Formális nyelvek III.
Általános és környezetfüggő nyelvek

Fülöp Zoltán

SZTE TTIK Informatikai Intézet
Számítástudomány Alapjai Tanszék
6720 Szeged, Árpád tér 2.

A Turing-gép

Definíció. Egy *Turing-gép* egy $M = (Q, \Sigma, \Gamma, q_0, \delta, F)$ rendszer, ahol

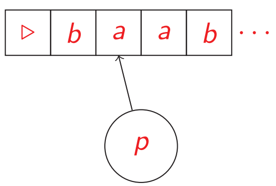
- Q véges halmaz, az *állapothalmaz*,
- Γ a *munka ábécé*, a $\triangleright$ (kezdet) és B (blank) Γ-ban van,
- Σ az *input ábécé*, $\Sigma \subseteq \Gamma - \{\triangleright, B\}$,
- $q_0 \in Q$ a *kezdőállapot*,
- $F \subseteq Q$, a *végállapotok halmaza*,
- $\delta : Q \times \Gamma \to Q \times \Gamma \times D$ egy parciális függvény, az *átmenetfüggvény*, ahol $D = \{\leftarrow, \rightarrow, \_\}$ a következő mozgási irányokat jelenti:
  $\leftarrow$: lépés balra, $\rightarrow$: lépés jobbra, $\_$: helyben maradás.

Parciális függvény: előfordulhat, hogy valamely $q \in Q$-ra és $a \in \Gamma$-ra $\delta(q, a)$ nem definiált!

A Turing-gép

Az M Turing-gép valamely állapotban van, inputja egy cellákra felosztott szalagon helyezkedik el. A cellákban Γ-beli szimbólumok vannak, a legelső cellában $\triangleright$ van. Az író-olvasó fej valamelyik cellában lévő szimbólumot olvassa. Az állapot és az olvasott szimbólum határozza meg a következő lépést.

Példa: tfh M a p állapotban van, az író-olvasó fej pedig a betűt olvas.



Ekkor $\delta(p, a) = \ldots$ határozza meg a következő lépést, ld. a következő diákon ...

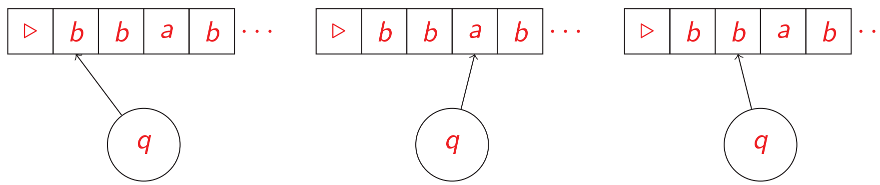
A Turing-gép

Példa folytatása:

1. eset: Ha

$$\delta(p, a) = (q, b, d),$$

akkor a-t felülírja b-vel, átmegy a q állapotba, és a $d \in \{\leftarrow, \rightarrow, \_\}$ értékétől függően balra lép egy cellányit, jobbra lép vagy helyben marad.



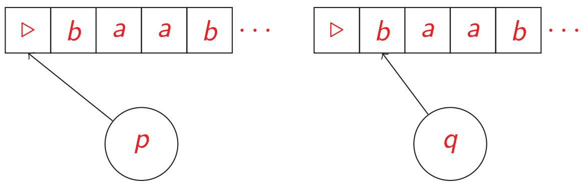
A Turing-gép

Példa folytatása:

Az első és az utolsó cella kezelése speciális: a szalagról nem lehet "leesni".

Az első cellában lévő $\triangleright$, nem írható felül és onnan mindig csak jobbra lehet lépni.

$$\delta(p, \triangleright) = (q, \triangleright, \rightarrow)$$

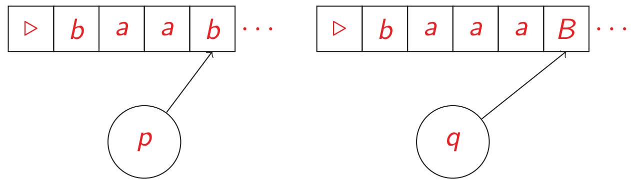


A Turing-gép

Példa folytatása:

Ha az utolsó cellát olvasta és jobbra lép, akkor az utolsó cella után ragaszt még egyet, amibe B-t ír. (Így a következő lépésben B-t olvas: $\delta(q, B) = \ldots$)

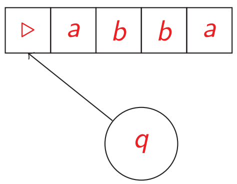
$$\delta(p, b) = (q, a, \rightarrow)$$



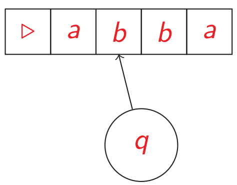
2. eset: Ha $\delta(p, a)$ nem definiált, akkor a Turing-gép megáll.

A Turing-gép

Egy konfigurációt úgy írunk le, hogy az állapotot azon betű elé írjuk, amire az író-olvasó fej mutat. Például:



leírva $q \triangleright abba$ és



leírva $\triangleright aqbba$

A Turing-gép

Példa Turing-gépre.

Megadunk egy Turing-gépet, amely a következőképpen dolgozik:

Az input egy $\{0, 1\}^*$-beli szó. Az inputban jobbra haladva megkeresi a legelső 1-est.

Ha megtalálta, átírja 0-ra, átmegy végállapotba, helyben marad és megáll.

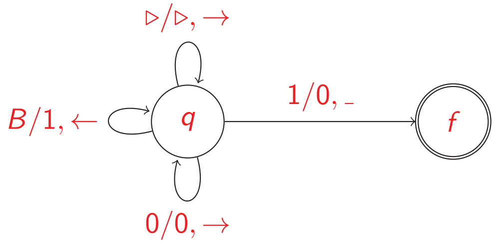
Ha B-vel találkozik, akkor átírja 1-re és balra lép.

Ez a következőképpen valósítható meg (következő dia).

A Turing-gép

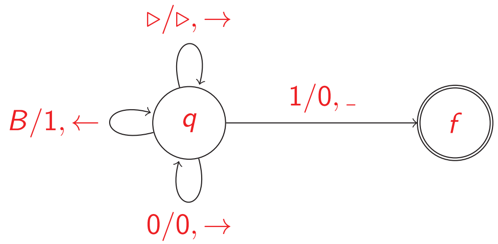
Példa Turing-gépre.
Ez a következőképpen valósítható meg:

- $Q = \{q, f\}$, kezdőállapot $= q$, végállapot $= f$
- $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, B, \rhd\}$,
- $\delta(q, \rhd) = (q, \rhd, \rightarrow)$, $\delta(q, 0) = (q, 0, \rightarrow)$, $\delta(q, 1) = (f, 0, \_)$, $\delta(q, B) = (q, 1, \leftarrow)$



A Turing-gép

Példa Turing-gépre.



