20. Automaták és formális nyelvek

A fordítóprogramok tervezésében és megvalósításában az automaták és formális nyelvek elmélete fontos szerepet játszik. A fejezet elején értelmezzük a formális nyelv és a nyelvtan fogalmát. A Chomsky-féle felosztás alapján tárgyaljuk a különböző nyelv- és nyelvtantípusokat. Részletesen foglalkozunk a véges automatákkal és az általuk felismert reguláris nyelvekkel. Befejezésül a veremautomatákról és a környezetfüggetlen nyelvekről lesz szó. A fejezet végén a gazdag szakirodalomból felsorolunk néhány fontosabb könyvet.

20.1. Nyelvek és nyelvtanok

Tetszőleges szimbólumok (jelek) nemüres, véges halmazát ***ábécének*** nevezzük. A halmaz elemeit az ábécé ***betűinek*** nevezzük, de sokszor használjuk a jel és szimbólum neveket is. Ábécé például a $\Sigma = \{a, b, c, d, 0, 1, \sigma\}$, amelynek betűi $a, b, c, d, 0, 1, \sigma$.

Az ábécé betűiből szavakat képezünk. Ha $a_1, a_2, \ldots, a_n \in \Sigma, n \geq 0$, akkor $a_1 a_2 \ldots a_n$ a Σ ábécé betűiből képzett ***szó*** (az a_i-k nem feltétlenül különbözőek). A szót alkotó betűk száma a szó ***hossza***. Ha $w = a_1 a_2 \ldots a_n$, akkor a w hossza $|w| = n$. Ha $n = 0$, akkor a szó egyetlen betűt sem tartalmaz, és ***üres szónak*** nevezzük. Jelölése: ε (sok könyvben λ). A Σ betűiből képezhető szavak halmazának jelölésére a Σ^* jelet használjuk:

$$\Sigma^* = \left\{ a_1 a_2 \ldots a_n \mid a_1, a_2, \ldots, a_n \in \Sigma,\, n \geq 0 \right\} .$$

A nemüres szavak halmaza $\Sigma^+ = \Sigma^* \setminus \{\varepsilon\}$. Az n hosszúságú szavak halmazát Σ^n-nel jelöljük, és természetesen $\Sigma^0 = \{\varepsilon\}$. Ekkor

$$\Sigma^* = \Sigma^0 \cup \Sigma^1 \cup \ldots \cup \Sigma^n \cup \ldots \qquad \text{és} \qquad \Sigma^+ = \Sigma^1 \cup \Sigma^2 \cup \ldots \cup \Sigma^n \cup \ldots$$

Azt mondjuk, hogy $u = v$, ha $u = a_1 a_2 \ldots a_m$ és $v = b_1 b_2 \ldots b_n$ esetében $m = n$ és $a_i = b_i,\ \ i = 1, 2, \ldots, n$.

A Σ^*-ban bevezetünk egy bináris műveletet, két szó konkatenációját. Az $u = a_1 a_2 \ldots a_m$ és $v = b_1 b_2 \ldots b_n$ szavak ***konkatenációján*** (illesztésén, szorzatán, összefűzésén) az $uv = a_1 a_2 \ldots a_m b_1 b_2 \ldots b_n$ szót értjük. Nyilvánvaló, hogy $|uv| = |u| + |v|$. Ez a művelet asszociatív, de nem kommutatív. Van egységeleme, az ε, mivel

$\varepsilon u = u\varepsilon = u$ tetszőleges $u \in \Sigma^*$ szóra. Tehát Σ^* ezzel a művelettel ***monoidot*** (egységelemes félcsoportot) képez.

Bevezetjük a szavak hatványozását. Ha $u \in \Sigma^*$ akkor

$u^0 = \varepsilon$

$u^n = u^{n-1}u$, ha $n \geq 1$.

Az $u = a_1 a_2 \dots a_n$ szó *tükörképe* $u^{-1} = a_n a_{n-1} \dots a_1$. A tükörkép jelölésére néha használják még a következőket is: u^R, $\tilde{u}$. Nyilvánvaló, hogy $\left(u^{-1}\right)^{-1} = u$ és $(uv)^{-1} = v^{-1}u^{-1}$.

A v szó ***részszava*** az u szónak, ha léteznek a p és q szavak úgy, hogy $u = pvq$. Ha $pq \neq \varepsilon$, akkor v *valódi részszava* u-nak. Azt mondjuk, hogy p az u ***kezdőszelete*** (prefixuma, prefixe), q pedig ***végszelete*** (szuffixuma, szuffixa). Természetesen, p és q is részszava u-nak.

A Σ^* tetszőleges L részhalmazát a Σ ábécé feletti ***nyelvnek*** nevezzük. Mivel a szavaknak nem tulajdonítunk értelmet, gyakran ***formális nyelvről*** beszélünk, hogy megkülönböztessük a természetes vagy mesterséges nyelvektől, amelyekben a szavakhoz valamilyen értelem is kapcsolódik. Az $L \subseteq \Sigma^*$ lehet véges vagy végtelen nyelv, attól függően, hogy L véges vagy végtelen halmaz. Megjegyezzük, hogy $\emptyset$ üres nyelv, míg a $\{\varepsilon\}$ az üres szóból álló nyelv.

20.1.1. Műveletek nyelvekkel

Nyelvekre értelmezzük a következő műveleteket, ahol L, L_1, L_2 egy-egy Σ feletti nyelvet jelöl:

- *egyesítés*
$$L_1 \cup L_2 = \{u \in \Sigma^* \mid u \in L_1 \text{ vagy } u \in L_2\} ,$$
- *metszet*
$$L_1 \cap L_2 = \{u \in \Sigma^* \mid u \in L_1 \text{ és } u \in L_2\} ,$$
- *különbség*
$$L_1 \setminus L_2 = \{u \in \Sigma^* \mid u \in L_1 \text{ és } u \notin L_2\} ,$$
- *komplementum (komplemens vagy komplementer nyelv)*
$$\overline{L} = \Sigma^* \setminus L \qquad \text{(szokásos jelölése még } \complement L \text{)} ,$$
- *szorzat*
$$L_1 L_2 = \{uv \mid u \in L_1, v \in L_2\} ,$$
- *nyelv hatványa*
$$L^0 = \{\varepsilon\}, \qquad L^n = L^{n-1}L, \text{ ha } n \geq 1 ,$$
- *nyelv iteráltja*
$$L^* = \bigcup_{i=0}^{\infty} L^i = L^0 \cup L \cup L^2 \cup \ldots \cup L^i \cup \ldots ,$$
- *tükrözés*
$$L^{-1} = \{u^{-1} \mid u \in L, \} ,$$

Érvényesek a következők:
$$\left(L^{-1}\right)^{-1} = L, \qquad \left(L^{-1}\right)^n = (L^n)^{-1} .$$
A hatványozás esetében használjuk még az $L^+ = L^* \setminus \{\varepsilon\}$ jelölést is.
Az egyesítés, szorzás, iteráció műveleteket ***reguláris műveleteknek*** nevezzük.

20.1.2. Nyelvek megadása

Nyelveket többféleképpen adhatunk meg:

1) felsoroljuk a szavait,

2) megadunk egy tulajdonságot, amellyel a nyelv minden szava rendelkezik, de más szavak nem,

3) nyelvtan segítségével.

Nyelvek megadása elemeik felsorolásával

Nyelvek például:

$L_1 = \{\varepsilon, 0, 1\}$,

$L_2 = \{a, aa, aaa, ab, ba, aba\}$,

$L_3 = \{\varepsilon, ab, aabb, aaabbb, aaaabbbb, \ldots\}$

Nyelvek megadása tulajdonság segítségével

Nyelvek a következő halmazok is:

$L_4 = \{a^n b^n \mid n = 0, 1, 2, \ldots\}$,

$L_5 = \{uu^{-1} \mid u \in \Sigma^*\}$,

$L_6 = \{u \in \{a, b\}^* \mid n_a(u) = n_b(u)\}$,

ahol $n_a(u)$ az u szóban levő a betűk számát, $n_b(u)$ pedig a b betűk számát jelöli.

Nyelvek megadása nyelvtannal

Értelmezzük a ***generatív nyelvtan*** vagy röviden ***nyelvtan*** (grammatika) fogalmát.

20.1. definíció. ***Generatív nyelvtan****nak nevezzük a* $G = (N, T, P, S)$ *rendezett négyest, ahol*

- N *a* ***változók*** *(****nemterminális jelek****) ábécéje,*
- T *a* ***terminális jelek*** *ábécéje, ahol* $N \cap T = \emptyset$,
- $P \subseteq (N \cup T)^* N (N \cup T)^* \times (N \cup T)^*$ *véges halmaz, vagyis* P *az* (u, v) *alakú* ***helyettesítési szabályok*** *vagy* ***produkciók*** *véges halmaza, ahol* $u, v \in (N \cup T)^*$, *és* u *tartalmaz legalább egy nemterminális jelet.*
- $S \in N$ *a nyelvtan* ***kezdőszimbóluma.***

Megjegyzés. Az (u, v) jelölés helyett gyakran használjuk az $u \to v$ jelölést, amely szemléletesebben utal a helyettesítésre (amint azt később látni fogjuk).

Az $u \to v$, azaz (u, v) szabályban u-t a szabály bal, v-t pedig a jobb oldalának nevezzük. Amennyiben a nyelvtanban több olyan szabály van, amelyeknek a bal oldala azonos, akkor ezeket egyszerűbben is írhatjuk.

$$u \to v_1,\ u \to v_2, \ldots, u \to v_r \quad \text{helyett} \quad u \to v_1 \mid v_2 \mid \ldots \mid v_r$$

Értelmezzük a $(N \cup T)^*$ halmazban a ***közvetlen levezetés*** relációt:

$$u \Longrightarrow v, \qquad \text{ha} \quad u = p_1 p p_2, \quad v = p_1 q p_2 \quad \text{és} \quad (p, q) \in P.$$

Tulajdonképpen u-ban p-t q-val helyettesítjük, és eredményül v-t kapjuk. A közvetlen levezetésre még szokás a $\vdash$ vagy a $\models$ jeleket is használni. Ha hangsúlyozni szeretnénk

a G nyelvtant, amelynek szabályait használjuk, akkor $\Longrightarrow$ helyett $\underset{G}{\Longrightarrow}$ szimbólumot írunk.

A $\Longrightarrow$ reláció reflexív és tranzitív lezártját $\overset{*}{\Longrightarrow}$ jelöli. Ennek a relációnak a neve ***levezetés***.

$u \overset{*}{\Longrightarrow} v$, ha léteznek a $w_0, w_1, \ldots, w_n \in (N \cup T)^*$, $n \geq 0$ szavak, és $u = w_0$, $w_0 \Longrightarrow w_1$, $w_1 \Longrightarrow w_2$, $\ldots, w_{n-1} \Longrightarrow w_n$, $w_n = v$. Ezt röviden így is írhatjuk: $u = w_0 \Longrightarrow w_1 \Longrightarrow w_2 \Longrightarrow \ldots \Longrightarrow w_{n-1} \Longrightarrow w_n = v$. A helyettesítések száma a levezetés hossza, jelen esetben n. Hasonlóképpen értelmezhető az $u \overset{+}{\Longrightarrow} v$ reláció, azzal a különbséggel, hogy itt $n \geq 1$, tehát elvégzünk legalább egy helyettesítést ($u \overset{*}{\Longrightarrow} v$ esetében, ha $n = 0$, akkor $u = v$).

20.2. definíció. *A $G = (N, T, P, S)$ nyelvtan által* ***generált nyelv****:*

$$L(G) = \{u \in T^* \mid S \overset{*}{\Longrightarrow} u\} .$$

Tehát $L(G)$ mindazokat a T ábécé betűiből képzett szavakat tartalmazza, amelyek az S kezdőszimbólumból levezethetők P szabályainak a segítségével.

20.1. példa. Legyen $G = (N, T, P, S)$, ahol
$N = \{S\}$,
$T = \{a, b\}$,
$P = \{S \to ab,\ S \to aSb\}$.
Ekkor, könnyű belátni, hogy $L(G) = \{a^n b^n \mid n \geq 1\}$, mivel

$$S \underset{G}{\Longrightarrow} aSb \underset{G}{\Longrightarrow} a^2Sb^2 \underset{G}{\Longrightarrow} \ldots \underset{G}{\Longrightarrow} a^{n-1}Sb^{n-1} \underset{G}{\Longrightarrow} a^n b^n,$$

ahol az utolsó előtti helyettesítésig mindig a második helyettesítési szabályt ($S \to aSb$) használtuk, míg az utolsónál az $S \to ab$ szabályt. Ezt a levezetést röviden így is írhatjuk: $S \overset{*}{\underset{G}{\Longrightarrow}} a^n b^n$. Tehát $a^n b^n$ tetszőleges n-re levezethető S-ből, és más szót nem lehet levezetni S-ből.

20.3. definíció. *Azt mondjuk, hogy a G_1 és G_2 nyelvtanok* ***ekvivalensek****, és ezt $G_1 \cong G_2$-vel jelöljük, ha $L(G_1) = L(G_2)$.*

20.2. példa. Adott a következő két nyelvtan,
$G_1 = (\{S\}, \{a, b\}, \{S \to aSb,\ S \to \varepsilon\}, S)$ és
$G_2 = (\{S\}, \{a, b\}, \{S \to aSb,\ S \to ab\}, S)$.
Ekkor $L(G_1) \setminus \{\varepsilon\} = L(G_2)$. A két nyelvtan nem ekvivalens.

20.3. példa. A következő két nyelvtan ekvivalens, mivel mindegyik az $\{a^n b^n c^n \mid n \geq 1\}$ nyelvet generálja.
$G_1 = (N_1, T, P_1, S_1)$, ahol
$N_1 = \{S_1, X, Y\}$, $T = \{a, b, c\}$,
$P_1 = \{S_1 \to abc,\ S_1 \to aXbc,\ Xb \to bX,\ Xc \to Ybcc,\ bY \to Yb,$

$aY \to aaX,\ aY \to aa\}$.

$G_2 = (N_2, T, P_2, S_2)$, ahol

$N_2 = \{S_2, A, B, C\}$,

$P_2 = \{S_2 \to aS_2BC,\ S_2 \to aBC,\ CB \to BC,\ aB \to ab,\ bB \to bb,$
$bC \to bc,\ cC \to cc\}$.

Először megmutatjuk teljes indukcióval, hogy $n \geq 2$-re $S_1 \overset{*}{\underset{G_1}{\Longrightarrow}} a^{n-1}Yb^nc^n$. Ha $n = 2$, akkor

$$S_1 \underset{G_1}{\Longrightarrow} aXbc \underset{G_1}{\Longrightarrow} abXc \underset{G_1}{\Longrightarrow} abYbcc \underset{G_1}{\Longrightarrow} aYb^2c^2$$

Feltételezzük, hogy $S_1 \overset{*}{\underset{G_1}{\Longrightarrow}} a^{n-2}Yb^{n-1}c^{n-1}$. Alkalmazzuk az $aY \to aaX$ szabályt, majd $(n-1)$-szer az $Xb \to bX$ szabályt, egyszer az $Xc \to Ybcc$ szabályt, utána pedig szintén $(n-1)$-szer a $bY \to Yb$ szabályt. Tehát

$$S_1 \underset{G_1}{\Longrightarrow} a^{n-2}Yb^{n-1}c^{n-1} \underset{G_1}{\Longrightarrow} a^{n-1}Xb^{n-1}c^{n-1} \overset{*}{\underset{G_1}{\Longrightarrow}} a^{n-1}b^{n-1}Xc^{n-1}$$

$$\underset{G_1}{\Longrightarrow} a^{n-1}b^{n-1}Ybc^n \overset{*}{\underset{G_1}{\Longrightarrow}} a^{n-1}Yb^nc^n$$

Ha most alkalmazzuk az $aY \to aa$ szabályt, azt kapjuk, hogy $S_1 \overset{*}{\underset{G_1}{\Longrightarrow}} a^nb^nc^n$, $n \geq 2$-re, de $S_1 \underset{G_1}{\Longrightarrow} abc$ az $S_1 \to abc$ szabály alapján, tehát $a^nb^nc^n \in L(G_1)$ tetszőleges $n \geq 1$-re. Be kell még bizonyítanunk, hogy a szabályok alkalmazásával más szót nem lehet generálni, csak $a^nb^nc^n$ alakút. Ezt könnyű belátni, hisz eredményes levezetés (amely csak terminális betűket tartalmazó szóban végződik) csak a fenti lehetséges.

Hasonlóképpen $n \geq 2$-re

$$S_2 \underset{G_2}{\Longrightarrow} aS_2BC \overset{*}{\underset{G_2}{\Longrightarrow}} a^{n-1}S_2(BC)^{n-1} \underset{G_2}{\Longrightarrow} a^n(BC)^n \overset{*}{\underset{G_2}{\Longrightarrow}} a^nB^nC^n$$

$$\underset{G_2}{\Longrightarrow} a^nbB^{n-1}C^n \overset{*}{\underset{G_2}{\Longrightarrow}} a^nb^nC^n \underset{G_2}{\Longrightarrow} a^nb^ncC^{n-1} \overset{*}{\underset{G_2}{\Longrightarrow}} a^nb^nc^n$$

Itt sorrendben a következő szabályokat alkalmaztuk: $S_2 \to aS_2BC$ $(n-1$-szer), $S_2 \to aBC$, $CB \to BC$ $(n-1$-szer), $aB \to ab$, $bB \to bb$ $(n-1$-szer), $bC \to bc$, $cC \to cc$ $(n-1$-szer). Ugyanakkor $S_2 \underset{G_2}{\Longrightarrow} aBC \underset{G_2}{\Longrightarrow} abC \underset{G_2}{\Longrightarrow} abc$, tehát $S_2 \overset{*}{\underset{G_2}{\Longrightarrow}} a^nb^nc^n$, $n \geq 1$. Itt is könnyű belátni, hogy más szavakat nem lehet generálni a G_2-ben.

20.4. tétel. *Létezik olyan formális nyelv, amelyet nem lehet nyelvtannal megadni.*

Bizonyítás. A bizonyításhoz kódoljuk a nyelvtanokat. Egy adott $G = (N, T, P, S)$ nyelvtan esetében legyen

$N = \{S_1, S_2, \ldots S_n\},\ T = \{a_1, a_2, \ldots a_m\}$ és $S = S_1$.

A kódolás a következő:

$$S_i \text{ kódja } 10\underbrace{11\ldots11}_{i\text{-szer}}01, \qquad a_i \text{ kódja } 100\underbrace{11\ldots11}_{i\text{-szer}}001$$

A jelek kódját a kódolásban 000 választja el, a nyíl jele 0000, a szabályokat pedig 00000 jelsorozattal választjuk el.

Nyílván, elég csak a szabályokat kódolni. Példaként vegyük a következő nyelvtant:

$G = (\{S\}, \{a, b\}, \{S \to aSb,\ S \to ab\}, S)$.
S kódja 10101, a kódja 1001001, b kódja 10011001. A nyelvtan kódja:

$$\underbrace{10101}0000\underbrace{1001001}000\underbrace{10101}000\underbrace{10011001}00000\underbrace{10101}0000\underbrace{1001001}$$
$$000\underbrace{10011001}$$

Ennek a kódolásnak az alapján a nyelvtanok sorrendbe[1] rendezhetők a kódjuk alapján: $G_1, G_2, \ldots, G_k, \ldots$.

Tekintsük most a Σ ábécé fölötti nyelvek halmazát: $\mathcal{L}_\Sigma = \{L \mid L \subseteq \Sigma^*\}$, azaz $\mathcal{L}_\Sigma = \mathcal{P}(\Sigma^*)$. A Σ^* halmaz megszámlálhatóan végtelen, hisz szavai sorrendbe írhatók. Legyen ez a sorrend $s_0, s_1, s_2, \ldots$, ahol természetesen $s_0 = \varepsilon$. Ekkor minden $L \in \mathcal{L}_\Sigma$ nyelvnek megfeleltethetünk egy $b_0, b_1, b_2, \ldots$ végtelen bináris sorozatot a következőképpen:

$$b_i = \begin{cases} 1, & \text{ha } s_i \in L \\ 0, & \text{ha } s_i \notin L \end{cases} \qquad i = 0, 1, 2, \ldots$$

Könnyű belátni, hogy az így keletkezett bináris sorozatok halmaza megszámlálhatatlanul végtelen, hisz minden sorozat elképzelhető úgy, mint egy 1-nél kisebb pozitív valós szám kettes számrendszerbeli ábrázolása (a tizedesvesszőt az első számjegy elé képzeljük). Fordítva is igaz, hogy minden 1-nél kisebb pozitív szám kettes számrendszerbeli ábrázolása, ha a tizedesvessző utáni részt vesszük, megfelel egy ilyen sorozatnak. Mivel a $[0, 1]$ intervallum kontinuum számosságú, az lesz a végtelen bináris sorozatok halmaza is. Tehát $\mathcal{L}_\Sigma$ is kontinuum számosságú.

Mivel a formális nyelvek halmaza kontinuum számosságú, a nyelvtanok megszámlálhatóan végtelen halmaza és a nyelvek kontinuum számosságú halmaza között nem létezik kölcsönös és egyértelmű megfeleltetés. Tehát, létezik olyan formális nyelv, amelyet nem lehet nyelvtannal megadni. ■

20.1.3. Chomsky-féle nyelvosztályok

Ha a helyettesítési szabályokra bizonyos megkötéseket teszünk, a következő négy nyelvtantípust különböztethetjük meg.

20.5. definíció. *A $G = (N, T, P, S)$ nyelvtan*

*0-típusú (**általános** vagy **mondatszerkezetű**), ha semmilyen megkötést nem teszünk a helyettesítési szabályaira.*

*1-típusú (**környezetfüggő**), ha minden szabálya $uAv \to uwv$ alakú, ahol $A \in N$, $u, v \in (N \cup T)^*$, $w \in (N \cup T)^+$. Ezenkívül megengedhető az $S \to \varepsilon$ szabály is, ha S nem szerepel egyetlen szabály jobb oldalán sem.*

*2-típusú (**környezetfüggetlen**), ha minden szabálya $A \to w$ alakú, ahol $A \in N$,*

[1] Ez a sorrend lehet alaprendezés szerinti, ha a szavakat előbb a hosszúságuk szerint rendezzük, majd azon belül ábécésorrendbe, az ábécé betűi közötti rendezés alapján. De lehet lexikografikus sorrend is, amely azt jelenti, hogy $u < v$ (u előbb van, mint v), ha u valódi kezdőszelete v-nek, vagy azonos hosszúság esetén létezik az $u = xay$ és $v = xby'$ felbontás, ahol x, y, y' részszavak, a és b betűk, és $a < b$.

$w \in (N \cup T)^+$. *Ezenkívül megengedhető az* $S \to \varepsilon$ *szabály is, ha* S *nem szerepel egyetlen szabály jobb oldalán sem.*

*3-típusú (**reguláris**), ha szabályai* $A \to aB$, $A \to a$ *alakúak, ahol* $a \in T$ *és* $A, B \in N$. *Ezenkívül megengedhető az* $S \to \varepsilon$ *szabály is, ha* S *nem szerepel egyetlen szabály jobb oldalán sem.*

Ha G *i-típusú nyelvtan, akkor az* $L(G)$ *nyelvet szintén i-típusúnak (általánosnak, környezetfüggőnek, környezetfüggetlennek, regulárisnak) nevezzük.*

Ez a felosztás Noam Chomsky nevéhez fűződik.

Egy L nyelv i-típusú ($i = 0, 1, 2, 3$), ha létezik egy i-típusú G nyelvtan, amelyik az L nyelvet generálja, vagyis $L = L(G)$.

Jelölje $\mathcal{L}_i$ ($i = 0, 1, 2, 3$) az i-típusú nyelvek osztályát. Ekkor bizonyítható, hogy

$$\mathcal{L}_0 \supset \mathcal{L}_1 \supset \mathcal{L}_2 \supset \mathcal{L}_3.$$

A nyelvtantípusok fenti értelmezése alapján a tartalmazás ($\supseteq$) nyilvánvaló, de a szigorú tartalmazás ($\supset$) bizonyításra szorul.

20.4. példa. Minden nyelvtantípusra adunk egy példát.

Környezetfüggő nyelvtan

$G_1 = (N_1, T_1, P_1, S_1)$, ahol $N_1 = \{S_1, A, B, C\}$, $\quad T_1 = \{a, 0, 1\}$

P_1 elemei:

$$\begin{array}{lcl} S_1 & \to & ACA \\ AC & \to & AACA \mid ABa \mid AaB \\ B & \to & AB \mid A \\ A & \to & 01 \end{array}$$

Az $L(G_1)$ nyelv olyan uav szavakból áll, ahol $u, v \in \{0, 1\}^*$ és $|u| \neq |v|$.

Környezetfüggetlen nyelvtan

$G_2 = (N_2, T_2, P_2, K)$, ahol $N_2 = \{K, T, F\}$, $\quad T_2 = \{+, *, (,), a\}$

P_2 elemei:

$$\begin{array}{lcl} K & \to & K + T \mid T \\ T & \to & T * F \mid F \\ F & \to & (K) \mid a \end{array}$$

Az $L(G_2)$ olyan algebrai kifejezésekből áll, amelyek a T_2 halmaz elemeiből képezhetők.

Reguláris nyelvtan

$G_3 = (N_3, T_3, P_3, S_3)$, ahol $N_3 = \{S_3, A, B\}$, $\quad T_3 = \{a, b\}$

P_3 elemei:

$$\begin{array}{lcll} S_3 & \to & aA & \\ A & \to & aB \mid a & lll \\ B & \to & aB \mid bB \mid a \mid b & \end{array}$$

Az $L(G_3)$ nyelv olyan, a és b betűkből képezhető szavakból áll, amelyek legalább két a-val kezdődnek.

20.1.4. Kiterjesztett nyelvtanok

1-típusú kiterjesztett nyelvtan

Minden szabály $\alpha \to \beta$ alakú, ahol $|\alpha| \leq |\beta|$, kivéve esetleg az $S \to \varepsilon$ szabályt.

