Írta:
ÉSIK ZOLTÁN

A SZÁMÍTÁSTUDOMÁNY ALAPJAI

Egyetemi tananyag

TYPOTEX

2011

LEKTORÁLTA: Dr. Dömösi Pál, Debreceni Egyetem Informatikai Kar Számítógéptudományi Tanszék

TÁMOGATÁS:
Készült a TÁMOP-4.1.2-08/1/A-2009-0008 számú, „Tananyagfejlesztés mérnök informatikus, programtervező informatikus és gazdaságinformatikus képzésekhez” című projekt keretében.

ISBN 978 963 279 496 9

KÉSZÜLT: a Typotex Kiadó gondozásában
FELELŐS VEZETŐ: Votisky Zsuzsa
AZ ELEKTRONIKUS KIADÁST ELŐKÉSZÍTETTE: Gerner József

KULCSSZAVAK:
formális nyelv, véges automata, reguláris nyelv, környezetfüggetlen nyelv, veremautomata, Turing-gép, rekurzívan felsorolható nyelv, Church–Turing-tézis, idő- és tárbonyolultság, polinomidőben megoldható problémák, polinomidőben verifikálható problémák, polinom tárral megoldható problémák.

ÖSSZEFOGLALÁS:
A jegyzet egységes rövid bevezetést ad a számítástudomány három területéhez. Ezek az automaták és formális nyelvek elmélete, a kiszámíthatóságelmélet és a bonyolultságelmélet.

Tartalomjegyzék

Bevezetés

A jegyzet célja az, hogy egységes rövid bevezetést adjon a számítástudomány három területéhez. Ezek az automaták és formális nyelvek elmélete, a kiszámíthatóságelmélet és a bonyolultságelmélet.

Az automaták és formális nyelvek elmélete az 1950-es évekre nyúlik vissza és mind a mai napig a számítástudomány egyik alappillérének tekinthető. A jegyzetben a véges automaták viselkedését jellemezzük a reguláris kifejezések és a jobblineáris nyelvtanok segítségével. Az egyik fő eredmény Kleene klasszikus tétele, mely szerint egy nyelv akkor és csak akkor ismerhető fel véges automatával, ha megadható reguláris kifejezéssel. A véges automaták tárgyalását a környezetfüggetlen nyelvek majd a Chomsky-féle hierarchia tárgyalása követi. Itt az egyik fő eredmény a környezetfüggetlen nyelvtanok és veremautomaták ekvivalenciája.

A veremautomatát tekinthetjük a (nemdeterminisztikus) Turing-gép speciális eseteként. A Turing-gép bevezetésére egészen általános okból került sor az 1930-as években. Ma már teljesen elfogadott az a tézis, hogy a Turing-gépet (és a vele ekvivalens számos más modellt) tekinthetjük az algoritmus fogalom matematikai megfelelőjének. Egy feladat, probléma akkor oldható meg algoritmikusan, ha megoldható Turing-géppel. Ismertetjük a Turing-gépekhez kapcsolódó alapfogalmakat és a Turing-gépeken alapuló kiszámíthatóságelmélet néhány alapvető eredményét. Több példát adunk Turing-géppel, és így algoritmuikusan megoldhatatlan feladatra is.

A bonyolultságelmélet a kiszámíthatóságelmélet kiterjesztése. Azt vizsgálja, hogy hogyan lehet osztályozni az algoritmikusan megoldható problémákat, feladatokat a megoldásukhoz szükséges erőforrások mennyisége szerint. Ismertetjük a bonyolultságelmélet néhány alapfogalmát, és részletesebben foglalkozunk az **P**, **NP** és **PSPACE** bonyolultsági osztályokkal.

A jegyzet azokat az ismereteket öleli fel, amelyet a Szegedi Tudományegyetem műszaki informatikai alapképzésben az Számítástudomány alapjai c. tárgy oktatásában 2005-től helyet kaptak. Számos olyan anyagrész kimaradt a jegyzetből, amely egy önálló automataelméleti bevezető kurzusban, vagy egy önálló kiszámíthatóságelméleti vagy bonyolultságelméleti bevezető kurzusban helyet szokott kapni. Törekedtünk arra, hogy a jegyzet anyaga egy félévben heti két óra előadással leadható legyen. További kiegészítő ismeretek tárgyalása megtalálható az irodalomjegyzékben felsorolt könyvekben és jegyzetekben, magyarul vagy idegen nyelven.

Köszönetemet fejezem ki Iván Szabolcsnak és Gazdag Zsoltnak a jegyzet ekészítésében nyújtott segítségükért.

Szeged, 2011. június.

Ésik Zoltán

1. fejezet

Véges automaták és reguláris nyelvek

1.1. Szavak és nyelvek

Legyen Σ véges nemüres halmaz, melyet *abc*-nek, és elemeit *betű*knek nevezzük. *(Σ-feletti) w szó*n a Σ betűiből képzett véges sorozatot értünk:

$$w = w_1 \dots w_n, \quad w_i \in \Sigma, \quad i = 1, \dots, n.$$

Az n nemnegatív egész számot a *w szó hosszának* nevezzük, ennek jelölése $|w|$. A 0 hosszúságú szót ε-nal je]öljük és *üres szó*nak nevezzük. A Σ-feletti szavak halmazát Σ^* jelöli. Példaként legyen $\Sigma = \{0,1\}$, ekkor $w_1 = 01001$ és $w_2 = 000 = 0^3$ szavak, továbbá $|w_1| = 5$ és $|w_2| = 3$.

Szavakon több művelet és reláció értelmezhető. A *konkatenáció* vagy *szorzás* művelete az u, v szavakhoz az $u \cdot v = uv$ szót rendeli, melyet úgy kapunk, hogy az u után leírjuk a v-t. A szavak Σ^* halmaza a konkatenáció műveletével és az üres szóval (szabad) *monoidot* alkot:

$$\begin{aligned} u(vw) &= (uv)w \\ u\varepsilon &= u = \varepsilon u \end{aligned}$$

minden u, v, w szóra.

Egy $w = w_1 \dots w_n \in \Sigma^*$ szó *tükörképe* a $w_n \dots w_1$ szó, ahol $w_1, \dots, w_n \in \Sigma$. Ennek jelölése w^{-1}. Világos, hogy $\varepsilon^{-1} = \varepsilon$ és $(uv)^{-1} = v^{-1}u^{-1}$. Azt mondjuk, hogy u a v szó *kezdőszelete* ha létezik olyan w szó, amelyre $v = uw$. Ha még $u \neq v$ is teljesül, akkor u a v *valódi kezdőszelete*. A *zárószelet* és a *valódi zárószelet* fogalmát hasonlóan definiáljuk. Nyilvánvalóan u akkor és csak akkor a v (valódi) kezdőszelete, ha u^{-1} a v^{-1} (valódi) zárószelete. Végül azt mondjuk, hogy az u szó a v *rész-szava*, ha léteznek olyan x, y szavak, hogy $v = xuy$. Ha $u \neq v$ is teljesül, akkor u a v valódi rész-szava.

Nyelven szavak halmazát értjük. Pontosabban, egy Σ-feletti L nyelv a Σ^* egy részhalmaza. Véges nyelvekre példák a $\{0,1\}$ *bináris abc* felett a $\{0, 01, 001\}$, $\{\varepsilon\}$, $\{\varepsilon, 10\}$ és $\emptyset$ halmazok. A későbbiekben számos olyan módszert ismerünk majd meg, amellyel végtelen nyelvek is megadhatóak. Most a nyelvet alkotó szavak tulajdonságaival adunk meg néhány végtelen nyelvet: $\{0^n 1^m : n, m \geq 0\}$, $\{0^n 1^n : n \geq 0\}$, $\{w \in \Sigma^* : |w|_0 = |w|_1\}$, $\{0^n 1^m 0^{n+m} : n, m \geq 0\}$, $\{0^p : p \text{ prímszám}\}$. Itt $|w|_0$ a w-ben előforduló 0-ák számát jelöli.

Adott Σ abc-re jelölje $P(\Sigma^*)$ az összes Σ-feletti nyelvek nem megszámlálható halmazát. Természetes módon értelmezhetjük nyelveken a halmazelméleti *egyesítés* ($L_1 \cup L_2$), *metszet* ($L_1 \cap L_2$) és *komplementerképzés* $\overline{L}$ műveleteket, melyek a jól ismert *Boole-algebra* azonosságoknak tesznek eleget. A Σ-feletti nyelvek tetszőleges halmazának is képezhetjük egyesítését és metszetét. Nyelveken is definiáljuk a konkatenáció (vagy szorzás) és tükörkép képzésének műveletét:

$$L_1 \cdot L_2 = L_1 L_2 = \{uv : u \in L_1,\ v \in L_2\} \quad \text{és} \quad L^{-1} = \{u^{-1} : u \in L\}$$

tetszőleges $L, L_1, L_2 \subseteq \Sigma^*$ nyelvekre. Például $\{01, 10\} \cdot \{00, 11\} = \{0100, 0111, 1000, 1011\}$.

A $P(\Sigma^*)$ halmaz a konkatenáció műveletével és a $\{\varepsilon\}$ nyelvvel monoidot alkot, és érvényes az $(L_1 L_2)^{-1} = L_2^{-1} L_1^{-1}$ azonosság is. Egy $L \subseteq \Sigma^*$ nyelv kezdőszeleteinek $\mathrm{pre}(L)$ halmaza az L szavainak kezdőszeleteiből áll: $\mathrm{pre}(L) = \{u : \exists v \in L\ u \in \mathrm{pre}(v)\}$. A zárószeletek $\mathrm{suf}(L)$ és a rész-szavak $\mathrm{sub}(L)$ nyelvét hasonlóan definiáljuk. Például $\mathrm{pre}(\{010, 01\}) = \{\varepsilon, 0, 01, 010\}$.

Egy további fontos művelet *Kleene iteráció*, amely az $L \subseteq \Sigma^*$ nyelvhez az $L^* = \bigcup_{n \geq 0} L^n$ nyelvet rendeli, ahol L^n az L nyelv önmagával vett n-szeres szorzata: $L^0 = \{\varepsilon\}$, $L^{n+1} = LL^n = L^n L$, ha $n \geq 0$. Tehát

$$L^* = \{w_1 \dots w_n : n \geq 0,\ w_1, \dots, w_n \in L\}.$$

Az L nyelv *pozitív iteráltja* az $L^+ = LL^* = L^* L = \bigcup_{n \geq 1} L^n$ nyelv. Vegyük észre, hogy L^* mindig tartalmazza az üres szót, míg L^+ csak akkor, ha az üres szó benne van L-ben. Továbbá $L^* = L^+ \cup \{\varepsilon\}$. Például $\emptyset^* = \{\varepsilon\}$, $\{\varepsilon\}^* = \{\varepsilon\}^+ = \{\varepsilon\}$, $\{01\}^* = \{(01)^n : n \geq 0\}$, $\{01\}^+ = \{(01)^n : n \geq 1\}$. A $\{00, 01, 10, 11\}^*$ nyelv az összes olyan bináris szóból áll, amely hossza páros.