Átmenetek:

$q \rhd 010 \vdash \rhd q010 \vdash \rhd 0q10 \vdash \rhd 0f00$
$q \rhd 00 \vdash \rhd q00 \vdash \rhd 0q0 \vdash \rhd 00qB \vdash \rhd 0q01 \vdash \rhd 00q1 \vdash \rhd 00f0$

Mindkét esetben megáll, mert $\delta(f, 0)$ nem definiált.

A Turing-gép

<u>Visszatérünk az általános esethez:</u>

$M = (Q, \Sigma, \Gamma, q_0, \delta, F)$ egy Turing-gép

$\rhd$ a szó eleje jel, B a blank. Kikötés, hogy ha $\delta(p, \rhd) = (q, b, d)$, akkor $b = \rhd$ és $d = \rightarrow$.

A Turing-gépeket ezen a kurzuson nyelvek felismerésére használjuk.

Konfigurációk. *M konfigurációi* a $p \rhd \alpha$ vagy $\rhd \alpha p a \beta$ alakú szavak, ahol $p \in Q, a \in \Gamma$ és $\alpha, \beta \in \Gamma^*$.

Megjegyzés: minden konfigurációban pontosan egy állapot szerepel, ami "olvas" egy Γ-beli betűt. Az első konfigurációban p olvassa $\rhd$-t, a másodikban p olvassa a-t.

A Turing-gép

Átmeneti reláció.

$\rhd \alpha p a \beta$	$\vdash$	$\rhd \alpha q b \beta$,	ha $\delta(p, a) = (q, b, \_)$,
$p \rhd \alpha$	$\vdash$	$\rhd q \alpha$,	ha $\delta(p, \rhd) = (q, \rhd, \rightarrow)$, $\alpha \neq \varepsilon$
$p \rhd$	$\vdash$	$\rhd q B$,	ha $\delta(p, \rhd) = (q, \rhd, \rightarrow)$, (!)
$\rhd \alpha p a \beta$	$\vdash$	$\rhd \alpha b q \beta$,	ha $\delta(p, a) = (q, b, \rightarrow)$, $\beta \neq \varepsilon$
$\rhd \alpha p a$	$\vdash$	$\rhd \alpha b q B$,	ha $\delta(p, a) = (q, b, \rightarrow)$, (!)
$\rhd \alpha c p a \beta$ minden	$\vdash$	$\rhd \alpha q c b \beta$, $c \in \Gamma$-ra	ha $\delta(p, a) = (q, b, \leftarrow)$,
$\rhd p a \beta$	$\vdash$	$q \rhd b \beta$,	ha $\delta(p, a) = (q, b, \leftarrow)$.

A Turing-gép

Észrevételek:
(1) Minden konfigurációra legfeljebb egy rákövetkező van (determinisztikus viselkedés).

(2) Általában nem minden konfigurációra van rákövetkező (mivel δ parciális függvény). Ilyenkor a Turing-gép megáll.

(3) Egy adott konfigurációból elinduló számolás eredményezhet végtelen konfiguráció-sorozatot, mert minden konfigurációnak lehet rákövetkezője (például $\delta(p, a) = (p, a, \_)$). Ilyenkor azt mondjuk, hogy az adott konfiguráción a Turing-gép nem áll meg.

(4) A szalag hossza jobb irányban tetszőleges nagyságúra kiterjedhet.

A Turing-gép

Kezdő konfiguráció: $q_0 \rhd x$ alakú, ahol $x \in \Sigma^*$.

Végkonfiguráció: $C = p \rhd \alpha$ vagy $C = \rhd \alpha p a \beta$ alakú, ahol $p \in F$ és nincs rákövetkező konfiguráció (mert $\delta(p, \rhd)$, illetve $\delta(p, a)$ nem definiált).

M felismeri az $x \in \Sigma^*$ szót, ha $q_0 \rhd x \vdash^* C$, ahol C végkonfiguráció.

Az $M = (Q, \Sigma, \Gamma, q_0, \delta, F)$ Turing-gép által felismert nyelv:

$L(M) = \{x \in \Sigma^* \mid q_0 \rhd x \vdash^* C,\ \text{ahol } C \text{ végkonfiguráció}\}$.

A Turing-gép

Példa Turing gépre:

Definíció: $(a_1 \dots a_n)^{-1} = a_n \dots a_1$,
pl. $(aab)^{-1} = baa$, $(abab)^{-1} = baba$.

Egy $w \in \Sigma^*$ szó palindróma, ha $w = w^{-1}$.

(Pl: $aba, abbabaababba$, stb. Indul a görög aludni.
Kis erek mentén, láp sík ölén, odavan a bánya rabja, jaj Baranyában a vadonélő Kis Pálnét nem keresik.)

Legyen $\Sigma = \{0, 1\}$ és $\mathrm{Pal}_\Sigma = \{w \in \Sigma^* \mid w = w^{-1}\}$.

Példa: Megadunk egy Turing-gépet, amely Pal_Σ-t ismeri fel.
(Ráadásul, minden inputon megáll). Ld. külön fájlban...

A Turing-gép változatai

1) k-szalagos Turing-gép
Egy szalag helyett k szalagon számol. Egy $M = (Q, \Sigma, \Gamma, q_0, \delta, F)$ rendszer, ahol az átmenetfüggvény most

$$\delta : Q \times \Gamma^k \to Q \times (\Gamma \times D)^k$$

alakú.
Ha M egy p állapotban van, a k darab író-olvasó fej pedig a szalagokról rendre az $a_1, \dots, a_k \in \Gamma$ betűket olvassa, akkor az egyes szalagokra írást és az azokon történő elmozdulásokat a

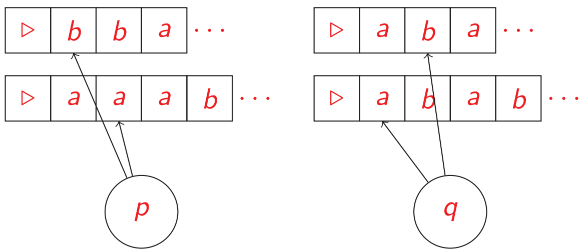
$$\delta(p, (a_1, \dots, a_k)) = (q, (b_1, d_1), \dots, (b_k, d_k))$$

érték határozza meg, ahol $b_1, \dots, b_k \in \Gamma$ és $d_1, \dots, d_k \in D$: az i-edik szalagon a_i helyére b_i-t ír és d_i irányban mozdul el.
Ha $a_i = \rhd$, akkor $b_i = \rhd$ és $d_i = \rightarrow$.