2-típusú kiterjesztett nyelvtan

Minden szabály $A \to \beta$ alakú, ahol $A \in N, \beta \in (N \cup T)^*$.

3-típusú kiterjesztett nyelvtan

Minden szabály $A \to uB$ vagy $A \to u$ alakú, ahol $A, B \in N, u \in T^*$.

20.6. tétel. *Tetszőleges kiterjesztett nyelvtanhoz megadható egy vele ekvivalens, ugyanolyan típusú nyelvtan.*

Bizonyítás. Jelöljük G_{ki}-vel a kiterjesztett nyelvtant és G-vel azt a nyelvtant, amelyet minden típusra külön értelmezünk, és amelyről meg akarjuk mutatni, hogy ekvivalens G_{ki}-vel.

Nyilvánvaló, hogy $L(G) \subseteq L(G_{ki})$, hisz az eredeti szabályok a kiterjesztett nyelvtanokban is szerepelhetnek. Minden típusra csak az $L(G_{ki}) \subseteq L(G)$ tartalmazást kell bizonyítani.

1-típus

Azt kell csak megmutatnunk, hogy tetszőleges $\alpha \to \beta$ szabály, ahol $|\alpha| \leq |\beta|$, mindig átírható $\gamma_1\delta\gamma_2 \to \gamma_1\gamma\gamma_2$ alakúra.

Legyen $X_1X_2 \dots X_m \to Y_1Y_2 \dots Y_n$ $(m \leq n)$ egy tetszőleges szabály, amely nem megfelelő alakú. Helyettesítsük ezt a következő szabályokkal, ahol $A_1, A_2, \dots A_m$ új változók:

$$
\begin{array}{lcl}
X_1X_2 \dots X_m & \to & A_1X_2X_3 \dots X_m \\
A_1X_2 \dots X_m & \to & A_1A_2X_3 \dots X_m \\
 & \dots & \\
A_1A_2 \dots A_{m-1}X_m & \to & A_1A_2 \dots A_{m-1}A_m \\
A_1A_2 \dots A_{m-1}A_m & \to & Y_1A_2 \dots A_{m-1}A_m \\
Y_1A_2 \dots A_{m-1}A_m & \to & Y_1Y_2 \dots A_{m-1}A_m \\
 & \dots & \\
Y_1Y_2 \dots Y_{m-2}A_{m-1}A_m & \to & Y_1Y_2 \dots Y_{m-2}Y_{m-1}A_m \\
Y_1Y_2 \dots Y_{m-1}A_m & \to & Y_1Y_2 \dots Y_{m-1}Y_mY_{m+1} \dots Y_n
\end{array}
$$

Könnyű belátni, hogy ezek a szabályok csak ebben a sorrendben alkalmazhatók, ezért $L(G_{ki}) \subseteq L(G)$.

2-típus

Legyen $G_{ki} = (N, T, P, S)$. Ki kell küszöbölnünk az $A \to \varepsilon$ alakú szabályokat. Csak $S \to \varepsilon$ maradhat, ha S nem szerepel szabály jobb oldalán. Ehhez felépítjük a következő halmazokat:

$U_0 = \{A \in N \mid (A \to \varepsilon) \in P\}$

$U_i = U_{i-1} \cup \{A \in N \mid (A \to W) \in P,\ W \in U_{i-1}^+\}$

Mivel N véges halmaz, léteznie kell egy olyan k-nak, amelyre $U_{k-1} = U_k$, és ezt a halmazt nevezzük el U-nak.

Az olyan $A \to \alpha$ alakú szabályok mellett, amelyek esetében α tartalmaz U-beli változókat, vegyük be azokat is, amelyeket úgy képezünk, hogy α-ból elhagyunk egy vagy több U-beli változót, de csak akkor, ha ezáltal a jobb oldal nem lesz ε. (Ha valamelyik $X \in U$ változóból az eredeti nyelvtanban csak az üres szó vezethető le, akkor az eredeti $A \to \alpha$ szabály elhagyható.)

Könnyű belátni, hogy ez a nyelvtan ugyanazt a nyelvet generálja, kivéve a ε szót, amelyet nem tudja generálni. Ha az eredeti nyelv tartalmazza ε-t akkor, ha a kezdőszimbólum nem szerepel egyetlen szabály jobb oldalán sem, az $S \to \varepsilon$ szabály

bevezetésével már az üres szót is generálni fogja. Ha S szerepel valamelyik szabály jobb oldalán, akkor egy új kezdőszimbólum bevezetésével ε is generálható lesz, ha bevesszük a szabályok közé az $S' \to S$ és $S' \to \varepsilon$ szabályokat.

3-típus

Először alkalmazzuk a 2-típus esetében használt eljárást az $A \to \varepsilon$ alakú szabályok kiküszöbölésére, majd küszöböljük ki az átnevezéseket ($A \to B$, $A, B \in N$), ha vannak ilyenek. Ha az átnevezések miatt léteznek $A \overset{+}{\Longrightarrow} B$ levezetések, akkor abban minden helyettesítés átnevezés, és akkor minden $B \to \alpha$ szabály mellé bevesszük az $A \to \alpha$ szabályt is. Végül elhagyjuk az átnevezéseket. Könnyű belátni, hogy az eredetivel ekvivalens nyelvtant kapunk.

Ezután minden $A \to a_1 a_2 \dots a_n B$ szabály helyett, ahol $B \in N \cup \{\varepsilon\}$, bevezetjük a következőket:

$$\begin{array}{lcl} A & \to & a_1 A_1, \\ A_1 & \to & a_2 A_2, \\ & \dots & \\ A_{n-1} & \to & a_n B, \end{array}$$

ahol $A_1, A_2, \dots A_{n-1}$ új változók. Könnyen igazolható, hogy ez a nyelvtan ekvivalens az eredetivel. ■

20.5. példa. Adva van a következő 1-típusú kiterjesztett nyelvtan: $G_{ki} = (N, T, P, S)$, ahol $N = \{S, B, C\}$, $T = \{a, b, c\}$ és P a következő szabályokból áll:

$$\begin{array}{lcllcl} S & \to & aSBC \mid aBC & CB & \to & BC \\ aB & \to & ab & bB & \to & bb \\ bC & \to & bc & cC & \to & cc. \end{array}$$

Az egyetlen szabály amely nem környezetfüggő, az a $CB \to BC$. Ehelyett bevezetjük a következőket, a bizonyításban adott módszer alapján:

$$\begin{array}{lcl} CB & \to & AB \\ AB & \to & AD \\ AD & \to & BD \\ BD & \to & BC \end{array}$$

Így az új nyelvtan, amely most már környezetfüggő: $G = (\{S, A, B, C, D\}, \{a, b, c\}, P', S)$, ahol P' elemei:

$$\begin{array}{lcllcl} S & \to & aSBC \mid aBC & & & \\ CB & \to & AB & aB & \to & ab \\ AB & \to & AD & bB & \to & bb \\ AD & \to & BD & bC & \to & bc \\ BD & \to & BC & cC & \to & cc. \end{array}$$

Igazolni lehet, hogy $L(G_{ki}) = L(G) = \{a^n b^n c^n \mid n \geq 1\}$.

20.6. példa. Legyen $G_{ki} = (\{S, B, C\}, \{a, b, c\}, P, S)$ 2-típusú kiterjesztett nyelvtan, ahol P elemei:

$$\begin{array}{lcl} S & \to & aSc \mid B \\ B & \to & bB \mid C \\ C & \to & Cc \mid \varepsilon. \end{array}$$

Ekkor $U_0 = \{C\}$, $U_1 = \{B, C\}$, $U_3 = \{S, B, C\} = U$. Az új nyelvtan szabályai:

$S \to aSc \mid ac \mid B$
$B \to bB \mid b \mid C$
$C \to Cc \mid c.$

Mivel az eredeti nyelvtan generálja az üres szót is, új kezdőszimbólumot kell bevezetnünk, és még két szabályt: $S' \to S \mid \varepsilon$. Tehát az eredetivel ekvivalens környezetfüggetlen nyelvtan: $G = (\{S', S, B, C\}, \{a, b, c\}, P', S')$ és a szabályok:

$S' \to S \mid \varepsilon$
$S \to aSc \mid ac \mid B$
$B \to bB \mid b \mid C$
$C \to Cc \mid c.$

Mindkét nyelvtan az $\{a^m b^n c^p \mid p \geq m \geq 0, n \geq 0\}$ nyelvet generálja.

20.7. példa. Legyen $G_{ki} = (\{S, A, B\}, \{a, b\}, P, S)$ a vizsgálandó 3-típusú kiterjesztett nyelvtan, ahol P:

$S \to abA$
$A \to bB$
$B \to S \mid \varepsilon.$

Először küszöböljük ki a $B \to \varepsilon$ szabályt. Mivel $U_0 = U = \{B\}$, a szabályok a következők lesznek:

$S \to abA$
$A \to bB \mid b$
$B \to S.$

Ez utóbbi szabályt (amely átnevezés) ki lehet küszöbölni, bevezetve helyette a $B \to abA$ szabályt. Hátravan még az első szabály jobb oldalának a rövidebbé tétele. Erre egy új változót kell bevezetnünk, és az $S \to abA$ helyett az $A \to aC$ és $C \to bA$ szabályokat használjuk. Az új nyelvtan: $G = (\{S, A, B, C\}, \{a, b\}, P', S)$, ahol P':

$S \to aC$
$A \to bB \mid b$
$B \to aC$
$C \to bA.$

Könnyű bizonyítani, hogy $L(G_{ki}) = L(G) = \{(abb)^n \mid n \geq 1\}$.

20.1.5. Zártsági tulajdonságok tetszőleges nyelvosztályra

Láttuk, hogy a reguláris nyelvek osztálya zárt a reguláris műveletekre, azaz az egyesítésre, a szorzatra és az iterációra. Bizonyítható, hogy ez igaz minden $\mathcal{L}_i$ osztályra ($i = 0, 1, 2, 3$).

20.7. tétel. *Az $\mathcal{L}_i$ ($i = 0, 1, 2, 3$) nyelvek osztálya zárt a reguláris műveletekre nézve.*

Bizonyítás. Kiterjesztett nyelvtanok segítségével végezzük a bizonyítást. Legyenek $G_1 = (N_1, T_1, P_1, S_1)$ és $G_2 = (N_2, T_2, P_2, S_2)$ i-típusú kiterjesztett nyelvtanok. Feltételezzük, hogy $N_1 \cap N_2 = \emptyset$.

Egyesítés

Legyen $G_\cup = (N_1 \cup N_2 \cup \{S\}, T_1 \cup T_2, P_1 \cup P_2 \cup \{S \to S_1, S \to S_2\}, S)$.

Könnyű igazolni, hogy $L(G_\cup) = L(G_1) \cup L(G_2)$. Ha $i = 0, 2, 3$, akkor abból, hogy G_1 és G_2 i-típusú, következik, hogy $G_\cup$ is az lesz. Ha $i = 1$, akkor, ha valamelyik nyelvtan generálja az üres szót, akkor a $G_\cup$ szabályaiból kivesszük a megfelelő

(esetleg mindkettő) $S_k \to \varepsilon$ $(k = 1, 2)$ szabályt és helyettesítjük $S \to \varepsilon$ szabállyal.

Szorzat

$G. = (N_1 \cup N_2 \cup \{S\}, T_1 \cup T_2, P_1 \cup P_2 \cup \{S \to S_1S_2\}, S)$.

Könnyű igazolni, hogy $L(G.) = L(G_1)L(G_2)$. Az $i = 0, 2$ típusoknál $G.$ is ugyanolyan típusú lesz. Az $i = 1$ típusnál, akárcsak az egyesítésnél, kivesszük a szabályok közül az $S_1 \to \varepsilon$ szabályt (ha létezik), és helyettesítjük az $S \to S_2$ szabállyal, és fordítva. Ha mindkét nyelvtan generálja az üres szót, akkor mindkét $S_k \to \varepsilon$ szabály kivesszük, és helyettesítjük az $S \to \varepsilon$ szabállyal.

Módosítani kell a szabályokat a reguláris nyelvtanok esetében, mert $S \to S_1S_2$ nem reguláris szabály. Helyette a következő nyelvtant használjuk:

$G. = (N_1 \cup N_2, T_1 \cup T_2, P_1' \cup P_2, S_1)$, ahol P_1' annyiban különbözik P_1-től, hogy az $A \to u, u \in T^*$ szabályok helyett $A \to uS_2$ kerül be a szabályok közé.

Iteráció

$G_* = (N_1 \cup \{S\}, T_1, P_1 \cup \{S \to S_1S,\ S \to \varepsilon\}, S)$.

2-típusú nyelvtanoknál G_* is 2-típusú lesz. A 3-típusnál, a szorzathoz hasonlóan átalakítjuk a szabályokat, $S \to S_1S \mid \varepsilon$ helyett $S_1 \to \varepsilon$, és az $A \to u$ $(u \in T^*)$ helyett $A \to uS_1$ lesz. S helyett pedig S_1 lesz a kezdőszimbólum.

$i = 0, 1$-re a következő szabályokat használjuk:

$$P_1 \cup \{S \to \varepsilon \mid S_1S'\} \cup \{aS' \to aS \mid a \in T_1\},$$

így elkerülhetjük, hogy olyan szabályokat is alkalmazzunk a levezetésben, amelyek nem megengedettek, ugyanis átnyúlnak a szavak határán az iteráció folytán. (Elég csak terminális jelekre megadni a fenti szabályokat, ha feltételezzük, hogy az eredeti nyelvtan szabályai csak $A \to a$ alak esetében tartalmaznak terminális jelet.) ■

Gyakorlatok

20.1-1. Adjunk meg egy nyelvtant, amely az $L = \left\{uu^{-1} \mid u \in \{a, b\}^*\right\}$ nyelvet generálja, és határozzuk meg a típusát.

20.1-2. Adott a $G = (N, T, P, S)$ kiterjesztett környezetfüggetlen nyelvtan, ahol

$$N = \{S\},\ T = \{a, b\},\ P = \{S \to \lambda, S \to aSa, S \to bSb\}.$$

Adjunk meg egy vele ekvivalens környezetfüggetlen nyelvtant.

20.1-3. Adott a $G = (N, T, P, S)$ kiterjesztett környezetfüggetlen nyelvtan, ahol

$N = \{S, A, C, D\},\ T = \{a, b, c, d, e\}$,

$P = \{S \to abCADe,\ C \to cC,\ C \to \lambda,\ D \to dD,\ D \to \lambda,\ A \to \lambda,$
$A \to dDcCA\}$.

Adjunk meg egy vele ekvivalens környezetfüggetlen nyelvtant.

20.1-4. Mutassuk meg, hogy Σ^* és Σ^+ reguláris nyelvek, tetszőleges Σ ábécére.

20.1-5. Adjunk meg egy nyelvtant az $L = \left\{u \in \{0, 1\}^* \mid n_0(u) = n_1(u)\right\}$ nyelv generálására, ahol $n_0(u)$ az u szóban lévő 0-k, $n_1(u)$ pedig az 1-ek számát jelenti.

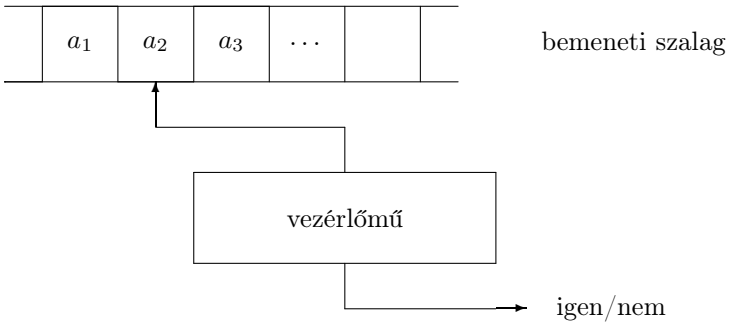
20.1-6. Adjunk meg egy nyelvtant, amely a természetes számokat generálja.

20.1-7. Adjunk meg egy-egy nyelvtant a következő nyelvek generálására.

$L_1 = \{a^n b^m c^p \mid n \geq 1, m \geq 1, p \geq 1\}$,

$L_2 = \{a^{2n} \mid n \geq 1\}$,

$L_3 = \{a^n b^m \mid n \geq 0, m \geq 0\ \}$,



20.1. ábra. Véges automata.

$L_4 = \{a^n b^m \mid n \geq m \geq 1\}$.

20.1-8. Adott a $G = (N, T, P, S)$ kiterjesztett nyelvtan, ahol $N = \{S, A, B, C\}$, $T = \{a\}$, P pedig a következő helyettesítési szabályokat tartalmazza:

$$S \to BAB,\ BA \to BC,\ CA \to AAC,\ CB \to AAB,\ A \to a,\ B \to \lambda.$$

Milyen típusú ez a nyelvtan? Adjunk meg egy vele ekvivalens, ugyanolyan típusú nem kiterjesztett nyelvtant. Milyen nyelvet generál a G nyelvtan?

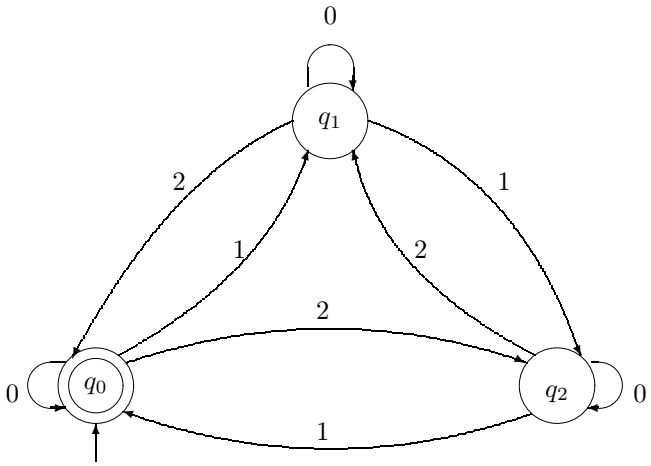
20.1-9. Adjunk példát arra, hogy reguláris nyelv részhalmaza lehet környezetfüggetlen.

20.2. Véges automaták és reguláris nyelvek

A véges automaták olyan matematikai gépek, amelyek rendelkeznek egy bemeneti szalaggal és több állapottal. Az állapotok között vannak kezdőállapotnak, illetve végállapotnak nevezett állapotok. Az automata kezdőállapotból indul, a szalagról sorra olvassa a betűket, miközben állapotot válthat. Érzékeli, ha végigolvasta a szót, és amennyiben az utolsó állapot végállapot, akkor azt mondjuk, hogy felismerte az adott szót (lásd 20.1. ábra). Az ilyen típusú automatákat ***felismerőknek*** nevezzük. Egy ilyen automata által felismert szavak halmazát az automata által felismert nyelvnek nevezzük. Léteznek más típusú véges automaták is, amelyek még egy kimeneti szalaggal is rendelkeznek, amelyre minden bemeneti betű beolvasásakor kiírnak egy betűt. Ebben az esetben ***átalakító*** automatákról van szó. Ezekkel a továbbiakban nem foglalkozunk.

20.8. definíció. ***Véges automatának*** *nevezzük az* $\mathrm{A} = (Q, \Sigma, E, I, F)$ *rendezett ötöst, ahol*

- Q *egy véges, nem üres halmaz, az* ***állapotok*** *halmaza,*
- Σ *a* ***bemeneti ábécé****,*
- E *az* ***átmenetek*** *(vagy* ***élek****) halmaza, ahol* $E \subseteq Q \times \Sigma \times Q$*,*
- $I \subseteq Q$ *a* ***kezdőállapotok*** *halmaza,*
- $F \subseteq Q$ *a* ***végállapotok*** *halmaza.*



20.2. ábra. A 20.8.. példában szereplő véges automata.

A véges automata tulajdonképpen egy olyan irányított gráf, amelynek éleihez egy-egy betűt rendelünk az adott ábécéből, és az állapotokat jelentő csúcsok között bizonyosak kezdő- és bizonyosak végállapotok. A kezdőállapotokat egy-egy befutó nyíl jelzi, míg a végállapotokat két-két koncentrikus kör. Ha két csúcs között több, ugyanolyan irányú él van, akkor ezeket helyettesítjük egyetlen éllel, amelyre a betűket vesszővel elválasztva írjuk.

20.8. példa.
Legyen $\mathrm{A} = (Q, \Sigma, E, I, F)$, ahol
$Q = \{q_0, q_1, q_2\}$,
$\Sigma = \{0, 1, 2\}$,
$E = \{(q_0, 0, q_0), (q_0, 1, q_1), (q_0, 2, q_2),$
$(q_1, 0, q_1), (q_1, 1, q_2), (q_1, 2, q_0),$
$(q_2, 0, q_2), (q_2, 1, q_0), (q_2, 2, q_1)\}$
$I = \{q_0\}$
$F = \{q_0\}$
Az automata a 20.2. ábrán látható.

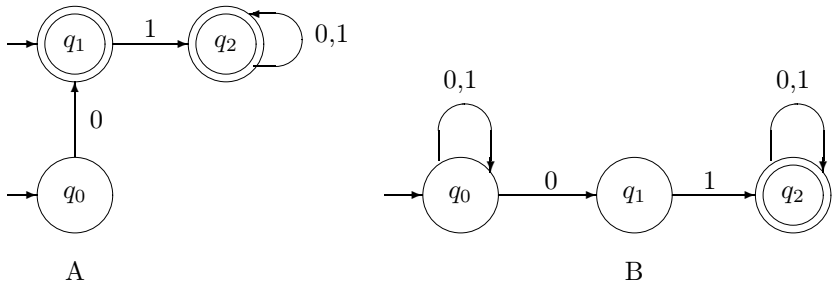
Egy (p, a, q) élnek p a kezdőpontja, q a végpontja, a pedig a címkéje. Értelmezzük a gráfoknál használatos ***séta*** fogalmát. A

$$(q_0, a_1, q_1), (q_1, a_2, q_2), \ldots, (q_{n-2}, a_{n-1}, q_{n-1}), (q_{n-1}, a_n, q_n)$$

élsorozat az automata egy sétája, amelynek címkéje az $a_1 a_2 \ldots a_n$ szó. A séta jelölése

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{k-1}} q_{k-1} \xrightarrow{a_k} q_n,$$

vagy ha $w = a_1 a_2 \ldots a_n$, akkor röviden: $q_0 \xrightarrow{w} q_n$. Itt q_0 a séta kezdőpontja, q_n pedig a végpontja. A séta elemei, természetesen, általában nem mind különbözőek.



20.3. ábra. Nemdeterminisztikus véges automaták.

Egy sétát ***produktívnak*** nevezünk, ha kezdőpontja kezdőállapot, végpontja pedig végállapot. Azt mondjuk, hogy az automata ***felismer*** egy szót, ha az a szó egy produktív séta címkéje. Egy automata által felismert szavak halmazát az automata által felismert nyelvnek mondjuk. Az A ***véges automata által felismert nyelv***

$$L(\mathrm{A}) = \{w \in \Sigma^* \mid \exists p \in I,\ \exists q \in F,\ \exists p \xrightarrow{w} q\}$$

Az A_1 és A_2 automaták ***ekvivalensek***, ha $L(\mathrm{A}_1) = L(\mathrm{A}_2)$.

Sokszor hasznos definiálni egy átmenetfüggvényt:

$$\delta : Q \times \Sigma \to \mathcal{P}(Q), \quad \delta(p,a) = \{q \in Q \mid (p,a,q) \in E\}.$$

Ez a függvény egy p állapotnak és egy a betűnek megfelelteti azt az állapothalmazt, amelynek állapotaiba átmehet az automata, ha a p állapotban van és az olvasófej az a betűre mutat. Jelöljük $|A|$-val az A halmaz elemeinek a számát.[2] Egy véges automatáról azt mondjuk, hogy ***determinisztikus***, ha

$$|I| = 1,\ \text{és}\ |\delta(q,a)| \le 1,\ \forall q \in Q,\ \forall a \in \Sigma,$$

és ***nemdeterminisztikus*** különben. A 20.2. ábrán látható automata determinisztikus.

A $|\delta(q,a)| \le 1$ feltételt helyettesíthetjük a következővel:

$$(p,a,q) \in E,\ (p,a,r) \in E \implies q = r.$$

Az átmenetfüggvény segítségével könnyen elkészíthetjük az automata átmenettáblázatát, amely például a a 20.2. ábra esetében a következő:

δ	0	1	2
q_0	$\{q_0\}$	$\{q_1\}$	$\{q_2\}$
q_1	$\{q_1\}$	$\{q_2\}$	$\{q_0\}$
q_2	$\{q_2\}$	$\{q_0\}$	$\{q_1\}$

[2]Nem értelemzavaró az, hogy ugyanazt a jelölést használjuk a halmaz számosságára, mint a szó hosszára, hisz a szót mindig kisbetűvel jelöljük, a halmazt pedig nagybetűvel. Kivétel csak a $\delta(q,a)$ jelölés, amely nem téveszthető össze szóval.

δ	0	1
q_0	$\{q_1\}$	$\emptyset$
q_1	$\emptyset$	$\{q_2\}$
q_2	$\{q_2\}$	$\{q_2\}$

A

δ	0	1
q_0	$\{q_0, q_1\}$	$\{q_0\}$
q_1	$\emptyset$	$\{q_2\}$
q_2	$\{q_2\}$	$\{q_2\}$

B

20.4. ábra. A 20.3. ábrán látható két automata átmenettáblázata.

A 20.3. ábrán látható két automata mindegyike nemdeterminisztikus, az első (az A automata) azért mert két kezdőállapota van, a második (a B automata) pedig azért mert a q állapotból 0-val a q_0 és q_1 állapotokba is el lehet jutni. A két automata átmenettáblázata a 20.4. ábrán látható. $L(A)$ azon szavak halmaza $\Sigma = \{0, 1\}$ felett, amelyek nem kezdődnek két 0-val (természetesen az ε is eleme a nyelvnek), $L(B)$ pedig azon szavaké, amelyekben van 01 részszó.

Determinisztikus véges automaták ekvivalenciájának vizsgálata
Determinisztikus véges automaták ekvivalenciáját vizsgáljuk az azonos címkéjű, kezdőállapottal kezdődő séták segítségével a két automatában. Ha egyik séta végállapottal végződik, a másik pedig nem, akkor a két automata nyilvánvalóan nem lehet ekvivalens.

