyelv fogalma is módosult. Egy NTG-re úgy gondolhatunk, hogy **több számítása is lehet egyazon szóra**. Akkor fogad el egy szót, ha az adott szóra **legalább egy számítása q_i-ben ér véget**.

Nemdeterminisztikus Turing-gép

Definíció

Egy M TG egy $u \in \Sigma^*$ inputjához tartozó **nemdeterminisztikus számítási fája** egy gyökeres fa, melynek csúcsai M konfigurációival címkézettek. $q_0 u \sqcup$ a gyökér címkéje. Ha C egy csúcs címkéje, akkor $|\{C' \mid C \vdash C'\}|$ gyereke van és ezek címkéi éppen $\{C' \mid C \vdash C'\}$ elemei.

Példa:



M elfogadja u-t, hiszen a $q_0 u \sqcup \vdash C_3 \vdash C_{32} \vdash C_{321}$ számítása elfogadó konfigurációba visz. **Egyetlen** elfogadó számítás is elég!

Nemdeterminisztikus Turing-gép

Tehát adott inputra több számítás is lehetséges, ezek lehetnek elfogadóak, elutasítóak, elakadóak (ha olyan C-be jut, melyre $\{C' \mid C \vdash C'\} = \emptyset$), illetve végtelenek.

Észrevétel: $u \in L(M) \Leftrightarrow$ az u-hoz tartozó nemdeterminisztikus számítási fának van olyan levele, ami elfogadó konfiguráció.

Definíció

Az M NTG **eldönti** az $L \subseteq \Sigma^*$ nyelvet, ha felismeri továbbá minden $u \in \Sigma^*$ input szóhoz tartozó nemdeterminisztikus számítási fa véges és a fa minden levele elfogadó vagy elutasító konfiguráció.

Definíció

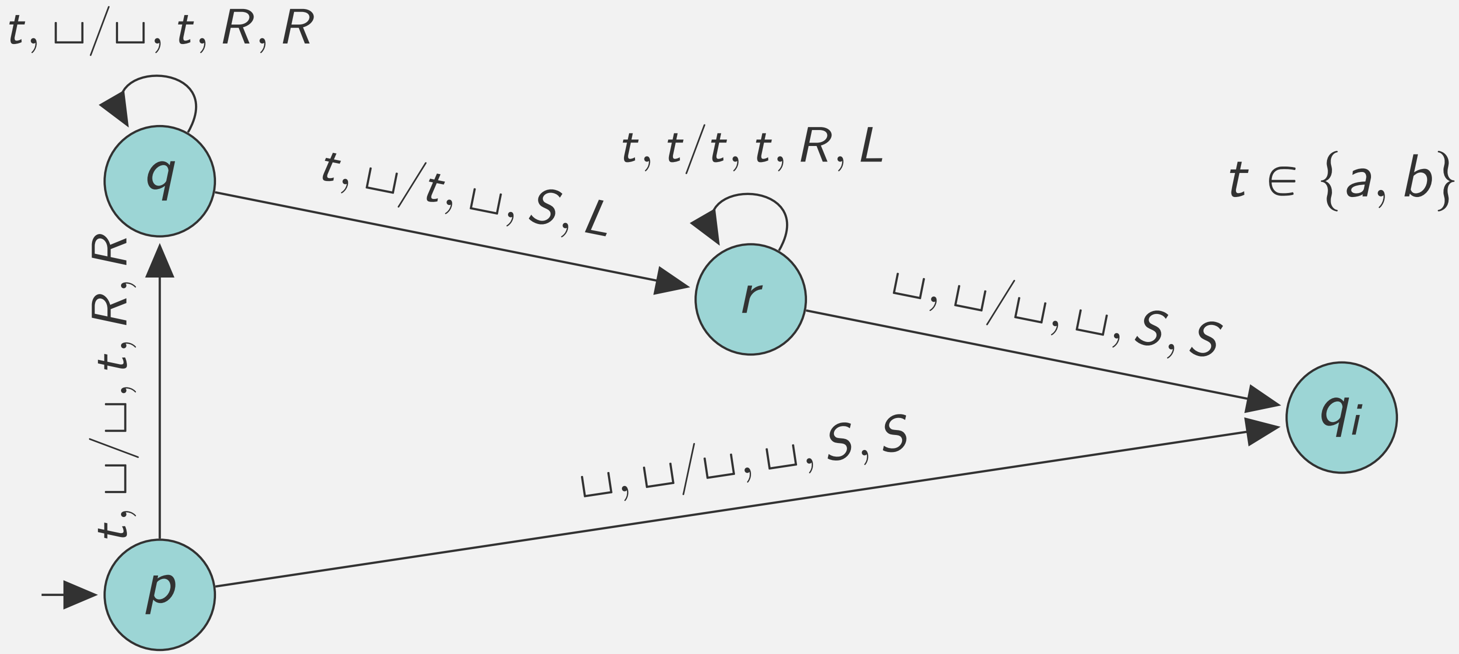
Az M NTG $f(n)$ **időkorlátos** (időigényű), ha minden $u \in \Sigma^*$ n hosszú szóra u számítási fája legfeljebb $f(n)$ magas.