A Turing-gép változatai

Példa: kétszalagos Turing gép

$$\delta(p, b, a) = (q, (a, \rightarrow), (b, \leftarrow))$$



A következő lépés: $\delta(q, b, a) = \dots$

A Turing-gép változatai

Általános eset, számolás:

Ha M a p állapotban van, akkor az i-edik szalaghoz tartozik egy $\alpha_i p a_i \beta_i$ konfiguráció: az i-edik szalag tartalma $\alpha_i a_i \beta_i$ és az i-edik író-olvasó fej a_i-t olvassa.
A k szalag-konfiguráció meghatározza M egy konfigurációját:

$$(\alpha_1 p a_1 \beta_1, \dots, \alpha_k p a_k \beta_k).$$

Kezdő konfiguráció: $(q_0 \rhd x, q_0 \rhd, \dots, q_0 \rhd)$ (az első szalagon van az x input, a többi szalagon semmi.)
Végkonfiguráció: minden olyan konfiguráció (ld fent), ahol $p \in F$ és $\delta(p, (a_1, \dots, a_k))$ nem definiált. Ezek után:

$$L(M) = \{x \in \Sigma^* \mid (q_0 \rhd x, q_0 \rhd, \dots, q_0 \rhd) \vdash C, \text{ ahol } C \text{ végkonf.}\}$$

Példa: Megadunk egy kétszalagos Turing-gépet, amely Pal_Σ-t ismeri fel. Ld. külön fájlban...

A Turing-gép változatai

Az általánosítás nem növeli meg a felismerő kapacitást:

Tétel. Egy nyelv akkor és csak akkor ismerhető fel k-szalagos Turing-géppel, ha felismerhető Turing-géppel.

Bizonyítás. Tetszőleges k-szalagos M Turing géphez megadható olyan (egyszalagos) M' Turing gép, melyre $L(M) = L(M')$.

Vázlat: M' az M szavainak konkatenációját egyetlen szóban tárolja: vagyis M egy $(\alpha_1 p a_1 \beta_1, \dots, \alpha_k p a_k \beta_k)$ konfigurációjának megfelel M'-nek egy

$$(p, \rhd, \alpha_1 \underline{a_1} \beta_1 \lhd \dots \alpha_k \underline{a_k} \beta_k \lhd \lhd)$$

konfigurációja.
Az input szimbólumok aláhúzott változataival lehet szimulálni M író-olvasó fejeinek pozícióját, a $\lhd$ szimbólum pedig szeparálja a k szalag tartalmát.

A Turing-gép változatai

Ezek után M' "két menetben" szimulálja M egy lépését:

(1) végigolvassa az input szót és az állapotai segítségével "megjegyzi", hogy M mely input betűket olvassa a k szalagon;

(2) az első menetben begyűjtött információ birtokában szimulálja M lépéseit az egyes szalagokon.

Minden technikai részlet kezelhető (pl. egy szalag kiegészítése B-vel, az utána lévő szalagoknak megfelelő rész egy pozícióval jobbra történő mozgatását jelenti). $\diamond$

Teljes bizonyítás: C. H. Papadimitriou: Számítási bonyolultság, Novodat Bt., 1999.

A Turing-gép változatai

2) Nemdeterminisztikus Turing-gép

Egy $M = (Q, \Sigma, \Gamma, q_0, \delta, F)$, rendszer, ahol most

$$\delta : Q \times \Gamma \to \mathcal{P}(Q \times \Gamma \times D)$$

tehát $\delta(p, a) = \{(q_1, b_1, d_1), \ldots, (q_k, b_k, d_k)\}$.

A következő lépést a $\delta(p, a)$ elemei közül választva (nemdeterminisztikus működés), az átmeneti relációt és a felismert nyelvet pedig ugyanúgy definiáljuk, mint a Turing-gép esetében.

$L(M) = \{x \in \Sigma^* \mid q_0 \rhd x \vdash^* C, \text{ ahol } C \text{ végkonfiguráció } \}$.

A Turing-gép változatai

Ez az általánosítás nem növeli meg a felismerő kapacitást:

Tétel. Egy nyelv akkor és csak akkor ismerhető fel nemdeterminisztikus Turing-géppel, ha felismerhető Turing-géppel.

Bizonyítás. Tetszőleges nemdeterminisztikus $M = (Q, \Sigma, \Gamma, q_0, \delta, F)$ Turing géphez megadható olyan (determinisztikus) M' Turing gép, melyre $L(M) = L(M')$.
Vázlat: M nemdeterminisztikussági foka a

$$d = \max\{|\delta(q, a)| \mid q \in Q, a \in \Sigma\}$$

szám. (Legfeljebb ennyi lehetőség közül választhat egy lépésben.)

A Turing-gép változatai

M-nek t lépése jellemezhető egy $c_1c_2 \ldots c_t$ választási sorozattal, ahol $1 \leq c_1, c_2, \ldots, c_t \leq d$: az első lépésben a c_1-edik, a másodikban a c_2-edik, stb lehetőséget választja (nemdeterminisztikus működés).

M' a következőképpen utánozza M-et: A $q_0 \rhd x$ kezdő konfigurációból indul, majd minden $t \geq 1$-re felsorolja az összes $c_1c_2 \ldots c_t$ választási sorozatot, vagyis az $\{1, \ldots, d\}$ ábécé feletti 1, 2, stb hosszúságú szavakat:

$$1, \ldots, d, 11, \ldots, 1d, \ldots, d1, \ldots, dd, 111, \ldots, ddd, 1111, \ldots.$$

Minden egyes $c_1c_2 \ldots c_t$ sorozat felírása után végrehajtja M-nek a kezdő konfigurációból induló t hosszúságú lépés sorozatát, melyben a nemdeterminisztikus lehetőségek közül rendre a c_1-edik, ..., c_t-edik lehetőséget válsztja.

A Turing-gép változatai

Így M' az x inputon M minden lehetséges nemdeterminisztikus választási sorozatát kipróbálja. Ha M felismeri x-et, akkor M' "megtalálja" a megfelelő lépés sorozatot, ezért $L(M) = L(M')$.

M' fent leírt működése megfelel egy $q_0 \rhd x$ gyökerű, minden csúcspontban d felé ágazó fa szélességi bejárásának. $\diamond$

Teljes bizonyítás: C. H. Papadimitriou: Számítási bonyolultság, Novodat Bt., 1999.

Rekurzív és r.e. nyelvek

Definíció. A Turing-géppel felismerhető nyelveket *rekurzívan felsorolható* (röviden csak r.e.) nyelveknek nevezzük. A rekurzívan felsorolható nyelvek osztályát $\mathcal{L}_{re}$-vel jelöljük.

Definíció. Egy nyelv *eldönthető* (vagy: *rekurzív*), ha felismerhető olyan Turing-géppel, amely minden input szón megáll. A rekurzív nyelvek osztályát $\mathcal{L}_r$-vel jelöljük.

Az olyan Turing-gépet, amelyik minden input szón megáll algoritmusnak is nevezzük.