A teljesség igénye nélkül megemlítünk néhány alapvető azonosságot.

$$\begin{aligned}
L(L_1 \cup L_2) &= LL_1 \cup LL_2 \\
(L_1 \cup L_2)L &= L_1 L \cup L_2 L \\
(L_1 \cup L_2)^{-1} &= L_1^{-1} \cup L_2^{-1} \\
\mathrm{pre}(L_1 \cup L_2) &= \mathrm{pre}(L_1) \cup \mathrm{pre}(L_2) \\
\mathrm{pre}(L_1 L_2) &= \mathrm{pre}(L_1) L_2 \cup L_1 \mathrm{pre}(L_2) \\
L^* &= LL^* \cup \{\varepsilon\} \\
(L_1 L_2)^* L_1 &= L_1 (L_2 L_1)^* \\
(L_1 \cup L_2)^* &= (L_1^* L_2)^* L_1^* \\
(L_1 \cup L_2)^* &= (L_1^* L_2^*)^* \\
L^* &= (L^n)^* (\{\varepsilon\} \cup L \cup \ldots \cup L^{n-1}) \quad (n \geq 2).
\end{aligned}$$

Amint azt már a fentiekben is láttuk, a műveletek újabb hatékony eszközt adnak nyelvek megadására. Legyen $\Sigma = \{a, b, c\}$. Ekkor $\overline{\Sigma^* abc \Sigma^*}$ az összes olyan Σ-feletti szavakól áll, amelyeknek nem rész-szava az abc szó. Az $(\{a, b\}^* c \{a, b\}^* c)^* \{a, b\}^*$ nyelv azon Σ-feletti szavakból áll, amelyekben c párosan fordul elő.

1.2. Véges determinisztikus automaták

Véges automatákkal olyan véges rendszereket írhatunk le, melyeknek véges sok állapotuk van, és amelyek diszkrét lépésekben állapotot válthatnak egy véges nemüres halmazból kikerülő bemenő jel hatására. Az átmeneteket kiterjeszthetjük bemenő jelekről bemenő szavakra. Kezdetben az automata egy kitüntetett kezdőállapotban van. Az automata viselkedését azon szavak alkotják, amelyek hatására a kezdőállapotból valamely kitüntetett végállapotba kerülhet. Annak megfelelően, hogy adott állapotból adott jel hatására egy vagy több állapotba mehet át közvetlenül az automata, determinisztikus és nemdeterminisztikus automatát különböztetünk meg. Néha azt is megengedjük, hogy az automata közvetlenül állapotot váltson egy szó, pld. az üres szó hatására.

Véges determinisztikus automata egy olyan

$$M = (Q, \Sigma, \delta, q_0, F)$$

rendszer, ahol

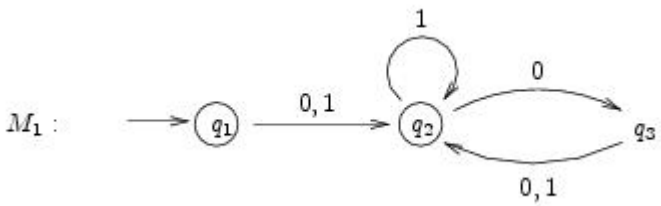
- Q az *állapotok* abc-je,
- Σ a *bemenő jelek* (vagy betűk) abc-je,
- $\delta : Q \times \Sigma \to Q$ az *átmeneti függvény* (vagy *átmenetfüggvény*),
- $q_0 \in Q$ a *kezdőállapot*,
- $F \subseteq Q$ a megkülönböztetett *végállapotok halmaza*.

Amennyiben $\delta(q,a) = q'$, akkor azt mondjuk, hogy az automata a q állapotból átmegy az a hatására a q' állapotba. Mivel az általunk bevezetett determinisztikus modellben δ teljesen definiált függvény, ezért adott állapotból adott jel hatására mindig pontosan egy átmenet van.

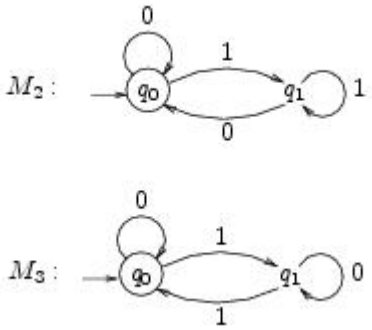
Az $M = (Q, \Sigma, \delta, q_0, F)$ véges determinisztikus automatát ábrázolhatjuk olyan címkézett irányított gráffal, amelynek csúcsai az automata állapotai, élei a bemenő jelekkel címkézettek, és a q, q' állapotokra akkor és csakis akkor létezik a-val címkézett irányított él a q állapotból a q' állapotba, ha $\delta(q,a) = q'$. A kezdőállapotot és a végállapotokat megkülönböztetjük. Pld. legyen az M_1 automatában $Q = \{q_1, q_2, q_3\}$, $\Sigma = \{0,1\}$, ahol q_1 a kezdőállapot és $\{q_1, q_2\}$ a végállapotok halmaza. Legyen δ az alábbi táblázattal adott átmenetfüggvény:

	0	1
q_1	q_2	q_2
q_2	q_3	q_2
q_3	q_2	q_2

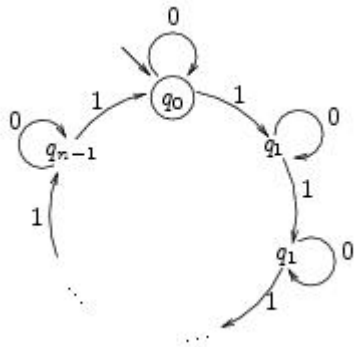
Az automata ábrája:



További véges determinisztikus automaták adottak az alábbi ábrákkal:



Az M_3 egy általánosítása az alábbi automata, ahol $n \geq 2$:



Legyen $M = (Q, \Sigma, \delta, q_0, F)$ véges determinisztikus automata, $q \in Q$, $w = w_1 \dots w_n \in \Sigma^*$ $(w_i \in \Sigma, i = 1, \dots, n)$. Azt mondjuk, hogy az

$$r_0, r_1, \dots, r_n$$

állapotsorozat az M q-ból induló *számítási sorozat*a a w szón, ha

1. $r_0 = q$,
2. $r_i = \delta(r_{i-1}, w_i) \quad i = 1, \dots, n$.

Sikeres (vagy elfogadáshoz vezet) az $r_0, r_1, \dots, r_n$ számítási sorozat, ha $r_n \in F$. Azt mondjuk, hogy M *elfogadja* a w szót, ha létezik a q_0 kezdőállapotból induló sikeres számítási sorozata a w szón. Végül az M által *felismert nyelv*: $L(M) = \{w \in \Sigma^* : M \text{ elfogadja } w\text{-ét}\}$. Két automatát *ekvivalens*nek nevezünk, ha ugyanazt a nyelvet ismerik fel.

Példaként tekintsük az M_1 automatát és az $u = 011011$ szót. A q_1-ből induló számítási sorozat az u szón a $q_1, q_2, q_2, q_2, q_3, q_2, q_2$ sorozat. Mivel q_2 végállapot, ez sikeres, és mivel q_1 a kezdőállapot, ezért $u \in L(M_1)$. Az M_2 által felismert nyelv az üres szóból és az összes olyan $\{0,1\}$-feletti szóból áll, amely 0-ra végződik: $L(M_2) = \{\varepsilon\} \cup \{0,1\}^*\{0\}$. Az M_3 által felismert nyelv az összes olyan $\{0,1\}$-feletti szóból áll, amelyben az 1 páros sokszor fordul elő.

Az M által felismert nyelvet a következő módon is megadhatjuk. Először kiterjesztjük az átmenetfüggvényt egy $\overline{\delta} : Q \times \Sigma^* \to Q$ függvénnyé az alábbi módon:

$$\begin{aligned} \overline{\delta}(q,\varepsilon) &= \varepsilon \\ \overline{\delta}(q,ua) &= \delta(\overline{\delta}(q,u),a), \end{aligned}$$

ahol $q \in Q$, $u \in \Sigma^*$ és $a \in \Sigma$. Mivel $\overline{\delta}(q,a) = \delta(q,a)$ minden $q \in Q$ állapotra és $a \in \Sigma$ betűre, ezért a továbbiakban a $\overline{\delta}$ jelölés helyett csak a δ jelölést használjuk, sőt, $\delta(q,u)$ helyett röviden gyakran csak qu-t írunk. A fenti definícióból világos, hogy $qu = q'$ akkor és csakis akkor teljesül, ha a q állapotból induló számítási sorozat az u szón a q' állapotban ér véget. Így $L(M) = \{u \in \Sigma^* : q_0 u \in F\}$.

*Felismerhető*nek nevezünk egy $L \subseteq \Sigma^*$ nyelvet, ha létezik olyan M véges determinisztikus automata, mely L-et felismeri, azaz amelyre $L = L(M)$. Mivel azok a véges determinisztikus automaták, amelyek *izomorfak*, tehát csak az állapotaik jelölésében különböznek, ugyanazt a nyelvet ismerik fel, ezért csak megszámlálhatóan sok felismerhető nyelv van egy tetszőleges Σ abc felett. A nyelvek „többsége” tehát nem felismerhető.

Példaként tekintsük a $\Sigma = \{0,1\}$ bináris abc-t. Az alábbi Σ-feletti nyelvek felismerhetők:

- $\{w \in \Sigma^* : w$ utolsó betűje $1\} = \{u1 : u \in \Sigma^*\}$,
- $\{w \in \Sigma^* : |w|_1 \equiv 0 \bmod 2\}$,
- $\{w \in \Sigma^* : 00$ és 11 nem fordulnak elő rész-szóként w-ben$\}$.

Később belátjuk majd, hogy a $\{0^n 1^n : n \geq 0\}$ nyelv nem felismerhető.

1.3. Felismerhető nyelvek zártsági tulajdonságai, I

Ebben a részben belátjuk, hogy a felismerhető nyelvek zártak a halmazelméleti egyesítés, metszet, és komplementerképzés műveleteire. A konkatenáció és iteráció műveleteit később vizsgáljuk.

1.1. Tétel. *Legyenek $L, L_1, L_2 \subseteq \Sigma^*$ felismerhető nyelvek. Ekkor $L_1 \cup L_2$, $L_1 \cap L_2$ és $\overline{L}$ felismerhetőek.*

Bizonyítás.

Komplemeterképzés. Legyen $L \subseteq \Sigma^*$ felismerhető nyelv, mondjuk $L = L(M)$, ahol $M = (Q, \Sigma, \delta, q_0, F)$ véges determinisztikus automata. Ekkor tetszőleges $u \in \Sigma^*$ szóra definiált a $q_0 u = \delta(q_0, u)$ állapot, és $q_0 u \in L$ akkor és csak akkor, ha $u \in L$. Legyen F' az F-nek a Q halmazra vett komplementere. Ekkor teszőleges $u \in \Sigma^*$ szóra, $q_0 u \in F'$ akkor és csakis akkor, ha $u \in \overline{L}$, ahol $\overline{L}$ az L nyelvnek a Σ^*-ra vett komplementere. Tehát $\overline{L} = L(\overline{M})$ az $\overline{M} = (Q, \Sigma, \delta, q_0, F')$ véges determinisztikus automatára. Így $\overline{L}$ felismerhető.