Megjegyzés: a nemdeterminisztikus Turing-gép definíciója értelemszerűen kiterjeszthető k-szalagos gépekre is, így beszélhetünk k-szalagos nemdeterminisztikus Turing-gépekről is.

Nemdeterminisztikus Turing-gép

Példa

Az alábbi M nemdeterminisztikus Turing-gépre
$L(M) = \{ww^{-1} \mid w \in \{a,b\}^*\}$.



$(p, \varepsilon, abba, \varepsilon, \sqcup) \vdash (q, \varepsilon, bba, a, \sqcup) \vdash (r, \varepsilon, bba, \varepsilon, a) \vdash$
$(q_n, \varepsilon, bba, \varepsilon, a)$

$(p, \varepsilon, abba, \varepsilon, \sqcup) \vdash (q, \varepsilon, bba, a, \sqcup) \vdash (q, \varepsilon, ba, ab, \sqcup) \vdash$
$(r, \varepsilon, ba, a, b) \vdash (r, b, a, \varepsilon, ab) \vdash (r, ba, \sqcup, \varepsilon, \sqcup ab) \vdash$
$(q_i, ba, \sqcup, \varepsilon, \sqcup ab)$

NTG szimulálása TG-pel

Tétel

Minden $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n \rangle$ $f(n)$ idejű NTG-hez megadható egy ekvivalens, $2^{O(f(n))}$ idejű M' determinisztikus TG.

A szimuláció alapötlete: Egy w bemenetre járja be M' az M gép w-hez tartozó nemdeterminisztikus számítási fájának csúcsait szélességi bejárással.

Minden csúcs megfeleltethető egy a gyökértől az adott csúcsig tartó parciális számításnak.

Ha a csúcsnak megfelelő számítás nem ért még véget vagy pedig egy elutasító számításnak felel meg, akkor vegyük a szélességi bejárás szerinti következő csúcsot.

Amennyiben a csúcs egy elfogadó számításnak felel meg M' fogadja el w-t.

Időigény: A legfeljebb n hosszúságú, a fa gyökeréből induló utak összhossza n-nek exponenciális függvénye.

Nemdeterminisztikus Turing-gép

Következmények:

1. Egy nyelv Turing-felismerhető, akkor és csak akkor, ha valamely nemdeterminisztikus Turing-gép felismeri.
2. Egy nyelv eldönthető, akkor és csak akkor, ha van azt eldöntő nemdeterminisztikus Turing-gép.

Turing-ekvivalens számítási modellek

Most bemutatunk néhány olyan számítási modellt, amelyeknek a számítási ereje megegyezik a Turing gépek számítási erejével.

Ezen modellek Turing-ekvivalenciája a Church-Turing tézis alátámasztásául szolgálnak és így egyúttal, ha a Church-Turing tézist igaznak gondoljuk, az algoritmus fogalmának alternatív matematikai modelljeit adhatják.

0-típusú grammatikák és a TG-ek kapcsolata

Tétel

Minden G grammatikához megadható egy $L(G)$-t felismerő NTG.

Bizonyítás: Legyen M-nek 3 szalagja, az első a TG bemenetét, a második a G grammatika szabályait tartalmazza. Ezeket a működés során csak olvassuk.

A harmadik szalagon mindig egy α mondatforma áll (kezdetben G kezdőszimbóluma).

A Turing gép nemdeterminisztikusan választ egy $p \to q$ szabályt és α-ban egy pozíciót. Ha az adott pozícióban éppen p kezdődik, azaz $\alpha = xpy$, akkor p-t q-ra cseréli, az új mondatforma xqy lesz.

Ha az 1. és a 3. szalag tartalma megegyezik a gép q_i-ben megáll. M ezt minden iteráció előtt ellenőrzi. Így $L(M) = L(G)$. $\square$

Következmény: Előző tételünk alapján determinisztikus TG is adható.

0-típusú grammatikák és a TG-ek kapcsolata

Tétel

Minden $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n \rangle$ determinisztikus TG-hez megadható egy $L(M)$-et generáló G grammatika.