Például, Pal_Σ rekurzív. A környezefüggetlen nyelvek is rekurzívak, mert az "Eleme-e" problémát eldöntő CYK algoritmust meg lehet valósítani Turing géppel.
Valójában minden "értelmes" nyelv rekurzív és nehéz olyan nyelvet megadni, amelyik nem az. Még nehezebb olyat, amelyik nem rekurzívan felsorolható. De nekünk mindkettő sikerülni fog :)

Rekurzív és r.e. nyelvek

A *rekurzívan felsorolható* és a *rekurzív* történelmi elnevezések. A definíciók közvetlen következménye az alábbi két állítás.

Következmény. $\mathcal{L}_r \subseteq \mathcal{L}_{re}$. (És majd belátjuk, hogy valódi része.)

Tétel. A rekurzív nyelvek zártak a komplementerre: ha L rekurzív, akkor $\overline{L}$ is rekurzív.
Bizonyítás. Tegyül fel, hogy L-et felismeri egy olyan M Turing-gép, amelyik minden input szón megáll. Módosítsuk M-et úgy, hogy az F helyébe $Q - F$ kerül, minden más marad. A kapott M' Turing-gép továbbra is megáll minden input szón és pontosan akkor kerül végállapotba, ha M nincs végállapotba.
Ezért M' az $\overline{L}$ nyelvet ismeri fel. $\diamond$

Rekurzív és r.e. nyelvek

Tétel. Egy L nyelv akkor és csak akkor rekurzív, ha mind L, mind $\overline{L}$ rekurzívan felsorolható.

Bizonyítás. Legyen $L \subseteq \Sigma^*$.

$\Rightarrow$: Tfh, hogy L rekurzív. Akkor – mint láttuk –, $\overline{L}$ is rekurzív, tehát mindkettő r.e.

$\Leftarrow$: Tfh, hogy mind L, mind $\overline{L}$ r.e., továbbá, hogy L-et az M_1, $\overline{L}$-et az M_2 Turing-gép ismeri fel.

Rekurzív és r.e. nyelvek

Megadunk egy olyan M Turing-gépet, amely L-et ismeri fel és minden input szón megáll.

M kétszalagos és a következőképpen működik. Legyen x egy input szó.

0) M átmásolja x-et a második szalagra, majd a következő ciklust hajtja végre.
1) $i = 1$
2) M szimulálja M_1 i-edik lépését az első szalagon. Ha M_1 felismeri x-et, akkor M végállapotba megy és megáll.
3) M szimulálja M_2 i-edik lépését a második szalagon. Ha M_2 felismeri x-et, akkor M megáll egy nem végállapotban.
4) $i = i + 1$; goto 2).

Rekurzív és r.e. nyelvek

M (csakúgy, mint M_1), az L nyelvet ismeri fel.

Továbbá, M minden x input szón megáll. Valóban, ha $x \in L$, akkor M_1 ismeri fel x-et, ezért M a 2) pont miatt megáll. Ha $x \in \overline{L}$, akkor M_2 ismeri fel x-et, ezért M a 3) pont miatt megáll. Tehát M algoritmus, ezért L rekurzív nyelv. $\diamond$

Megjegyzés: ha M úgy működne, hogy először M_1, majd M_2 teljes működését szimulálná, akkor az $x \in \overline{L}$ input szavakon nem biztos, hogy megállna. Ezért van szükség M_1 és M_2 lépésenkénti párhuzamos szimulálására.

Nem rekurzív és nem r.e. nyelvek megadása

A következőkben megadunk egy olyan nyelvet, amelyik rekurzívan felsorolható, de nem rekurzív (és ezzel igazoljuk, hogy $\mathcal{L}_r \subset \mathcal{L}_{re}$) és egy olyat is, amelyik nem rekurzívan felsorolható.

Előkészületek 1): A $\{0,1\}$ feletti szavak (bináris szavak) sorba rendezése.

Egy w bináris szó sorszáma legyen az $1w$ bináris szám decimális értéke. Pl, 1 a harmadik, 01 az ötödik és 001 a kilencedik bináris szó lesz, mert $11_{10} = 3$, $101_{10} = 5$ és $1001_{10} = 9$.
Az 1 első számjegyre azért van szükség, mert a 0-val kezdődő szavak miatt több bináris szónak is ugyanaz lenne a sorszáma.

Tehát a bináris szavak sorba rendezése:
$w_1 = \varepsilon$, $w_2 = 0$, $w_3 = 1$, $w_4 = 00$, $w_5 = 01$, $w_6 = 10$, ...

Nem rekurzív és nem r.e. nyelvek megadása

Előkészületek 2): A $\{0,1\}$ input ábécével rendelkező Turing gépek kódolása bináris szavakkal.

Az általánosság korlátozása nélkül feltehető, hogy a Turing gép állapotai, input és munka szimbólumai és az irányok a következő sorozatok "prefixei":

állapotok: q_1 (kezdő áll.), q_2 (végáll.), q_3, q_4, ...
szimbólumok: $X_1 = 0$, $X_2 = 1$, $X_3 = \triangleright$, $X_4 = B$, X_5, ...
irányok: $d_1 = \leftarrow$, $d_2 = \rightarrow$, $d_3 = \lrcorner$

Ugyancsak az általánosság korlátozása nélkül feltehető, hogy csak egy végállapot van.

Nem rekurzív és nem r.e. nyelvek megadása

Előkészületek 2): A $\{0,1\}$ input ábécével rendelkező Turing gépek kódolása bináris szavakkal.
A Turing gép minden átmenete

$$\delta(q_i, X_j) = (q_k, X_l, d_m)$$

alakú, ahol $i,j,k,l \geq 1$ és $1 \leq m \leq 3$. Legyen ennek az átmenetnek a kódja a

$$0^i 1 0^j 1 0^k 1 0^l 1 0^m$$

bináris szó. (Nem tartalmaz 11 alakú rész-szót!)
Magának a Turing gépnek a kódja legyen

$$\text{kód}_1 11 \text{kód}_2 11 \ldots 11 \text{kód}_n,$$

ahol n az átmenetek száma, kód_i pedig az i-edik átmenet kódja.

Nem rekurzív és nem r.e. nyelvek megadása

Előkészületek 2): A $\{0,1\}$ input ábécével rendelkező Turing gépek kódolása bináris szavakkal.

Minden Turing gép a kódja $\text{kód}_1 11 \text{kód}_2 11 \ldots 11 \text{kód}_n$ alakú.

Nyilvánvaló, hogy meg tudunk adni egy olyan algoritmust (minden inputon megálló Turing-gépet) amely minden $w \in \{0,1\}^*$ szóról el tudja dönteni, hogy Turing-gép kódja-e.

(Vázlat: Ha tartalmaz 111 alakú rész-szót, akkor nem Turing gép kódja. Ha nem, akkor megvizsgáljuk, hogy az 11 alakú rész-szavak által szeparált részek átmenetek kódjai-e.)