Adott két determinisztikus véges automata ugyanazon ábécé fölött: $A = (Q, \Sigma, E, I, F)$ és $A' = (Q', \Sigma, E', I', F')$. Készítünk egy táblázatot, amelynek első oszlopában (q, q') állapotpárok szerepelnek, ahol $q \in Q$ és $q' \in Q'$. Ezután a táblázatban az ábécé minden betűjének megfeleltetünk egy oszlopot. A táblázat (q, q') sorának és az a betű oszlopának a találkozásánál a $(\delta(q, a), \delta'(q', a))$ állapotpár szerepel.

	$\dots$	a	$\dots$
$\dots$		$\dots$	
(q, q')		$(\delta(q, a), \delta'(q', a))$	
$\dots$		$\dots$	

Mivel az automaták determinisztikusak, a $(\delta(q, a), \delta'(q', a))$ párok esetében elhagyjuk a halmazt jelölő zárójeleket, tehát például $(\{p\}, \{q\})$ helyett (p, q) párt írunk. Amennyiben van olyan q állapot és a bemeneti jel, hogy $\delta(q, a) = \emptyset$, akkor egy új s állapotot vezetünk be a következőképpen: $\delta(q, a) = \{s\}$ és $\delta(s, a) = \{s\}$ bármely $a \in \Sigma$ betűre. Ezzel elérjük, minden állapotpárban pontosan két állapot legyen.

A táblázat első sorának első oszlopába a (q_0, q_0') állapotpár kerül. Ha az első sor valamelyik oszlopában megjelenik egy olyan pár, amelyre egyik állapot végállapot, a másik meg nem, akkor az algoritmust befejezzük: ***a két automata nem ekvivalens***. Amennyiben nincs ilyen pár, minden új párt beírunk az első oszlopba, és folytatjuk az algoritmust. Ha már nem jelenik meg új állapotpár, és minden párra igaz, hogy mindkét eleme végállapot vagy egyik sem az, akkor az algoritmus szintén befejeződik, és ***a két automata ekvivalens***.

AUTOMATA-EKVIVALENCIA(A, A′)

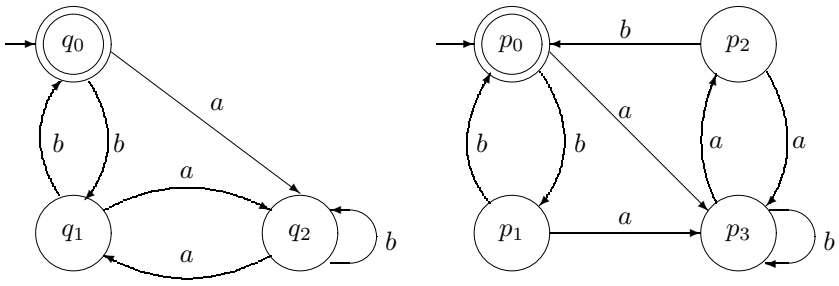
```
1  a táblázat első sorának első oszlopába beírjuk a (q_0, q'_0) állapotpárt.
2  i ← 0
3  repeat
4     i ← i + 1
5     legyen (q, q') a táblázat i-edik sorának első oszlopában levő állapotpár.
6     for minden a ∈ Σ betűre
8        do a táblázat i-edik sora a jelzésű oszlopába beírjuk a
              (δ(q, a), δ'(q', a)) állapotpárt.
9           if (δ(q, a), δ'(q', a)) egyik állapota végállapot, a másik nem
10             then return NEM
11             else (δ(q, a), δ'(q', a)) párt beírjuk az első oszlop következő
                    üres sorába, ha még nem szerepel az első oszlopban.
12 until (i + 1)-edik sor első eleme üres
13 return IGEN
```

Ha n jelöli az A állapotainak számát, n' az A′ állapotainak számát és m a bemeneti ábécé betűinek számát, akkor, figyelembe véve, hogy a ***repeat*** ciklust legrosszabb esetben nn'-szer kell elvégezni, a ***for*** ciklust pedig m-szer, könnyen kiszámítható, hogy a maximális lépésszám legrosszabb esetben $O(nn'm)$, vagy ha $n = n'$, akkor $O(n^2m)$.

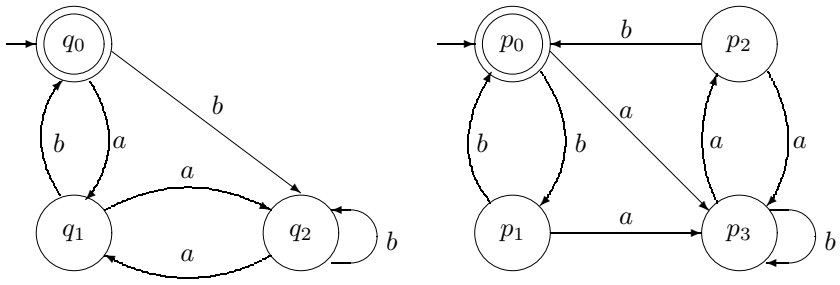
20.9. példa. Vizsgáljuk meg, hogy a 20.5. ábrán látható két automata ekvivalens-e. Elkészítjük az állapotpárok következő táblázatát.

	a	b
(q_0, p_0)	(q_2, p_3)	(q_1, p_1)
(q_2, p_3)	(q_1, p_2)	(q_2, p_3)
(q_1, p_1)	(q_2, p_3)	(q_0, p_0)
(q_1, p_2)	(q_2, p_3)	(q_0, p_0)

A két automata ekvivalens, mivel minden lehetséges állapotpárt figyelembe vettünk, és minden pár mindkét eleme végállapot vagy egyik sem az.



20.5. ábra. Nem ekvivalens véges automaták (20.9. példa).



20.6. ábra. Ekvivalens véges automaták (20.10. példa).

20.10. példa. Tekintsük a 20.6. ábrán lévő két automatát. Az állapotpárok táblázata a következőképpen néz ki:

	a	b
(q_0, p_0)	(q_1, p_3)	(q_2, p_1)
(q_1, p_3)	(q_2, p_2)	(q_0, p_3)
(q_2, p_1)		
(q_2, p_2)		

A két automata nem ekvivalens, mivel a második sor utolsó oszlopában a (q_0, p_3) állapotpár első eleme végállapot, a második pedig nem az.

Elérhetetlen állapotok kizárása

A következő algoritmus meghatározza egy véges automatában az elérhető állapotokat az U_0, U_1, $U_2, \ldots$ halmazok felépítésével, ahol U_0 a kezdőállapotok halmaza, és tetszőleges $i \geq 1$ természetes számra U_i azon állapotok halmaza, amelyekbe el lehet jutni közvetlenül az U_{i-1} halmaz állapotaiból.

ELÉRHETŐ-ÁLLAPOTOK(A, U)

```
1  U_0 ← I
2  i ← 0
3  repeat
4     i ← i + 1
5     U_i ← U_{i−1}
6     for minden q ∈ Q
7        do for minden a ∈ Σ
8              do U_i ← U_i ∪ δ(q, a)
9  until U_i = U_{i−1}
10 U ← U_i
11 return U
```

A $Q \setminus U$ halmaz elemei nem elérhető állapotok, ezért kizárhatók az automatából anélkül, hogy befolyásolnák annak a működését.

Ha az állapotok száma n és a betűké pedig m, akkor a fenti algoritmus lépésszáma $O(n^2m)$, mivel a két egymásba ágyazott ciklus lépésszáma nm, a **repeat** ciklust pedig legrosszabb esetben n-szer hajtjuk végre (ha minden alkalommal csupán egy új állapot kerül be az U_i halmazokba).

Nemproduktív állapotok kizárása
A produktív állapotok meghatározására szolgáló következő algoritmus használja a δ^{-1} függvényt, amelynek definíciója a következő:

$$\delta^{-1} : Q \times \Sigma \to \mathcal{P}(Q), \quad \delta^{-1}(p,a) = \{q \mid (q,a,p) \in E\}.$$

Ez a függvény megadja azt az állapothalmazt, amelynek elemeiből el lehet jutni a p állapotba, ha az olvasófej az a betűn található.

PRODUKTÍV-ÁLLAPOTOK(A, V)

```
1  V_0 ← F
2  i ← 0
3  repeat
4     i ← i + 1
5     V_i ← V_{i-1}
6     for minden q ∈ Q
7        do for minden a ∈ Σ
8              do V_i ← V_i ∪ δ^{-1}(q, a)
9  until V_i = V_{i-1}
10 V ← V_i
11 return V
```

A $Q \setminus V$ halmaz elemei nem produktív állapotok, ezért kizárhatók az automatából, anélkül, hogy befolyásolnák annak a működését.

Ha n az állapotok száma és m a betűk száma, akkor a lépésszám ebben az esetben is $O(n^2m)$, akárcsak a ELÉRHETŐ-ÁLLAPOTOK algoritmus esetében.

Nemdeterminisztikus automata átalakítása determinisztikus automatává
A következőkben megmutatjuk, hogy tetszőleges nemdeterminisztikus automata mindig átalakítható olyan determinisztikus automatává, amelyik ekvivalens az eredetivel.

20.9. tétel. *Tetszőleges nemdeterminisztikus véges automatához mindig megadható egy vele ekvivalens determinisztikus véges automata.*

Bizonyítás. Legyen $\mathrm{A} = (Q, \Sigma, E, I, F)$ egy nemdeterminisztikus véges automata. Értelmezzük a következő determinisztikus automatát:
$\overline{\mathrm{A}} = (\overline{Q}, \Sigma, \overline{E}, \overline{I}, \overline{F})$, ahol

$\overline{Q} = \mathcal{P}(Q)$,

$\overline{E}$ élei: $(S, a, R) = \big\{(p, a, q) \mid p \in S, q \in R\big\}$ az összes lehetséges $S, R \subseteq Q$ nem üres részhalmazra,

$\overline{I} = \{I\}$

$\overline{F} = \{S \subseteq Q \mid S \cap F \neq \emptyset\}$

Be kell bizonyítanunk, hogy $L(\mathrm{A}) = L(\overline{\mathrm{A}})$.

a) Bebizonyítjuk, hogy $L(\mathrm{A}) \subseteq L(\overline{\mathrm{A}})$. Legyen $w = a_1 a_2 \dots a_k \in L(\mathrm{A})$. Ekkor létezik a

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{k-1}} q_{k-1} \xrightarrow{a_k} q_k, \quad q_0 \in I,\ q_k \in F$$

séta. Képezzük a következő halmazokat, felhasználna az $\overline{\mathrm{A}}$ automata $\overline{\delta}$ átmenetfüggvényét: $S_0 = \{q_0\}$, $\overline{\delta}(S_0, a_1) = S_1$, $\dots \overline{\delta}(S_{k-1}, a_k) = S_k$. Nyilvánvaló, hogy $q_1 \in S_1, \dots, q_k \in S_k$, és mivel $q_k \in F$, következik, hogy $S_k \cap F \neq \emptyset$, tehát $S_k \in \overline{F}$. Így $w \in L(\overline{\mathrm{A}})$, mivel létezik az

$$S_0 \xrightarrow{a_1} S_1 \xrightarrow{a_2} S_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{k-1}} S_{k-1} \xrightarrow{a_k} S_k, \quad S_0 \in I,\ S_k \in F$$

séta. Tehát $L(\mathrm{A}) \subseteq L(\overline{\mathrm{A}})$.

b) Bebizonyítjuk, hogy $L(\overline{\mathrm{A}}) \subseteq L(\mathrm{A})$. Legyen $w = a_1 a_2 \dots a_k \in L(\overline{\mathrm{A}})$. Ekkor létezik a

$$\overline{q_0} \xrightarrow{a_1} \overline{q_1} \xrightarrow{a_2} \overline{q_2} \xrightarrow{a_3} \cdots \xrightarrow{a_{k-1}} \overline{q}_{k-1} \xrightarrow{a_k} \overline{q}_k, \quad \overline{q}_0 \in \overline{I},\ \overline{q}_k \in \overline{F}$$

séta.

Ekkor az $\overline{F}$ definíciója alapján $\overline{q}_k \cap F \neq \emptyset$, azaz létezik $q_k \in \overline{q}_k \cap F$, tehát $q_k \in F$ és $\overline{q}_k$ definíciója alapján létezik q_{k-1} úgy, hogy $(q_{k-1}, a_k, q_k) \in E$. Hasonlóképpen, léteznek a $q_{k-2}, \dots, q_1, q_0$ állapotok úgy, hogy $(q_{k-2}, a_k, q_{k-1}) \in E, \dots, (q_0, a_1, q_1) \in E$ ahol $q_0 \in \overline{q}_0 = I$, ezért létezik a

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{k-1}} q_{k-1} \xrightarrow{a_k} q_k, \quad q_0 \in I,\ q_k \in F$$

séta, tehát $L(\overline{\mathrm{A}}) \subseteq L(\mathrm{A})$. ■

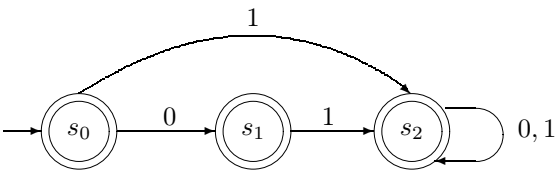
20.11. példa. Alkalmazzuk a 20.9. tételt a 20.3. ábra A automatájára. Az új automata átmenettáblázata a következő:

$\overline{\delta}$	0	1
$\{q_0\}$	$\{q_1\}$	$\emptyset$
$\{q_1\}$	$\emptyset$	$\{q_2\}$
$\{q_2\}$	$\{q_2\}$	$\{q_2\}$
$\{q_0, q_1\}$	$\{q_1\}$	$\{q_2\}$
$\{q_0, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$
$\{q_1, q_2\}$	$\{q_2\}$	$\{q_2\}$
$\{q_0, q_1, q_2\}$	$\{q_1, q_2\}$	$\{q_2\}$

Az így kapott automatában sok elérhetetlen állapot van. Mielőtt alkalmaznánk algoritmusunkat az elérhetetlen állapotok kizárására, vezessük be következő jelöléseket, hogy leegyszerűsítsük az állapotok írását:

$s_0 := \{q_0, q_1\}$,
$s_1 := \{q_0\}$, $s_2 := \{q_1\}$, $s_3 := \{q_2\}$,
$s_4 := \{q_0, q_2\}$, $s_5 := \{q_1, q_2\}$, $s_6 := \{q_0, q_1, q_2\}$,

amelyek közül s_0 a kezdőállapot. Ekkor az átmenettáblázat így is írható:



20.7. ábra. A 20.3. ábra A automatájával ekvivalens determinisztikus véges automata.

$\overline{\delta}$	0	1
s_0	$\{s_2\}$	$\{s_3\}$
s_1	$\{s_2\}$	$\emptyset$
s_2	$\emptyset$	$\{s_3\}$
s_3	$\{s_3\}$	$\{s_3\}$
s_4	$\{s_5\}$	$\{s_3\}$
s_5	$\{s_3\}$	$\{s_3\}$
s_6	$\{s_5\}$	$\{s_3\}$

Az Elérhető-állapotok algoritmus szerint az automata elérhető állapotai következők szerint határozható meg.

$U_0 = \{s_0\}$,
$U_1 = \{s_0, s_2, s_3\}$,
$U_2 = \{s_0, s_2, s_3\} = U_1 = U$

Az s_0 kezdőállapot és egyben végállapot is. Az s_2 és s_3 mindegyike végállapot. Az s_1, s_4, s_5, s_6 elérhetetlen állapotok, ezért kizárjuk őket az automatából, ekkor a kapott automata átmenettáblázata a következő lesz.

$\overline{\delta}$	0	1
s_0	$\{s_2\}$	$\{s_3\}$
s_2	$\emptyset$	$\{s_3\}$
s_3	$\{s_3\}$	$\{s_3\}$

Az ennek megfelelő determinisztikus automata átmenetgráfja a 20.7 ábrán látható.

A 20.9. tétel által nyújtott algoritmus egyszerűsíthető. Nem kell az állapothalmaz minden részhalmazát figyelembe venni. Az $\overline{\mathrm{A}}$ automata állapotait fokozatosan kapjuk meg úgy, hogy elindulunk a $\overline{q}_0 = I$ állapottal, meghatározzuk a $\overline{\delta}(q_0, a)$ állapotokat, minden $a \in \Sigma$ elemre. Az újonnan kapott állapotokra szintén meghatározzuk az átmenetek alapján az állapotokat. Ezt addig folytatjuk, amíg már nem kapunk új állapotokat.

Lássuk ezt az előző példánkon!

$\overline{q}_0 := \{q_0, q_1\}$, innen
$\overline{\delta}(\overline{q}_0, 0) = \{q_1\}$, és legyen $\overline{q}_1 := \{q_1\}$,
$\overline{\delta}(\overline{q}_0, 1) = \{q_2\}$, és legyen $\overline{q}_2 := \{q_2\}$.
Ezután $\overline{\delta}(\overline{q}_1, 0) = \emptyset$, $\overline{\delta}(\overline{q}_1, 1) = \{q_2\}$.
Majd $\overline{\delta}(\overline{q}_2, 0) = \{q_2\} = \overline{q}_2$, $\overline{\delta}(\overline{q}_2, 1) = \{q_2\} = \overline{q}_2$.
Azaz, az átmenettáblázat a következő:

$\overline{\delta}$	0	1
$\overline{q}_0$	$\overline{q}_1$	$\overline{q}_2$
$\overline{q}_1$	$\emptyset$	$\overline{q}_2$
$\overline{q}_2$	$\overline{q}_2$	$\overline{q}_2$

amely azonos az előbb kapott automata átmenettáblázatával.

A következő algoritmus egy nemdeterminisztikus véges automatához hozzárendeli a vele ekvivalens determinisztikus véges automata átmenettáblázatát, de nem tartalmazza annak megállapítását, hogy egy állapot végállapot-e vagy sem, de ez könnyűszerrel beépíthető. Az algoritmusban a BENNEVAN$(\overline{q}, \overline{Q})$ értéke igaz, ha a $\overline{q}$ állapot már szerepel a $\overline{Q}$ halmazban, és hamis ellenkező esetben.

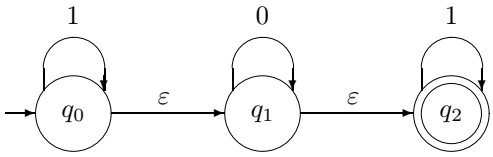
NEMDET-DET$(\mathrm{A}, \overline{\mathrm{A}})$

```
1  q̄_0 ← I
2  Q̄ ← {q̄_0}
3  i ← 0                                          ▷ i a sorokat számolja
4  k ← 0                                          ▷ k az állapotokat számolja
5  repeat
6     j ← 0                                       ▷ j az oszlopokat számolja
7     for minden a ∈ Σ
8         do j ← j + 1
9            q̄ ← ⋃_{p∈q̄_i} δ(p, a)
10           if q̄ ≠ ∅
11              then if BENNEVAN(q̄, Q̄)
12                      then A[i, j] ← {q̄}
13                      else k ← k + 1
14                           q̄_k ← q̄
15                           A[i, j] ← {q̄_k}
16                           Q̄ ← Q̄ ∪ {q̄_k}
17              else A[i, j] ← ∅
18    i ← i + 1
19 until i = k + 1
20 return Ā átmenettáblázata
```

Mivel a **repeat** ciklust annyiszor végezzük el, ahány állapota van az új automatának, legrosszabb esetben ez exponenciális érték is lehet, hisz ha az eredeti automata állapotainak száma n, akkor az eredményautomatának lehet akár $2^n - 1$ állapota is. (2^n egy n elemű halmaz részhalmazainak a száma, beleértve az üres halmazt is.)

20.2.1. ε-lépéses véges automaták

A ε-lépéses véges automata annyiban különbözik a véges automatától, hogy megengedjük azt, hogy az automata üres lépést is végezzen, azaz átmenjen egyik állapotból a másikba, anélkül, hogy valamilyen bemeneti jelet olvasna. A ε-lépéses véges automata átmeneteinek a halmaza $E \subseteq Q \times \big(\Sigma \cup \{\varepsilon\}\big) \times Q$



20.8. ábra. ε-lépéses véges automata.

A nemdeterminisztikus ε-lépéses véges automata átmenetfüggvénye a következő:

$$\delta : Q \times \big(\Sigma \cup \{\varepsilon\}\big) \to \mathcal{P}(Q), \quad \delta(p,a) = \{q \in Q \mid (p,a,q) \in E\}.$$

A 20.8. ábrán látható ε-lépéses automata az uvw alakú szavakat ismeri fel, ahol $u \in \{1\}^*$, $v \in \{0\}^*$ és $w \in \{1\}^*$.

20.10. tétel. *Tetszőleges ε-lépéses véges automatához mindig hozzárendelhető egy vele ekvivalens nemdeterminisztikus véges automata, amely ε-lépés nélküli.*

Bizonyítás helyett megadunk egy algoritmust, amely elvégzi a megfelelő átalakítást. Legyen adott az $\mathrm{A} = (Q, \Sigma, E, I, F)$ ε-lépéses véges automata. Értelmezzük az $\overline{\mathrm{A}} = (Q, \Sigma, \overline{E}, I, \overline{F})$ véges automatát, amely ε-lépés nélküli, és amely A-val ekvivalens. Algoritmusunk az $\overline{F}$ és az $\overline{E}$ halmazokat határozza meg.

Egy q állapotra jelöljük $\Lambda(q)$-val azon állapotok halmazát, amelyekbe el lehet jutni q-ból csupa ε-lépéssel (beleértve magát q-t is). Terjesszük ki ezt definíciót halmazokra is, azaz legyen

$$\Lambda(S) = \bigcup_{q \in S} \Lambda(q), \quad \forall S \subseteq Q.$$

Epszilon-mentesítés$(\mathrm{A}, \overline{\mathrm{A}})$

1 **if** $\Lambda(I) \cap F \neq \emptyset$
2 **then** $\overline{F} \leftarrow F \cup I$
3 **else** $\overline{F} \leftarrow F$
4 **for** minden $q \in Q$
5 **do for** minden $a \in \Sigma$
6 **do** $\Delta(q,a) \leftarrow \bigcup_{p \in \Lambda(q)} \delta(p,a)$
7 $\overline{\delta}(q,a) \leftarrow \Delta(q,a) \cup \left(\bigcup_{p \in \Delta(q,a)} \Lambda(p) \right)$
8 $\overline{E} \leftarrow \big\{(p,a,q), \mid p,q \in Q,\ a \in \Sigma, q \in \overline{\delta}(p,a)\big\}$

A fenti algoritmus az átmenetek meghatározását a $\overline{\delta}$ átmenetfüggvény segítségével végzi, amelyet az 7. sorban értelmeztük.

Ha $|Q| = n$ és $|\Sigma| = m$, a 4–7. sorokból látszik, hogy az algoritmus lépésszáma legrosszabb esetben $O(n^2 m)$.

20.12. példa. Tekintsük a 20.8. ábrán lévő automatát, amelynek átmenettáblázata a következő:

δ	0	1	ε
q_0	$\emptyset$	$\{q_0\}$	$\{q_1\}$
q_1	$\{q_1\}$	$\emptyset$	$\{q_2\}$
q_2	$\emptyset$	$\{q_2\}$	$\emptyset$

Alkalmazzuk az Epszilon-mentesítés algoritmust.
$\Lambda(q_0) = \{q_0, q_1, q_2\}$
$\Lambda(q_1) = \{q_1, q_2\}$
$\Lambda(q_2) = \{q_2\}$
$\Lambda(I) = \Lambda(q_0)$, és ennek metszete az F-fel nem üres, ezért $\overline{F} = F \cup \{q_0\} = \{q_0, q_2\}$.
$\Delta(q_0, 0) = \delta(q_0, 0) \cup \delta(q_1, 0) \cup \delta(q_2, 0) = \{q_1\}, \quad \{q_1\} \cup \Lambda(q_1) = \{q_1, q_2\}$
$\overline{\delta}(q_0, 0) = \{q_1, q_2\}$.

$\Delta(q_0, 1) = \delta(q_0, 1) \cup \delta(q_1, 1) \cup \delta(q_2, 1) = \{q_0, q_2\}, \quad \{q_0, q_2\} \cup (\Lambda(q_0) \cup \Lambda(q_2)) = \{q_0, q_1, q_2\}$
$\overline{\delta}(q_0, 1) = \{q_0, q_1, q_2\}$

$\Delta(q_1, 0) = \delta(q_1, 0) \cup \delta(q_2, 0) = \{q_1\}, \quad \{q_1\} \cup \Lambda(q_1) = \{q_1, q_2\}$
$\overline{\delta}(q_1, 0) = \{q_1, q_2\}$

$\Delta(q_1, 1) = \delta(q_1, 1) \cup \delta(q_2, 1) = \{q_2\}, \quad \{q_2\} \cup \Lambda(q_2) = \{q_2\}$
$\overline{\delta}(q_1, 1) = \{q_2\}$

$\Delta(q_2, 0) = \delta(q_2, 0) = \emptyset$
$\overline{\delta}(q_2, 0) = \emptyset$

$\Delta(q_2, 1) = \delta(q_2, 1) = \{q_2\}, \quad \{q_2\} \cup \Lambda(q_2) = \{q_2\}$
$\overline{\delta}(q_2, 1) = \{q_2\}$

Tehát a $\overline{\mathrm{A}}$ automata átmenettáblázata a következő:

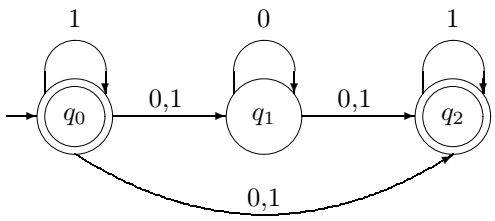
$\overline{\delta}$	0	1
q_0	$\{q_1, q_2\}$	$\{q_0, q_1, q_2\}$
q_1	$\{q_1, q_2\}$	$\{q_2\}$
q_2	$\emptyset$	$\{q_2\}$

az átmenetdiagram pedig a 20.9. ábrán látható.