Bizonyítás: G mondatformái M konfigurációit fogják kódolni. A G grammatika éppen fordítottan fog haladni. Nemdeterminisztikusan előállít egy elfogadó konfigurációt, majd ebből megpróbál egy kezdőkonfigurációt levezetni.

Legyen $G = \langle (\Gamma \backslash \Sigma) \cup Q \cup \{S, A, \rhd, \lhd\}, \Sigma, P, S \rangle$.
P szabályai:

1. $S \to \rhd A q_i A \lhd$
2. $A \to aA \,|\, \varepsilon$ $\quad (\forall a \in \Gamma)$
3. $bq' \to qa$, ha $\delta(q, a) = (q', b, R)$
4. $q'b \to qa$, ha $\delta(q, a) = (q', b, S)$
5. $q'cb \to cqa$, ha $\delta(q, a) = (q', b, L)$ $\quad (\forall c \in \Gamma)$
6. $\sqcup \lhd \to \lhd$, $\lhd \to \varepsilon$, $\rhd \sqcup \to \rhd$, $\rhd q_0 \to \varepsilon$

0-típusú grammatikák és a TG-ek kapcsolata

1. $S \to \rhd A q_i A \lhd$
2. $A \to aA \,|\, \varepsilon$ $\qquad (\forall a \in \Gamma)$
3. $bq' \to qa$, ha $\delta(q, a) = (q', b, R)$
4. $q'b \to qa$, ha $\delta(q, a) = (q', b, S)$
5. $q'cb \to cqa$, ha $\delta(q, a) = (q', b, L)$ $\qquad (\forall c \in \Gamma)$
6. $\sqcup\lhd \to \lhd$, $\lhd \to \varepsilon$, $\rhd\sqcup \to \rhd$, $\rhd q_0 \to \varepsilon$

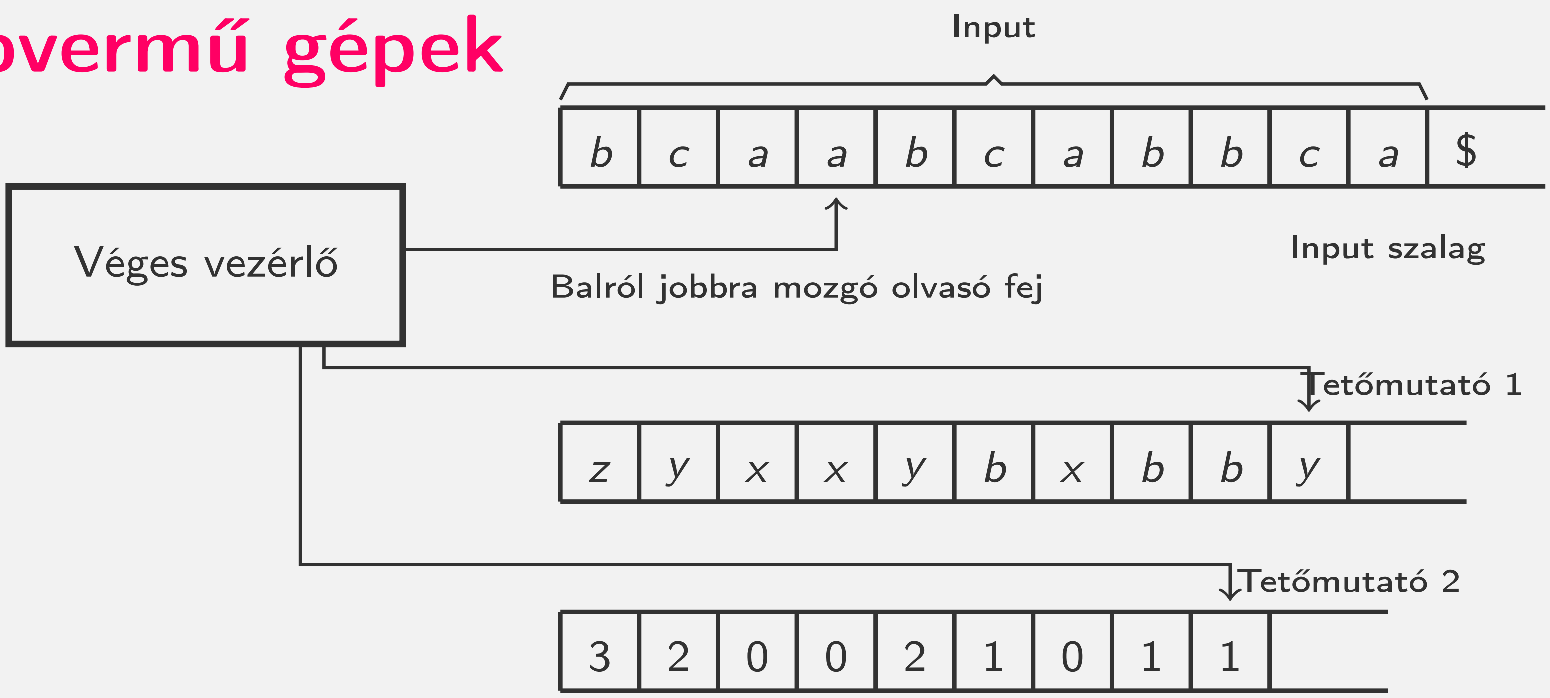
1-2. generálunk egy tetszőleges elfogadó konfigurációt