Nem rekurzív és nem r.e. nyelvek megadása

Tétel. $\mathcal{L}_r \subset \mathcal{L}_{re}$. (Van olyan r.e. nyelv, amelyik nem rekurzív.)
Bizonyítás. Tekintsük a $\{0,1\}$ input ábécével rendelkező Turing gépeket és definiáljuk minden $i = 1, 2, \ldots$-re az i-edik Turing-gépet a következőképpen:

Legyen $w_1, w_2, w_3, \ldots$ a bináris szavak ismert sorbarendezése. (Nem mindegyik Turing-gép kódja!)

Legyen az i-edik, Turing-gép
- a w_i által kódolt Turing-gép, ha w_i egy Turing-gép kódja,
- különben legyen egy olyan Turing-gép, amely az $\emptyset$-t ismeri fel.

Nem rekurzív és nem r.e. nyelvek megadása

Tekintsük a következő mátrixot:

	w_1	w_2	w_3	$\cdots$
M_1				$\cdots$
M_2				$\cdots$
M_3				$\cdots$
$\vdots$	$\vdots$	$\vdots$	$\vdots$	

és legyen az (i,j)-edik elem 1, ha $w_j \in L(M_i)$, különben legyen 0.
Legyen

$$L_d = \{w_i \mid i \geq 1, w_i \in L(M_i)\}$$

(a diagonális nyelv, Cantor féle diagonalizálás). Megmutatjuk, hogy L_d rekurzívan felsorolható, de nem rekurzív.

Nem rekurzív és nem r.e. nyelvek megadása

1. Állítás. L_d r.e. (vagyis felismerhető Turing-géppel).

Bizonyítás. Az L_d-t felismerő M Turing-gép a következő:

1. szalag: $x = w_i$ input szó
2. szalag: M előállítja M_i Turing-gépet (lásd fentebb)
3. szalag: M szimulálja M_i működését x-en.

Ha M_i felismeri x-et (és ezért megáll), akkor M is ismerje fel x-et.
Tehát $L_d = L(M)$.

Nem rekurzív és nem r.e. nyelvek megadása

Ahhoz, hogy L_d nem rekurzív, elegendő megmutatni, hogy

$$\overline{L}_d = \{w_i \mid i \geq 1, w_i \notin L(M_i)\}$$

nem rekurzívan felsorolható (az előző tétel miatt).

2. Állítás. $\overline{L}_d$ nem r.e.

Bizonyítás. Tfh $\overline{L}_d$ r.e. Ez azt jelenti, hogy $\overline{L}_d = L(M_j)$ valamelyik j-re. Megvizsgáljuk, hogy $w_j \in \overline{L}_d$ teljesül-e. Kapjuk, hogy

$w_j \in \overline{L}_d \Rightarrow w_j \in L(M_j)$ (mert $\overline{L}_d = L(M_j)$) $\Rightarrow w_j \notin \overline{L}_d$ ($\overline{L}_d$ def.) és

$w_j \notin \overline{L}_d \Rightarrow w_j \in L(M_j)$ ($\overline{L}_d$ def.) $\Rightarrow w_j \in \overline{L}_d$ (mert $\overline{L}_d = L(M_j)$),

ami ellentmondás. Tehát $\overline{L}_d$ nem r.e. $\diamond$

Nem rekurzív és nem r.e. nyelvek megadása

Jegyezzük meg!

A bizonyítás során definiált L_d és $\overline{L}_d$ nyelvek tulajdonságai:

- L_d nem rekurzív, de rekurzívan felsorolható,
- $\overline{L}_d$ nem rekurzívan felsorolható.

Rekurzív és r.e. nyelvek nyelvek zártsági tulajdonságai

Tétel. A rekurzív nyelvek és a rekurzívan felsorolható nyelvek is zártak az egyesítésre.

Bizonyítás. Tfh, hogy $L_1 = L(M_1)$ és $L_2 = L(M_2)$, ahol M_1 és M_2 Turing gépek.

Egy kétszalagos M Turing gépet konstruálunk, amely a második szalagra másolja az x inputot, majd a szalagokon lépésenként (párhuzamosan) szimulálja M_1 és M_2 működését.

Ha M_1 vagy M_2 elfogad, akkor M is elfogad.

Ha M_1 és M_2 minden inputon megáll, akkor M is. $\diamond$

Rekurzív és r.e. nyelvek nyelvek zártsági tulajdonságai

Tétel. A rekurzív nyelvek és a rekurzívan felsorolható nyelvek is zártak a metszésre.

Bizonyítás. Tfh, hogy $L_1 = L(M_1)$ és $L_2 = L(M_2)$, ahol M_1 és M_2 Turing gépek.

Egy kétszalagos M Turing gépet konstruálunk, amely a második szalagra másolja az x inputot, majd a szalagokon lépésenként (párhuzamosan) szimulálja M_1 és M_2 működését.

Ha M_1 és M_2 elfogad, akkor M is elfogad.

Ha M_1 és M_2 minden inputon megáll, akkor M is. $\diamond$

Rekurzív és r.e. nyelvek nyelvek zártsági tulajdonságai

Azt már láttuk, hogy a rekurzív nyelvek zártak a komplementerre. Ezzel szemben:

Tétel. A rekurzívan felsorolható nyelvek nem zártak a komplementerre.

Bizonyítás. Mint láttuk egy L akkor és csak akkor rekurzív, ha mind L, mind $\overline{L}$ r.e.

Ebből következik, hogy ha a rekurzívan felsorolható nyelvek zártak lennének a komplementerre, akkor $\mathcal{L}_r = \mathcal{L}_{re}$. Ez viszont nem igaz, mert $\mathcal{L}_r \subset \mathcal{L}_{re}$. ◇

Rekurzív és r.e. nyelvek nyelvek zártsági tulajdonságai

Összefoglalás

Művelet	**Zártság**	**Zártság**
	$\mathcal{L}_r$	$\mathcal{L}_{re}$
$\cup$	igen	igen
konkatenáció	igen	igen
$*$	igen	igen
$\cap$	igen	igen
komplementer	igen	nem

Általános nyelvek

Definíció. Egy $G = (N, \Sigma, P, S)$ nyelvtan 0-*típusú* (vagy *kifejezés struktúrájú* vagy *általános*), ha a szabályaira semmilyen korlátozás nincs.

Megmutatjuk, hogy az általános nyelvek pontosan a Turing-gépekkel felismerhető nyelvek.

Lemma. $\mathcal{L}_0 \subseteq \mathcal{L}_{re}$. (Minden általános nyelvtannal generálható nyelv felismerhető Turing-géppel.)
Bizonyítás. Legyen $L = L(G)$, ahol $G = (N, \Sigma, P, S)$ egy 0-típusú nyelvtan. Konstruálunk egy 2 szalagos, nemdeterminisztikus Turing-gépet, amely L-et ismeri fel.
M az első szalagon az $x \in \Sigma^*$ input szót tartja, a második szalagon pedig előállítja az $S \Rightarrow^* \alpha$ derivációkkal kapható α mondatformákat és a következőképpen működik.