Láttuk, hogy tetszőleges nemdeterminisztikus véges automatához mindig hozzárendelhető egy vele ekvivalens determinisztikus véges automata. Ezért a nemdeterminisztikus véges automaták ugyanazt a nyelvosztályt ismerik fel, mint a determinisztikus véges automaták. A következők két tétel azt mutatja, hogy ez a nyelvosztály nem más, mint a reguláris nyelvek halmaza.

20.11. tétel. *Ha L egy tetszőleges determinisztikus véges automata által felismert nyelv, akkor létezik olyan reguláris nyelvtan, amelyik az L nyelvet generálja.*

Bizonyítás. Legyen $\mathrm{A} = (Q, \Sigma, E, \{q_0\}, F)$ az L nyelvet felismerő determinisztikus véges automata, azaz $L = L(\mathrm{A})$. Értelmezzük a $G = (Q, \Sigma, P, q_0)$ reguláris nyelvtant a következő szabályokkal:



20.9. ábra. A 20.8. ábrán lévő ε-lépéses véges automatával ekvivalens ε-lépésmentes véges automata.

• Ha $(p, a, q) \in E$ valamilyen $q, p \in Q$ és $a \in \Sigma$, akkor tegyük be a szabályok közé a $p \to aq$ szabályt.

• Ha $(p, a, q) \in E$ és $p \in F$, akkor a fenti szabály mellé még tegyük be a szabályok közé a $p \to a$ szabályt is.

Bebizonyítjuk, hogy $L(G) = L(\mathrm{A}) \setminus \{\varepsilon\}$.

Legyen $u = a_1 a_2 \dots a_n \in L(\mathrm{A})$ és $u \neq \varepsilon$. Ekkor, mivel az A automata felismeri az u szót, létezik a

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{n-1}} q_{n-1} \xrightarrow{a_n} q_n, \quad q_n \in F$$

séta. Ekkor a G-ben léteznek a következő szabályok:

$$q_0 \to a_1 q_1, \quad q_1 \to a_2 q_2, \quad \dots, \quad q_{n-2} \to a_{n-1} q_{n-1}, \quad q_{n-1} \to a_n$$

(az utóbbi szabálynál hiányzik q_n mivel $q_n \in F$), tehát létezik a

$$q_0 \Longrightarrow a_1 q_1 \Longrightarrow a_1 a_2 q_2 \Longrightarrow \dots \Longrightarrow a_1 a_2 \dots a_{n-1} q_{n-1} \Longrightarrow a_1 a_2 \dots a_n$$

levezetés. Ezért $u \in L(G)$.

Fordítva, legyen $u = a_1 a_2 \dots a_n \in L(G)$, és $u \neq \varepsilon$. Ekkor létezik a

$$q_0 \Longrightarrow a_1 q_1 \Longrightarrow a_1 a_2 q_2 \Longrightarrow \dots \Longrightarrow a_1 a_2 \dots a_{n-1} q_{n-1} \Longrightarrow a_1 a_2 \dots a_n$$

levezetés, amelyben a

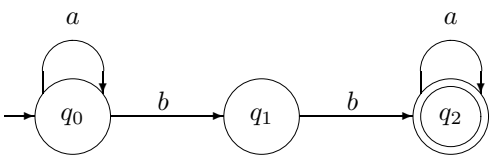
$$q_0 \to a_1 q_1, \quad q_1 \to a_2 q_2, \quad \dots, \quad q_{n-2} \to a_{n-1} q_{n-1}, \quad q_{n-1} \to a_n$$

szabályokat használtuk, amelyek, értelmezés szerint azt jelentik, hogy az automatában létezik a következő séta:

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{n-1}} q_{n-1} \xrightarrow{a_n} q_n,$$

és mivel q_n végállapot, következik, hogy $u \in L(\mathrm{A}) \setminus \{\varepsilon\}$.

Ha az automata ε-t is felismeri, akkor a fenti nyelvtan annyiban módosul, hogy bevezetünk egy új q_0' kezdőszimbólumot q_0 helyett, és minden $q_0 \to \alpha$ szabály mellé bevesszük a $q_0' \to \alpha$ szabályt is. ■



20.10. ábra. A 20.13. példa véges automatája.

20.13. példa. Adott az $\mathrm{A} = (\{q_0, q_1, q_2\}, \{a, b\}, E, \{q_0\}, \{q_2\})$ véges automata, ahol $E = \{(q_0, a, q_0), (q_0, b, q_1), (q_1, b, q_2), (q_2, a, q_2)\}$. Az automata átmenettáblázata a következő:

δ	a	b
q_0	$\{q_0\}$	$\{q_1\}$
q_1	$\emptyset$	$\{q_2\}$
q_2	$\{q_2\}$	$\emptyset$

Az A átmenetgráfja a 20.10. ábrán látható.

Könnyű észrevenni, hogy $L(\mathrm{A}) = \{a^m bba^n \mid m \geq 0, n \geq 0\}$. Ekkor 20.11. tétel alapján a $G = (\{q_0, q_1, q_2\}, \{a, b\}, P, q_0)$ reguláris nyelvtan P szabályai a következők:

$q_0 \to aq_0 \mid bq_1$
$q_1 \to bq_2 \mid b$
$q_2 \to aq_2 \mid a$.

A 20.11. tétel bizonyításában megadott módszert könnyen átírhatjuk algoritmussá. Az eredmény a $G = (Q, \Sigma, P, q_0)$ reguláris nyelvtan, amelynek a helyettesítési szabályait a következő algoritmus határozza meg.

AUTOMATÁBÓL-REGULÁRISNYELV(A, G)

```
1  P ← ∅
2  for minden p ∈ Q
3      do for minden a ∈ Σ
4          do for minden q ∈ Q
5              do if (p, a, q) ∈ E
6                  then P ← P ∪ {p → aq}
7                       if q ∈ F
8                          then P ← P ∪ {p → a}
```

Könnyű belátni, hogy az algoritmus bonyolultsága $\Theta(m^2 n)$, ha az állapotok száma m és a betűk száma n. A 2–4. sorokban lévő három ciklus helyett lehet csupán egyet venni, ha az E elemeit vizsgáljuk, ekkor a bonyolultság legrosszabb esetben Θp, ahol p az átmenetek száma. Ez szintén $O(m^2 n)$, mivel lehetséges, hogy minden átmenet jelen van. Az algoritmus a következőképpen írható le:

AUTOMATÁBÓL-REGULÁRISNYELV'(A, G)

```
1  P ← ∅
2  for minden (p, a, q) ∈ E
3      do P ← P ∪ {p → aq}
4          if q ∈ F
5              then P ← P ∪ {p → a}
```

20.12. tétel. *Ha $L = L(G)$ reguláris nyelv, akkor létezik olyan nemdeterminisztikus véges automata, amely a L nyelvet felismeri.*

Bizonyítás. Legyen $G = (N, T, P, S)$ az L nyelvet generáló reguláris nyelvtan. Értelmezzük az $\mathrm{A} = (Q, T, E, \{S\}, F)$ nemdeterminisztikus véges automatát a következőképpen.

- $Q = N \cup \{Z\}$, ahol $Z \notin N \cup T$ (vagyis egy új szimbólum),
- Minden $A \to aB$ szabályra értelmezzük az (A, a, B) átmenetet.
- Minden $A \to a$ szabályra értelmezzük az (A, a, Z) átmenetet.
- $F = \begin{cases} \{Z\} & \text{ha } G\text{-ben nem szerepel } S \to \varepsilon \text{ szabály,} \\ \{Z, S\} & \text{ha } G\text{-ben van } S \to \varepsilon \text{ szabály.} \end{cases}$

Bebizonyítjuk, hogy $L = L(A)$.

Legyen $u = a_1 a_2 \dots a_n \in L(G)$, $u \neq \varepsilon$. Ekkor létezik u-nak egy levezetése:

$$S \Longrightarrow a_1 A_1 \Longrightarrow a_1 a_2 A_2 \Longrightarrow \dots \Longrightarrow a_1 a_2 \dots a_{n-1} A_{n-1} \Longrightarrow a_1 a_2 \dots a_n.$$

Ez a levezetés a következő szabályok alapján történt:

$$S \to a_1 A_1, \;\; A_1 \to a_2 A_2, \;\; \dots, \;\; A_{n-2} \to a_{n-1} A_{n-1}, \;\; A_{n-1} \to a_n.$$

Ekkor az automata átmeneteinek értelmezése alapján létezik az

$$S \xrightarrow{a_1} A_1 \xrightarrow{a_2} A_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{n-1}} A_{n-1} \xrightarrow{a_n} Z, \;\; z \in F$$

séta. Ez azt jelenti, hogy $u \in L(\mathrm{A})$. Ha $\varepsilon \in L(G)$, akkor van $S \to \varepsilon$ szabály, de ekkor a kezdőállapot végállapot is, tehát $\varepsilon \in L(A)$.

Legyen most $u = a_1 a_2 \dots a_n \in L(\mathrm{A})$, $u \neq \varepsilon$. Ez azt jelenti, hogy létezik a

$$S \xrightarrow{a_1} A_1 \xrightarrow{a_2} A_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{n-1}} A_{n-1} \xrightarrow{a_n} Z, \;\; Z \in F$$

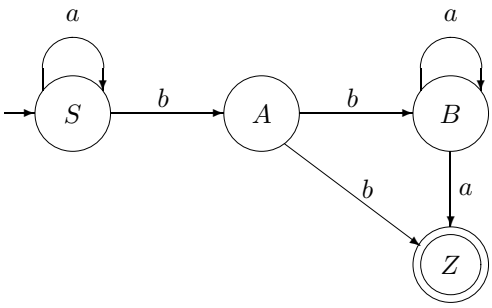
séta. (S-sel nem végződhet, csak ha u az üres szó.) Tehát G-ben szerepelnek a következő szabályok:

$$S \to a_1 A_1, \;\; A_1 \to a_2 A_2, \;\; \dots, \;\; A_{n-2} \to a_{n-1} A_{n-1}, \;\; A_{n-1} \to a_n,$$

és így létezik az

$$S \Longrightarrow a_1 A_1 \Longrightarrow a_1 a_2 A_2 \Longrightarrow \dots \Longrightarrow a_1 a_2 \dots a_{n-1} A_{n-1} \Longrightarrow a_1 a_2 \dots a_n$$

levezetés, tehát $u \in L(G)$. ∎



20.11. ábra. A 20.14. példa G nyelvtanához rendelt automata.

20.14. példa. Adott a $G = (\{S, A, B\}, \{a, b\}, \{S \to aS|bA,\ A \to bB|b,\ B \to aB|a\}, S)$ reguláris nyelvtan. A hozzá rendelt automata $A = (\{S, A, B, Z\}, \{a, b\}, E, S, \{Z\})$, ahol $E = \{(S, a, S), (S, b, A), (A, b, B), (A, b, Z), (B, a, B), (B, a, Z)\}$. Az automata átmenettáblázata a következő.

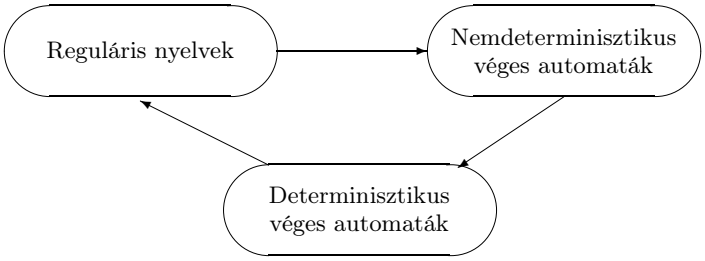
δ	a	b
S	$\{S\}$	$\{A\}$
A	$\emptyset$	$\{B, E\}$
B	$\{B, E\}$	$\emptyset$
E	$\emptyset$	$\emptyset$

Az átmenetgráf 20.11. ábrán látható. Ez az automata egyszerűsíthető. A B és Z állapotok összevonhatók.

Az előbbi tétel eredménye algoritmikusan is megfogalmazható. A REGULÁRISNYELVBŐL-AUTOMATA algoritmus hozzárendeli a $G = (N, T, P, S)$ reguláris nyelvtanhoz az $\mathrm{A} = (Q, T, E, \{S\}, F)$ véges automatát.

REGULÁRISNYELVBŐL-AUTOMATA(G, A)

```
1  E ← ∅
2  Q ← N ∪ {Z}
3  for minden A ∈ N
4      do for minden a ∈ T
5             do if (A → a) ∈ P
6                    then E ← E ∪ {(A, a, Z)}
7                for minden B ∈ N
8                    do if (A → aB) ∈ P
9                           then E ← E ∪ {(A, a, B)}
10 if (S → ε) ∉ P
11    then F ← {Z}
12    else  F ← {Z, S}
```



20.12. ábra. Kapcsolat véges automaták és reguláris nyelvek között. Tetszőleges reguláris nyelvhez megadható egy olyan nemdeterminisztikus véges automata, amely felismeri az adott reguláris nyelvet. Minden nemdeterminisztikus automata átalakítható determinisztikussá. A determinisztikus véges automata reguláris nyelvet ismer fel.

Akárcsak az AUTOMATÁBÓL-REGULÁRISNYELV algoritmus esetében, a bonyolultság ebben az esetben is $\Theta(m^2 n)$, ha a nemterminálisok száma m és a terminálisoké n. Lehetne a 3., 4. és 7. sorokban lévő ciklusokat helyettesíteni eggyel, amelyik a helyettesítési szabályokon megy végig. Ez sok esetben javítja az algoritmus hatékonyságát, amely ekkor $\Theta(p)$ lesz, ha p a szabályok száma. Az algoritmus a következő:

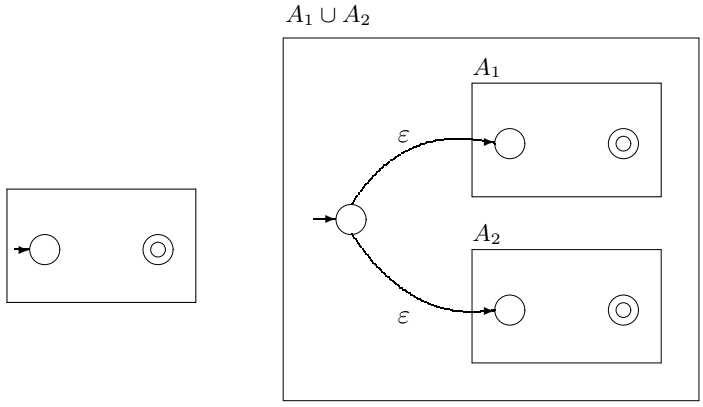
REGULÁRISNYELVBŐL-AUTOMATA'(G, A)

```
1  E ← ∅
2  Q ← N ∪ {Z}
3  for minden (A → u) ∈ P
4       do if u = a
5             then E ← E ∪ {(A, a, Z)}
6          if (u = aB) ∈ P
7             then E ← E ∪ {(A, a, B)}
8  if (S → ε) ∉ P
9     then F ← {Z}
10    else  F ← {Z, S}
```

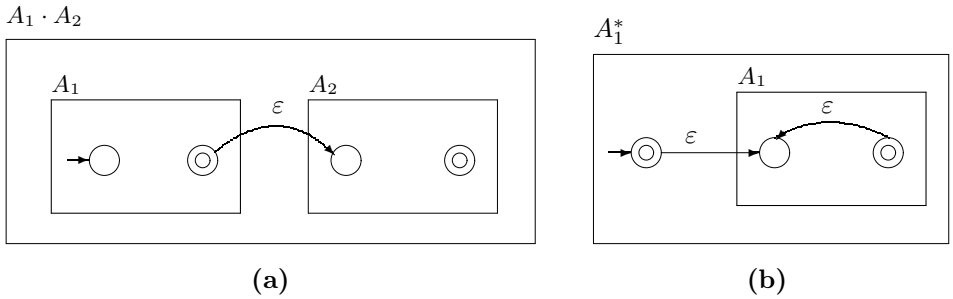
A 20.9., 20.11. és 20.12. tételek segítségével bebizonyítottuk, hogy a reguláris nyelvek halmaza egybeesik a véges (determinisztikus vagy nem) automaták által felismert nyelvek halmazával. A három tétel eredményét a 20.12. ábra foglalja össze.

20.2.2. Műveletek véges automatákkal

A következőkben értelmezzük a véges automatákon a reguláris műveleteket: egyesítés, szorzat, iteráció. Ezeket a műveleteket a legegyszerűbben a ε-lépések segítségével definiálhatjuk, az eredményautomata nemdeterminisztikus. A műveleteket szemléletesen ábrákkal is megadjuk. Egy véges automatát a 20.13(a). ábrán látható módon jelölünk. Egy nyíllal ellátott körrel jelöljük a kezdőállapotokat, és két koncentrikus körből álló jellel a végállapotokat.



20.13. ábra. (a) Véges automata jelölése. Egy bemenő nyíl jelzi a kezdőállapotokat, míg két koncentrikus kör a végállapotokat. **(b)** Két véges automata egyesítése.



20.14. ábra. (a) Két véges automata szorzata. **(b)** Véges automata iteráltja.

Ha $\mathrm{A}_1 = (Q_1, \Sigma_1, E_1, I_1, F_1)$ és $\mathrm{A}_2 = (Q_2, \Sigma_2, E_2, I_2, F_2)$, akkor a művelet eredményét A-val jelöljük: $\mathrm{A} = (Q, \Sigma, E, I, F)$.

Egyesítés

$\mathrm{A} = \mathrm{A}_1 \cup \mathrm{A}_2$, ahol

$$
\begin{aligned}
Q &= Q_1 \cup Q_2 \cup \{q_0\},\\
\Sigma &= \Sigma_1 \cup \Sigma_2,\\
I &= \{q_0\},\\
F &= F_1 \cup F_2,\\
E &= E_1 \cup E_2 \cup \bigcup_{q \in I_1 \cup I_2} \big\{(q_0, \varepsilon, q)\big\}.
\end{aligned}
$$

Az eredményautomata a 20.13(b). ábrán látható. Ugyanazt az eredményt kapjuk, ha kezdőállapothalmaznak vesszük a $I_1 \cup I_2$ halmazt egy újabb kezdőállapot helyett. Ekkor egyáltalán nem lesznek ε-lépések.

Szorzat
$\mathrm{A} = \mathrm{A}_1 \cdot \mathrm{A}_2$, ahol

$$\begin{aligned} Q &= Q_1 \cup Q_2, \\ \Sigma &= \Sigma_1 \cup \Sigma_2, \\ F &= F_2, \\ I &= I_1, \\ E &= E_1 \cup E_2 \cup \bigcup_{\substack{p \in F_1 \\ q \in I_2}} \{(p, \varepsilon, q)\} \end{aligned}$$

Az eredményautomata a 20.14(a). ábrán látható.

Iteráció
$\mathrm{A} = \mathrm{A}_1{}^*$, ahol

$$\begin{aligned} Q &= Q_1 \cup \{q_0\}, \\ \Sigma &= \Sigma_1, \\ F &= F_1 \cup \{q_0\}, \\ I &= \{q_0\} \\ E &= E_1 \cup \bigcup_{p \in I_1} \{(q_0, \varepsilon, p)\} \cup \bigcup_{\substack{q \in F_1 \\ p \in I_1}} \{(q, \varepsilon, p)\} \end{aligned}$$

Az automata iteráltja a 20.14(b). ábrán látható.

Ha az eredményül kapott A automatához hozzárendelünk egy vele ekvivalens ε-lépés nélküli automatát, akkor a kapott automatát úgy is felfoghatjuk, mint a művelet eredményét ε-lépés nélküli megfogalmazásban.

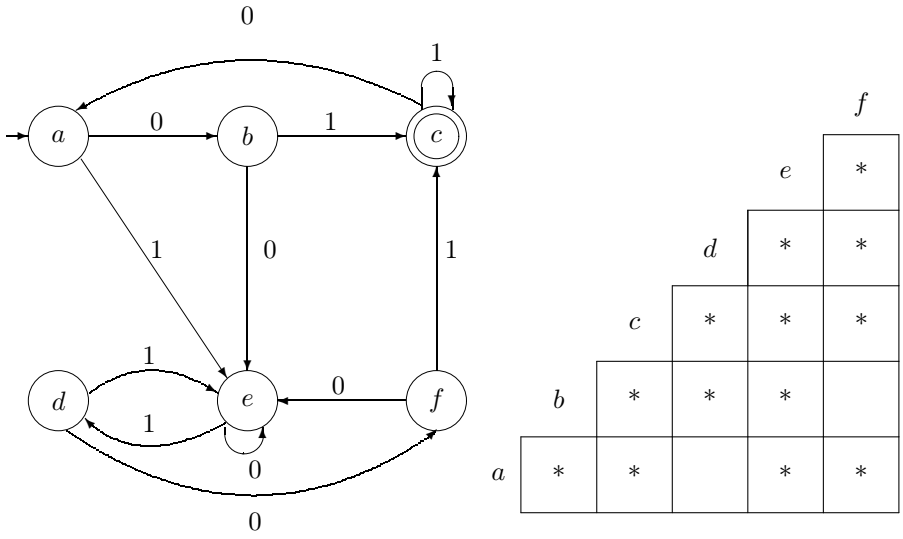
20.2.3. Véges automaták minimalizálása

Megadunk egy algoritmust, amely tetszőleges teljes determinisztikus véges automatához hozzárendel egy vele ekvivalens, de minimális számú állapotot tartalmazó automatát, ha a vizsgált automata már nem minimális számú állapotot tartalmaz.

Az automata p és q állapotát *ekvivalensnek* mondjuk, ha tetszőleges u szóra, mindkettőből végállapotba vagy mindkettőből nem végállapotba jutunk, azaz

$$p \equiv q \text{ ha } \begin{cases} p \xrightarrow{u} r,\ r \in F \text{ és } q \xrightarrow{u} s,\ s \in F & \text{vagy} \\ p \xrightarrow{u} r,\ r \notin F \text{ és } q \xrightarrow{u} s,\ s \notin F \end{cases}$$

tetszőleges $u \in \Sigma^*$ szóra. Ha két állapot nem ekvivalens, akkor azt mondjuk, hogy megkülönböztethetők. Az alábbi algoritmusban megcsillagozzuk a megkülönböztethető állapotokat, az egymással ekvivalenseket pedig összevonjuk. Az algoritmus során bizonyos állapotpárokhoz állapotpárokból álló listát rendelünk a későbbi megcsillagozás reményében. Az alábbi algoritmust olyan automatára alkalmazzuk, amelyből már kizártuk az elérhetetlen állapotokat. Mivel az automata determinisztikus és teljes a $\delta(p, a)$ pontosan egy elemet tartalmaz, az algoritmusban szereplő $(\delta(p, a), \delta(q, a))$ párokban $(\{p\}, \{q\})$ helyett mindenhol (p, q)-t írunk.



20.15. ábra. Véges automata minimalizálása.

AUTOMATA-MINIMALIZÁLÁSA(A)

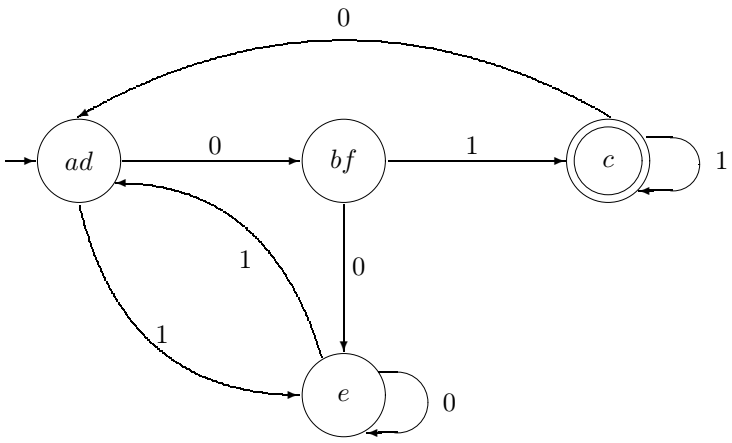
1 jelöljük meg egy-egy csillaggal az összes olyan (p, q) állapotpárt, amelyre $p \in F$ és $q \notin F$ vagy fordítva.
2 minden jelöletlen (p, q) állapotpárra végezzük el a következőket: minden $a \in \Sigma$ betűre vizsgáljuk meg az összes $(\delta(p, a), \delta(q, a))$ állapotpárt. Ha valamelyik ezek közül meg van csillagozva, akkor csillagozzuk meg a (p, q) párt is, egyetemben a már előzőleg a (p, q) párhoz rendelt lista elemeivel. Ha a fenti állapotpárok közül egy sincs megcsillagozva, akkor rendeljük hozzá a (p, q) párt egy-egy listában a $(\delta(p, a), \delta(q, a))$ párok mindegyikéhez, ha ezen párok elemei különbözőek (a későbbi megcsillagozás reményében).
3 a megjelöletlen párok ekvivalensek, és összevonhatók.

Megjegyzés. Algoritmusunk abban az esetben is alkalmazható, ha az automata nem teljes, azaz vannak olyan állapotok, amelyekből adott bemeneti jelre nincs átmenet. Ekkor $(\emptyset, \{q\})$ pár is előfordulhat, és ha q végállapot, úgy tekintjük, mintha ez a pár meg lenne csillagozva.

20.15. példa. Tekintsük a 20.15. ábrán látható automatát. Az algoritmus alkalmazásához egy táblázatot használunk, amelyben a csillagozást végezzük. A (p, q) állapotpár megjelölését (megcsillagozását) a p sor és q oszlop (vagy q sor és p oszlop) találkozásnál elhelyezett csillag jelzi.

Először megcsillagozzuk az (c, a), (c, b), (c, d), (c, e) és (c, f) párokat (mivel c az egyetlen végállapot). Ezután sorra vesszük a meg nem csillagozott állapotokat, és az algoritmus szerint megvizsgáljuk őket.

Kezdjük az (a, b) párral. Hozzárendeljük a következő állapotpárokat: $(\delta(a, 0), \delta(b, 0))$, $(\delta(a, 1), \delta(b, 1))$, azaz (b, e), (e, c), mivel (e, c) már meg van jelölve, megjelöljük (a, b)-t is.



20.16. ábra. A 20.15. ábrán látható automata minimalizált változata.