Egyesítés és metszet. Legyen $L_i = L(M_i) \subseteq \Sigma^*$ felismerhető nyelv, ahol $M_i = (Q_i, \Sigma, \delta, {}_i q_i, F_i)$, $i = 1,2$, véges determinisztikus automata, $i = 1,2$. Képezzük az alábbi

$$\begin{aligned} M_\cup &= (Q_1 \times Q_2, \Sigma, \delta, (q_1,q_2), F_\cup) \\ M_\cap &= (Q_1 \times Q_2, \Sigma, \delta, (q_1,q_2), F_\cap) \end{aligned}$$

automatákat, ahol csak a végállapot halmazok különböznek és

$$\delta((q_1',q_2'),a) = (\delta_1(q_1',a),\delta_2(q_2',a)) = (q_1'a,q_2'a)$$

minden $(q_1',q_2') \in Q_1 \times Q_2$ és $a \in \Sigma$ esetén. A végállapot halmazok az alábbi összfüggésekkel adottak:

$$\begin{aligned} F_\cup &= F_1 \times Q_2 \cup Q_1 \times F_2 \\ F_\cap &= F_1 \times F_2. \end{aligned}$$

Az $u \in \Sigma^*$ szó hossza szerinti teljes indukcióval könnyen beláthatjuk, hogy

$$(q_1,q_2')u = \delta((q_1',q_2'),u) = (\delta_1(q_1',u),\delta_2(q_2',u)) = (q_1'u,q_2'u)$$

tetszőleges $(q_1',q_2') \in Q_1 \times Q_2$ esetén. Így

$$\begin{aligned} u \in L(M_\cup) &\Leftrightarrow (q_1,q_2)u \in F_\cup \\ &\Leftrightarrow (q_1u,q_2u) \in F_1 \times Q_2 \cup Q_1 \times F_2 \\ &\Leftrightarrow q_1u \in F_1 \text{ vagy } q_2u \in F_2 \\ &\Leftrightarrow u \in L(M_1) \text{ vagy } u \in L(M_2) \\ &\Leftrightarrow u \in L_1 \cup L_2. \end{aligned}$$

Tehát $M_\cup$ az $L_1 \cup L_2$ nyelvet ismeri fel. Ehhez hasonlóan adódik, hogy $M_\cap$ az $L_1 \cap L_2$ nyelvet ismeri fel. □

1.4. Véges nemdeterminisztikus automaták

Véges determinisztikus automaták felhasználásával is beláthatnánk, hogy a felismerhető nyelvek zártak a szorzás és iteráció műveletére. Ugyanakkor egy egyéb vonatkozásban is fontos fogalom felhasználásával erre egyszerűbb bizonyítás adható. Ez a véges nemdeterminisztikus automata fogalma. Valójában ennek is két fajtáját vezetjük be annak megfelelően, hogy megengedjük-e azt, hogy az automata spontán, azaz az üres szó hatására is állapotot váltson.

Véges nemdeterminisztikus spontán átmenetekkel rendelkező automata egy $M = (Q,\Sigma,\delta,q_0,F)$ rendszer, ahol δ kivételével minden komponens ugyanaz, mint véges determinisztikus automata esetén. Legyen $\Sigma_\varepsilon = \Sigma \cup \{\varepsilon\}$. A δ függvény a $Q \times \Sigma_\varepsilon$ halmazt képezi a Q hatványhalmazába, azaz a Q összes részhalmazának $P(Q)$ halmazába:

$$\delta : Q \times \Sigma_\varepsilon \rightarrow P(Q).$$

Amennyiben $\delta(q,a)$ tartalmazza a q' állapotot, ahol $q,q' \in Q$ és $a \in \Sigma_\varepsilon$, akkor azt mondjuk, hoy M a q állapotból az a hatására (közvetlenül) átmehet a q' állapotba. Amennyiben $\delta(q,\varepsilon) = \emptyset$ minden $q \in Q$ állapotra, a δ függvényt felfoghatjuk $Q \times \Sigma \rightarrow P(Q)$ leképezésként is, és ekkor M-et *véges nemdeterminisztikus automatá*nak nevezzük.

Tekintsük az $M = (Q,\Sigma,\delta,q_0,F)$ véges nemdeterminisztikus spontán átmenetekkel rendelkező automatát. Legyenek $q,q' \in Q$ tetszőleges állapotok és $u \in \Sigma^*$. Azt mondjuk, hogy

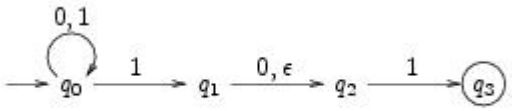
az M az u hatására eljuthat a q-ból a q' állapotba, ha létezik olyan $p_0,\ldots,p_m$ állapotsorozat és az u szónak olyan $u=u_1\ldots u_m$ felbontása, ahol $u_i\in\Sigma_\varepsilon$, $i=1,\ldots,m$, hogy $p_0=q$, $p_m=q'$ és $p_i\in\delta(p_{i-1},u_i)$ minden $i=1,\ldots,m$ esetén. Ha ez fennáll, a $p_0,\ldots,p_m$ sorozatot q-ból induló (és q'-be vezető) számítási sorozatnak nevezzük az u szón (vagy az u szó hatására). A számítási sorozat sikeres, ha $q'\in F$. Az M által felismert $L(M)\subseteq\Sigma^*$ nyelv az összes olyan $u\in\Sigma^*$ szóból áll, amelyre létezik a q_0 kezdőállapotból induló sikeres számítási sorozat.

Vegyük észre, hogy minden véges determinisztikus automata felfogható véges nemdeterminisztikus automataként. De amíg egy véges determinisztikus automata adott állapotból adott jel hatására mindig pontosan egy rákövetkező állapotba mehet át, úgy nemdeterminisztikus esetben esetleg több, vagy egyetlen állapotba sem léphet tovább az automata. Mivel minden véges determinisztikus automata egyben véges nemdeterminisztikus automata is, és a felismert nyelv definíciója nem változik azzal, hogy egy determinisztikus automatát nemdeterminisztikus automataként fogunk fel, minden felismerhető nyelv felismerhető véges nemdeterminisztikus automatával.

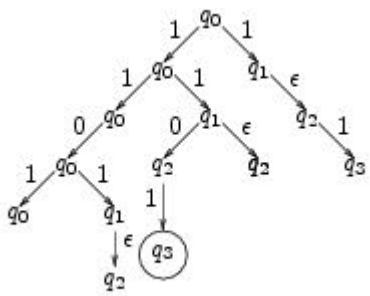
Példaként tekintsük azt a véges nemdeterminisztikus spontán átmenetekkel rendelkező automatát, amelyben $Q=\{q_0,q_1,q_2,q_3\}$, $\Sigma=\{0,1\}$, a kezdőállapot q_0, és q_3 az egyetlen végállapot. A δ átmenetfüggvény az alábbi táblázattal adott:

δ	0	1	ε
q_0	$\{q_0\}$	$\{q_0,q_1\}$	$\emptyset$
q_1	$\{q_2\}$	$\emptyset$	$\{q_2\}$
q_2	$\emptyset$	$\{q_3\}$	$\emptyset$
q_3	$\emptyset$	$\emptyset$	$\emptyset$

Az automatát az alábbi ábra szemlélteti:



Néhány q_0-ból induló számítási sorozatot láthatunk az alábbi ábrán:



Ezek közül az 1101 szóhoz tartozó sorozatok:

- q_0, q_0, q_0, q_0, q_0
- q_0, q_0, q_0, q_0, q_1
- $q_0, q_0, q_0, q_0, q_1, q_2$
- q_0, q_0, q_1, q_2, q_3

A legutolsó sorozat sikeres, tehát 1101 az automata által felismert nyelvben van. Az automata által felismert nyelv az összes olyan bináris szóból áll, amely 11-re vagy 101-re végződik.

Két automatát *ekvivalens*nek nevezünk, ha ugyanazt a nyelvet ismerik fel.

Legyen $M = (Q, \Sigma, \delta, q_0, F)$ spontán átmenetekkel rendelkező véges nemdeterminisztikus automata. A δ függvényt kiterjesztjük egy $\overline{\delta} : Q \times \Sigma^* \to P(Q)$ leképezéssé. Ehhez először tekintsük minden $X \subseteq Q$ halmaz esetén azon s állapotok $\widehat{X}$ halmazát, amelyekre van olyan X-beli q állapotból induló számítási sorozat az ε szón, amely s-ben végződik. Tehát azon s állapotok tartoznak az $\widehat{X}$ halmazhoz, amelyekbe el lehet jutni X-ből ε-nal címkézett élek mentén. Speciálisan $X \subseteq \widehat{X}$. Formálisan,

$$\widehat{X} = \{s \in Q : \exists r_0, r_1, \ldots, r_n, n \geq 0,\ r_0 \in X,\ r_n = s,\ r_i \in \delta(r_{i-1}, \varepsilon),\ i = 1, \ldots n\}.$$

(Az előző példában, ha $X = \{q_0, q_1\}$, akkor $\widehat{X} = \{q_0, q_1.q_2\}$.) A $\widehat{X}$ halmazt az X ε-*lezártjának* nevezzük. Amennyiben M véges nemdeterminisztikus automata (tehát nem rendelkezik egyetlen ε-átmenettel sem), akkor $\widehat{X} = X$ minden $X \subseteq Q$ halmazra. Ezek után a $\overline{\delta} : Q \times \Sigma^* \to Q$ leképezést a következő két szabállyal adjuk meg:

$$\begin{aligned} \overline{\delta}(q, \varepsilon) &= \widehat{X} \quad \text{ahol } X = \{q\} \\ \overline{\delta}(q, ua) &= \widehat{Y} \quad \text{ahol } Y = \bigcup_{s \in \overline{\delta}(q,u)} \delta(s, a) \end{aligned}$$

valahányszor $q \in Q$, $u \in \Sigma^*$ és $a \in \Sigma$. Vegyük észre, hogy amennyiben léteznek spontán átmenetek, nem feltétlenül teljesül a $\overline{\delta}(q, a) = \delta(q, a)$ egyenlőség egy q állapotra és egy $a \in \Sigma$ betűre. (Az előző példában $\overline{\delta}(q_0, \varepsilon) = \{q_0\}$, $\overline{\delta}(q_0, 0) = \{q_0\}$, $\overline{\delta}(q_0, 01) = \{q_1, q_2\}$.) A fenti definícióból világosan adódik, hogy $\overline{\delta}(q, u)$ az összes olyan s állapot halmaza, amelyre létezik az u szó hatására q-ból induló, s-ben végződő számítási sorozat. Ezt a halmazt qu-val is jelöljük. Így

$$L(M) = \{u \in \Sigma^* : q_0 u \cap F \neq \emptyset\}.$$

1.2. Tétel. *Az alábbi állítások ekvivalensek egy $L \subseteq \Sigma^*$ nyelvre.*

1. *L felismerhető.*
2. *L felismerhető véges nemdeterminisztikus automatával.*
3. *L felismerhető véges nemdeterminisztikus spontán átmenetekkel rendelkező automatával.*

Bizonyítás. Az nyilvánvaló , hogy minden felismerhető nyelv felismerhető véges nemdeterminisztikus automatával, és hogy minden véges nemdeterminisztikus automatával felismerhető nyelv felismerhető véges nemdeterminisztikus, spontán átmenetekkel rendelkező automatával.