Általános nyelvek

<u>M működése:</u>

0) Első szó : $x \in \Sigma^*$, második szó: S (α kezdőértéke).
1) Nemdeterminisztikusan kiválaszt egy $1 \leq i \leq |\alpha|$ pozíciót α-ban és G-nek egy $\beta \to \gamma$ szabályát.
2) Ha α-ban az i-edik pozíción kezdődő rész-szó β, akkor felülírja azt γ-val. (Ha $|\beta| \neq |\gamma|$, akkor α részeit el kell ,,csúsztatni'' a szalagon balra vagy jobbra.)
Különben M megáll (nem végállapotban).
3) Összehasonlítja x-et α-val. Ha $x = \alpha$, akkor felismeri x-et, különben goto 1).

<u>Észrevétel:</u> M akkor és csak akkor ismeri fel x-et, ha $S \Rightarrow^* x$. ◇

Általános nyelvek

Lemma. $\mathcal{L}_{re} \subseteq \mathcal{L}_0$. (Minden Turing-géppel felismerhető nyelv generálható általános nyelvtannal.)

Bizonyítás. Legyen $M = (Q, \Sigma, \Gamma, q_0, \delta, F)$ egy Turing-gép. Vegyünk fel egy $\lhd$ végjelet és írjuk M átmeneteit ,,szabály alakban''.

$pa \to qb$		(helyben maradás)
$pac \to bqc$	$\forall c \in \Gamma$	(lépés jobbra)
$pa\lhd \to bqB\lhd$		(lépés jobbra és kiterjesztés B-vel)
$cpa \to qcb$	$\forall c \in \Gamma$	(lépés balra)

Legyen R ezen szabályok halmaza.

Általános nyelvek

Konstruáljuk meg a $G = (N, \Sigma, P, S)$ nyelvtant, ahol

- $N = Q \cup (\Gamma - \Sigma) \cup \{S, A, A', E\}$, ahol S, A, A', E új betűk,
- P a legszűkebb halmaz, melyre:
  (1) Minden $\alpha \to \beta \in R$ esetén $\beta \to \alpha \in P$;
  (2) A következő szabályok P-ben vannak:
    (-) $S \to \rhd A$ és $S \to q \rhd A'\lhd$, ha $q \in F$ és $\delta(q, \rhd)$ nem definiált;
    (-) $A \to aA$, $A' \to aA'|\varepsilon$ minden $a \in \Gamma$-ra,
    (-) $A \to E\lhd$,
    (-) $E \to Ea$, minden $a \in \Gamma$-ra,
    (-) $E \to pa$, minden $p \in F, a \in \Gamma$-ra, melyre $\delta(p, a)$ nem definiált;
  (3) $q_0\rhd \to \varepsilon$ és $\lhd \to \varepsilon$ P-ben vannak.

Általános nyelvek

Észrevételek:

- Az (1) típusú szabályokkal G nyelvtan M ,,visszafelé mozgását'' szimulálja.

- A (2) típusú szabályokkal S-ből M végkonfigurációi vezethetők le.

Például:

$S \Rightarrow \rhd A \Rightarrow^* \rhd\alpha A \Rightarrow \rhd\alpha E\lhd \Rightarrow^* \rhd\alpha E\beta\lhd \Rightarrow \rhd\alpha pa\beta\lhd$,

ahol $\alpha, \beta \in \Gamma^*$, $a \in \Gamma$ és $p \in F$. Tehát $\rhd\alpha pa\beta$ végkonfiguráció.

Hasonlóan:

$S \Rightarrow q \rhd A'\lhd \Rightarrow^* q \rhd \alpha\lhd$,

ahol $\alpha \in \Gamma^*$ és $p \in F$. Tehát $q \rhd \alpha$ végkonfiguráció.

Általános nyelvek

- Tehát először a (2) majd az (1) típusú szabályok alkalmazásával S-ből levezethetők M kezdő konfigurációi.

Összefoglalva:

Minden $x \in \Sigma^*$-ra:

$x \in L(M)$	$\iff$	$q_0 \rhd x\lhd$	$\vdash^*$	$C\lhd$,	ahol C végkonf.
	$\iff$	S	$\Rightarrow^*_G$	$C\lhd$	(2) szabályokkal és
			$\Rightarrow^*_G$	$q_0 \rhd x\lhd$	(1) szabályokkal és
			$\Rightarrow^2_G$	x	(3) szabályokkal
	$\iff$				
$x \in L(G)$.					

Az első és a harmadik ekvivalencia definíció szerint áll fenn. Tehát $L(M) = L(G)$.

Általános nyelvek

A két lemmát összevetve kapjuk a következő eredményt.

Tétel. $\mathcal{L}_0 = \mathcal{L}_{re}$, vagyis:

a rekurzívan felsorolható nyelvek megegyeznek a 0-típusú (általános) nyelvtanokkal generálható nyelvekkel.

A Chomsky nyelvosztályok gépi reprezentációi

Gépi reprezentáció:

Nyelvosztály	Gépi repr.	Det. vs nemdet.
$\mathcal{L}_3$ (reguláris nyelvek)	véges automaták	det. $=$ nemdet.
$\mathcal{L}_2$ (k. független nyelvek)	veremautomaták	det. $\subset$ nemdet.
$\mathcal{L}_1$ (k. függő nyelvek)	?	?
$\mathcal{L}_0$ (általános nyelvek)	Turing gépek	det. $=$ nemdet.

Környezetfüggő nyelvek

Definíció. Egy $G = (N, \Sigma, P, S)$ nyelvtan 1 típusú (vagy *környezetfüggő*), ha P-ben minden szabály $\alpha A \beta \to \alpha \delta \beta$ alakú, ahol $\delta \neq \varepsilon$. Kivétel, az $S \to \varepsilon$ szabály, ekkor azonban az S nem szerepelhet semelyik szabály jobb oldalán.

Definíció. Egy $G = (N, \Sigma, P, S)$ nyelvtan *monoton*, ha minden $\alpha \to \beta \in P$ szabály esetén $|\alpha| \leq |\beta|$. Kivétel, az $S \to \varepsilon$ szabály, ekkor azonban az S nem szerepelhet semelyik szabály jobb oldalán.

Környezetfüggő nyelvek

Tétel. A környezetfüggő nyelvek osztálya megegyezik a monoton nyelvek osztályával.

Bizonyítás.

a) Ha $\delta \neq \varepsilon$, akkor $|\alpha A \beta| \leq |\alpha \delta \beta|$. Tehát minden környezetfüggő nyelvtan egyben monoton is és ezért minden környezetfüggő nyelv monoton.