3-5. a konfigurációátmeneteket fordított irányban szimuláljuk. Pl. ha $\alpha cqa\beta \vdash \alpha q'cb\beta$ egy $\delta(q, a) = (q', b, L)$ szabály szerint, akkor most a grammatikában az 5-ös pont szerint $q'cb$ íródhat át cqa-ra.

6. Ha a mondatformánk egy kezdőkonfiguráció (esetleg néhány extra $\sqcup$-el), akkor ezek takarítják el a már felesleges jeleket.
A levezetés hosszára vonatkozó indukcióval megmutatható, hogy

$q_0 w \vdash^* \alpha q_i \beta$ a.cs.a. $S \Rightarrow^* \rhd \alpha q_i \beta \lhd \Rightarrow^* \rhd \sqcup^i q_0 w \sqcup^j \lhd \Rightarrow^* w$. $\square$

Többvermű gépek



- Egy k-vermű gép egy determinisztikus veremautomata k veremmel.
- Az inputszót a veremautomatához hasonlóan kapja. Van egy véges vezérlőegysége, amely egy véges halmazból veszi fel az állapotát.
- Adott egy minden verem számára közös veremábécé.
- A többvermű gép átmenete függ: (1) a vezérlőegység állapotától, (2) az olvasott inputszimbólumtól (3) az egyes vermek tetején lévő veremszimbólumtól.

Többvermű gépek

- A többvermű gép egy állapot-átmenet során: (1) állapotot vált, (2) az egyes vermek tetején lévő legfelső veremszimbólumot valahány (a 0-t is beleértve) veremszimbólum sorozatával helyettesíti, amely általában különböző az egyes vermek esetében.
- Egy tipikus konfiguráció- (állapot-)átmenet: $\delta(q, a, X_1, \ldots, X_k) = (p, \gamma_1, \ldots, \gamma_k)$.

  q állapotban X_i szimbólum hatására az i-dik verem tetején, $i = 1, \ldots, k$, az automata az a inputszimbólumot (vagy valamilyen inputszimbólum, vagy ε) feldolgozza az inputból, p állapotba megy és X_i szimbólumot az i-edik verem tetején γ_i-re cseréli, minden $i = 1, \ldots, k$-ra).
- A többvermű gép a végállapotába jutva fogadja el az egyes szavakat.

A többvermű gépek számítási ereje

Feltesszük, hogy az input végén (nem része annak) van egy \$ speciális szimbólum, amelyet **végjelzőnek (endmarker)** is hívunk. A végjelző jelzi az input összes betűjének a feldolgozását.

Jelölje $\mathcal{L}_{k\mathrm{V}}$ a k-veremmel felismerhető nyelvek osztályát. Nyilvánvalóan

$$\mathcal{L}_{0\mathrm{V}} \subseteq \mathcal{L}_{1\mathrm{V}} \subseteq \cdots \subseteq \mathcal{L}_{k\mathrm{V}} \subseteq \mathcal{L}_{(k+1)\mathrm{V}} \subseteq \cdots$$

Mivel a 0-vermek a véges automaták és az 1-vermek a determinisztikus veremautomaták, ezért

$$\mathcal{L}_{0\mathrm{V}} = \mathcal{L}_3, \quad \mathcal{L}_3 \subset \mathcal{L}_{1\mathrm{V}} \subset \mathcal{L}_2.$$

A többvermű gépek számítási ereje

Állítás

Minden $k \in \mathbb{N}$-re $\mathcal{L}_{k\mathrm{V}} \subseteq \mathrm{RE}$

Bizonyítás: Könnyű szimulálni egy k-vermet egy $(k+1)$-szalagos TG-pel: az első szalagon tároljuk a bemenetet, a többin pedig az egyes vermek tartalmát. Ha egy lépésben k darab betűt írunk valamelyik verembe, azt k lépésben, betűnként szimulálhatjuk.