Tekintsünk most egy $M = (Q, \Sigma, \delta, q_0, F)$ véges nemdeterminisztikus spontán átmenetekkel rendelkező automatát és az általa felismert $L = L(M)$ nyelvet. Célunk egy olyan M' véges determinisztikus automata megadása, amely szintén az L nyelvet ismeri fel. Legyen

$$M' = P(M) = (P(Q), \Sigma, \delta', Q_0, \mathcal{F})$$

ahol

$$\delta' : P(Q) \times \Sigma \to P(Q), \qquad \delta'(X,a) = \widehat{Y},\ Y = \bigcup_{q \in X} \delta(q,a),\ X \subseteq Q,\ a \in \Sigma,$$

$$Q_0 = \widehat{\{q_0\}},$$

$$\mathcal{F} = \{X \subseteq Q :\ X \cap F \neq \emptyset\},$$

Ekkor minden $X \subseteq Q$ halmazra és $u \in \Sigma^*$ szóra fennáll a

$$\delta'(\widehat{X}, u) \quad = \quad \overline{\delta}(X,u) = \bigcup_{q \in X} \overline{\delta}(q,u)$$

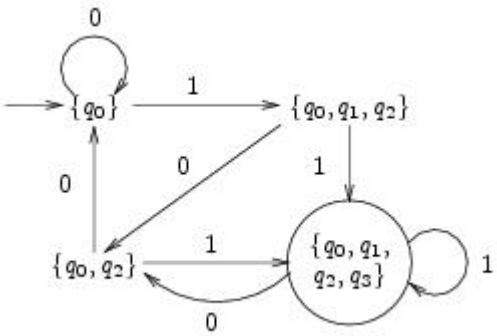
összefüggés. Itt a baloldalon az a halmaz áll, amelybe M' az $\widehat{X}$ állapotából eljut az u szó hatására, a jobboldalon pedig az összes olyan állapot halmaza szerepel, amelybe M eljuthat X-beli állapotból az u szó hatására. Így egy $u \in \Sigma^*$ szóra,

$$\begin{aligned} u \in L(M') \quad &\Leftrightarrow \quad \delta'(Q_0, u) \in \mathcal{F} \\ &\Leftrightarrow \quad \overline{\delta}(\{q_0\}, u) \cap F \neq \emptyset \\ &\Leftrightarrow \quad u \in L(M). \end{aligned}$$

Tehát $L(M') = L(M)$. □

Megjegyezzük, hogy $P(M)$ állapothalmazának választhattuk volna a $\{\widehat{X} : X \subseteq Q\}$ halmazt is.

A fentiekben megadott $M' = P(M)$ „hatványhalmaz automata” állapotainak száma $|P(Q)| = 2^{|Q|}$. Azonban elegendő ennek az „elérhető részét” venni, amely az M' azon állapotaiból áll, amelyek $\delta'(Q_0, u)$ alakban írhatóak, azaz „elérhetőek” a Q_0 kezdőállapotból. Ennek illusztrálására a korábbi véges nemdeterminisztikus automatára a következő ekvivalens véges determinisztikus automatát kapjuk:



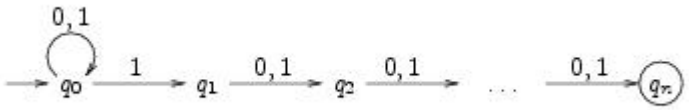
A determinizálásban az állapotok számának exponenciális növekedése nem kerülhető el.

1.3. Tétel. *Adott $n \geq 1$ esetén legyen*

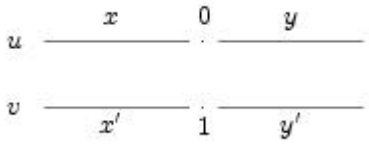
$$L_n = \{0,1\}^* \cdot \{1\} \cdot \{0,1\}^{n-1} \subseteq \{0,1\}^*.$$

Ekkor L felismerhető $n+1$ állapottal rendelkező véges nemdeterminisztikus automatával, de minden L-et felismerő véges determinisztikus automatának legalább 2^n állapota van.

Bizonyítás. Tehát L_n az összes olyan $\{0,1\}$-feletti szavakól áll, melyek hossza legalább n és a hátulról n-dik betű 1. Egy L_n-et felismerő $n+1$ állapotú véges nemdeterminisztikus automata az alábbi:



Tegyük most fel, hogy $M = (Q, \{0,1\}, \delta, q_0, F)$ véges determinisztikus automata mely felismeri L_n-et. Minden $u \in \{0,1\}^*$ szóra tekintsük a q_0u állapotot. Állítjuk, hogy ha $u \neq v$ n-hosszú szavak $\{0,1\}^*$-ban, akkor $q_0u \neq q_0v$. Ehhez tekintsük az alábbi ábrát:



$$|x| = |x'| = i,\ |y| = |y'| = n-i-1$$

Mivel $u \neq v$, van egy olyan pozíció, mondjuk az $(i+1)$-dik, ahol u és v eltérnek egymástól. Szimmetria miatt feltehetjük, hogy a tekintett pozíción u a 0 jelet, v pedig az 1 jelet tartalmazza. Ha $q_0u = q_0v$, akkor $q_0u0^i = q_0v0^i$, így $u0^i \in L_n \Leftrightarrow v0^i \in L_n$, ami ellentmondás.

Beláttuk tehát, hogy n hosszú u szavakra a q_0u állapotok mind különböznek, így $|Q| \geq 2^n$. □

1.5. Felismerhető nyelvek zártsági tulajdonságai, II

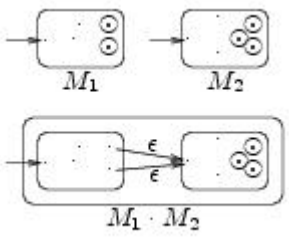
1.4. Tétel. *A felismerhető nyelvek osztálya zárt a konkatenációra: Ha $L_1, L_2 \subseteq \Sigma^*$ felismerhetők, akkor $L_1 \cdot L_2$ is felismerhető.*

Bizonyítás. Legyen $L_i \subseteq \Sigma^*$ az $M_i = (Q_i, \Sigma, \delta_i, q_i, F_i)$ véges nemdeterminisztikus spontán átmenetekkel rendelkező automata által felismert nyelv, ahol $i = 1,2$. Megadunk egy olyan véges nemdeterminisztikus spontán átmenetekkel rendelkező M automatát, amely az L_1L_2 nyelvet ismeri fel. Az általánosság megszorítása nélkül feltehető, hogy $Q_1 \cap Q_2 = \emptyset$.

Definiáljuk az $M_1 \cdot M_2 = (Q_1 \cup Q_2, \Sigma, \delta, q_1, F_2)$ automatát a

$$\delta(q,a) = \begin{cases} \delta_1(q,a) & q \in Q_1 - F_1 \\ \delta_1(q,a) & q \in F_1, a \neq \varepsilon \\ \delta_1(q,a) \cup \{q_2\} & q \in F_1, a = \varepsilon \\ \delta_2(q,a) & q \in Q_2 \end{cases}$$

szabályokkal. A konstrukció az alábbi ábrával szemléltethető:



Ha $u \in \Sigma^*$ olyan szó, amely hatására az automata q_1 kezdőállapotából el tud jutni valamely F_2-beli állapotba, akkor szükseg képpen „áthalad" egy olyan ε-nal címkézett élen, amely valamely F_1-beli állapotból q_2-be vezet. Ez azt jelenti, hogy u felbontható u_1u_2 alakban úgy, hogy az $M_1 \cdot M_2$ automatában létezik olyan olyan q_1-ből induló számítási sorozat, mely

- először az u_1 szóra a q_1 állapotból egy F_1-beli q állapotba vezet,
- majd q-bol közvetlenül q_2-be vezet az üres szóra,
- végül az u_2 szó hatására q_2-ből F_2-beli állapotba vezet.

Mivel a számítási sorozat első része az M_1 számítási sorozata, utolsó része pedig az M_2 számítási sorozata is, ezért $u_1 \in L_1$ és $u_2 \in L_2$. Tehát $u \in L_1L_2$. Mivel az u szó tetszőleges volt, beláttuk, hogy $L(M_1 \cdot M_2) \subseteq L_1L_2$.

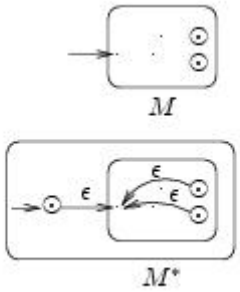
Fordítva, ha $u_i \in L_i$, $i = 1,2$, akkor tekintsünk az M_i-ben az u_i szóra olyan számítási sorozatokat, amely q_i-ből valamely $s_i \in F_i$ állapotba vezet. Felhasználva M-nek a $q_2 \in \delta(s_1, \varepsilon)$ átmenetét, a két számítási sorozatból összerakhatunk egy olyan számítási sorozatot, amely $M_1 \cdot M_2$-ben q_1-ből s_2-be vezet az u_1u_2 hatására. Ezért $L_1L_2 \subseteq L(M_1 \cdot M_2)$. □

1.5. Tétel. *A felismerhető nyelvek zártak a Kleene-féle iterációra: Ha $L \subseteq \Sigma^*$ felismerhető, akkor L^* is felismerhető.*

Bizonyítás. Legyen L az $M = (Q, \Sigma, \delta, q_0, F)$ véges nemdeterminisztikus spontán átmenetekkel rendelkező automata által felismert nyelv. Legyen s_0 egy új állapot, és definiáljuk az $M^* = (Q \cup \{s_0\}, \Sigma, \delta_*, s_0, F \cup \{s_0\})$ nemdeterminisztikus spontán átmenetekkel rendelkező automata átmeneteit az alábbi szabályokkal:

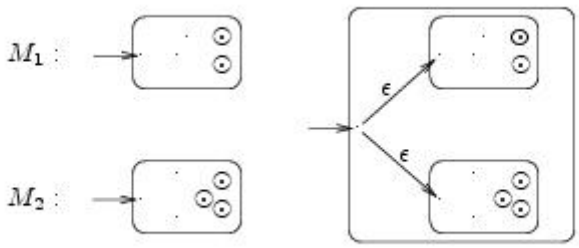
$$\delta_*(q,a) = \begin{cases} \delta(q,a) & q \in Q \text{ és } q \notin F \\ \delta(q,a) & q \in F \text{ és } a \neq \varepsilon \\ \delta(q,a) \cup \{q_0\} & q \in F \text{ és } a = \varepsilon \\ \{q_0\} & q = s_0 \text{ és } a = \varepsilon \\ \emptyset & q = s_0 \text{ és } a \neq \varepsilon \end{cases}$$

Az M^* konstrukcióját szemlélteti az alábbi ábra:



Könnyen belátható, hogy $L(M^*) = L^*$. □

Nemdeterminisztikus automatákat felhasználva új, egyszerűbb bizonyítás adható arra, hogy a felismerhető nyelvek zártak az egyesítésre. Ezt az alábbi ábrával szemléltetjük:



Az új $M_1 \cup M_2$ automatára $L(M_1 \cup M_2) = L(M_1) \cup L(M_2)$.