Környezetfüggő nyelvek

b) Legyen $G = (N, \Sigma, P, S)$ monoton nyelvtan. Megkonstruálunk egy vele ekvivalens $G' = (N', \Sigma, P', S)$ környezetfüggő nyelvtant.

Feltehető, hogy terminális betűk csak $A \to a$ alakú szabályokban szerepelnek (lásd a Chomsky normálalak bizonyítását).

Az $S \to \varepsilon$ és az $A \to a$ alakú (vagyis a terminális betűket tartalmazó) szabályokat egyszerűen áttesszük P-ből P'-be. Továbbá, ...

Környezetfüggő nyelvek

Minden $A_1 \ldots A_m \to B_1 \ldots B_n$, $m \leq n$, csak nemterminális betűket tartalmazó P-beli szabályra, legyenek az

- $A_1 \ldots A_m \to C_1 A_2 \ldots A_m$,
- $C_1 A_2 \ldots A_m \to C_1 C_2 A_3 \ldots A_m$,
- $\ldots$
- $C_1 \ldots C_{m-1} A_m \to C_1 \ldots C_{m-1} B_m \ldots B_n$,
- $C_1 \ldots C_{m-1} B_m \ldots B_n \to B_1 C_2 \ldots C_{m-1} B_m \ldots B_n$,
- $B_1 C_2 \ldots C_{m-1} B_m \ldots B_n \to B_1 B_2 C_3 \ldots C_{m-1} B_m \ldots B_n$,
- $\ldots$
- $B_1 \ldots B_{m-2} C_{m-1} B_m \ldots B_n \to B_1 \ldots B_n$, szabályok P'-ben, ahol $C_1, \ldots, C_{m-1}$ új nemterminálisok.

Környezetfüggő nyelvek

Legyen N' az összes régi és új nemterminálisok halmaza.

Nyilvánvaló, hogy az így megkonstruált G' nyelvtan környezefüggő. Továbbá:
$L(G) \subseteq L(G')$, mivel az új P'-beli szabályokkal szimulálhatók a P-beli szabályok, és
$L(G') \subseteq L(G)$, mivel az új P'-beli szabályokkal újabb terminális szavak nem vezethetők le.

Tehát $L(G) = L(G')$.

Környezetfüggő nyelvek

Tétel. $\mathcal{L}_1 \subseteq \mathcal{L}_r$, vagyis minden környezetfüggő nyelv rekurzív.

Bizonyítás. Legyen $L = L(G)$, ahol $G = (N, \Sigma, P, S)$ egy monoton nyelvtan. Konstruálunk egy 2 szalagos Turing-gépet, amely eldönti L-et (felismeri L-et és minden input szón megáll).

Környezetfüggő nyelvek

M működése:

0) Első szalag : $x \in \Sigma^*$, második szalag: üres
1) A második szalagra M felsorolja az összes $\alpha_0, \alpha_1, \ldots, \alpha_n$ sorozatot, ahol
(a) $n \geq 1$ és minden $0 \leq i \leq n$-re, $\alpha_i \in (N \cup \Sigma)^*$,
(b) $\alpha_0 = S$, $\alpha_n = x$,
(c) minden $0 \leq i \leq n-1$-re $|\alpha_i| \leq |\alpha_{i+1}|$ (monoton!),
(d) minden $0 \leq i < j \leq n$-re, $\alpha_i \neq \alpha_j$.
2) Egy sorozat felírása után megnézi, hogy teljesül-e $\alpha_i \Rightarrow \alpha_{i+1}$, minden $0 \leq i \leq n-1$-re. Ha igen, akkor felismeri x-et, különben felírja a következő sorozatot.
3) Ha 2) egyetlen sorzatra sem teljesül, akkor megáll (de nem ismeri fel x-et).

Környezetfüggő nyelvek

Észrevételek:

1) M (az általános nyelvet felismerő Turing-géppel ellentétben) minden input szón megáll, mert a nyelvtan monotonitása miatt egy adott x-re csak véges számú olyan sorozat van, amely rendelkezik az (a), (b), (c) és (d) tulajdonságokkal.

2) Ezért M akkor és csak akkor ismeri fel x-et, ha $S \Rightarrow^* x$, tehát M eldönti $L(G)$-t.

Környezetfüggő nyelvek

A tétel általánosabb formában is igaz:

Tétel. $\mathcal{L}_1 \subset \mathcal{L}_r$.

Bizonyítás. A tartalmazás valódisága úgy igazolható, hogy a Cantor-féle diagonalizálási eljárással megadunk egy alkalmas L nyelvet, amely rekurzív, de nem környezetfüggő.

A Chomsky-féle nyelvhierarchia

Tétel. $\mathcal{L}_3 \subset \mathcal{L}_2 \subset \mathcal{L}_1 \subset \mathcal{L}_0$.

Bizonyítás. Az $\mathcal{L}_3 \subset \mathcal{L}_2$ és $\mathcal{L}_2 \subset \mathcal{L}_1$ tartalmazásokat már igazoltuk. Továbbá, igazoltuk, hogy:

$$\mathcal{L}_1 \subseteq \mathcal{L}_r \subset \mathcal{L}_{re} = \mathcal{L}_0.$$

Emlékeztető:

- $\{a^n b^n \mid n \geq 0\} \in \mathcal{L}_2 - \mathcal{L}_3$
- $\{a^n b^n c^n \mid n \geq 0\} \in \mathcal{L}_1 - \mathcal{L}_2$
- $L_d \in \mathcal{L}_{re} - \mathcal{L}_r$, ezért $L_d \in \mathcal{L}_0 - \mathcal{L}_1$

Megjegyezzük, hogy $\overline{L_d}$ még $\mathcal{L}_0$-ban sincs benne.

Lineárisan korlátos automaták

A lineárisan korlátos automata egy olyan speciális Turing-gép, amely (az általános Turing-géppel ellentétben) nem léphet túl az input szó jobb oldali végén sem. A szó jobb oldali végét is jelzi egy speciális betű, amely visszafordítja az író-olvasó fejet. A lineárisan korlátos automaták pontosan a környezetfüggő nyelveket ismerik fel.

Lineárisan korlátos automaták

Definíció. A lineárisan korlátos automata egy $M = (Q, \Sigma, \Gamma, q_0, R, F)$ rendszer, ahol

- Q az *állapothalmaz*,
- Σ az *input ábécé*,
- Γ a *munka ábécé*, ahol $\Sigma \subseteq \Gamma$ és $\rhd, \lhd \in \Gamma - \Sigma$ (kezdet, vég),
- $q_0 \in Q$ *a kezdőállapot*,
- $F \subseteq Q$, a *végállapothalmaz*,
- R a következő alakú *átmenetek* halmaza, ...