Tétel

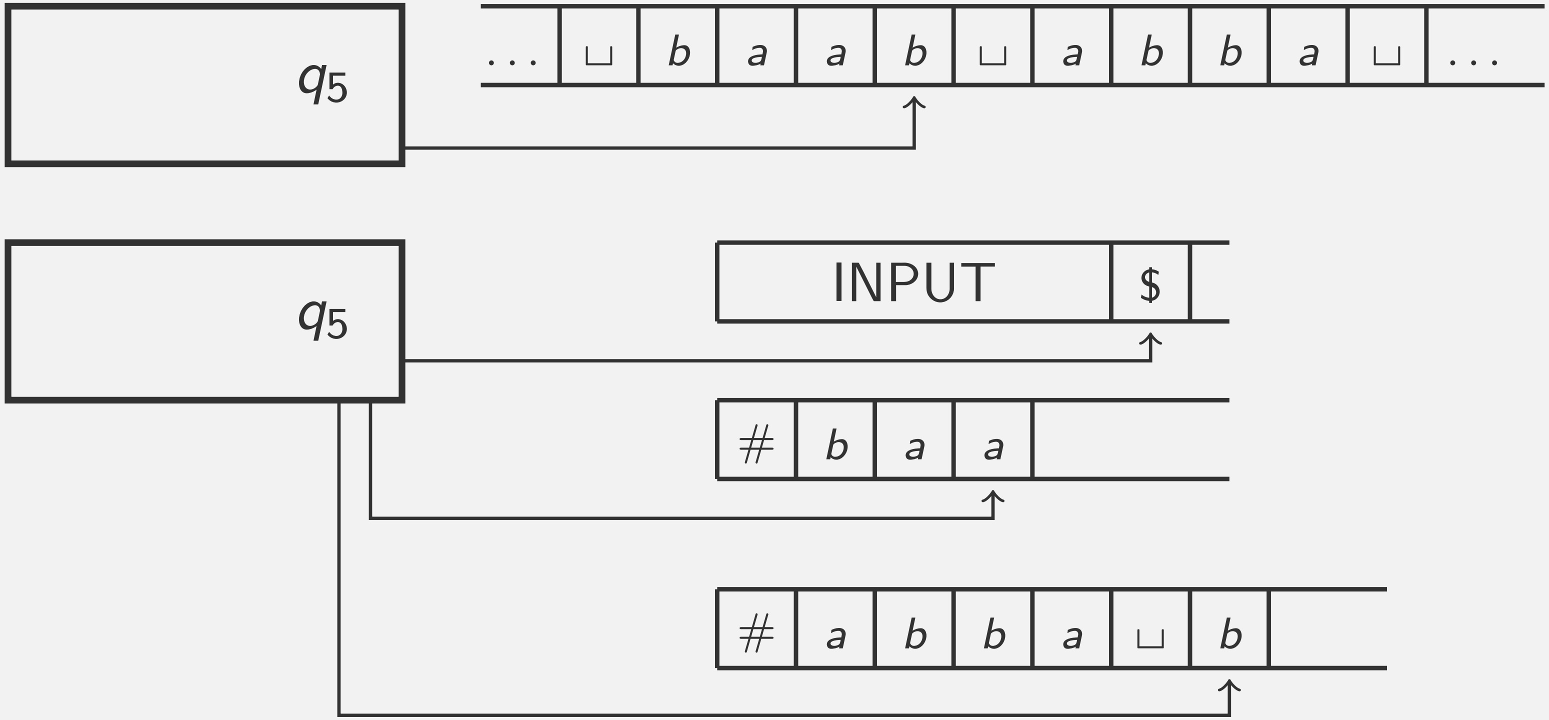
$\mathrm{RE} \subseteq \mathcal{L}_{2\mathrm{V}}$.

Következmény

Minden $k \geqslant 2$-re $\mathcal{L}_{k\mathrm{V}} = \mathrm{RE}$.

A többvermű gépek számítási ereje

A tétel bizonyítása:



Ötlet: Az egyik verem az író-olvasó fejtől balra, a másik az attól jobbra lévő szimbólumokat tárolja.