Az (a, d) pár esetében a két új pár (b, f) és (e, e). A (b, f) párhoz hozzárendeljük az (a, d)-t. Most (b, f)-fel folytatva, az (e, e) és (c, c) párokat kapjuk, amelyekhez az algoritmus szerint semmit sem rendelünk. Folytatjuk az (a, e) párral. A hozzárendelt párok (b, e) és (e, d). Egyik sincs megcsillagozva, ezért hozzájuk rendeljük az (a, e) párt. Most a (b, e) párral folytatva az (e, e), (c, d) párokat kapjuk, és mivel ez utóbbi meg van jelölve csillaggal, megjelöljük a (b, e) és az (a, e) párokat is. Így folytatva, eljutunk a fenti táblázathoz, azaz azt kapjuk, hogy $a \equiv d$ és $b \equiv f$. Ezeket összevonva, az 20.16. ábrán látható automatát kapjuk, amelyik ekvivalens az eredetivel.

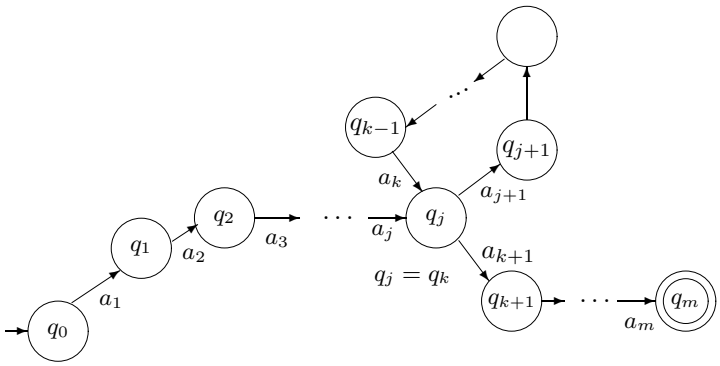
20.2.4. Iterációs lemma reguláris nyelvekre (Bar-Hillel-lemma)

A következő tétel, amelyet *iterációs lemmának*, *Bar-Hillel-lemmának* vagy *pumpáló lemmának* nevezünk (a *lemma = segédtétel* elnevezés történeti okokból maradt fenn) jól használható arra, hogy egy nyelvről bebizonyítsuk, hogy nem reguláris. Ez a tétel a reguláris nyelveket jellemzi.

20.13. tétel (Iterációs lemma). *Bármely L reguláris nyelv esetében létezik olyan $n \geq 1$ természetes szám (amely csak L-tól függ) úgy, hogy az L bármely legalább n hosszúságú u szava felbontható $u = xyz$ alakba úgy, hogy*

1) $|xy| \leq n$,
2) $|y| \geq 1$,
3) $xy^i z \in L$ minden $i = 0, 1, 2 \ldots$ értékre.

Bizonyítás. Ha L reguláris nyelv, akkor létezik olyan determinisztikus véges automata, amely felismeri az L nyelvet. Legyen ez az automata $\mathrm{A} = (Q, \Sigma, E, \{q_0\}, F)$, tehát $L = L(\mathrm{A})$. Jelöljük n-nel az állapotok számát, azaz $|Q| = n$. Legyen $u = a_1 a_2 \ldots a_m \in L$ és $m \geq n$. Ekkor, mivel az automata az u szót felismeri,



20.17. ábra. Az iterációs lemma bizonyításában használt automata rajza.

léteznek a $q_0, q_1, \ldots, q_m$ állapotok és a

$$q_0 \xrightarrow{a_1} q_1 \xrightarrow{a_2} q_2 \xrightarrow{a_3} \cdots \xrightarrow{a_{m-1}} q_{m-1} \xrightarrow{a_m} q_m,\ q_m \in F$$

séta. Mivel összesen csak n állapotunk van és $m \geq n$, a skatulya-elv[3] alapján, a $q_0, q_1, \ldots, q_m$ állapotok között van legalább két egyenlő (20.17. ábra). Legyen $q_j = q_k$, ahol $j < k$. Ekkor $j < k \leq n$. Bontsuk fel az u szót a következőképpen:

$x = a_1 a_2 \ldots a_j$

$y = a_{j+1} a_{j+2} \ldots a_k$

$z = a_{k+1} a_{k+2} \ldots a_m$.

Rögtön látszik, hogy $|xy| \leq n$ és $|y| \geq 1$. Bebizonyítjuk, hogy $xy^i z \in L$ tetszőleges i-re.

Ha $u = xyz \in L$, akkor létezik a

$$q_0 \xrightarrow{x} q_j \xrightarrow{y} q_k \xrightarrow{z} q_m,\ q_m \in F$$

séta. De mivel $q_j = q_k$, ez így is írható

$$q_0 \xrightarrow{x} q_j \xrightarrow{y} q_j \xrightarrow{z} q_m,\ q_m \in F,$$

és a $q_j \xrightarrow{y} q_j$ séta elhagyható a $q_0 \xrightarrow{xyz} q_m$ sétából vagy többször is beilleszthető. Tehát léteznek a következő séták:

$$q_0 \xrightarrow{x} q_j \xrightarrow{z} q_m,\ q_m \in F.$$

$$q_0 \xrightarrow{x} q_j \xrightarrow{y} q_j \xrightarrow{y} \ldots \xrightarrow{y} q_j \xrightarrow{z} q_m,\ q_m \in F.$$

És ebből következik, hogy $xy^i z \in L$ tetszőleges i-re, és ezzel bebizonyítottuk a lemmát. ■

[3] *Skatulya-elv:* Ha k-nál több elemet k dobozba kell elhelyeznünk, akkor legalább egy dobozba egynél több elem kerül.

20.16. példa. Bebizonyítjuk, hogy $L_1 = \{a^k b^k \mid k \geq 1\}$ nem reguláris. Ha L_1 reguláris lenne, akkor érvényes lenne rá a Bar-Hillel-lemma. Ekkor például az $u = a^n b^n$ szó, ahol n a lemmabeli érték, felbontható lenne a lemma szerinti módon. Bebizonyítjuk, hogy ez ellentmondáshoz vezet. Feltételezzük, hogy u felbontható a kért módon, azaz $u = xyz$. A lemma szerint ekkor $|xy| \leq n$, tehát x is és y is csak a-t tartalmazhatnak, és mivel $|y| \geq 1$, y legalább egy a-t tartalmaz. Ekkor $xy^i z$, ha $i \neq 1$-re, különböző számú a-t és b-t tartalmaz, tehát $xy^i z \notin L_1$ tetszőleges $i \neq 1$ értékre. Ez ellentmond a lemma állításának, tehát az a feltevésünk, hogy L_1 reguláris, hamis. Tehát $L_1 \notin \mathcal{L}_3$.

Mivel a $G_1 = (\{S\}, \{a, b\}, \{S \to ab, S \to aSb\}, S)$ környezetfüggetlen nyelvtan generálja L_1-et, így $L_1 \in \mathcal{L}_2$. E két állításból rögtön következik, hogy $\mathcal{L}_3 \subset \mathcal{L}_2$.

20.17. példa. Bebizonyítjuk, hogy $L_2 = \{u \in \{0,1\}^* \mid n_0(u) = n_1(u)\}$ nem reguláris. ($n_0(u)$ az u-ban lévő nullák, $n_1(u)$ pedig az 1-esek számát jelenti).

Az előbbi példához hasonlóan járunk el az $u = 0^n 1^n$ szóval.

Úgy tűnhet, hogy ha $x = \varepsilon,\ y = 0^n 1^n,\ z = \varepsilon$, akkor a lemma alkalmazható lenne, de nem így van, mivel $|xy| > n$.

20.18. példa. Bebizonyítjuk, hogy $L_3 = \{uu \mid u \in \{a, b\}^*\}$ nem reguláris. Legyen $w = a^n b a^n b = xyz$, ahol n itt is a lemmabeli érték. Mivel, $|xy| \leq n$, következik, hogy y csak a betűket tartalmazhat, és legalább egyet tartalmaz is. De, ekkor a lemma szerint $xz \in L_3$, ami lehetetlen. Tehát L_3 nem reguláris.

Az iterációs lemmának több érdekes következménye van.

20.14. következmény. *Az L reguláris nyelv akkor és csakis akkor nem üres, ha létezik $u \in L$, $|u| < n$, ahol n az iterációs lemmabeli n.*

Bizonyítás. Az állítás egyik irányba nyilvánvaló: ha létezik n-nél rövidebb szó L-ben, akkor $L \neq \emptyset$. Fordítva, legyen $L \neq \emptyset$, és legyen $u \in L$. Ha $|u| < n$, állításunk igaz. Ha $|u| \geq n$, alkalmazzuk az iterációs lemmát, tehát $u = xyz$ és $xz \in L$. Ha $|xz| < n$, állításunk be van bizonyítva, ha nem, akkor alkalmazzuk rá is az iterációs lemmát. Így egyszer eljutunk egy n-nél rövidebb szóhoz. ■

20.15. következmény. *Van olyan algoritmus, amely eldönti, hogy egy reguláris nyelv üres-e.*

Bizonyítás. Ha L reguláris, létezik olyan determinisztikus véges automata, amely felismeri. A iterációs lemma szerinti n-re megvizsgáljuk az összes n-nél rövidebb szavakat, ha ezek közül legalább egyet elfogad az automata, akkor $L \neq \emptyset$. ■

20.16. következmény. *Egy L reguláris nyelv akkor és csakis akkor végtelen, ha létezik $u \in L$ úgy, hogy $n \leq |u| < 2n$, ahol n az iterációs lemmabeli n.*

Bizonyítás. Ha L végtelen, van $2n$-nél hosszabb szava, legyen ez u. Mivel L reguláris, alkalmazható rá a Bar-Hillel-lemma, tehát $u = xyz$, ahol $|xy| \leq n$, tehát $|y| \leq n$

is igaz. A lemma szerint $u' = xz \in L$. Ha $u' \geq 2n$, alkalmazzuk újra a lemmát (u-nak véve az u' szót). Ha $|u'| < 2n$, akkor $|u'| \geq |u| - n \geq 2n - n = n$.

Fordítva, ha létezik $u \in L$ úgy, hogy $n \leq |u| < 2n$, akkor alkalmazva rá az iterációs lemmát, következik, hogy $u = xyz$ és $xy^iz \in L$ tetszőleges i-re, tehát L végtelen. ■

Könnyű bebizonyítani, hogy minden véges nyelv reguláris. A nyelvtan szabályait úgy adjuk meg, hogy generálják az összes szót. Például, ha $u = a_1a_2 \dots a_k$ eleme a nyelvnek, akkor bevezetjük a következő szabályokat: $A_1 \to a_1A_2$, $A_2 \to a_2A_3$, $\dots A_{n-2} \to a_{n-2}A_{n-1}$, $A_{n-1} \to a_{n-1}a_n$.

Feltehetjük a kérdést, hogyan alkalmazzuk egy véges nyelvre a Bar-Hillel-lemmát, hisz az iterálással végtelen sok szót kapunk? A nyelvet felismerő véges automata állapotainak száma nagyobb, mint a nyelv leghosszabb szavának a hossza. Ezért a nyelvben nincs n-nél hosszabb szó, amelyre alkalmazhatnánk a lemmát. Így a lemma csak üres halmazra alkalmazható.

20.2.5. Műveletek reguláris nyelvekkel

A reguláris nyelvek $\mathcal{L}_3$ halmaza zárt a reguláris műveletekre, azaz ha L_1, L_2 reguláris, akkor regulárisak a következő nyelvek is: $L_1 \cup L_2$, L_1L_2, L_1^*. Ezek könnyen bizonyíthatók automaták segítségével, hisz két véges automata egyesítése, szorzata is véges automata, ugyanígy egy automata tetszőleges hatványai, tehát a csillag-művelet eredménye is.

Ezenkívül a reguláris nyelvekre igazak a következő állítások is.

Egy reguláris nyelvnek a komplementuma is reguláris.

Ez könnyen igazolható automaták segítségével. Legyen az L nyelvet felismerő véges automata A. Ha A-ban megváltoztatjuk az állapotokat oly módon, hogy minden végállapot nem lesz végállapot, és minden állapot, amely nem volt végállapot legyen az, minden más elem változatlan marad. Az így kapott új automata azokat a szavakat ismeri fel, amelyeket A nem ismer fel, és fordítva. Így $\overline{L}$ is reguláris.

Két reguláris nyelvnek a metszete is reguláris.

Mivel $L_1 \cap L_2 = \overline{\overline{L_1} \cup \overline{L_2}}$, a metszet is reguláris.

Két reguláris nyelvnek a különbsége is reguláris.

Mivel $L_1 - L_2 = L_1 \cap \overline{L_2}$, a különbség is reguláris.

20.2.6. Reguláris kifejezések. Kleene tétele

Tekintsünk két egyszerű nyelvet: $L_1 = \{a\}$, és $L_2 = \{b\}$ (mindkettő egy-egy betűből áll). Képezzük az iteráltjukat: $L_1^* = \{a\}^*$, $L_2^* = \{b\}^*$, amelyek szintén nyelvek. Ugyanígy nyelv $L_1^* \cup L_2^* = \{a\}^* \cup \{b\}^*$. Ezeket egyszerűbben is írhatjuk: a^*, b^*, $a^* + b^*$, és ezek alatt rendre a $\{a\}^*$, $\{b\}^*$, $\{a\}^* \cup \{b\}^*$ nyelveket (szóhalmazokat) értjük. A fenti egyszerűbb alakú kifejezéseket *reguláris kifejezéseknek* nevezzük, és a következőképpen értelmezzük rekurzívan.

20.17. definíció. *Adott egy Σ ábécé. Rekurzívan értelmezzük a Σ feletti reguláris*

$$
\begin{aligned}
x+y &= y+x \\
(x+y)+z &= x+(y+z) \\
(xy)z &= x(yz) \\
(x+y)z &= xz+yz \\
x(y+z) &= xy+xz \\
(x+y)^* = (x^*+y)^* &= (x+y^*)^* = (x^*+y^*)^* \\
(x+y)^* = (xy^*)^* &= (x^*y)^* = (x^*y^*)^* \\
(x^*)^* &= x^* \\
x^*x &= xx^* \\
xx^*+\varepsilon &= x^*
\end{aligned}
$$

20.1. táblázat. Reguláris kifejezések tulajdonságai.

kifejezéseket.

- $\emptyset$ *reguláris kifejezés és az üres halmazt jelöli.*
- ε *reguláris kifejezés és a* $\{\varepsilon\}$ *halmazt jelöli.*
- *Ha* $a \in \Sigma$, *a reguláris kifejezés és az* $\{a\}$ *halmazt jelöli.*
- *Ha* x, y *reguláris kifejezések és az* X, *illetve* Y *halmazokat jelölik, akkor* $x+y$, xy, x^* *is reguláris kifejezések és rendre az* $X \cup Y$, XY *és* X^* *halmazokat jelölik.*

Ha szükséges, bizonyos kifejezéseket zárójelbe tehetünk, pl. $(a+b)^*$.

Minden reguláris kifejezés a fenti négy szabály véges sokszori alkalmazásával hozható létre. Két reguláris kifejezés *ekvivalens*, ha ugyanazt a halmazt jelölik. Az ekvivalenciára az $=$ jelet használjuk. A 20.1. táblázatban felsorolunk néhány ekvivalenciát.

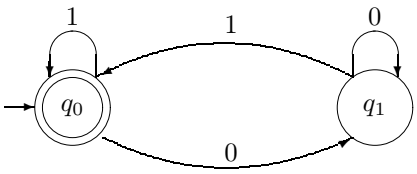
A következőkben bebizonyítjuk Kleene tételét, amely kapcsolatot teremt a reguláris nyelvek és a reguláris kifejezések között.

20.18. tétel (Kleene tétele). *Az* $L \subseteq \Sigma^*$ *nyelv pontosan akkor reguláris, ha van olyan* Σ *feletti reguláris kifejezés, amely éppen* L*-et jelöli.*

Bizonyítás. Legyen $\Sigma = \{a_1, a_2, \ldots, a_k\}$ és legyen x egy reguláris kifejezés, valamint L az a nyelv, amelyet x jelöl.

Először bebizonyítjuk, hogy ha x reguláris kifejezés, akkor az L nyelv, amelyet x jelöl, szintén reguláris. L-et a következő halmazok egyesítéséből, szorzatából, iteráltjából kapjuk: $\emptyset, \{\varepsilon\}, \{a_1\}, \ldots, \{a_k\}$, így L reguláris, mert reguláris nyelvekből reguláris műveletekkel képeztük.

Fordítva, bebizonyítjuk, hogy ha L reguláris nyelv, akkor hozzárendelhető egy x reguláris kifejezés, amely éppen az L nyelvet jelöli. Ha L reguláris, akkor létezik egy $\mathrm{A} = (Q, \Sigma, E, I, F)$ véges automata, amelyre $L = L(\mathrm{A})$. Legyenek A állapotai $q_0, q_1, \ldots, q_n$. Értelmezzük bizonyos szavak halmazát: R_{ij}^k azon szavak halmaza, amelyek hatására az A automata a q_i állapotból a q_j állapotba kerül úgy, hogy közben nem használja a k-nál nagyobb indexű állapotokat. Az állapotgráf esetében ez azt



20.18. ábra. A 20.19. példában szereplő véges automata, amelyhez reguláris kifejezést rendelünk az 1. módszer alapján.

jelenti, hogy nem megyünk át a $q_{k+1}, \dots q_n$ állapot egyikén sem, miközben végigolvassuk a szó betűit. Az R_{ij}^k halmazokat formálisan is leírhatjuk:

$R_{ij}^{-1} = \{a \in \Sigma \mid (q_i, a, q_j) \in E\}$, ha $i \neq j$,

$R_{ii}^{-1} = \{a \in \Sigma \mid (q_i, a, q_j) \in E\} \cup \{\varepsilon\}$,

$R_{ij}^k = R_{ij}^{k-1} \cup R_{ik}^{k-1} \left(R_{kk}^{k-1}\right)^* R_{kj}^{k-1}$ minden $i, j, k \in \{0, 1, \dots, n\}$ értékre.

Be lehet bizonyítani indukcióval, hogy az R_{ij}^k halmazok leírhatók reguláris kifejezésekkel. Ha $F = \{q_{i_1}, q_{i_2} \dots . q_{i_p}\}$ az A automata végállapotainak halmaza, akkor $L = L(\mathrm{A}) = R_{0i_1}^n \cup R_{0i_2}^n \cup \dots \cup R_{0i_p}^n$ is megadható reguláris kifejezésekkel. ■

A következőkben eljárásokat adunk meg, amelyek segítségével tetszőleges reguláris kifejezéshez megadhatjuk a megfelelő véges automatát, és fordítva, tetszőleges véges automatához hozzárendelhetjük a megfelelő reguláris kifejezést.

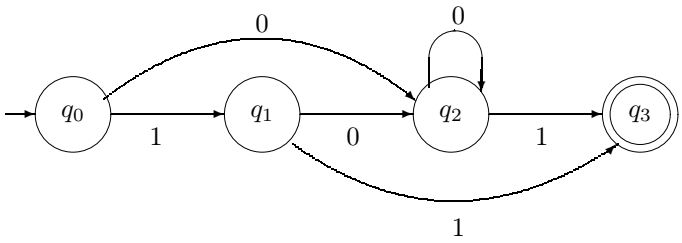
Reguláris kifejezés hozzárendelése véges automatához

Három módszert mutatunk be, amelyek mindegyike tetszőleges véges automatához hozzárendeli a megfelelő reguláris kifejezést.

1. módszer

Felhasználjuk Kleene tételének az eredményét, azaz megkonstruáljuk az R_{ij}^k halmazokat.

20.19. példa. Tekintsük a 20.18. ábrán látható automatát!



20.19. ábra. A 20.20. példában szereplő véges automata, amelyhez reguláris kifejezést rendelünk az 1. módszer alapján. A számításokat a 20.2 táblázat tartalmazza.

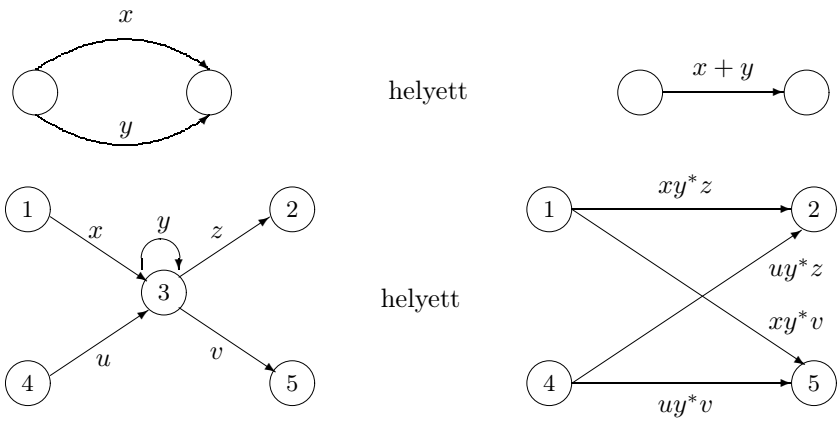
$L(\mathrm{A}) = R_{00}^1 = R_{00}^0 \cup R_{01}^0 \left(R_{11}^0\right)^* R_{10}^0$

	$k=-1$	$k=0$	$k=1$	$k=2$	$k=3$
R_{00}^k	ε	ε	ε	ε	
R_{01}^k	1	1	1	1	
R_{02}^k	0	0	$0+10$	$(0+10)0^*$	
R_{03}^k	$\emptyset$	$\emptyset$	11	$11+(0+10)0^*1$	$11+(0+10)0^*1$
R_{11}^k	ε	ε	ε	ε	
R_{12}^k	0	0	0	00^*	
R_{13}^k	1	1	1	$1+00^*1$	
R_{22}^k	$0+\varepsilon$	$0+\varepsilon$	$0+\varepsilon$	0^*	
R_{23}^k	1	1	1	0^*1	
R_{33}^k	ε	ε	ε	ε	

20.2. táblázat. A 20.19. ábra véges automatájához rendelt reguláris kifejezés meghatározása az R_{ij}^k halmazok segítségével.

R_{00}^0 : $1^*+\varepsilon=1^*$
R_{01}^0 : 1^*0
R_{11}^0 : $11^*0+\varepsilon+0=(11^*+\varepsilon)0+\varepsilon=1^*0+\varepsilon$
R_{10}^0 : 11^*
Ekkor az $L(\mathrm{A})$-nak megfelelő reguláris kifejezés a következő:
$1^*+1^*0(1^*0+\varepsilon)^*11^*=1^*+1^*0(1^*0)^*11^*$.

20.20. példa. Keressük meg a 20.19. ábrán lévő automatához rendelt reguláris kifejezést! A számításokat a 20.2. táblázat tartalmazza. Az R_{03}^3-nak megfelelő reguláris kifejezés a



20.20. ábra. Lehetséges redukálások véges automatához rendelhető reguláris kifejezés meghatározására.

következő: $11 + (0 + 10)0^*1$.

2. módszer

A véges automatát könnyen általánosíthatjuk oly módon, hogy az átmenetgráf éleihez nem betűket, hanem reguláris kifejezéseket rendelünk. Az ilyen automata (illetve átmenetgráf) megfelelő átalakításával elérhetjük, hogy az egyetlen élből álljon, amelyhez épp a megfelelő reguláris kifejezést rendeltük. Először átalakítjuk a véges automatát megfelelő ε-átmenetekkel, hogy egyetlen kezdőállapota és egyetlen végállapota legyen. Ezután a 20.20. ábrán látható redukálásokat végezzük. Amennyiben az ábrán látható 1, 2, 4, 5 pontok közül bármelyik kettő egybeesik, a végeredményben ezeket összevonjuk, azaz hurokél is megjelenik.

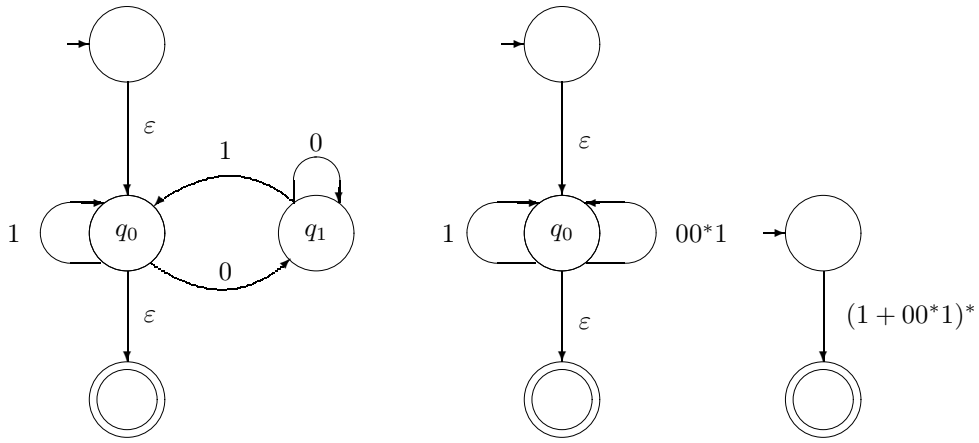
20.21. példa. A 20.18. ábra esetében a 20.21. ábrán látható lépesekkel jutunk el az eredményhez.

Tehát az eredmény $(1 + 00^*1)^*$, amely, habár más alakú, de ugyanazt a nyelvet jelenti, mint az előbbi módszerrel kapott kifejezés (20.19. példa).