A véges nemdeterminisztikus (spontán átmenettel rendelkező) automata fogalma tovább általánosítható a felismerhető nyelvek osztályának növelése nélkül úgy, hogy több kezdőállapotot is megengedünk. Ennek felhasználásával az is könnyen belátható, hogy a felismerhető nyelvek zártak a tükörkép képzésre, hiszen csak az átmeneteket kell megfordítanunk és a kezdő- és végállapotokat felcserélnünk. Az is belátható, hogy ha $L \subseteq \Sigma^*$ felismerhető, akkor $\mathrm{pre}(L)$, $\mathrm{suf}(L)$ és $\mathrm{pre}(L)$ is felismerhetők.

1.6. Reguláris nyelvek és Kleene tétele

Az egyesítés, metszet és iteráció műveleteit *reguláris műveleteknek nevezzük*. Egy $L \subseteq \Sigma^*$ nyelvet *reguláris*nak nevezünk, ha előállítható a Σ^* véges részhalmazaiból a reguláris műveletek segítségével. Egy másik ekvivalens megfogalmazás az, hogy a nyelv előállítható az $\emptyset$ és $\{a\}$, $a \in \Sigma$ nyelvekből a reguláris műveletek segítségével, vagy az, hogy megadható reguláris kifejezéssel.

Legyen Σ véges, nemüres halmaz. Azt mondjuk, hogy R *reguláris kifejezés* (Σ-felett), ha:

1. $R = a$ valamely $a \in \Sigma$-ra, és ekkor R az $\{a\}$ nyelvet jelöli, vagy

2. $R = \emptyset$, és ekkor R az üres nyelvet jelöli, vagy

3. $R = (R_1 + R_2)$, és ekkor R az R_1 és R_2 által jelölt nyelvek egyesítését jelöli, vagy

4. $R = (R_1 \cdot R_2)$, és ekkor R az R_1 és R_2 által jelölt nyelvek konkatenációját jelöli, vagy

5. $R = (R_1^*)$, és ekkor R az R_1 által jelölt nyelv iterációját jelöli,

ahol R_1, R_2 már reguláris kifejezések. Egy Σ-feletti R reguláris kifejezésre legyen $|R|$ az R által jelölt nyelv. Az R_1 és R_2 kifejezések *ekvivalensek*, ha $|R_1| = |R_2|$.

Példákban a felesleges zárójeleket általában elhagyjuk és megegyezünk abban, hogy * erősebben köt, mint $\cdot$, ami erősebben köt, mint a $+$ művelet. A $\cdot$ jelet általában elhagyjuk. Az $\emptyset^*$ kifejezés helyett ε-t írunk. Vegyük észre, hogy a $\emptyset$ szimbólumot kétféle módon is használjuk, mint reguláris kifejezést, és az üres halmaz jelét. A kétféle felhasználás kompatibilis. Hasonló észrevétel érvényes az ε szimbólumra is. Néhány példa reguláris kifejezésekre (a $\{0,1\}$ halmaz felett), és az általuk jelölt nyelvekre:

kifejezés	**jelölt nyelv**
$0(01+10)1+\varepsilon$	$\{\varepsilon, 0011, 0101\}$
$0(0+1)^*1$	$\{w : w$ 0-val kezdődik, 1-el végződik$\}$
$(0+\varepsilon)(10)^*(1+\varepsilon)$	$\{w : 00$ és 11 nem rész-szavak$\}$
$(0^2)^*$	$\{w : w$ páros hosszú és csak 0-t tartalmaz$\}$
$(0^*10^*1)^*0^*$	$\{w : w$ páros sok 1-est tartalmaz$\}$

1.6. Tétel. *Minden reguláris nyelv felismerhető.*

Bizonyítás. Legyen R egy Σ-feletti reguláris kifejezés. Az R felépítése szerinti indukcióval igazoljuk, hogy $|R| \subseteq \Sigma^*$ felismerhető. Az alapeset az, amikor $R = \emptyset$ vagy $R = a$ valamely $a \in \Sigma$ betűre. Ekkor $|R| = \emptyset$ vagy $R = \{a\}$. De könnyen megadhatóak olyan véges determinisztikus, vagy véges nemdeterminisztikus automaták, amelyek ezeket a nyelveket ismerik fel. Az üres nyelvet felismerő nemdeterminisztikus automatának egyetlen állapota van, ami a kezdőállapot de nem végállapot, és egyetlen átmenettel sem rendelkezik. Az $\{a\}$ nyelvet felismerő nemdeterminisztikus automatának két állapota van, mondjuk q_0 és q_1, és egyetlen átmenete, amellyel q_0-ból az a hatására q_1-be jutunk. Tehát a δ átmenetfüggvényre $\delta(q_0, a) = \{q_1\}$ és $\delta(q, b) = \emptyset$ különben, azaz ha $q \neq q_0$ vagy $b \neq a$. A q_0 a kezdőállapot, és q_1 az egyetlen végállapot. A két automata szemléltetése:

$$\to \cdot \qquad \text{és} \qquad \to \cdot \xrightarrow{a} \odot$$

Az indukiós lépésben 3 esetet különböztetünk meg.

- $R = (R_1 + R_2)$. Ekkor $|R| = |R_1| \cup |R_2|$, és az állítás következik az indukciós feltevésből és abból, hogy a felismerhető nyelvek zártak az egyesítésre.

- $R = (R_1 \cdot R_2)$. Most az indukciós feltevést használjuk, és azt, hogy a felismerhető nyelvek zártak a konkatenációra.

- $R = (R_1^*)$. Az indukciós feltevést és a felismerhető nyelvek iterációra való zártságát használjuk. □

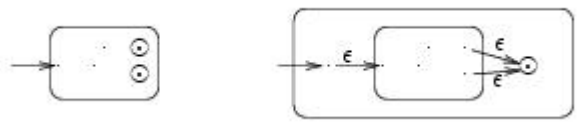
Mielőtt az előző tétel megfordítását is igazolnánk, belátunk egy egyszerű állítást.

1.1. Lemma. *Minden felismerhető nyelv felismerhető olyan véges nemdeterminisztikus spontán átmenetekkel rendelkező automatával, melynek pontosan egy végállapota van, a végállapotból nem indul átmenet, a kezdőállapotba nem vezet átmenet, továbbá a kezdőállapot különbözik a végállapottól.*

Bizonyítás. Az állítás az, hogy minden $L \subseteq \Sigma^*$ felismerhető nyelvhez megadható olyan $M = (Q, \Sigma, \delta, q_0, \{q_f\})$ véges nemdeterminisztikus automata, amelyre $L = L(M)$, $q_0 \neq q_f$, továbbá

$$\begin{aligned} \delta(q_f, a) &= \emptyset \quad a \in \Sigma_\varepsilon \\ q_0 &\notin \delta(q, a) \quad q \in Q, a \in \Sigma_\varepsilon. \end{aligned}$$

Kiindulva egy teszőleges L-et felismerő véges nemdeterminisztikus (spontán átmenetekkel rendelkező) automatából, ilyen tulajdonságokkal rendelkező automatát kapunk akkor, ha felveszünk egy új kezdőállapotot, egy új végállapotot, továbbá egy új átmenetet ε hatására az új kezdőállapotból a régi kezdőállapotba, valamint minden egyes régi végállapotból egy új átmenetet ε hatására az új végállapotba, amely az egyedüli végállapot lesz.



□

A továbbiakban olyan véges irányított gráfokat fogunk tekinteni, melyeknek:

1. Ki van jelölve egy q_0 „bemenő” csúcsa.
2. Ki van jelölve egy q_f „kimenő” csúcsa, $q_0 \neq q_f$.
3. A q_0 csúcsba nem vezet él és q_f-ből nem indul él.
4. Ettől eltekintve bármely két q_1, q_2 csúcsra pontosan egy q_1-ből q_2-be vezető él van.
5. Valamely rögzített Σ-ra minden él egy Σ-feletti reguláris kifejezéssel van címkézve.

A példákban nem tüntetjük fel azokat az éleket, melyek címkéje $\emptyset$. A q_0-tól és q_f-től különböző csúcsokat *belső csúcsok*nak nevezzük.

Minden ilyen gráfhoz hozzárendelhetünk egy reguláris nyelvet, melyet a gráf reprezentál. Ezt azon u szavak alkotják, amelyeket úgy kapunk, hogy bemenő csúcsból az élek mentén (egy él többször is felhasználható) egy úton elmegyünk a kimenő csúcsba, és valahányszor egy élen áthaladunk, leírunk egy olyan u_i szót, amely abba a reguláris nyelvbe esik, amelyet az él címkéje jelöl. Az $u = u_1 \dots u_n$ szó az u_i szavak szorzata.

1.7. Tétel. *Minden felismerhető nyelv reguláris.*

Bizonyítás. Legyen $L = L(M)$, ahol $M = (Q, \Sigma, \delta, q_0, \{q_f\})$ eleget tesz az előző lemmában megkövetelt tulajdonságoknak. Megadunk egy algoritmust, amellyel M-et reguláris kifejezéssé konvertáljuk.

Először vegyük észre, hogy M-et felfoghatjuk úgy, mint egy, a fentiekben leírt címkézett irányított gráfot. Adott q, q' állapotokra, $(q \neq q_f,\ q' \neq q_0)$ a $q \longrightarrow q'$ él címkéje a

$$\sum_{a \in \Sigma_\varepsilon} (a :\ q' \in \delta(q,a))$$

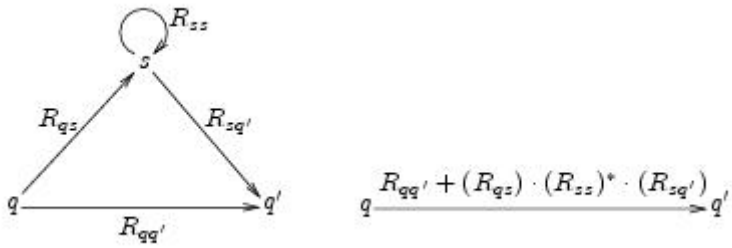
reguláris kifejezés. (Ha q-ból nem indul ki átmenet, akkor ez $\emptyset$.) Ez a gráf az L nyelvet reprezentálja.