A többvermű gépek számítási ereje

- Kezdetben S vermei a $\#$ veremalja szimbólummal indulnak (Feltehető, hogy $\#$ nem eleme M szalagábécéjének.). $\#$ -t csak a verem aljának jelzésére használjuk.
- Tegyük fel, hogy S bemenetén ott van a w szó. Másolja S a w szót az első vermébe és fejezze be a másolást, ha az input végét jelző \$-t olvasta.
- S egyesével kiveszi a szimbólumokat az első verméből és belerakja az összeset a másodikba. (ε-átmenetekkel, innentől minden átmenet ilyen.) Ekkor az első verem üres lesz, a második fogja w -t tartalmazni, w első betűje lesz a verem tetején.
- S az M kezdőállapotának megfeleltetett állapotába lép.

A többvermű gépek számítási ereje

S M egy átmenetét a következőképpen szimulálja:

- Feltehető, hogy S ismeri M állapotát, amelyet jelöljünk q-val. Ehhez S-ben M minden állapotához legyen egy saját állapot a véges vezérlőegységében.
- S ismeri az M író-olvasó által éppen pásztázott X szimbólumot is: ez S második vermének a legfelső szimbóluma. Kivételt képez az az eset, amikor a második verem csak a veremalja szimbólumot tartalmazza, ilyenkor M épp egy üres szimbólumra ért.
- Tehát S ismeri M következő átmenetét. S vezérlőegysége M következő állapotának megfelelően kerül a saját, ennek megfelelő állapotába.

A többvermű gépek számítási ereje

1. eset:

Ha M az X szimbólumot Y -nal helyettesíti és a fej **jobbra mozog**, akkor S Y-t az első vermébe rakja, ami azt fejezi ki, hogy Y az M író-olvasó fejétől balra van, X -t pedig kivesszük S második vereméneк tetejéről. Előfordulhat az alábbi két kivételes eset.

- Ha a második verem csak a $\#$-t tartalmazza (azaz $X = \sqcup$), akkor a második verem tartalma nem változik. (Hiszen M újabb üres cella felé mozdul jobbra.)
- Ha $Y = \sqcup$ ÉS az első verem tetején $\#$ áll, akkor az első verem tartalma változatlan marad. (M író-olvasó fejétől balra továbbra is minden cella üres.)

2. eset:

Ha M az X szimbólumot Y -nal helyettesíti és a fej **nem mozog**, akkor S Y-ra cseréli X -t a második verem tetején.

A többvermű gépek számítási ereje

3. eset:

Ha M az X szimbólumot Y -nal helyettesíti és a fej **balra mozog**, akkor S az első verem legfelső szimbólumát, nevezzük Z -nek, eltávolítja a verem tetejéről, és X -et YZ -vel helyettesíti a második veremben (Z lesz legfelül.). Ez azt jelenti, hogy ami egy pozícióval az író-olvasó fejétől balra volt, most az író-olvasó fej alatt lesz.

- Kivételt képez az az eset, amikor Z a veremalja szimbólum. Ekkor S X-et $Y\sqcup$-re cseréli a második veremben (azaz $\sqcup$ kerül a második verem tetejére), az elsőből pedig nem távolít el semmit.

Ezen felül S természetesen M állapotváltozásait lekövetve aktualizálja az állapotát.

S elfogad, ha M új állapota elfogadó, egyébként S M következő lépésének szimulálásával folytatja működését. $\square$

Számláló gépek

A **számláló gépek (counter machines)** olyan matematikai gépek, amelyek véges sok (de előre rögzített számú) **regiszterrel** rendelkeznek, amelyek korlát nélküli méretű természetes számokat tárolhatnak. A számláló gépek tipikusan egy nagyon limitált **utasításkészlettel** működtethetők. Ezen utasításkészlet 1-2 aritmetikai továbbá 1-2 vezérlő utasításból áll.

A számláló gépek a **regiszter gépek** legegyszerűbb, speciális esete. Később találkozni fogunk majd a RAM-géppel is, amelyik szintén speciális regiszter gép, a számláló gép olyan általánosítása, ahol indirekt címzésű utasítások is rendelkezésre állnak.

Számláló gépek

A számláló gépek egy lehetséges reprezentációja egy szekvenciális programkód, ahol a programsorok címkézettek.

Az utasításkészlet alapján többfajta számláló gép ismeretes. Mi **Marvin Minsky** egy 1967-es modelljét tekintjük, ahol mindössze 3 utasítás létezik:

- INC(r): 1-gyel növeli az r regiszter értékét
- DEC(r): 1-gyel csökkenti az r regiszter értékét
- IF r regiszter értéke $= 0$ THEN GOTO z: ha a regiszter értéke 0, akkor a z címkéjű utasításra ugrik, különben a következő utasítással folytatja.