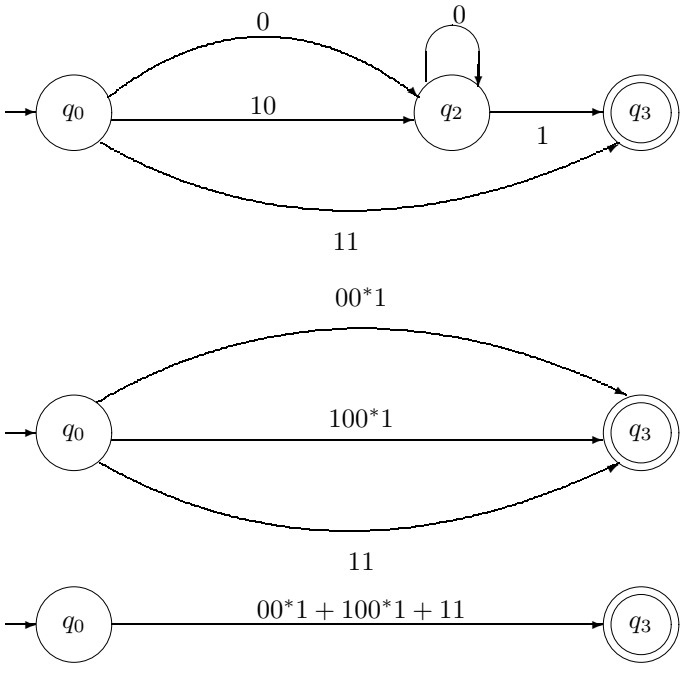
20.22. példa. A 20.19. ábra esetében nem szükséges új kezdő- és végállapotot bevezetni. Az átalakítás lépései a 20.22. ábrán láthatók. A kapott reguláris kifejezés még így is írható: $(0 + 10)0^*1 + 11$, amely azonos az előbbi módszerrel kapott kifejezéssel.

3. módszer

Egy másik módszer a reguláris kifejezés felírására a formális egyenletek módszere. Minden állapothoz hozzárendelünk egy-egy változót. A 20.19. ábra esetében legyenek ezek X, Y, Z, U, amelyek a q_0, q_1, q_2, q_3 állapotoknak felelnek meg. Az egyes állapotokba befutó nyilaknak megfelelően egyenletek írunk fel. Ha egy nyíl az X változónak megfelelő állapotból az Y-nak megfelelő állapotba mutat, és a betűvel jelölt, akkor az egyenletbe, amelynek bal oldalán Y szerepel, bekerül egy Xa tag. A



20.21. ábra. A 20.18. ábrán lévő véges automata átalakítása.



20.22. ábra. A 20.22. példa lépései.

20.19. ábra esetében ezek az egyenletek a következők:

$X = \varepsilon$
$Y = X1$
$Z = X0 + Y0 + Z0$
$U = Y1 + Z1$

Az egyenleteket, átalakítással igyekszünk $X = X\alpha + \beta$ alakra hozni (α, β szavak). Könnyű ellenőrizni, egyszerű behelyettesítéssel, hogy egy ilyen egyenletnek $X = \beta\alpha^*$ megoldása.

A fenti egyenletrendszerben, az első egyenlet segítségével azt kapjuk, hogy $Y = 1$.

Innen $Z = 0 + 10 + Z0$ azaz $Z = Z0 + (0 + 10)$,

és ezt megoldva, azt kapjuk, hogy

$Z = (0 + 10)0^*$.

Innen pedig U egyszerűen megkapható:

$U = 11 + (0 + 10)0^*1$.

Ezt a módszert alkalmazva a 20.18. ábra esetében, a következő egyenletekhez jutunk:

$X = \varepsilon + X1 + Y1$
$Y = X0 + Y0$

Kiemelés után:

$X = \varepsilon + (X + Y)1$
$Y = (X + Y)0$

Összeadva a két egyenletet, a következő egyenlethez jutunk:

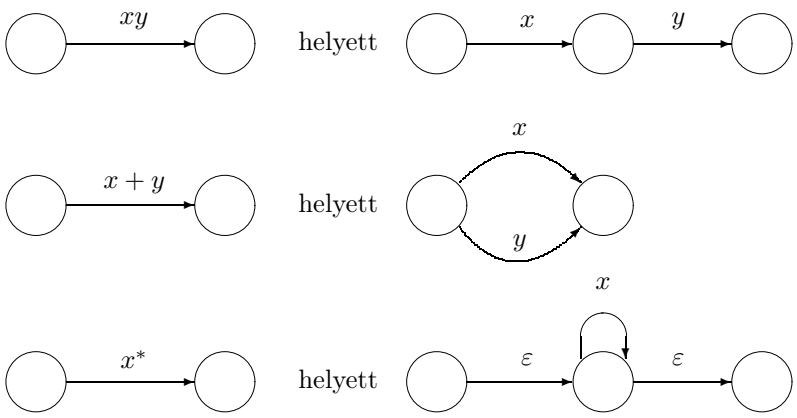
$X + Y = \varepsilon + (X + Y)(0 + 1)$, ahonnan ($\varepsilon$-t β-nak, $(0+1)$-et α-nak tekintve), eredményül kapjuk a következőt:

$X + Y = (0 + 1)^*$.

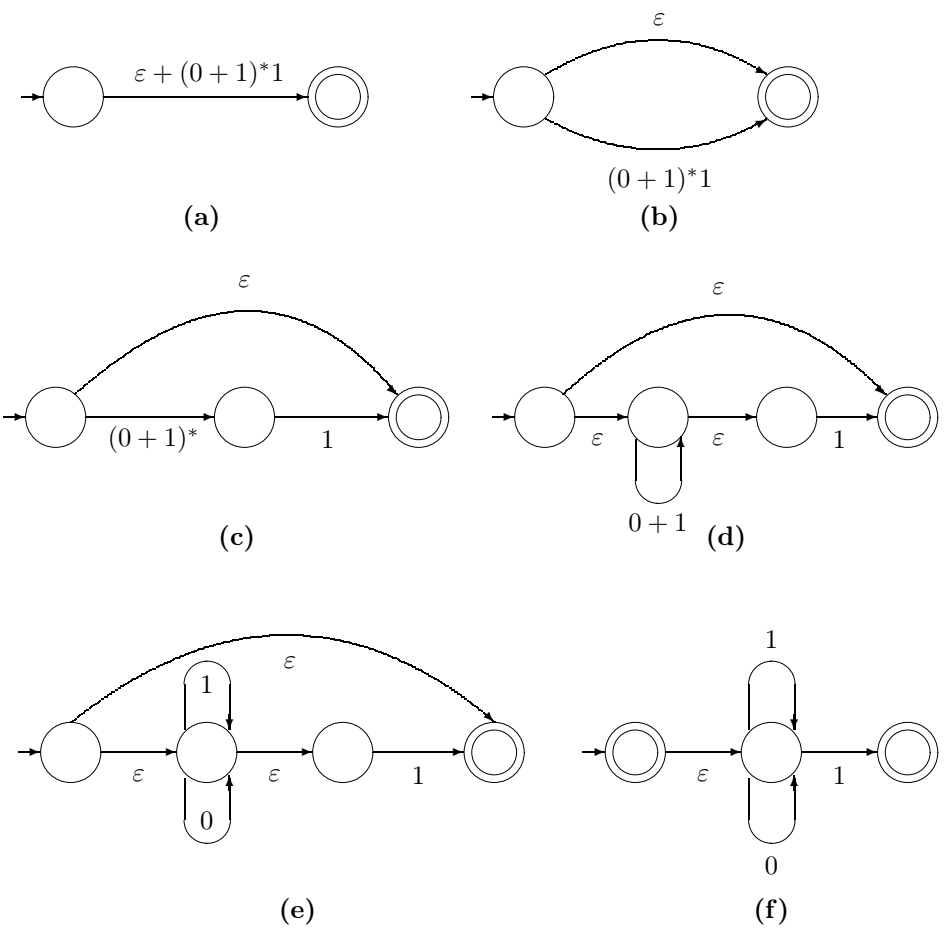
Majd innen, behelyettesítés után, megkapjuk X értékét:

$X = \varepsilon + (0 + 1)^*1$,

amely ekvivalens a másik módszerrel kapott kifejezéssel.

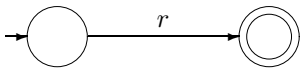


20.23. ábra. Lehetséges átalakítások reguláris kifejezéshez rendelt automata meghatározásához.



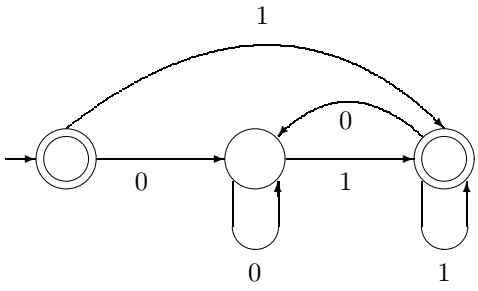
20.24. ábra. Véges automata hozzárendelése a $\varepsilon + (0+1)^*1$ reguláris kifejezéshez.

Véges automata hozzárendelése reguláris kifejezéshez
A r reguláris kifejezéshez hozzárendelünk egy általánosított véges automatát:



Azután lépésenként alkalmazzuk a 20.23. ábrán látható átalakításokat.

20.23. példa. Induljunk el a $\varepsilon + (0+1)^*1$ reguláris kifejezésből. Az átalakítás lépései a 20.24(a)-(e) ábrákon láthatók. Az utolsó automata (a 20.24(e) ábrán) egyszerűbb alakban is magadható, ez a 20.24(f) ábrán látható. Ha ebből kiküszöböljük a ε-lépést, átalakítjuk determinisztikussá, akkor a 20.25. ábrán látható automatát kapjuk eredményül, amelyről be lehet bizonyítani, hogy ekvivalens a 20.18. ábrán látható automatával.



20.25. ábra. A $\varepsilon + (0+1)^*1$ kifejezéshez rendelt véges automata.

Gyakorlatok

20.2-1. Adjunk meg egy determinisztikus véges automatát, amely a 9-cel osztható természetes számokat ismeri fel.

20.2-2. Adjunk meg egy-egy determinisztikus véges automatát, amely

a. a páros számú 0-t és páros számú 1-et tartalmazó szavakból álló nyelvet ismeri fel,

b. a páros számú 0-t és páratlan számú 1-et tartalmazó szavakból álló nyelvet ismeri fel,

c. a páratlan számú 0-t és páros számú 1-et tartalmazó szavakból álló nyelvet ismeri fel,

d. a páratlan számú 0-t és páratlan számú 1-et tartalmazó szavakból álló nyelvet ismeri fel.

20.2-3. Adjunk meg egy-egy determinisztikus véges automatát a következő nyelvek felismerésére.

$L_1 = \{a^n b^m \mid n \geq 1, m \geq 0\}$, $\quad L_2 = \{a^n b^m \mid n \geq 1, m \geq 1\}$,
$L_3 = \{a^n b^m \mid n \geq 0, m \geq 0\}$, $\quad L_4 = \{a^n b^m \mid n \geq 0, m \geq 1\}$.

20.2-4. Adjunk meg egy nemdeterminisztikus véges automatát, amely a legalább két 0-át és tetszőleges számú 1-et tartalmazó szavakat ismeri fel. Adjunk meg egy vele ekvivalens determinisztikus véges automatát.

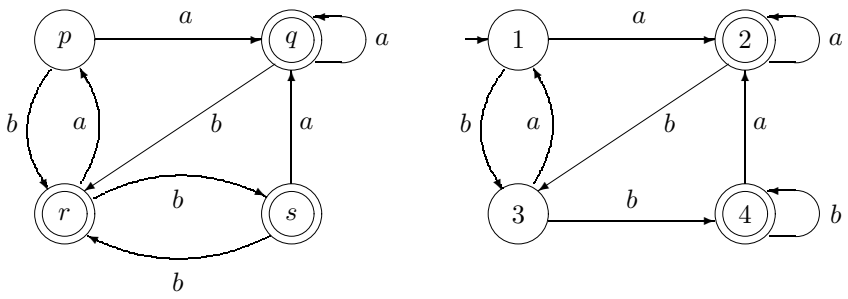
20.2-5. Minimalizáljuk a 20.26. ábrán lévő automatákat.

20.2-6. Mutassuk meg, hogy a 20.27.(a) ábrán automata nem minimalizálható.

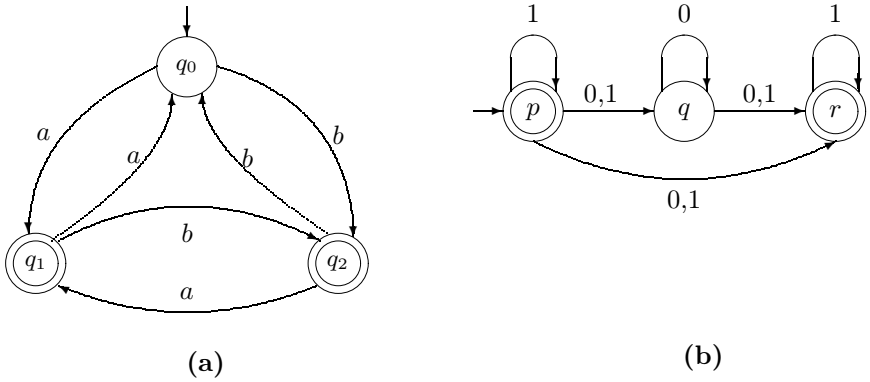
20.2-7. Alakítsuk át a 20.27.(b) ábrán látható automatát determinisztikussá, majd minimalizáljuk.

20.2-8. Értelmezzük az A_1 automatát, amely felismeri a $0(10)^n$ alakú szavakat ($n \geq 0$), az A_2-t, amely pedig az $1(01)^n$ ($n \geq 0$) alakúakat. Adjuk meg az $A_1 \cup A_2$ egyesített automatát, majd küszöböljük ki a λ-lépéseket.

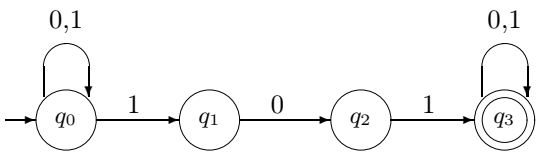
20.2-9. Adjuk meg a 20.28. ábrán látható automatához rendelhető reguláris kifejezést.



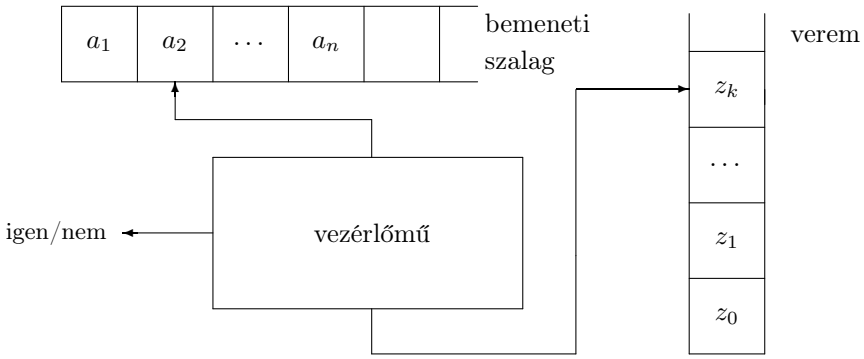
20.26. ábra. Minimizálandó véges automaták a 20.2-5. gyakorlathoz.



20.27. ábra. Véges automaták a 20.2-6. és 20.2-7. gyakorlatokhoz.



20.28. ábra. Véges automata a 20.2-9. gyakorlathoz.



20.29. ábra. Veremautomata.

20.2-10. Adjuk meg az $ab^*ba^* + b + ba^*a$ reguláris kifejezéshez rendelhető véges automatát.

20.2-11. Bizonyítsuk be, felhasználva az iterációs lemmát, hogy a következő nyelvek egyike sem reguláris.

$L_1 = \{a^n cb^n \mid n \geq 0\}$, $L_2 = \{a^n b^n a^n \mid n \geq 0\}$, $L_3 = \{a^p \mid p \text{ prím}\}$.

20.2-12. Bizonyítsuk be, hogy ha L reguláris nyelv, akkor az $\{u^{-1} \mid u \in L\}$ nyelv is reguláris.

20.2-13. Bizonyítsuk be, hogy ha $L \subseteq \Sigma^*$ reguláris nyelv, akkor regulárisak a következő nyelvek is:

$\text{pre}(L) = \{w \in \Sigma^* \mid \exists u \in \Sigma^*, wu \in L\}$, $\text{suf}(L) = \{w \in \Sigma^* \mid \exists u \in \Sigma^*, uw \in L\}$.

20.2-14. Mutassuk meg, hogy az alábbi nyelvek mind regulárisak.

$L_1 = \{ab^n cd^m \mid n > 0, m > 0\}$,

$L_2 = \{(ab)^n \mid n \geq 0\}$,

$L_3 = \{a^{kn} \mid n \geq 0,\ k \text{ állandó}\}$.

20.3. Veremautomaták és környezetfüggetlen nyelvek

20.3.1. Veremautomaták

Láttuk, hogy a véges automaták felismerőként működnek, de vannak olyan véges automaták is amelyek átalakítók. A következőkben egy újabb típusú automatával ismerkedünk meg, amely a környezetfüggetlen nyelveket ismeri fel. Ez a *veremautomata*, amely a bemeneti szalag mellett használ egy belső memóriát, amely veremszerkezetű, azaz olvasáskor csak a tetején levő elem hozzáférhető (20.29. ábra).

Az automata működése során egy bemeneti jel (amely az üres szó is lehet), az aktuális állapot és a verem tetején lévő elem alapján állapotot vált és beír egy szót a verembe. A bemeneti szót akkor ismeri fel, ha annak beolvasása után végállapotba kerül vagy kiüríti a vermet. Meg fogjuk mutatni, hogy a kétféle felismerés ekvivalens egymással, azaz ha egy veremautomata végállapottal ismer fel egy nyelvet, akkor

mindig hozzárendelhetünk egy olyan veremautomatát, amely üres veremmel ismeri fel ugyanazt a nyelvet és fordítva.

A determinisztikus és a nemdeterminisztikus veremautomaták, ellentétben a véges automatákkal, nem ugyanazt a nyelvosztályt ismerik fel. Míg a nemdeterminisztikus változat a környezetfüggetlen nyelveket ismeri fel, addig a determinisztikus csak azok egy részhalmazát.

Akárcsak a véges automaták esetében, itt is értelmezhetünk olyan átalakító veremautomatákat, amelyek környezetfüggetlen nyelveket alakítanak át környezetfüggetlen nyelvekké.

20.19. definíció. ***Nemdeterminisztikus veremautomatának*** *nevezzük a*

$$\mathrm{P} = (Q, \Sigma, W, E, q_0, z_0, F)$$

rendezett hetest, ahol

- Q *az* ***állapotok*** *véges, nem üres halmaza,*
- Σ *a* ***bemeneti ábécé****,*
- W *a* ***veremábécé****,*
- $E \subseteq Q \times \big(\Sigma \cup \{\varepsilon\}\big) \times W \times W^* \times Q$ *az* ***átmenetek*** *vagy* ***élek*** *halmaza,*
- $q_0 \in Q$ *a* ***kezdőállapot****,*
- $z_0 \in W$ *a* ***veremmemória kezdőjele****,*
- $F \subseteq Q$ *a* ***végállapotok*** *halmaza.*

A veremautomata kezdőállapotból indul, a bemeneti szalagról elolvas egy jelet (amely után elmozdul egyet jobbra) vagy nem olvas semmit (és ekkor nem mozdul el), elolvassa a verem tetején lévő elemet, és mindezek hatására állapotot vált, és egy szót (amely akár ε is lehet) ír a verembe. Az automata akkor áll meg, ha nem tud továbblépni (mivel nincs átmenet az adott állapotból az aktuális bemeneti és veremjelre), vagy ha, miután elolvasta a bemeneti szót, végállapotba kerül, vagy pedig az utolsó állapot nem végállapot, de a verem üres.

Egy (p, a, z, w, q) átmenet azt jelenti, hogy az automata aktuális állapota p, a bemeneti szalagról az a jelet (amely az üres szó is lehet), míg a verem tetejéről a z jelet olvassa, és ezek hatására beírja a verembe a w szót, és átvált a q állapotba. A w szó beírása a verembe követi a természetes sorrendet, azaz először a szó első betűje, utoljára pedig az utolsó betűje kerül be a verembe. Ez azt jelenti, hogy ekkor a szó utolsó betűje olvasható a veremből. A veremből való olvasás egyben azt is jelenti, hogy a kiolvasott jelet kitöröljük a veremből. A könnyebb olvashatóságért a (p, a, z, w, q) átmenet helyett a $\big(p, (a, z/w), q\big)$ jelölést használjuk, amely utal arra, hogy az a bemeneti jel elolvasása hatására a veremben kicseréljük z-t w-re.

Akárcsak a véges automaták esetében, most is értelmezhetünk egy átmenetfüggvényt a következőképpen:

$$\delta \;:\; Q \times (\Sigma \cup \{\varepsilon\}) \times W \to \mathcal{P}(W^* \times Q),$$

amely az aktuális állapothoz, bemeneti jelhez és verem tetején lévő elemhez hozzárendel (w, q) állapotpárokat, ahol w a verembe írt szó, q pedig az új állapot.

Mivel a veremautomata nemdeteminisztikus, az átmenetfüggvény esetében

$\delta(q, a, z) = \{(\gamma_1, p_1), \ldots, (\gamma_k, p_k)\}$ (ha az automata a bemeneti szalagról olvas egy jelet, majd az olvasófej továbblép), vagy

$\delta(q, \varepsilon, z) = \{(\gamma_1, p_1), \ldots, (\gamma_k, p_k)\}$ (ha nem mozdul el az olvasófej a bemeneti szalagon).

A veremautomata ***determinisztikus***, ha tetszőleges $q \in Q$ és $z \in \Sigma$ esetében

- $|\delta(q, a, z)| \leq 1$, $\forall a \in \Sigma \cup \{\varepsilon\}$
- Ha $\delta(q, \varepsilon, z) \neq \emptyset$, akkor $\delta(q, a, z) = \emptyset$, $\forall a \in \Sigma$.

A veremautomatához mindig megadható egy átmenettáblázat, akárcsak a véges automaták esetében. Ugyanakkor könnyen értelmezhetjük az átmenetgráfot is, amely csupán annyiban különbözik a véges automatáknál használt gráftól, hogy a $\big(p, (a, z/w), q\big)$ átmenetnek megfelelően a (p, q) élre $(a, z/w)$ kerül.

20.24. példa. $\mathrm{P}_1 = (\{q_0, q_1, q_2\}, \{a, b\}, \{z_0, z_1\}, E, q_0, z_0, \{q_0\})$. Az E halmaz elemei:

$\big(q_0, (\varepsilon, z_0/z_0), q_0\big)$ $\quad$ $\big(q_0, (a, z_0/z_0z_1), q_1\big)$
$\big(q_1, (a, z_1/z_1z_1), q_1\big)$ $\quad$ $\big(q_1, (b, z_1/\varepsilon), q_2\big)$
$\big(q_2, (b, z_1/\varepsilon), q_2\big)$ $\quad$ $\big(q_2, (\varepsilon, z_0/z_0), q_0\big)$

Az átmenetfüggvény:

$\delta(q_0, \varepsilon, z_0) = \{(\varepsilon, q_0)\}$ $\quad$ $\delta(q_1, b, z_1) = \{(\varepsilon, q_0)\}$
$\delta(q_0, a, z_0) = \{(z_0z_1, q_1)\}$ $\quad$ $\delta(q_2, b, z_1) = \{(\varepsilon, q_2)\}$
$\delta(q_1, a, z_1) = \{(z_1z_1, q_1)\}$ $\quad$ $\delta(q_2, \varepsilon, z_0) = \{(\varepsilon, q_2)\}$

Az átmenettáblázat:

	a		b		ε	
	z_0	z_1	z_0	z_1	z_0	z_1
q_0	(z_0z_1, q_1)				(ε, q_0)	
q_1		(z_1z_1, q_1)		(ε, q_2)		
q_2				(ε, q_2)	(ε, q_0)	

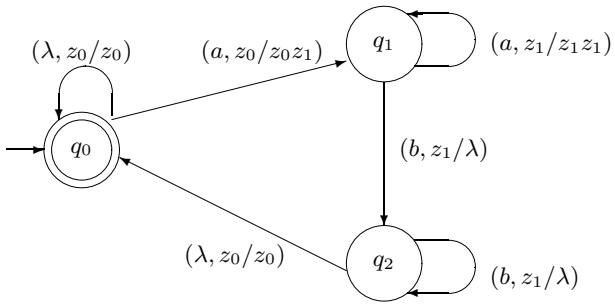
A fenti átmenettáblázatban, minden négyzetben csak egy elem szerepel, mivel minden halmaz csak egy elemet tartalmaz. Általában egymás alá írjuk a halmaz különböző elemeit, és nem használjuk a halmaz szokásos jelölését.

A veremautomata átmenetgráfja a 20.30. ábrán látható.

Az aktuális állapot, a bemeneti szalagon a még elolvasandó szó (vagy szórészlet) és a verem tartalma együtt képezik a veremautomata egy *konfigurációját*, vagyis

$$(q, u, v) \text{ egy konfiguráció, ahol } q \in Q,\ u \in \Sigma^*,\ v \in W^*.$$

Ha $u = a_1a_2 \ldots a_k$ és $v = x_1x_2 \ldots x_m$, akkor a veremautomata kétféleképpen léphet



20.30. ábra. Példa veremautomatára.

(azaz konfigurációt válthat).
- $(q,\ a_1a_2\dots a_k,\ x_1x_2\dots x_{m-1}x_m) \Longrightarrow (p,\ a_2a_3\dots a_k,\ x_1, x_2\dots x_{m-1}w)$, ha $\big(q,(a_1,x_m/w),p\big) \in E$
- $(q,\ a_1a_2\dots a_k,\ x_1x_2\dots x_m) \Longrightarrow (p,\ a_1a_2\dots a_k,\ x_1, x_2\dots x_{m-1}w)$, ha $\big(q,(\varepsilon,x_m/w),p\big) \in E$.

A $\Longrightarrow$ jel helyett szokták még használni a $\vdash$ jelet is.

Az automata működése: elindulunk a $(q_0,\ a_1a_2\dots a_n,\ z_0)$ kezdeti konfigurációból, majd meghatározzuk az összes lehetséges következő konfigurációt, majd ezekre a rákövetkezőket, és így tovább, ameddig lehet.