Ezek után a gráfot a belső csúcsok eliminálásával átalakítjuk olyan gráffá, amelynek egyetlen éle van (mely a bemenő csúcsból a kimenő cscsba vezet), és amely L-et jelöli. Az átalakítás minden lépésében garantáltan ekvivalens, tehát az L nyelvet reprezentáló gráfot kapunk.

Redukciós lépés. Mindaddig, amíg még van belső csúcs, válasszunk ki egyet, mondjuk az s csúcsot. Ezek után elhagyjuk az s csúcsot, és minden ettől különböző q, q' csúcsra, amelyre $q \neq q_0$ vagy $q' \neq q_f$, a $q \to q'$ él $R_{qq'}$ címkéjét az

$$R_{qq'} + (R_{qs}) \cdot (R_{ss})^* \cdot (R_{sq'})$$

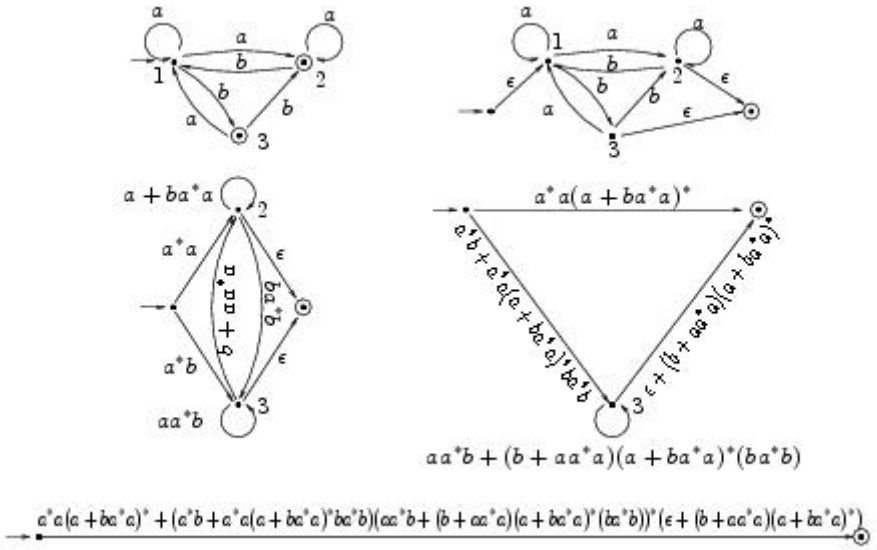
kifejezésre változtatjuk.



Világos, hogy ekvivalens gráfot kapunk.

Az eljárás gyorsítható azzal, hogy kezdetben elhagyjuk az összes olyan állapotot, amely nem érhető el q_0-ból, vagy amelyből nem érhető el q_f (az élekre illeszkedő csúcsokkal együtt). □

Eljárásunkat az alábbi példával szemléltetjük:



1.1. Következmény. (Kleene tétele.) *Egy nyelv akkor és csak akkor felismerhető, ha reguláris.*

Kleene tételének mindkét irányát konstruktívan bizonyítottuk. A bizonyítások egyben eljárást is adnak véges automata ekvivalens reguláris kifejezésbe való átalakítására, és megfordítva, reguláris kifejezés automatába való alakítására. Az a módszer, amellyel reguláris kifejezéshez készítettünk automatát a kifejezés hosszában lineáris állapotszámú véges nemdeterminisztikus automatát eredményezett. A fordított irányú konstrukció azonban az állapotok számában akár exponenciális hosszú kifejezést is eredményezhet. Ismert, hogy ez általában nem is kerülhető el.

A reguláris kifejezés fogalma kiterjeszthető úgy, hogy a metszet és komplementerképzést is megengedjük a reguláris műveletek mellett. Az ilyen *általánosított reguláris kifejezések*kel továbbra is csak a reguláris nyelvek jelölhetőek, mivel a reguláris nyelvek zártak ezekre a műveletekre is. Ugyanakkor segítségükkel akár exponenciálisan rövidebben adhatunk meg nyelveket. Egy általánosított reguláris kifejezés *iterációs foka* az a legnagyobb szám, ahányszor az iteráció művelete be van skatulyázva. (Itt célszerű az $\emptyset^*$ kifejezés fokát nem 1-nek, hanem 0-nak választani.) Egy L reguláris nyelv (általánosított) iterációs foka a legkisebb olyan n szám, amelyre létezik olyan n iterációs fokú (általánosított) reguláris kifejezés, amely L-et jelöli. Így egy reguláris nyelv iterációs foka akkor és csak akkor 0, ha véges. A páros sok 1-est tartalmazó szavak $L \subseteq \{0,1\}^*$ nyelvének iterációs foka 2, általánosított iterációs foka 1. Minden $n \geq 0$ számra létezik olyan reguláris nyelv, melynek iterációs foka n, de nem ismert, hogy létezik-e 2 általánosított iterációs fokú reguláris nyelv. Az általánosított reguláris kifejezésekhez hasonló kifejezések írhatók számos UNIX utasításban.

1.7. A reguláris nyelvek pumpáló lemmája

Ebben a részben a reguláris nyelvek egy fontos kombinatorikus tulajdonságával ismerkedünk meg. Segítségével beláthatjuk bizonyos nyelvekről azt, hogy nem regulárisak.

1.8. Tétel. *Minden $L \subseteq \Sigma^*$ reguláris nyelvhez létezik olyan $p \geq 1$ szám, hogy valahányszor az $u \in L$ szó hossza legalább p, u felírható*

$$u = xyz$$

alakban úgy, hogy

1. $xy^iz \in L$ *minden* $i \geq 0$ *számra,*
2. $|y| > 0$,
3. $|xy| \leq p$.

Bizonyítás. Legyen $L = L(M)$, ahol $M = (Q, \Sigma, \delta, q_0, F)$ véges determinisztikus automata. Legyen $p = |Q|$. Ha $u \in \Sigma^*, |u| \geq p$ és $u \in L$, akkor tekintsük a q_0-ból induló $q_0, q_1, \ldots, q_n$ számítási sorozatot az u szón. Fennállnak a következők:

1. $|u| = n \geq p$,
2. $q_n \in F$,
3. $\exists i, j,\ 0 \leq i < j \leq p,\ q_i = q_j$.

Legyen x az u i hosszú kezdőszelete, y az x-et követő $j-i$ hosszú rész-szó, z az u $n-j$ hosszú zárószelete. Ekkor az $u = xyz$ felbontásra teljesülnek a tétel állításai. □

1.1. Állítás. *Az $L = \{0^n1^n : n \geq 0\} \subseteq \{0,1\}^*$ nyelv nem reguláris.*

Bizonyítás. Belátjuk, hogy

$$\forall p\ \exists u \in L, |u| \geq p\ \forall x,y,z \quad [(u = xyz \wedge |xy| \leq p \wedge |y| > 0) \quad \Rightarrow \quad \exists i\ xy^iz \notin L].$$

Legyen $p \geq 1$ tetszőleges. Tekintsük az $u = 0^p1^p$ szót. Tegyük fel, hogy x, y, z olyan szavak, melyekre $u = xyz$, $|xy| \leq p$ és $|y| > 0$. Ekkor xy csupa 0-ból áll, és y tartalmaz legalább egy 0-át. Ha tehát $i \neq 1$, akkor az xy^iz szóban a 0-ák száma különbözik az 1-ek számától, így $i \neq 1$ esetén $xy^iz \notin L$. □

2. fejezet

Környezetfüggetlen nyelvek és veremautomaták

Ebben a fejezetben egy újabb nyelvosztállyal, a környezetfüggetlen nyelvek osztályával ismerkedünk meg, amely valódi módon tartalmazza a reguláris nyelveket. A környezetfüggetlen nyelveket Chomsky vezette be az 1960-as években a természetes nyelvek bizonyos aspektusainak leírására. Mindeddig legátütőbb alkalmazásukat azonban a mesterséges nyelvek, ezen belül a programozási nyelvek adták.

2.1. Környezetfüggetlen nyelvtanok és nyelvek

Nyelveket gyakran *rekurzióval* adunk meg. Példaként tekintsük az előzőekben már szerepelt $L = \{0^n 1^n : n \geq 0\} \subseteq \{0,1\}^*$ nyelvet. Ekkor L a legszűkebb olyan $L' \subseteq \{0,1\}^*$ nyelv, amelyre

- $\varepsilon \in L'$ és
- $u \in L' \quad \Rightarrow \quad 0u1 \in L'$.

Valóban, egyrészt az L nyelvre teljesül a fenti két tulajdonság. Másrészt ha az $L' \subseteq \{0,1\}^*$ nyelvre is teljesül, akkor teljes indukcióval könnyen beláthatjuk, hogy $0^n 1^n \in L'$ minden $n \geq 0$ számra, tehát $L \subseteq L'$.

Egy másik példaként tekintsük a helyes zárójelezések $L \subseteq \{),(\}^*$ nyelvét. Ez a legszűkebb olyan L' nyelv, amelyre

- $\varepsilon \in L'$,
- $u \in L' \quad \Rightarrow \quad (u) \in L'$,
- $u, v \in L \quad \Rightarrow \quad uv \in L'$.

Ennek belátásához persze szükség lenne a helyes zárójelezések nyelvének egy másik, mindenki számára nyilvánvaló megadására, amitől most eltekintünk.

A környezetfüggetlen nyelvtanok a példákban adott rekurzív definíciókat modellezik. *Környezetfüggetlen nyelvtan* egy

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

rendszer, ahol

- V véges, nemüres halmaz: a *változók* vagy *nemterminálisok* abc-je,
- Σ véges, nemüres halmaz: a *terminálisok* abc-je, ahol $V \cap \Sigma = \emptyset$,
- $R: A \to w$ alakú *átírási szabályok* véges halmaza, ahol $A \in V,\ w \in (V \cup \Sigma)^*$,
- $S \in V$ a *kezdőszimbólum*.

Legyen $G = (V, \Sigma, R, S)$ környezetfüggetlen nyelvtan, $u, v \in (V \cup \Sigma)^*$.

1. Azt mondjuk, hogy *u közvetlenül deriválja* a *v* szót, vagy *v közvetlenül levezethető u*-ból, $u \Rightarrow v$, ha léteznek az $u = u_1 A u_2$ és $v = u_1 w u_2$ felbontások úgy, hogy $A \to w \in R$.
2. Egy $u_0, u_1, \ldots, u_n \quad (n \geq 0)$ sorozatot a *v szó u-ból való derivációjának* vagy *levezetésének* nevezünk, ha
   - $u_0 = u,\ u_n = v$
   - $u_{i-1} \Rightarrow u_i \quad i = 1, \ldots, n.$
3. Azt mondjuk, hogy *v deriválható* vagy *levezethető u*-ból, ha létezik a *v*-nek *u*-ból való derivációja. Ennek jelölése: $u \Rightarrow^* v$.
4. A *G által generált nyelv*:

$$L(G) = \{w \in \Sigma^* : S \Rightarrow^* w\}.$$

Két nyelvtant *ekvivalens*nek nevezünk, ha ugyanazt a nyelvet generálják. Végül egy $L \subseteq \Sigma^*$ nyelvet *környezetfüggetlen*nek nevezünk, ha generálható környezetfüggetlen nyelvtannal, azaz létezik olyan G környezetfüggetlen nyelvtan (melyben a terminálisok halmaza Σ) úgy, hogy $L = L(G)$.