Matematikai gépként is reprezentálhatjuk. Ilyenkor a gép következő állapota függ az aktuális állapotától, az inputszimbólumtól, valamint attól, hogy valamely számlálója nulla-e vagy sem. Egy állapot-átmenet során a gép megváltoztathatja állapotát továbbá hozzáadhat vagy kivonhat 1-et bármely számlálójából, feltéve, hogy az nem 0-val egyenlő.

Számláló gépek k-verem reprezentációja

Egy lehetséges (valamivel általánosabb) gépként történő reprezentáció:

Definíció

A **számláló gép** (counter machine) egy korlátozott többvermű gép a következő megszorításokkal:

- Csak kétfajta veremszimbólum létezik: Z_0 (veremalja szimbólum) és X.
- Kezdetben egyetlen Z_0 van minden veremben.
- Z_0 -t csak Z_0X^i alakú szó helyettesítheti valamely $i \geqslant 0$-ra.
- X -t csak X^i helyettesítheti valamely $i \geqslant 0$-ra.

Ez azt jelenti, hogy Z_0-ból legfeljebb 1 lehet minden egyes veremben mégpedig a verem alján. Z_0-n kívül csak X-ek lehetnek a vermekben.

Így egy számláló 0 voltát úgy ellenőrizhetjük, hogy megnézzük Z_0 vagy X van-e a verem tetején.

A számláló gépek számítási ereje

Észrevételek:

- A számláló gép által elfogadott bármely nyelv rekurzívan felsorolható. Ez következik abból, az előző tételünkből, hogy többvermes gépeket lehet TG-pel szimulálni.
- Az egyszámlálós gép által elfogadott bármely nyelv környezetfüggetlen. A számlálót tekinthetjük úgy, mint egy vermet. Következésképpen a számlálógép speciális esete az egyvermű gépnek, a veremautomatának.

Tétel

Minden rekurzívan felsorolható nyelv felismerhető egy 3 számlálóval rendelkező számláló géppel.

A számláló gépek számítási ereje

Bizonyítás:
Elegendő egy tetszőleges 2-vermet 3-számlálós géppel szimulálni.

Tegyük fel, hogy a kétvermű gép (közös) veremábécéje $r-1$ elemű. Ezen szimbólumokat azonosíthatjuk 1 és $r-1$ közötti számjegyekkel, az $u = X_n \cdots X_1$ szóval leírt veremtartalmat (a verem tetején u utolsó betűje áll) pedig tekinthető egy r-es számrendszerbeli alakban felírt számnak. Azaz a verem tartalma (amelynek a teteje u utolsó betűje) dekódolhatóan reprezentálható egy $X_n r^{n-1} + X_{n-1} r^{n-2} + \cdots + X_2 r + X_1$ természetes számmal.

Az első két számláló tárolja tehát a két verem tartalmát, egy-egy természetes számként. A harmadik számláló egy segédszámláló. Arra szolgál, hogy a két másik számlálót igazítsa, amennyiben ez szükséges, pl. ha valamelyik számlálót r -rel szeretnénk osztani vagy szorozni.

A számláló gépek számítási ereje

A vermen végrehajtható három alapművelet a következő:

(1) a legfelső elem eltávolítása (pop),

(2) a legfelső elem átírása

(3) egy elemnek a verembe történő helyezése (push).

A k-verem műveletei ezen alapműveletek kompozíciói, így elegendő ezeket számlálógéppel szimulálni.

Például, ha a legfelső X szimbólumot k darab szimbólumból álló szóval szeretnénk helyettesíteni, akkor ezt a műveletet k részre bonthatjuk, X helyettesítésére és $k - 1$ darab push műveletre.

Ezen alapműveletek végrehajtását a számláló segítségével a következőképpen oldhatjuk meg:

A számláló gépek számítási ereje

(1): pop művelet

A legfelső elemnek a veremből való eltávolítása megfelel a neki megfelelő számláló i értékének $\lfloor i/r \rfloor$ -rel való helyettesítésének és a maradék (X_1) eldobásának. Kezdjünk el 1-eket kivonni ebből a számlálóból. A 0 kezdeti értékű harmadik számlálóhoz minden r. kivonás után adjunk hozzá 1-et. Amikor a csökkenő számláló értéke 0 lesz, akkor a harmadik számlálóé éppen $\lfloor i/r \rfloor$. Ezután ismételten növeljük az eredeti számlálót 1-gyel és a harmadikat csökkentsük 1-gyel, amíg a harmadik számláló újra 0 nem lesz. Ekkor az a számláló, amelyik i-t tárolta előzőleg, $\lfloor i/r \rfloor$ lesz.