Lineárisan korlátos automaták

... ahol $p, q \in Q$ és $a, b \in \Gamma - \{\rhd, \lhd\}$:

$pa \to qb$		(helyben maradás)	(1)
$pa \to bq$		(jobbra lépés)	(2)
$p\rhd \to \rhd q$		(jobbra lépés)	(3)
$cpa \to qcb$	$(\forall c \in \Gamma - \{\lhd\})$	(balra lépés)	(4)
$cp\lhd \to qc\lhd$	$(\forall c \in \Gamma - \{\lhd\})$	(balra lépés)	(5).

Az átmeneteket ,,szabály alakban'' írjuk.

Lineárisan korlátos automaták

M konfigurációi $p \rhd \alpha \lhd$ vagy $\rhd \alpha p a \beta \lhd$ alakúak, ahol $\alpha, \beta \in (\Gamma - \{\rhd, \lhd\})^*$. A $\vdash$ átmeneti relációt a következőképpen definiáljuk:

$\rhd\alpha pa\beta\lhd$	$\vdash$	$\rhd\alpha qb\beta\lhd$	ha (1) átmenet R-ben van,
$\rhd\alpha pa\beta\lhd$	$\vdash$	$\rhd\alpha bq\beta\lhd$	ha (2) átmenet R-ben van,
$p \rhd \alpha\lhd$	$\vdash$	$\rhd q\alpha\lhd$	ha (3) átmenet R-ben van,
$\rhd\alpha cpa\beta\lhd$	$\vdash$	$\rhd\alpha qcb\beta\lhd$	ha (4) átmenet R-ben van és $c \neq \rhd$,
$\rhd pa\beta\lhd$	$\vdash$	$q \rhd b\beta\lhd$	ha (4) átmenet R-ben van és $c = \rhd$,
$\rhd\alpha cp\lhd$	$\vdash$	$\rhd\alpha qc\lhd$	ha (5) átmenet R-ben van.

Lineárisan korlátos automaták

Észrevételek:

(1) Nemdeterminisztikus viselkedés.

(2) Nem minden konfigurációra van rákövetkező (a számolás elakadhat).

(3) Ugyanakkor, egy adott konfigurációból elinduló számolás nem mindig fejeződik be, mert minden konfigurációnak lehet rákövetkezője, például $pa \to pa$.

(4) A számolás a $\rhd$ és $\lhd$ jelek között történik, tehát a számolás hossza rögzített.

Lineárisan korlátos automaták

Kezdő konfiguráció: $q_0 \rhd x \lhd$ alakú, ahol $x \in \Sigma^*$.

Végkonfiguráció: $C = p \rhd \alpha \lhd$ vagy $C = \rhd \alpha p a \beta \lhd$ alakú, ahol $p \in F$ és nincs rákövetkező konfiguráció.

M felismeri az $x \in \Sigma^*$ szót, ha $q_0 \rhd x \lhd \vdash^* C$, ahol C végkonfiguráció.

Az M által felismert nyelv:

$L(M) = \{x \in \Sigma^* \mid q_0 \rhd x \lhd \vdash^* C \text{ végkonfiguráció } \}$.

Lineárisan korlátos automaták

Tétel. Minden környezetfüggő nyelv felismerhető lineárisan korlátos automatával.

Bizonyítás. (Vázlat.) Legyen $G = (N, \Sigma, P, S)$ egy monoton nyelvtan. Megadunk egy M lineárisan korlátos automatát, melyre $L(G) = L(M)$.

Legyenek $\alpha_1 \to \beta_1, \ldots, \alpha_n \to \beta_n$ a P-beli szabályok. Mivel G monoton, ezért minden $1 \leq i \leq n$ esetén $|\alpha_i| \leq |\beta_i|$. Egészítsük ki α_i-t $|\beta_i| - |\alpha_i|$ számú B betűvel, (ahol $B \notin (N \cup \Sigma)$). Legyen az így kapott szó α_i', melyre nyilván $|\alpha_i'| = |\beta_i|$.

M munka ábécéje $N \cup \Sigma \cup \{B\}$, input ábécéje Σ.

Lineárisan korlátos automaták

M-nek hat „fő”-állapota van: <u>start</u>, <u>choice</u>, <u>match</u>$_i$, $1 \leq i \leq n$, <u>return</u>, <u>check</u> és <u>accept</u>, melyek közül <u>start</u> a kezdőállapot és <u>accept</u> az egyedüli végállapot.

Mindegyik állapottal egy tevékenységet hajtunk végre. (Ezen tevékenységek további „segéd”-állapotokkal realizálhatók, amelyeket nem részletezünk.)

Az egyes állapotokkal végrehajtott tevékenységek a következők.

Lineárisan korlátos automaták

start: goto choice vagy goto check, nemdeterminisztikusan.

choice: Kiválaszt az inputban egy pozíciót, és egy $1 \leq i \leq n$ szabály-indexet, majd goto match_i.

match_i: Az input kiválsztott pozíciójától kezdve az i-edik szabály β_i jobb oldalát betűről-betűre összehasonlítja az input megfelelő betűjével (átugorva az inputban szereplő B betűket). Ha a két betű megegyezik, akkor az input betűt felülírja α_i' megfelelő betűjével. Ha a teljes β_i-t sikerült illeszteni, akkor goto return.

return: Visszamegy az input szó elejére és goto start.

check: Megvizsgálja, hogy az input szó eleme-e $\{B\}^*S\{B\}^*$-nek. Ha igen, goto accept.

accept: Nincs tevékenység.

Lineárisan korlátos automaták

M az x inputból kiindulva, G szabályainak nemdeterminisztikus módon történő, ,,visszafelé'' alkalmazásával megpróbál S-hez, pontosabban, egy $|x|$ hosszúságú $\{B\}^*S\{B\}^*$-beli szóhoz eljutni. Mivel G monoton, elegendő egy $|x|$ hosszúságú munkaterületen dolgozni (ellentétben az általános nyelvtan esetével).

Ez akkor és csak akkor sikerül, ha $x \in L(G)$. Tehát $L(M) = L(G)$.

Lineárisan korlátos automaták

Fordítva is igaz (bizonyítás nélkül):

Tétel. Minden lineárisan korlátos automatával felismerhető nyelv környezetfüggő.

Egy nyitott kérdés: megegyeznek-e a lineárisan korlátos automatákkal felismerhető nyelvek a lineárisan korlátos determinisztikus automatákkal felismerhető nyelvekkel?

A Chomsky nyelvosztályok gépi reprezentációi

Gépi reprezentáció:

Nyelvosztály	Gépi repr.	Det. vs nemdet.
$\mathcal{L}_3$ (reguláris nyelvek)	véges automaták	det. = nemdet.
$\mathcal{L}_2$ (k. független nyelvek)	veremautomaták	det. $\subset$ nemdet.
$\mathcal{L}_1$ (k. függő nyelvek)	lin. korl. automaták	det. ? nemdet.
$\mathcal{L}_0$ (általános nyelvek)	Turing gépek	det. = nemdet.

Utolsó dia

VÉGE