Példákban nyelvtanokat általában úgy adunk meg, hogy soronként megadjuk minden egyes A nemterminálisra az A baloldalú szabályokat. A szabályok jobboldalait a | jellel választjuk el. Az első sorban szerepelnek a kezdőszimbólumra vonatkozó szabályok. Minden olyan betű terminális, amely szabály jobboldalán előfordul, de nem fordul elő szabály baloldalán.

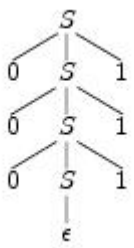
Első példaként tekintsük az

$$S \quad \to \quad 0S1 \mid \varepsilon$$

nyelvtant. Egy derivációra példa:

$$S \Rightarrow 0S1 \Rightarrow 00S11 \Rightarrow 000S111 \Rightarrow 000111$$

Ez a deriváció egy fával is ábrázolható:



Belátjuk, hogy a nyelvtan az $L = \{0^n1^n : n \geq 0\}$ nyelvet generálja. Legyen $n \geq 0$ és tekintsük az $u = 0^n1^n$ szót. Teljes indukcióval igazoljuk, hogy $u \in L(G)$.

Ha $n = 0$, akkor $u = \varepsilon$. Mivel $S \Rightarrow \varepsilon$, ezért $u \in L(G)$. Tegyük most fel, hogy $n > 0$, és hogy $0^{n-1}1^{n-1} \in L(G)$. Ekkor $S \Rightarrow^* u^{n-1}v^{n-1}$ és ezért $0S1 \Rightarrow^* 0^n1^n$, hiszen ha a $0^{n-1}1^{n-1}$ S-ből való valamely levezetésében minden egyes szót egy 0 és egy 1 közé teszünk, akkor 0^n1^n levezetését kapjuk $0S1$-ből. Mivel $S \Rightarrow 0S1 \Rightarrow^* 0^n1^n$, ezért $S \Rightarrow^* u^nv^n = u$.

Ezzel beláttuk, hogy $L \subseteq L(G)$. Az $L(G) \subseteq L$ tartalmazás igazolásához tegyük most fel, hogy $S \Rightarrow^* u$ teljesül az $u \in \{0,1\}^*$ szóra. Tekintsünk egy

$$S = u_0 \Rightarrow u_1 \Rightarrow \ldots \Rightarrow u_{n+1} = u$$

levezetést, ahol $n \geq 0$. Belátjuk n szerinti indukcióval, hogy $u = 0^n1^n$. Ha $n = 0$, akkor a levezetés egyetlen lépésből áll. Mivel u terminális szó, csak az $S \to \varepsilon$ szabályt alkalmazhattuk. Tehát $u = \varepsilon = 0^n1^n$, hiszen $n = 0$. Az indukciós lépésben tegyük fel, hogy $n > 0$. Most az első lépésben alkalmazott szabály csak az $S \to 0S1$ lehet. Ezért az $u_1, \ldots, u_{n+1}$ szavak mindegyike 0-val kezdődik és 1-gyel végződik. Léteznek tehát olyan v_i, $i = 1, \ldots, n+1$ szavak, hogy $u_i = 0v_i1$ minden i-re. Azt már megjegyeztük, hogy $v_1 = S$, és az is világos, hogy

$$S = v_1 \Rightarrow \ldots \Rightarrow v_{n+1}.$$

Az indukciós feltevésből így $v_{n+1} = 0^{n-1}1^{n-1}$, tehát $u = 0v1 = 0^n1^n$.

Mivel az L nyelv nem reguláris, ezzel azt is beláttuk, hogy létezik olyan környezetfüggetlen nyelvtan, amely nem reguláris.

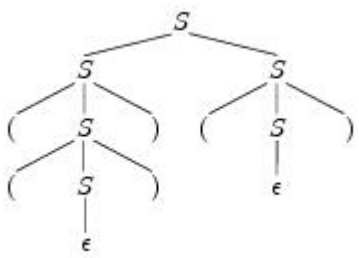
Második példaként tekintsük az

$$S \quad \to \quad (S) \mid SS \mid \varepsilon$$

nyelvtant. Ez a helyes zárójelezések nyelvét generálja. Egy levezetés:

$$S \Rightarrow SS \Rightarrow (S)S \Rightarrow ((S))S \Rightarrow (())S \Rightarrow (())(S) \Rightarrow (())()$$

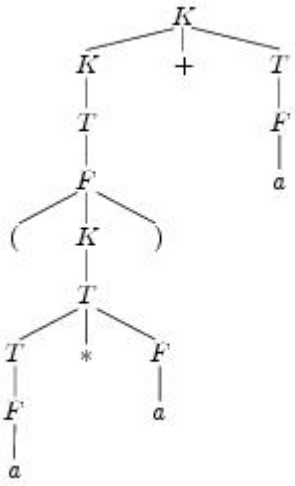
A levezetéshez tartozó fa:



Egy újabb példa az „aritmetikai kifejezések” nyelvtana, ahol egyszerűség kedvéért csak egy azonosítót engedtünk meg.

$$K \to K+T \,|\, T$$
$$T \to T*F \,|\, F$$
$$F \to (K) \,|\, a$$

$$\begin{aligned} K &\Rightarrow K+T \Rightarrow T+T \\ &\Rightarrow F+T \Rightarrow (K)+T \\ &\Rightarrow (T)+T \Rightarrow (T*F)+T \\ &\Rightarrow (F*F)+T \Rightarrow (F*a)+T \\ &\Rightarrow (F*a)+F \Rightarrow (a*a)+F \\ &\Rightarrow (a*a)+a \end{aligned}$$



2.2. Derivációs fák

Az előző példákban derivációkat fákkal szemléltettünk. A derivációs fa fogalma általánosan is bevezethető.

Legyen $G = (V, \Sigma, R, S)$ környezetfüggetlen nyelvtan. G feletti *derivációs fa* olyan véges, irányított, rendezett fa, melynek csúcsai a $V \cup \Sigma_\varepsilon$ halmaz elemeivel cimkézettek úgy, hogy valahányszor egy csúcs és leszármazottainak címkéi rendre $X, X_1, \ldots, X_n \quad (n \geq 1)$, mindannyiszor $X_1, \ldots, X_n \in V \cup \Sigma$ és $X \to X_1 \ldots X_n \in R$, vagy $n = 1, X_1 = \varepsilon$ és $X \to \varepsilon \in R$. Továbbá minden levél címkéje a Σ_ε halmazban van. Ha a gyökér címkéje $X \in V \cup \Sigma_\varepsilon$, akkor azt is mondjuk, hogy a *derivációs fa X-ből indul*. A derivációs fa leveleinek, illetve a levelek címkéinek sorozata a *derivációs fa határa*. Ez egy Σ^*-beli szó.

2.2. Állítás. *Ha $X \Rightarrow^* u$, ahol $X \in V \cup \Sigma_\varepsilon$ és $u \in \Sigma^*$, akkor létezik olyan X-ből induló derivációs fa, melynek határa u.*

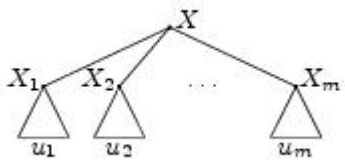
Bizonyítás vázlat. Tegyük fel, hogy $X = u_0 \Rightarrow u_1 \Rightarrow \ldots \Rightarrow u_n = u$ az u levezetése X-ből.

Ha $n = 0$ akkor $X = u \in \Sigma_\varepsilon$, és a megfelelő derivációs fának egyetlen csúcsa van, a gyökér, mely u-val címkézett:

$$\cdot u$$

Tegyük fel, hogy $n > 0$. Legyen $u_1 = X_1 \ldots X_m$, ahol $X_i \in V \cup \Sigma$, $i = 1, \ldots, m$. Ekkor u felbontható $u_1 \ldots u_m$ alakban úgy, hogy minden i-re $X_i \Rightarrow^* u_i$ n-nél rövidebb levezetéssel. Így minden $i = 1, \ldots, m$ esetén létezik olyan X_i-ből induló derivációs fa, melynek határa u_i:

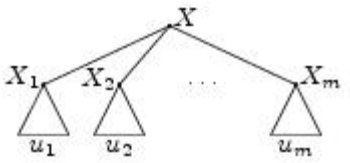
Ezek felhasználásával összerakhatunk egy olyan X-ből induló derivációs fát, melynek határa $u = u_1 \ldots u_m$:



□

2.3. Állítás. *Ha létezik olyan X-ből induló derivációs fa, melynek határa az $u \in \Sigma^*$ szó, akkor $X \Rightarrow^* u$.*

Bizonyítás. A derivációs fa n mélysége szerinti indukcióval igazoljuk az állítást. (Fa mélysége: leghosszabb úton lévő élek száma.) Ha $n = 0$, akkor $X = u \in \Sigma_\varepsilon$. Ugyanakkor nyilvánvalóan $X \Rightarrow^* u$, hiszen a $\Rightarrow^*$ reláció reflexív. Tegyük fel, hogy $n > 0$. Ekkor a fa



alakú. Az indukciós feltevés szerint $X_i \Rightarrow^* u_i$, $i = 1, \ldots, m$. Így

$$X \Rightarrow X_1 \ldots X_m \Rightarrow^* u_1 X_2 \ldots X_m \Rightarrow^* u_1 u_2 X_3 \ldots X_m \Rightarrow^* u_1 u_2 \ldots u_m = u.$$

Mivel $\Rightarrow^*$ tranzitív reláció, mely tartalmazza a $\Rightarrow$ relációt, $X \Rightarrow^* u$. □

Az előző bizonyítással *baloldali derivációhoz* jutunk. Azt mondjuk, hogy egy

$$u_0 \Rightarrow u_1 \Rightarrow \ldots \Rightarrow u_n$$

derivációban baloldali, ha minden $i < n$ számra u_{i+1} úgy áll elő az u_i szóból, hogy az u_i-ben előforduló első nemterminálist írjuk át. *Jobboldali derivációk* hasonlóan definiálhatóak. *Baloldali derivációra jelölése:*

$$u_0 \Rightarrow_l u_1 \Rightarrow_l \ldots \Rightarrow_l u_n, \quad \text{vagy} \quad u_0 \Rightarrow_l^* u_n.$$

2.2. Következmény. *Legyen $G = (V, \Sigma, R, S)$ környezetfüggetlen nyelvtan. A következők ekvivalensek egy $u \in \Sigma^*$ szóra:*

1. *$u \in L(G)$,*
2. *$S \Rightarrow_l^* u$,*
3. *Létezik olyan S-ből induló derivációs fa, melynek határa u.*

Megjegyezzük, hogy egy $u \in L(G)$ szónak az S-ből induló derivációs fái és az S-ből induló baloldali derivációi között kölcsönösen egyértelmű kapcsolat van. A $G = (V, \Sigma, R, S)$ nyelvtant *egyértelmű*nek nevezzük, ha minden $u \in L(G)$ szónak pontosan egy S-ből induló baloldali levezetése (derivációs fája) van. A nyelvtan egyértelműsége azért fontos feltétel általában, mert különböző derivációs fákhoz különböző jelentés társítható.