(2): a tetőelem megváltoztatása

Ahhoz, hogy X -et Y -ra cseréljük a verem tetején, amelyet i reprezentál a számoló gépnél, növelni vagy csökkenteni kell i-t egy r -nél nem nagyobb értékkel. Ha $Y > X$ akkor i-t $Y - X$ -szel növeljük. Ha $Y < X$, akkor i-t $X - Y$ -nal csökkentjük.

A számláló gépek számítási ereje

(3) push művelet

Ahhoz, hogy X -et azon verembe helyezzük, amely eredetileg i-t tartalmazza, i-t $ir + X$ -szel kell helyettesíteni. Először r -rel szorzunk. Ehhez folyamatosan csökkentjük i-t 1-gyel, és növeljük a harmadik számlálót (amely 0-ról indul) r -rel. Amikor az eredeti számláló értéke 0 lesz, akkor a harmadik számlálóé ir. A harmadik számlálót visszamásoljuk az eredetibe (lépésenként eggyel növelve az eredeti, és eggyel csökkentve a harmadik számláló értékét). Ekkor a harmadik számláló értéke újra 0 lesz. Végül az eredeti számlálót X -szel megnöveljük.

Inicializálás:

A konstrukció befejezéséhez szükséges még a számlálók inicializálása a vermek kezdeti állapotának szimulálásához, amikor még csak a kezdőszimbólumot tartalmazzák a vermek. Ez a lépés megoldható úgy, hogy a két számlálót valamilyen 1 és $r - 1$ közötti értékkel növeljük, amely megfelel a kezdőszimbólumnak. $\square$

A számláló gépek számítási ereje

Tétel

Minden rekurzívan felsorolható nyelv elfogadható kétszámlálós géppel.

Bizonyítás:

Elegendő annak bizonyítása, hogyan szimuláljunk három számlálót kettővel. Az ötlet az, hogy a három számlálót egy egész számmal fogjuk reprezentálni. Jelöljük a három szóbanforgó számlálót i-vel, j-vel és k-val és legyen a választott egész szám $m = 2^i 3^j 5^k$.

Az egyik számláló ezt a számot fogja tárolni, míg egy másik szorozni vagy osztani fogja m-et valamelyik előbbi prímszámmal (2-vel, 3-mal, 5-tel).

A háromszámlálós gép szimulálása a következőképpen történik:

A számláló gépek számítási ereje

(1) Növeljük 1-gyel i-t, j-t vagy k-t.

i 1-gyel való növeléséhez, szorozzuk m-et 2-vel. Az előbb láttuk, hogyan kell egy számot valamely r konstanssal megszorozni egy második számlálót (most a másodikat) használva. Hasonlóan, j növeléséhez, szorozzuk m-et 3-mal, k növeléséhez pedig 5-tel.

(2) i, j vagy k valamelyike 0-val egyenlő-e?

Ahhoz, hogy megmondjuk, hogy $i = 0$-e, meg kell határoznunk, hogy vajon m osztható-e 2-vel. Másoljuk m-et a második számlálóba egyesével és használjuk a számlálógép állapotát annak megjegyzésére, hogy m-et páros vagy páratlan sokszor csökkentettük-e. Ha m-et páratlan sokszor csökkentettük, mikor 0-vá válik, akkor $i = 0$. Ekkor m-et visszaállítjuk a második számláló tartalmának az elsőbe másolásával. Hasonlóan tesztelhetjük, hogy $j = 0$-e (m osztható-e 3-mal), illetve, hogy $k = 0$-e (m osztható-e 5-tel).

A számláló gépek számítási ereje

(3) Csökkentsük 1-gyel i-t, j-t vagy k-t.

i, j illetve k csökkentéséhez osszuk m-et rendre 2-vel, 3-mal vagy 5-tel. Láttuk korábban hogyan kell az osztást végrehajtani egy második számlálót használva. Ha nem 0 az osztás maradéka, akkor elfogadás nélkül termináljunk mivel a számláló értéke 0 volt. Amennyiben sikerült maradék nélkül osztani, akkor éppen a megfelelő számláló 1-gyel való csökkentését szimuláltuk. $\square$

Miért pont 2,3,5? Gondoljuk meg, miért fontos prímek (általánosabban: relatív prímek) használata!

Összegzés a számláló gépekről:

- A k számlálós számláló gépek számítási ereje $k \geqslant 2$-re megegyezik a TG-ek számítási erejével.
- Az egyetlen számlálóval rendelkező számláló gépek CF egy valódi részhalmazát alkotják. Például $\{a^n b^n \mid n \in \mathbb{N}\}$ felismerhető 1 számlálós géppel, de $\{a^n b^k a^k b^n \mid n, k \in \mathbb{N}\}$ nem.