20.20. definíció. *Azt mondjuk, hogy a* P *veremautomata* ***végállapottal*** *felismer egy w szót, ha van* P*-beli konfigurációknak olyan sorozata, amelyre teljesülnek a következők:*
- *a sorozat első eleme (q_0, w, z_0),*
- *a sorozat minden eleméből van átmenet a sorozat következő elemébe,*
- *a sorozat utolsó eleme (p, ε, γ), ahol $p \in F$ és $\gamma \in W^*$.*

20.21. definíció. *Azt mondjuk, hogy a* P *veremautomata* ***üres veremmel*** *felismer egy w szót, ha van* P*-beli konfigurációknak olyan sorozata, amelyre teljesülnek a következők:*
- *a sorozat első eleme (q_0, w, z_0),*
- *a sorozat minden eleméből van átmenet a sorozat következő elemébe,*
- *a sorozat utolsó eleme $(p, \varepsilon, \varepsilon)$, és p tetszőleges állapot.*

Ha P végállapottal ismeri fel w-t, akkor $(q_0, w, z_0) \Longrightarrow \dots \Longrightarrow (p, \varepsilon, \gamma),\ p \in F$, azaz röviden $(q_0, w, z_0) \overset{*}{\Longrightarrow} (p, \varepsilon, \gamma),\ p \in F$. A P veremautomata által végállapottal felismert szavak halmazát (nyelvet) L(P)-vel jelöljük.

Ha P üres veremmel ismeri fel w-t, akkor $(q_0, w, z_0) \overset{*}{\Longrightarrow} (p, \varepsilon, \varepsilon)$. A P veremautomata által üres veremmel felismert szavak halmazát L_ε(P)-vel jelöljük.

20.25. példa. Ha a 20.24. példa P_1 automatáját megvizsgáljuk, észrevehetjük, hogy az $\{a^nb^n \mid n \geq 0\}$ nyelvet ismeri fel végállapottal. Végezzük el a levezetést a következő szavakra: *aaabbb* és *abab*.

Az a^3b^3 szót felismeri az automata, mivel:
$(q_0, aaabbb, z_0) \Longrightarrow (q_1, aabbb, z_0z_1) \Longrightarrow (q_1, abbb, z_0z_1z_1) \Longrightarrow (q_1, bbb, z_0z_1z_1z_1)$
$\Longrightarrow (q_2, bb, z_0z_1z_1) \Longrightarrow (q_2, b, z_0z_1) \Longrightarrow (q_2, \varepsilon, z_0) \Longrightarrow (q_0, \varepsilon, \varepsilon)$, és mivel q_0 végállapot, a veremautomata felismeri a szót.

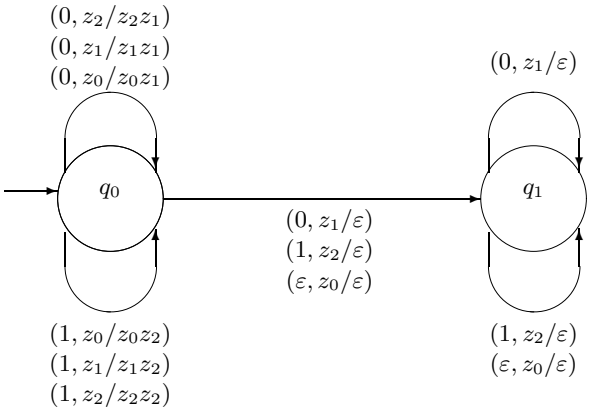
Hogy bebizonyítsuk, hogy az *abab* szót nem ismeri fel, szükségünk van az összes lehetőség megvizsgálására. Könnyű belátni, hogy ebben az esetben csak a következő két lehetőség van:
$(q_0, abab, z_0) \Longrightarrow (q_1, bab, z_0z_1) \Longrightarrow (q_2, ab, z_0)$, de innen nincs átmenet.
$(q_0, abab, z_0) \Longrightarrow (q_0, abab, \varepsilon)$, és innen sincs tovább átmenet. Ezért az automata nem ismeri fel az *abab* szót.

20.26. példa. $\mathrm{P}_2 = (\{q_0, q_1\}, \{0, 1\}, \{z_0, z_1, z_2\}, E, q_0, z_0, \emptyset)$

Az átmenettáblázat:

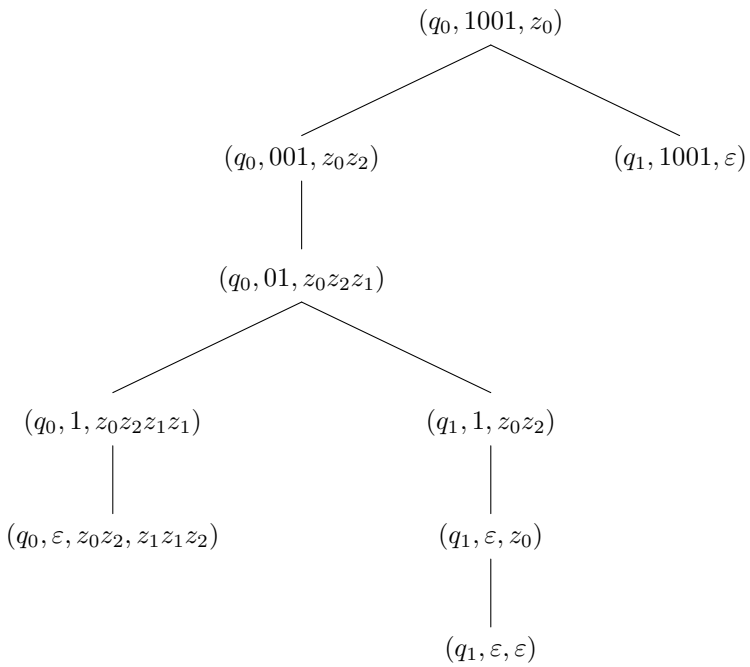
	0			1			ε
	z_0	z_1	z_2	z_0	z_1	z_2	z_0
q_0	(z_0z_1, q_0)	(z_1z_1, q_0) (ε, q_1)	(z_2z_1, q_0)	(z_0z_2, q_0)	(z_1z_2, q_0)	(z_2z_2, q_0) (ε, q_1)	(ε, q_1)
q_1		(ε, q_1)				(ε, q_1)	(ε, q_1)

Az átmenetgráf a 20.31. ábrán látható.



20.31. ábra. A 20.26. példa átmenetgráfja.

Az P_2 veremautomata a $\{uu^{-1} \mid u \in \{0, 1\}$ nyelvet ismeri fel (vagyis a $\{0, 1\}$ halmazon értelmezett tükörszavakat). Lévén, hogy nemdeterminisztikus, egy szó felismerésének levezetése egy fa segítségével történik, mivel figyelembe kell venni minden átmenetet. Nézzük



20.32. ábra. Az 1001 szó felismerésének levezetése (20.26. példa).

meg például, hogy 1001 szót felismeri-e! A levezetés a 20.32. ábrán látható. Mivel a fa egyik levele a $(q_1, \varepsilon, \varepsilon)$ konfigurációt tartalmazza, az P_2 automata felismeri üres veremmel az 1001 szót.

A 20.33. ábrán látható fa annak bizonyítéka annak, hogy az P_2 veremautomata nem ismeri fel az 101 szót. A levelekben lévő konfigurációkat nem lehet folytatni, és egyik sem $(q, \varepsilon, \varepsilon)$ alakú.

20.22. tétel. *A* P_1 *nemdeterminisztikus veremautomata pontosan akkor ismeri fel üres veremmel az* L *nyelvet, ha létezik egy olyan* P_2 *nemdeterminisztikus veremautomata, amelyik ugyanazt az* L *nyelvet végállapottal ismeri fel.*

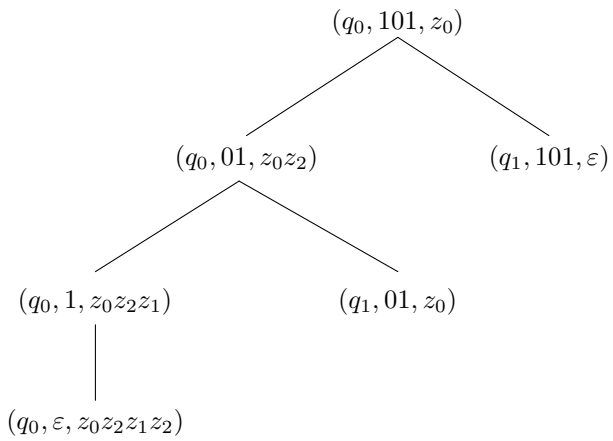
Bizonyítás. a) Legyen $P_1 = (Q, \Sigma, W, E, q_0, z_0, \emptyset)$ veremautomata, amely üres veremmel ismeri fel az L nyelvet. Definiáljuk az P_2 automatát.

$P_2 = (Q \cup \{p_0, p\}, \Sigma, W \cup \{x\}, E', p_0, x, \{p\})$,
ahol $p, p_0 \notin Q,\ , x \notin W$.

$$E' = E \cup \Big\{\big(p_0, (\varepsilon, x/xz_0), q_0\big)\Big\} \cup \Big\{\big(q, (\varepsilon, x/\varepsilon), p\big) \mid q \in Q\Big\}$$

P_2 működése: P_2 először, egy ε-lépéssel átmegy az P_1 kezdőállapotába, ettől kezdve úgy működik, mint P_1. Ha P_1 egy adott szóra kiüríti a saját vermét, akkor P_2-nél még mindig marad egy x a veremben, amelyet P_2 ε-lépéssel töröl és végállapotba kerül. P_2 csak akkor kerülhet végállapotba, ha P_1 kiürítette a vermét.

b) Legyen $P_2 = (Q, \Sigma, W, E, q_0, z_0, F)$ egy veremautomata, amely az L nyelvet



20.33. ábra. A 20.26. példa veremautomatája nem ismeri fel az 101 szót.

végállapottal ismeri fel. Definiáljuk az P_1 veremautomatát:

$\mathrm{P}_1 = (Q \cup \{p_0, p\}, \Sigma, W \cup \{x\}, E', p_0, x, \emptyset)$,

ahol $p_0, p \notin Q$, $x \notin W$.

$$E' = E \cup \Big\{ \big(p_0, (\varepsilon, x/xz_0), q_0\big) \Big\} \cup \Big\{ \big(q, (\varepsilon, x/\varepsilon), p\big) \mid q \in F, z \in W \cup \{x\} \Big\}$$
$$\cup \Big\{ \big(p, (\varepsilon, x/\varepsilon), p\big) \mid z \in W \cup \{x\} \Big\}$$

P_1 működése: P_1 először ε-lépéssel beírja a verembe x mellé z_0-t, P_2 kezdőállapotát, ettől kezdve mint P_2 működik, azaz végállapotba jut minden felismert szóra. Innen P_1 ε-lépéssel kiüríti a vermet. P_1 csak akkor ürítheti ki a vermet, ha P_2 végállapotba kerül. ■

A következő két tétel azt bizonyítja, hogy a nemdeterminisztikus veremautomaták által felismert nyelvek halmaza éppen a környezetfüggetlen nyelvek halmaza.

20.23. tétel. *Ha G környezetfüggetlen nyelvtan, akkor létezik egy olyan* P *nemdeterminisztikus veremautomata, amely üres veremmel felismeri az $L(G)$ nyelvet, azaz $L_\varepsilon(P) = L(G)$.*

Csak a bizonyítás ötletét adjuk meg. Legyen $G = (N, T, P, S)$ egy környezetfüggetlen nyelvtan, és $\varepsilon \notin L(G)$. Értelmezzük az P veremautomatát a következőképpen:

$\mathrm{P} = (\{q\}, T, N \cup T, E, q, S, \emptyset)$,

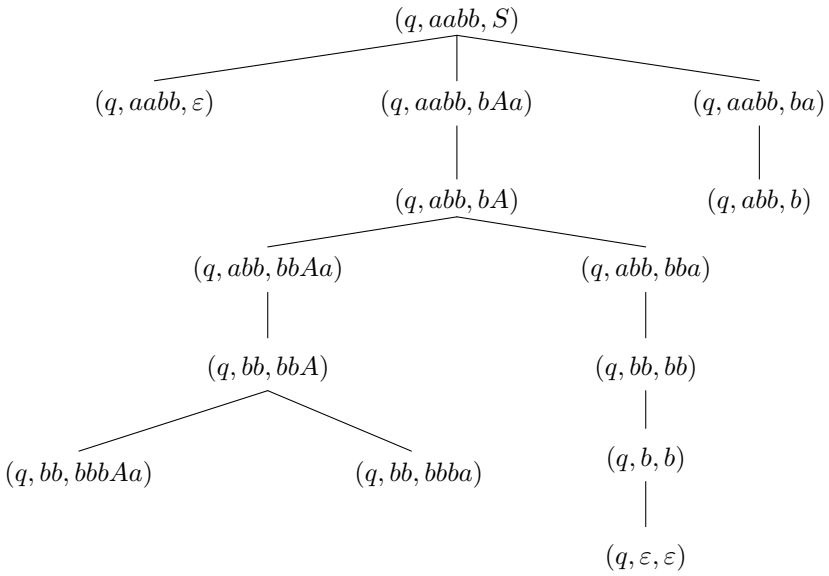
ahol $q \notin N \cup T$.

Az E átmenethalmaz értelmezése:

• Ha létezik a G nyelvtan szabályai között $A \to \alpha$ szabály, akkor tegyük be E-be a $\big(q, (\varepsilon, A/\alpha^{-1}), q\big)$ átmenetet,

• Minden $a \in T$ jelre tegyük be E-be a $\big(q, (a, a/\varepsilon), q\big)$ átmenetet.

Ha van $S \to \alpha$ szabály G-ben, az automata egy ε-lépéssel beírja a verembe az α tükörképét. Ha a beolvasott betű egyezik a verem tetején lévővel, akkor törli azt a



20.34. ábra. Szó felismerése üres veremmel. 20.27. példa

veremből. Ha a verem tetején az A nemterminális betű van, akkor beviszi a verembe valamelyik A-val kezdődő szabály jobb oldalának a tükörképét. Ha a szó beolvasása végén a verem kiürül, a veremautomata felismerte a szót.

20.27. példa. Legyen $G = (\{S, A\}, \{a, b\}, \{S \to \varepsilon \mid ab \mid aAb, A \to aAb \mid ab\}, S)$. Ekkor $\mathrm{P} = (\{q\}, \{a, b\}, \{a, b, A, S\}, E, q, S, \emptyset)$, a a következő átmenettáblázattal.

	a				b				ε			
	a	b	S	A	a	b	S	A	a	b	S	A
q	(ε, q)					(ε, q)					$(\varepsilon, q,)$ (ba, q) (bAa, q)	(bAa, q) (ba, q)

Nézzük meg, hogyan ismeri fel az P veremautomata az $aabb$ szót, amelyet a G nyelvtanban a következőképpen lehet levezetni.

$$S \Longrightarrow aAb \Longrightarrow aabb,$$

ahol alkalmaztuk az $S \to aAb$ és $A \to ab$ szabályokat. A felismerés üres veremmel történik (20.34. ábra).

20.24. tétel. *Ha* P *nemdeterminisztikus veremautomata, akkor létezik egy olyan G környezetfüggetlen nyelvtan, hogy* P *üres veremmel felismeri az $L(G)$ nyelvet, azaz $L_\varepsilon(P) = L(G)$.*

Bizonyítás helyett megadjuk, hogyan kell definiálni a G nyelvtant. Legyen P $= (Q, \Sigma, W, E, q_0, z_0, \emptyset)$. Ekkor $G = (N, T, P, S)$, ahol
$N = \{S\} \cup \{S_{p,z,q} \mid p, q \in Q, z \in W\}$ és $T = \Sigma$.
A P szabályok pedig a következők:

- $S \to S_{q_0,z_0,q}, \ \forall q \in Q,$
- $S_{q,z,p_{k+1}} \to aS_{p_1z_1p_2}S_{p_2z_2p_3}\dots S_{p_kz_kp_{k+1}},$
  $\forall q, p_1, p_2, \dots, p_{k+1} \in Q, \ z_1, z_2, \dots z_k \in W, \ \forall a \in \Sigma \cup \{\varepsilon\},$
  ha $\big(q, (a, z/z_k \dots z_2z_1), p_1\big) \in E$
- $S_{q,z,p} \to a,$
  $\forall p, q \in Q, z \in W, \forall a \in \Sigma \cup \{\varepsilon\},$
  ha $\big(q(a, z/\varepsilon), p\big) \in E.$

A determinisztikus veremautomaták a környezetfüggetlen nyelveknek egy részhalmazát, az $LR(1)$ nyelveket ismerik fel.

20.28. példa. Példaként, tekintsük az előbbi példában (20.27. példa) meghatározott P veremautomatát: P $= (\{q\}, \{a, b\}, \{a, b, A, S\}, E, q, S, \emptyset)$. A G nyelvtan a következő:

$$G = (\{S, S_a, S_b, S_S, S_A, \}, \{a, b\}, P, S),$$

ahol $S_{q,z,q}$ helyett egyszerűen csak S_z-t írtunk. Felírjuk az automata átmeneteit:

$$\begin{array}{lll} \big(q, (a, a/\varepsilon), q\big), & \big(q, (b, b/\varepsilon), q\big), & \\ \big(q, (\varepsilon, S/\varepsilon), q\big), & \big(q, (\varepsilon, S/ba), q\big), & \big(q, (\varepsilon, S/bAa), q\big) \\ \big(q, (\varepsilon, A/ba), q\big), & \big(q, (\varepsilon, A/bAa), q\big). & \end{array}$$

Ezek alapján a következő szabályokat definiálhatjuk:

$S \ \to S_S$
$S_a \ \to a$
$S_b \ \to b$
$S_S \to \varepsilon \mid S_aS_b \mid S_aS_AS_b$
$S_A \to S_aS_AS_b \mid S_aS_b$

Könnyen belátható, hogy S_S-t kiiktathatjuk, így a szabályok, miután az elsőt elhagyjuk a következő lesznek:

$S \ \to \varepsilon \mid S_aS_b \mid S_aS_AS_b$
$S_A \to S_aS_AS_b \mid S_aS_b$
$S_a \ \to a, \ \ S_b \to b,$

ezek a szabályok pedig helyettesíthetők a következőkkel:

$S \to \varepsilon \mid ab \mid aAb$
$A \to aAb \mid ab.$

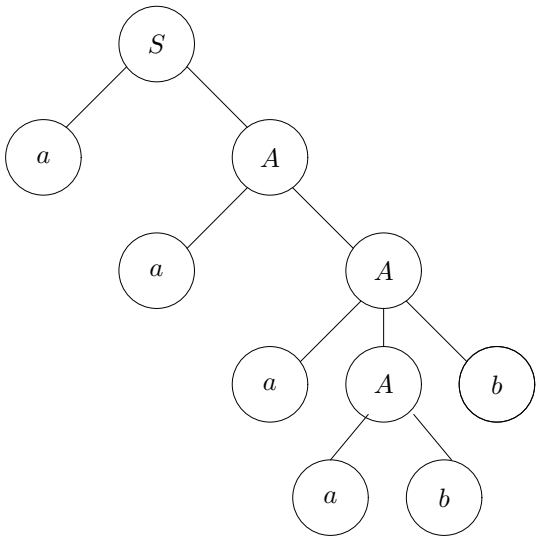
20.3.2. Környezetfüggetlen nyelvek

Amint azt az 20.1. alfejezetben láttuk, egy környezetfüggetlen nyelvtan olyan $G = (N, T, P, S)$ nyelvtan, amelynek a P szabályai $A \to w$ alakúak, $A \in N$, $w \in (N \cup T)^+$. Megengedhető az $S \to \varepsilon$ szabály is, amennyiben S nem szerepel egyetlen szabály jobb oldalán sem. Az

$$L(G) = \{u \in T \mid S \overset{*}{\underset{G}{\Longrightarrow}} u\}$$

nyelv a G nyelvtan által generált környezetfüggetlen nyelv.

Például a $G = (\{S, A\}, \{a, b\}, \{S \to aA \mid a \mid \varepsilon, A \to aA \mid aAb \mid ab \mid b\}, S)$



20.35. ábra. Az *aaaabb* szó levezetési fája.

nyelvtan környezetfüggetlen. Ez a nyelvtan az $L(G) = \{a^n b^m \mid n \geq m \geq 0\}$ nyelvet generálja. Az $a^4b^2 \in L(G)$ szó levezetése a következő:

$$S \Longrightarrow aA \Longrightarrow aaA \Longrightarrow aaaAb \Longrightarrow aaaabb.$$

Ezt a levezetést ábrázolhatjuk a 20.35. ábrán lévő fával.

Az ilyen fát *levezetési fának* nevezzük. A levezetési fában a gyökér címkéje mindig az S kezdő szimbólum, minden belső csúcsának címkéje egy nemterminális jel, minden levelének címkéje terminális szimbólum. Ha egy belső csúcs címkéje A, akkor a leszármazottjai egy $A \to a_1 a_2 \dots a_k$ szabály jobb oldalának a_1, a_2, $\dots a_k$ jeleivel címkézett csúcsok. A levezetési fa *eredménye* az a szó, amelyet a fa levelei tartalmaznak, balról jobbra olvasva. Az előbbi levezetési fa eredménye az *aaaabb* szó.

Minden levezetéshez hozzárendelhető egy levezetési fa. Fordítva, egy tetszőleges levezetési fához több levezetés is rendelhető. Tekintsük a következő nyelvtant:

$G = (\{S, A\}, \{a, b, c\}, \{S \to bA \mid bAS \mid a,\ A \to cS \mid a\}, S)$.

Nézzük meg a következő levezetéseket:

$S \Longrightarrow bAS \Longrightarrow baS \Longrightarrow baa$

$S \Longrightarrow bAS \Longrightarrow bAc \Longrightarrow baa.$

Ugyanazt a *baa* szót két különböző levezetés adta meg. Ha azonban, megegyezünk abban, hogy mindig az első nemterminális jelet helyettesítjük, akkor csak az első levezetést fogadjuk el. Az ilyen levezetésre azt mondjuk, hogy *legbaloldalibb.*

20.25. definíció. *Az* $\alpha_0 \Longrightarrow \alpha_1 \Longrightarrow \dots \Longrightarrow \alpha_n$ *levezetést* ***legbaloldalibb levezetésnek*** *nevezzük, ha*

$$\alpha_i = u_i A_i \beta_i, \qquad \alpha_{i+1} = u_i \gamma_i \beta_i, \quad i \in \{1, 2, \dots, n-1\}$$

$$u_i \in T^*, \ \ A_i \in N, \ \ \beta_i, \gamma_i \in (N \cup T)^*, \ \ (A_i \to \gamma_i) \in P$$

.

Az előbbi nyelvtanban a *bcbaa* szónak két, egymástól különböző legbaloldalibb levezetése van:
$S \Longrightarrow bA \Longrightarrow bcS \Longrightarrow bcbAS \Longrightarrow bcbaS \Longrightarrow bcbaa$,
$S \Longrightarrow bAS \Longrightarrow bcSS \Longrightarrow bcbAS \Longrightarrow bcbaS \Longrightarrow bcbaa$.

20.26. definíció. *Egy G környezetfüggetlen nyelvtant* ***többértelműnek*** *nevezünk, ha létezik $L(G)$-ben olyan szó, amelynek létezik két különböző legbaloldalibb levezetése.*

Az előbbi G nyelvtan többértelmű, mert a *bcbaa* szónak találtunk két legbaloldalibb levezetését. Ha egy nyelvtan nem többértelmű, akkor ***egyértelműnek*** mondjuk. Egy egyértelmű nyelvtan tetszőleges levezetési fájához csak egy legbaloldalibb levezetés rendelhető.

Egy nyelvet több nyelvtan is generálhat, és ezek között lehetnek egyértelműek és többértelműek is.

20.29. példa. Vizsgáljuk meg a következő két nyelvtant.
A $G_1 = (\{S\}, \{a, +, *\}, \{S \to S + S \mid S * S \mid a\}, S)$ többértelmű, mert
$S \Longrightarrow S + S \Longrightarrow a + S \Longrightarrow a + S * S \Longrightarrow a + a * S \Longrightarrow a + a * S + S$
$\Longrightarrow a + a * a + S \Longrightarrow a + a * a + a$ és
$S \Longrightarrow S * S \Longrightarrow S + S * S \Longrightarrow a + S * S \Longrightarrow a + a * S \Longrightarrow +a * S + S$
$\Longrightarrow a + a * a + S \Longrightarrow a + a * a + a$.
A $G_2 = (\{S, A\}, \{a, *, +\}, \{S \to A + S \mid A,\ A \to A * A \mid a\}, S)$ nyelvtan egyértelmű. Be lehet bizonyítani, hogy $L(G_1) = L(G_2)$.

20.27. definíció. *Egy környezetfüggetlen nyelv* ***örökletesen többértelmű****, ha nem létezik egyetlen egyértelmű nyelvtan sem, amely generálja.*

A környezetfüggetlen nyelvtan értelmezése szerint megengedettek az $A \to B$ $(A, B \in N)$ alakú szabályok is. Az ilyen szabályokat **átnevezéseknek** nevezzük. Minden olyan környezetfüggetlen nyelvtanhoz, amely tartalmaz átnevezéseket, hozzárendelhetünk egy vele ekvivalens, de átnevezéseket nem tartalmazó nyelvtant.

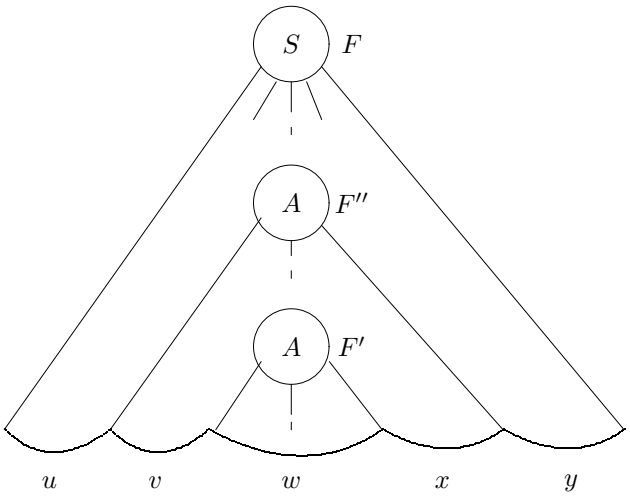
Ha $G = (N, T, P, S)$ átnevezéseket tartalmazó környezetfüggetlen nyelvtan, akkor definiáljuk a vele ekvivalens, de átnevezéseket nem tartalmazó $G' = (N, T, P', S)$ nyelvtant.

P' **értelmezése:** P' tartalmazza P-nek mindazon szabályait, amelyek nem átnevezések. Ezenkívül, minden $A \overset{+}{\underset{G}{\Longrightarrow}} B$ $(A, B \in N)$ levezetésre, amelynek hossza nem nagyobb, mint $|N|$ (azaz a nemterminálisok száma), tegyük be P'-be az $A \to \alpha$ szabályt, ha P-ben létezik a $B \to \alpha$ szabály (ha $\alpha \notin N$). Könnyű belátni, hogy a két nyelvtan által generált nyelv azonos.

20.3.3. Iterációs lemma környezetfüggetlen nyelvekre

Környezetfüggetlen nyelvekre is létezik egy Bar-Hillel-szerű lemma.

20.28. tétel (Iterációs lemma). *Tetszőleges környezetfüggetlen nyelvhez létezik*



20.36. ábra. Levezetési fa az iterációs lemma bizonyításához.

olyan n természetes szám (amely csak az adott nyelvtől függ) úgy, hogy a nyelv minden n-nél hosszabb α szavát fel lehet írni $uvwxy$ alakban, és igazak a következők:

1) $|w| \geq 1$,
2) $|vx| \geq 1$,
3) $|vwx| \leq n$,
4) uv^iwx^iy *is eleme a nyelvnek, minden* $i \geq 0$ *értékre.*

Bizonyítás. Legyen $G = (N, T, P, S)$, amelyben nincsenek átnevezések. Legyen $m = |N|$‘, a nemterminális jelek száma, és legyen ℓ a P-beli szabályok jobb oldalainak maximális hossza. Legyen $n = \ell^{m+1}$ és $\alpha \in L(G)$ úgy, hogy $|\alpha| > n$. Ekkor létezik egy olyan levezetési fa (20.36. ábra), amelynek eredménye pontosan α. Ennek a F fának a magassága h (a szintek száma $h + 1$), és mivel minden szinten legfeljebb ℓ csúcs van, a legalsó szinten nincs több mint ℓ^h betű. Fehát $|\alpha| \leq \ell^h$. De mivel $|\alpha| > \ell^{m+1}$, azt kapjuk, hogy: $h > m + 1$. Ebből következik, hogy van olyan út a fában a gyökértől egy levélig, amelyben több mint $m+1$ csúcs van, és az utolsót kivéve mindegyik nemterminális betűvel van címkézve. Mivel azonban a nyelvtannak csak m nemterminális jele van, ebben az útban legalább két nemterminális betű azonos. Legyen ez a betű az A.

Keressük meg A első előfordulását az úton alulról fölfelé haladva, és jelöljük F'-tel azt részfát, amelynek ez az A a gyökere. Hasonlóképpen az A következő előfordulása által adott részfa legyen F''. A levezetési fa és annak $uvwxy$ eredménye a 20.36. ábrán látható. A F' eredménye w, a F'' eredménye vwx, míg F eredménye $uvwxy = \alpha$.

Mivel G-ben nincsenek átnevezések, minden belső csúcsból legalább két él fut ki, ezért $|w| \geq 1$ és $|vx| \geq 1$. Mivel a F'' fa magassága legfeljebb $m + 1$, és a vizsgált út F''-be eső része különböző jeleket tartalmaz (a gyökeret nem számítva), következik, hogy $|vwx| \leq \ell^{m+1} = n$. Ha F-ből eltávolítjuk a F'' csúcsait, csak a gyökeret hagyva

meg, akkor az így kapott fa eredménye uAy, azaz
$$S \overset{*}{\underset{G}{\Longrightarrow}} uAy.$$
Hasonlóképpen kapjuk F''-ben, hogy $A \overset{*}{\underset{G}{\Longrightarrow}} vAx$, és F'-ben, hogy $A \overset{*}{\underset{G}{\Longrightarrow}} w$.

Tehát $S \overset{*}{\underset{G}{\Longrightarrow}} uAy$, $A \overset{*}{\underset{G}{\Longrightarrow}} vAx$, $A \overset{*}{\underset{G}{\Longrightarrow}} w$.

Innen
$$S \overset{*}{\underset{G}{\Longrightarrow}} uAy \overset{*}{\underset{G}{\Longrightarrow}} uwy$$
és
$$S \overset{*}{\underset{G}{\Longrightarrow}} uAy \overset{*}{\underset{G}{\Longrightarrow}} uvAxy \overset{*}{\underset{G}{\Longrightarrow}} \ldots \overset{*}{\underset{G}{\Longrightarrow}} uv^iAx^iy \overset{*}{\underset{G}{\Longrightarrow}} uv^iwx^iy$$
tetszőleges $i \geq 1$ értékre. Tehát $S \overset{*}{\underset{G}{\Longrightarrow}} uv^iwx^iy$ tetszőleges $i \geq 0$-ra. ■

Bemutatjuk az iterációs lemma két következményét.

20.29. következmény. $\mathcal{L}_2 \subset \mathcal{L}_1$.

Bizonyítás. A következmény azt állítja, hogy létezik olyan környezetfüggő nyelv, amely nem környezetfüggetlen, tehát a tartalmazás szigorú. Ennek bizonyításához, elég ha találunk egy olyan környezetfüggő nyelvet, amelyre az előbbi iterációs lemma nem teljesül. Legyen ez
$$L = \{a^mb^mc^m \mid m \geq 1\}.$$
Legyen n a lemmabeli érték, és legyen $\alpha = a^nb^nc^n$. Nyilvánvaló, hogy $|\alpha| = 3n > n$. Bebizonyítjuk, hogy α egyetlen felbontására sem igaz a lemma minden állítása. Legyen $\alpha = a^nb^nc^n = uvwxy$, ahol $|vx| \geq 1$, tehát v és x legalább egyik nem ε.

Ha v és x szavak legalább egyike egynél többféle betűt tartalmaz, akkor létezik olyan i, hogy uv^iwx^iy-ban felbomlik a betűk sorrendje és egymásmellettisége.

Ha mindkettő csak egyféle betűt tartalmaz, akkor a lemmabeli $i = 0$-ra $uwy \notin L$, mert különbözik a betűk hatványa.

Hogy a fenti nyelv környezetfüggő, azt könnyen lehet igazolni, megfelelő nyelvtan megadásával. A 20.3. példában megadott két nyelvtan mindegyike kiterjesztett környezetfüggő nyelvtan. De tudjuk, hogy tetszőleges i típusú kiterjesztett nyelvtanhoz mindig hozzárendelhető ugyanolyan típusú és vele ekvivalens nyelvtan. ■

20.30. következmény. *Léteznek olyan környezetfüggetlen nyelvek, amelyeknek metszete nem környezetfüggetlen.*

Bizonyítás. Legyen $G_1 = (N, T, P_1, S)$ és $G_2 = (N, T, P_2, S)$, ahol $N = \{S, A, B\}$, $T = \{a, b, c\}$ és

P_1: $S \to AB$, $A \to aAb \mid ab$, $B \to cB \mid c$.

P_2: $S \to AB$, $A \to Aa \mid a$, $B \to bBc \mid bc$.

$L(G_1) = \{a^nb^nc^m \mid n \geq 1, m \geq 1\}$, és $L(G_2) = \{a^nb^mc^m \mid n \geq 1, m \geq 1\}$, tehát mindkét nyelv környezetfüggetlen. De
$$L(G_1) \cap L(G_2) = \{a^nb^nc^n \mid n \geq 1\}$$

nem környezetfüggetlen, amint azt az iterációs lemmával bizonyítottuk. ■

Chomsky-féle normálalak

20.31. definíció. *Egy* $G = (N, T, P, S)$ *környezetfüggetlen nyelvtan Chomsky-féle normálalakban van megadva, ha minden szabálya* $A \to a$ *vagy* $A \to BC$ *alakú, ahol* $A, B, C \in N,\ a \in T$.

20.30. példa. A $G = (\{S, A, B, C\}, \{a, b\}, \{S \to AB \mid CB,\ C \to AS,\ A \to a,\ B \to b\}, S)$ nyelvtan Chomsky-féle normálalakú és $L(G) = \{a^n b^n \mid n \geq 1\}$.

Minden ε-mentes környezetfüggetlen nyelvtanhoz megadható egy vele ekvivalens Chomsky-féle normálalakba írt nyelvtan. Erre adunk egy algoritmust.

CHOMSKY-ALAK

1 kiküszöböljük a szabályokban az átnevezéseket.
2 a szabályokban minden a_i terminális jelet helyettesítünk egy új A_i nemterminálissal, és betesszük a szabályok közé az $A_i \to a_i$ új szabályokat is.
3 minden $B \to A_1 A_2 \dots A_k$ alakú szabályt, ahol minden $A_i \in N$ és $k \geq 3$, helyettesítünk a következőkkel:
$$\begin{aligned} B &\to A_1 C_1, \\ C_1 &\to A_2 C_2, \\ &\dots \\ C_{k-3} &\to A_{k-2} C_{k-2}, \\ C_{k-2} &\to A_{k-1} A_k, \end{aligned}$$
ahol minden C_i új nemterminális szimbólum.

20.31. példa. Legyen $G_1 = (\{S\}, \{a, b\}, \{S \to ab \mid aSb\}, S)$. Könnyű belátni, hogy $L(G_1) = \{a^n b^n \mid n \geq 1\}$. Alakítsuk át ezt a nyelvtant, hogy Chomsky-féle normálalakú legyen! Az algoritmus lépései:

1. Nincsenek átnevezések.
2. $S \to AB,\ \ S \to ASB,\ \ A \to a,\ \ B \to b$
3. $S \to ASB$ helyett betesszük a szabályok közé a következőket:
   $S \to AC$,
   $C \to SB$

Tehát, az új nyelvtan
$G_2 = (\{S, A, B, C\}, \{a, b\}, \{S \to AB \mid AC, C \to SB, A \to a, B \to b\}, S)$
és $L(G_1) = L(G_2)$.

Greibach-féle normálalak

20.32. definíció. *Egy* $G = (N, T, P, S)$ *környezetfüggetlen nyelvtan Greibach-féle normálalakban van megadva, ha minden szabálya* $A \to aW$ *alakú, ahol* $A \in N$,

$a \in T,\ W \in N^*$.

20.32. példa. A $G = (\{S, B\}, \{a, b\}, \{S \to aB \mid aSB,\ B \to b\}, S)$ nyelvtan Greibach-alakú. $L(G) = \{a^n b^n \mid n \geq 1\}$

Minden ε-mentes környezetfüggetlen nyelvhez megadható egy vele ekvivalens Greibach-féle normálalakba írt nyelvtan.

GREIBACH-ALAK

1 a Chomsky-féle normálalakból indulunk el. A nemterminális jelek: $A_1, A_2, \ldots, A_n$. Az $A_i \to A_j A_k$ alakú szabályokat kell átalakítani.
2 ha $A_i \to A_j A_k$ és létezik $A_j \to a$ ($a \in T$) szabály úgy, hogy A_j csak ilyen alakú szabályok bal oldalán fordul elő, akkor az $A_i \to A_j A_k$ szabályt helyettesítjük az $A_i \to aA_k$ szabállyal.
3 az $A \to AX_1, \quad \ldots, \quad A \to AX_r, \quad$ ahol $\quad X_i \in V^+,\ i = 1, 2, \ldots, r$
$A \to Y_1, \quad \ldots, \quad A \to Y_s, \quad$ ahol $\quad Y_i \in TV^*$
alakú szabályok helyett a következőket használjuk:
$A \to Y_i \quad 1 \leq i \leq s,$
$A \to Y_j B, \quad 1 \leq j \leq s,$
$B \to X_i, \quad 1 \leq i \leq r,$
$B \to X_i B, \quad 1 \leq i \leq r,$
ahol B egy új nemterminális.
4 ha már minden szabály
$A_i \to A_j X,$
$A_i \to aX,\ a \in T, X \in N^*,$
$B_i \to X,\ X \in N'^*$ (ahol N' tartalmazza az összes változót),
alakú, akkor legyen k az a legnagyobb szám, amelyre teljesül, hogy minden k-nál kisebb i-re igaz, hogy minden $A_i \to A_j X$ szabályra $i < j$.
Ha létezik $A_k \to A_j X$, ahol $k \geq j$, akkor $A_j \to Y_h$ alakú szabályokat használunk helyettesítésre.
Ezt többször megismételve, minden szabály $A_k \to A_j X$ alakú lesz, ahol $k \leq j$.
Egyenlőség esetén alkalmazzuk a 3. lépést.
5 ha még marad olyan szabály, amely nemterminálissal kezdődik, akkor ezeknél helyettesítéseket végzünk.

20.33. példa. A $G_1 = (\{S, A, B, C, D\}, \{0, 1\}, P_1, S)$ nyelvtan P_1 szabályai:
$S \to AA,\ S \to BB,\ S \to CA,\ S \to DB,\ C \to AS,\ D \to BS$
$A \to 0,\ B \to 1,$
amelyik már Chomsky-alakú. Legyen a betűk sorrendje S, A, B, C, D (azaz $A_1 = S$, $A_2 = A$, $A_3 = B$, $A_4 = C$, $A_5 = D$).
Ekkor, az 2. lépés alapján:
$S \to 0A$
$C \to 0S$
$S \to 1B$
$D \to 1S$
A 4. lépés alapján

$S \to CA$, helyettesítés után $S \to 0SA$.
$S \to DB$, helyettesítés után $S \to 1SB$.
Tehát Greibach-féle alakban: $G_2 = (\{S, A, B\}, \{0, 1\}, P_2, S)$, ahol P_2 elemei:
$S \to 0A \mid 1B \mid 0SA \mid 1SB$
$A \to 0,\ B \to 1.$
$L(G_1) = L(G_2) = \{uu^{-1} \mid u \in \{0, 1\}\}$.

20.34. példa. Nézzünk meg egy kicsivel bonyolultabb példát! Megadunk egy nyelvtant a következő nyelv generálására.

$$L = \left\{a^n b^k c^{n+k} \mid n \geq 0, k \geq 0, n + k > 0.\right\}$$

Bebizonyítható, hogy a következő nyelvtan generálja L-et.

$$G = \left\{\{S, R\}, \{a, b, c\}, \{S \to aSc \mid ac \mid R, R \to bRc \mid bc\}, S\right\}$$

Először kiküszöböljük az átnevezéseket (itt most csupán egy van), azután megadunk egy vele ekvivalens Chomsky-alakú nyelvtant, majd egy ezzel ekvivalens Greibach-alakút. Az $S \to R$ átnevezés kiküszöbölése után a következő szabályokot kapjuk:
$S \to aSc \mid ac \mid bRc \mid bc$
$R \to bRc \mid bc.$
Bevezetjük az $A \to a, B \to b, C \to c$ szabályokat, majd a terminálisokat helyettesítjük minden szabály jobb oldalán a megfelelő változóval:
$S \to ASC \mid AC \mid BRC \mid BC,$
$R \to BRC \mid BC,$
$A \to a,\ B \to b,\ C \to c.$
Két új változó (D, E) bevezetése után:
$S \to AD \mid AC \mid BE \mid BC,$
$D \to SC,$
$E \to RC,$
$R \to BE \mid BC,$
$A \to a,\ B \to b,\ C \to c.$
Ez már Chomsky-féle normálalak. Induljunk ki ebből a nyelvtanból, miután átírjuk a változókat A_i alakúra, hogy könnyebben alkalmazhassuk az algoritmust. Tehát, a következő átnevezés után
S helyett A_1, A helyett A_2, B helyett A_3, C helyett A_4, D helyett A_5,
E helyett A_6, R helyett A_7,
nyelvtanunk a következő szabályokat tartalmazza:
$A_1 \to A_2A_5 \mid A_2A_4 \mid A_3A_6 \mid A_3A_4,$
$A_2 \to a,\ A_3 \to b,\ A_4 \to c,$
$A_5 \to A_1A_4,$
$A_6 \to A_7A_4,$
$A_7 \to A_7A_6 \mid A_3A_4.$
Az algoritmus első lépésének többszöri alkalmazása után:
$A_1 \to aA_1A_4 \mid aA_4 \mid bA_7A_4 \mid bA_4,$
$A_4 \to c,$
$A_7 \to A_7A_7A_4 \mid bA_4.$
Az utolsó sor szabályaira alkalmazzuk az algoritmus 2. lépését, és helyette a következőt kapjuk:
$A_7 \to bA_4, \mid A_7 \to bA_4B,$

$B \to A_7A_4 \mid B \to A_7A_4B.$

A_7 behelyettesítése után a Greibach-féle normálalak a következő:

$A_1 \to aA_1A_4 \mid aA_4 \mid bA_7A_4 \mid bA_4,$
$A_4 \to c,$
$A_7 \to bA_4, \mid bA_4B,$
$B \to bA_4A_4 \mid bA_4BA_4.$
$B \to bA_4A_4B \mid bA_4BA_4B.$

Gyakorlatok

20.3-1. Adjunk meg egy-egy veremautomatát a következő nyelvek felismerésére:

$L_1 = \{a^ncb^n \mid n \geq 0\},$
$L_2 = \{a^nb^{2n} \mid n \geq 1\},$
$L_3 = \{a^{2n}b^n \mid n \geq 0\} \cup \{a^nb^{2n} \mid n \geq 0\},$
$L_4 = \{u \in \{0,1\}^* \mid n_0(u) = n_1(u)\}.$

20.3-2. Adjunk meg egy olyan környezetfüggetlen nyelvtant, amely az $L = \{a^nb^nc^m \mid n \geq 0, m \geq 0\}$ nyelvet generálja, majd írjuk át Chomsky-, illetve Greibach-féle normálalakúvá. Adjunk meg egy veremautomatát, amely felismeri az L nyelvet.

20.3-3. Ugyanaz a feladat az $L = \{a^nb^nc^m \mid n > m > 0\}$ nyelv esetében.

20.3-4. Milyen nyelvet generálnak a következő környezetfüggetlen nyelvtanok?

$G_1 = (\{S\}, \{a,b\}, \{S \to SSa \mid b\}, S),$ $G_2 = (\{S\}, \{a,b\}, \{S \to SaS \mid b\}, S)$

20.3-5. Adjunk meg egy környezetfüggetlen nyelvtant, amely olyan szavakat generál, amelyben egyenlő számban vannak az a és b betűk.

20.3-6. Bizonyítsuk be az iterációs lemma alkalmazásával, hogy az a nyelv, amelynek minden szava ugyanannyi a, b és c betűt tartalmaz, nem környezetfüggetlen.

20.3-7. Adott a következő nyelvtan: $G = (V, T, P, S)$, ahol

$V = \{S\},$
$T = \{$*if, then, else, a, c*$\},$
$P = \{S \to$ *if a then* $S, \quad S \to$ *if a then* S *else* $S, \quad S \to c,$

Mutassuk meg, hogy az *if a then if a then c else c* szónak létezik két különböző legbaloldalibb levezetése.

20.3-8. Bizonyítsuk be, hogy ha L környezetfüggetlen, akkor $L^{-1} = \{u^{-1} \mid u \in L\}$ is környezetfüggetlen.

Feladatok

20-1. Átalakító véges automaták

Értelmezzünk olyan véges automatát, amelyeknek kimeneti szalagja is van, és amely reguláris nyelvet reguláris nyelvvé alakít át.

20-2. Átalakító veremautomaták

Értelmezzünk olyan veremautomatát, amelyeknek kimeneti szalagja is van, és amely környezetfüggetlen nyelvet környezetfüggetlen nyelvvé alakít át.

Megjegyzések a fejezethez

A véges automata definíciójában eltértünk a hagyományos értelmezéstől, az átmenetfüggvény helyett az átmenetgráfot használtuk. Ezt a szemléletmódot követtük a veremautomata esetében is, amely hasznosnak bizonyult sok esetben, nagyban egyszerűsítvén a bizonyításokat.

Az automatákról és a formális nyelvekről sok klasszikus könyv létezik. Ezek közül megemlítjük a következőket: Aho és Ullman két könyve [1][2] 1972-ből és 1973-ból, Gécseg Ferenc és Peák István [5] angol nyelvű könyve 1972-ből, Salomaa két könyve [12][13] 1969-ból és 1973-ból, Hopcroft és Ullmann [7] könyve 1979-ből, Manna [10] könyve, amely 1981-ben magyar fordításban is megjelent. Megemlítjük még Sipser [14] 1997-es könyvét. Lothaire (francia szerzők közös neve) [9] szókombinatorikai könyvében egyéb típusú automatákról is olvashatunk. Giammarresi és Montalbano cikke [6] az általánosított véges automatákkal foglalkozik.

Magyar nyelven is több jegyzet és könyv tárgyalja az automaták és formális nyelvek témáját. Megemlítjük ezek közül Révész György [11] könyvét 1979-ből, a Demetrovics-Denev-Pavlov szerzőhármas jegyzetét [3] 1985-ből, amely 1999-ben tankönyvként is megjelent, Fülöp Zoltán [4] 1999-es könyvét, valamint Hunyadvári László [8] jegyzetét 1996-ból.

Irodalomjegyzék

[1] A. V. Aho, J.D. Ullman. *The Theory of Parsing, Translation and Compiling Vol. I.* Prentice-Hall, 1972. 662

[2] A. V. Aho, J.D. Ullman. *The Theory of Parsing, Translation and Compiling Vol. II.* Prentice-Hall, 1973. 662

[3] Demetrovics János, Denev J., Pavlov R. *A számítástudomány matematikai alapjai.* Nemzeti Tankönyvkiadó, Budapest, 1999. 662

[4] Fülöp Zoltán. *Formális nyelvek és szintaktikus elemzésük.* Polygon, Szeged, 1999. 662

[5] F. Gécseg, I. Peák. *Algebraic Theory of Automata.* Akadémiai Kiadó, Budapest, 1972. 662

[6] D. Giammarresi, R. Montalbano. Geterministic generalized automata. *Theoretical Computer Science*, 215:191–208, 1999. 662

[7] J.E. Hopcroft, J.D. Ullmann. *Introduction to Automata Theory, Languages and Computation.* Addison-Wesley, 1979. 662

[8] Hunyadvári László, Manhertz Tamás. *Automaták és formális nyelvek.* ELTE, Egyetemi jegyzet, Budapest, 1996. 662

[9] M. Lothaire. *Algebraic Combinatorics on Words.* Cambridge University Press, 2002. 662

[10] Z. Manna. *Mathematical Theory of Computation.* McGraw-Hill Book Co., 1974 (Magyarul: *Programozáselmélet*, Műszaki Könyvkiadó, 1981.). 662

[11] Révész György. *Bevezetés a formális nyelvek elméletébe.* Akadémiai Kiadó, Budapest, 1979. 662

[12] A. Salomaa. *Theory of Automata.* Pergamon Press, 1969. 662

[13] A. Salomaa. *Formal Languages.* Academic Press, 1973. 662

[14] M. Sipser. *Introduction to the Theory of Computation.* PWS Publishing Company, 1997. 662

Tárgymutató

ábécé, 601
állapotok
  elérhetetlen, 617
  produktív, 618

Automatából-regulárisnyelv', *626*
Automatából-regulárisnyelv, *625*
automaták
  ekvivalenciája, 615
  minimalizálása, 630
automata
  ε-lépéses véges, 621
  véges, 612
  veremautomata, 645
Automata-ekvivalencia, *616*
Automata-minimalizálása, *631*

Bar-Hillel-lemma, 632, 655

Chomsky-féle nyelvosztályok, 606
Chomsky-alak, *658*

Elérhető-állapotok, *617*
Epszilon-mentesítés, *622*

grammatika, 603
Greibach-alak, *659*

helyettesítési szabály, 603

iterációs lemma
  környezetfüggetlen nyelvekre, 655
  reguláris nyelvekre, 632

kezdőszimbólum, 603
kiterjesztett nyelvtan, 607
Kleene tétele, 636
konkatenáció, 601

levezetés, 604

műveletek
  nyelvekkel, 602
  reguláris nyelvekkel, 635

Nemdet-det, *621*
normálalak
  Chomsky-féle, 658
  Greibach-féle, 658
nyelv
  hatványa, 602
  iteráltja, 602
  környezetfüggő, 607
  környezetfüggetlen, 607, 645, 653
  komplementuma, 602
  mondatszerkezetű, 607
  0-,1-,2-,3-típusú, 607
  reguláris, 607, 612
  tükrözése, 602
nyelvek
  egyesítése, 602
  különbsége, 602
  megadása, 603
  metszete, 602
  szorzata, 602
nyelvtan, 603
  generatív, 603
  környezetfüggő, 606
  környezetfüggetlen, 606
  mondatstruktúrájú, 606
  0-,1-,2-,3-típusú, 607
  reguláris, 607

produkció, 603
Produktív-állapotok, *618*
pumpáló lemma, 632

reguláris
  kifejezés, 636
  műveletek, 602
  nyelv, 612
Regulárisnyelvből-automata', *628*
Regulárisnyelvből-automata, *627*

szó, 601
  hatványozása, 602

véges automata, 612
  minimalizálása, 630
veremautomata, 645

Névmutató

Aho, Alfred V., 664

Bar-Hillel, Yehoshua, 630

Demetrovics János, 664
Denev, Jordan, 664

Fülöp Zoltán, 664

Gécseg Ferenc, 664
Giammarresi, Dora., 664

Hopcroft, John E., 664
Hunyadvári László, 664

Kleene, Stephen C., 634

Lothaire, M., 664

Manhertz Tamás, 664
Manna, Zohar, 664
Montalbano, Rosa, 664

Pavlov, Radiszlav, 664
Peák István, 664

Révész György, 664

Salomaa, Arto, 664
Sipser, Michael, 664

Ullmann, Jeffrey D., 664

Tartalomjegyzék

20. Automaták és formális nyelvek (Kása Zoltán) **601**
20.1. Nyelvek és nyelvtanok 601
20.2. Véges automaták és reguláris nyelvek 612
20.3. Veremautomaták és környezetfüggetlen nyelvek 645
Irodalomjegyzék **663**
Tárgymutató **664**
Névmutató **665**