2.3. Környezetfüggetlen nyelvek zártsági tulajdonságai

Ebben a részben belátjuk, hogy a környezetfüggetlen nyelvek zártak a reguláris műveletekre. Később meg fogjuk mutatni, hogy ezzel szemben nem zártak a metszet és komplementerképzés műveleteire. Eredményeinkből könnyen adódik, hogy minden reguláris nyelv környezetfüggetlen.

2.9. Tétel. *Ha $L, L_1, L_2 \subseteq \Sigma^*$ környezetfüggetlen nyelvek, akkor $L_1 \cup L_2$, $L_1 L_2$ és L^* is azok.*

Bizonyítás. Legyenek $G_i = (V_i, \Sigma, R_i, S_i)$, $i = 1, 2$, olyan környezetfüggetlen nyelvtanok, amelyek az L_1 és L_2 nyelveket generálják. Feltehetjük, hogy V_1 és V_2 diszjunkt halmazok. Legyen S egy újabb szimbólum, amely nincs a $V_1 \cup V_2 \cup \Sigma$ halmazban.

Legyen

$$\begin{aligned} G_\cup &= (V_1 \cup V_2 \cup \{S\}, \Sigma, R_1 \cup R_2 \cup \{S \to S_1, S \to S_2\}, S) \\ G_\bullet &= (V_1 \cup V_2 \cup \{S\}, \Sigma, R_1 \cup R_2 \cup \{S \to S_1 S_2\}, S). \end{aligned}$$

Ekkor $L(G_\cup) = L_1 \cup L_2$ és $L(G_\bullet) = L_1 L_2$. Ez nyilvánvaló abból, hogy $G_\cup$ S-ből induló derivációs fái pontosan azok a fák, amelyek előállnak úgy, hogy vesszük a G_1 vagy G_2 egy S_1-ből ill. S_2-ből induló derivációs fáját, és azt kiegészítjük egy új gyökérrel, amelynek címkéje S. Az $L(G_\bullet)$ S-ből induló derivációs fái pontosan azok a fák, amelyek előállnak úgy, hogy vesszük a G_1 és G_2 egy S_1-ből ill. S_2-ből induló T_1 ill. T_2 derivációs fáját, majd felveszünk egy új S címkéjű gyökeret, melynek két leszármazottja a T_1 és T_2 gyökere.

Végül tekintsünk egy olyan $G = (V, \Sigma, R, S)$ nyelvtant, amely az L nyelvet generálja. Legyen S' új szimbólum, és tekintsük a $G^* = (V \cup \{S'\}, \Sigma, S', R \cup \{S' \to SS', S \to \varepsilon\}, S')$ nyelvtant. Könnyen belátható, hogy $L(G^*) = L^*$. □

Nem nehéz belátni az sem, hogy ha L környezetfüggetlen, akkor az L^{-1}, $\mathrm{pre}(L)$, $\mathrm{suf}(L)$ és $\mathrm{sub}(L)$ nyelvek is azok.

2.3. Következmény. *Minden reguláris nyelv környezetfüggetlen.*

Bizonyítás. Minden véges nyelv környezetfüggetlen, hiszen egy

$$\{u_1, \ldots, u_n\} \subseteq \Sigma^*$$

nyelv generálható azzal a nyelvtannal, melynek szabályai

$$S \quad \to \quad u_1 \mid \ldots \mid u_n$$

Mivel minden reguláris nyelv előáll a véges nyelvekből a reguláris műveletekkel, és mivel a környezetfüggetlen nyelvek zártak a reguláris műveletekre, minden Σ-feletti reguláris nyelv környezetfüggetlen. □

Megadható egy olyan egyszerű eljárás is, amely egy véges nemdeterminisztikus (spontán átmenetekkel rendelkező) automatát környezetfüggetlen nyelvtanra konvertál. Legyen $L = L(M) \subseteq \Sigma^*$, ahol $M = (Q, \Sigma, \delta, q_0, F)$ véges nemdeterminisztikus automata. Tekintsük azt a $G_M = (Q, \Sigma, R, q_0)$ nyelvtant, amelyben

$$R = \{q \to aq' : q' \in \delta(q, a)\} \cup \{q \to \varepsilon : q \in F\}.$$

Ekkor könnyen igazolható a deriváció hossza szerinti indukcióval, hogy amennyiben $q_0 \Rightarrow^* uq$ valamely $q \in Q$ állapotra és $u \in \Sigma^*$ szóra, akkor létezik az u szóra q_0-ból q-ba vezető számítási sorozat, azaz $q \in \overline{\delta}(q_0, u)$. Fordítva, az is könnyen igazolható (a számítási sorozat hossza szerinti indukcióval), hogy amennyiben létezik az u szóra q_0-ból q-ba vezető számítási sorozat, akkor $q_0 \Rightarrow^* uq$. Ezt felhasználva, $u \in L(G_M)$ akkor és csakis akkor, ha van olyan $q \in F$, amelyre $q_0 \Rightarrow^* uq$ akkor és csakis akkor, ha $\overline{\delta}(q_0, u) \cap F \neq \emptyset$, azaz ha $u \in L(M)$. □

Nevezzünk egy $G = (V, \Sigma, R, S)$ környezetfüggetlen nyelvtant *jobblineáris*nak, ha minden szabálya $A \to uB$ vagy $A \to u$ alakú, ahol $u \in \Sigma^*$ és $A, B \in V$. Továbbá nevezzünk egy $L \subseteq \Sigma^*$ nyelvet jobblineárisnak, ha generálható jobblineáris nyelvtannal. Az M-ből előállított G_M nyelvtan *jobblineáris*, tehát minden reguláris nyelv jobblineáris. Belátható az is, hogy a jobblineáris nyelvek osztálya pontosan megegyezik a reguláris nyelvek osztályával, ld. az alábbiakat.

2.4. Chomsky-féle normálforma

Ebben a részben belátjuk, hogy minden környezetfüggetlen nyelvtan egyszerűbb, ún. Chomsky-féle normálformára hozható. Azt mondjuk, hogy a $G = (V, \Sigma, R, S)$ környezetfüggetlen nyelvtan *Chomsky normálformában adott*, ha R minden szabálya

$$A \to BC \quad \text{vagy} \quad A \to a$$

alakú, ahol $A, B, C \in V$, $a \in \Sigma$, esetleg az $S \to \varepsilon$ szabály kivételével, de ekkor S nem fordul elő egyetlen szabály jobboldalán sem.

A Chomsky-féle normálformára hozást 3 lépésben végezzük el.

1. Az ε jobboldalú szabályok eliminálása.
2. A láncszabályok eliminálása.
3. A jobboldalak átalakítása.

Itt láncszabályon egy $A \to B$ alakú szabályt értünk, ahol A, B nemterminálisok.

2.4. Állítás. *Minden környezetfüggetlen nyelvtanhoz megadható olyan ekvivalens környezetfüggetlen nyelvtan, melyben egyetlen szabály jobboldala sem az üres szó, esetleg az $S \to \varepsilon$ szabály kivételével, ahol S a kezdőszimbólum, de ekkor S nem fordul elő egyetlen szabály jobboldalán sem.*

Bizonyítás. Legyen $G = (V, \Sigma, R, S)$. Először meghatározzuk azon $A \in V$ nemterminálisok halmazát, amelyekre $A \Rightarrow^* \varepsilon$. Ehhez kezdetben megjelöljük azokat az A nemterminálisokat, amelyekre $A \to \varepsilon$ szabály, majd mindaddig megjelölünk újabb A nemterminálist, amíg találunk olyan $A \to u$ szabályt, hogy $u \in V^+$ és u minden nemterminális betűje már korábban megjelölt. A végül megjelölt nemterminálisok pontosan azok, amelyekből levezethető az üres szó.

Ezek után két esetet különböztetünk meg. Ha S nincs megjelölve, akkor legyen $G' = (V, \Sigma, R', S)$, ahol R' az összes olyan $A \to u$ szabályból áll, amelyre $u \neq \varepsilon$, és létezik olyan

$A \to v$ szabály R-ben, hogy u megkapható v-ből megjelölt nemterminálisok törlésével. Ha S megjelölt, akkor G' tartalmazza még az S' új nemterminálist és az $S' \to S$ és $S' \to \varepsilon$ szabályokat. A kezdőszimbólum S'.

Ez a módszer legrosszabb esetben exponenciálisan megnöveli a nyelvtan szabályainak számát. A módszer egy olyan finomítását ismertetjük, ahol a szabályok száma csak polinomiálisan nő.

Legyen

$$V' \quad = \quad V \cup \{[v] : v \in (V \cup \Sigma)^+,\ \exists u \in (V \cup \Sigma)^*\ A \to uv \in R\},$$

és legyen R'' az alábbi szabályhalmaz:

1. Minden $A \to p \in R$, $p \neq \varepsilon$ szabályra legyen $A \to [p] \in R''$.
2. Valahányszor $[X_1 \dots X_n] \in V'$, ahol $n > 1$, legyen
$$[X_1 \dots X_n] \to X_1[X_2 \dots X_n] \in R''.$$
3. Valahányszor $[X_1 \dots X_n] \in V'$, ahol $n > 1$ és X_1 megjelölt nemterminális, legyen
$$[X_1 \dots X_n] \to [X_2 \dots X_n] \in R''.$$
4. Valahányszor $[X_1 \dots X_n] \in V'$, ahol $n \geq 1$ és $X_2, \dots, X_n$ mindegyike megjelölt nemterminális, legyen
$$[X_1 \dots X_n] \to X_1 \in R''.$$

Végül legyen $G' = (V', \Sigma, R'', S)$, amennyiben S végül nincs megjelölve. Ha S is megjelölt, akkor $G' = (V' \cup \{S'\}, \Sigma, R'', S')$ ahol S' új szimbólum, R'' pedig az előző esetben megadott szabályokon kívül még az $S' \to S$ és $S' \to \varepsilon$ szabályokból áll.

Azt, hogy G és G' ekvivalensek, csak az első konstrukcióra és csak abban az esetben mutatjuk meg, amikor S nem kerül megjelölésre. Mivel S nincs megjelölve, ezért $\varepsilon \notin L(G)$. Világos, hogy $\varepsilon \notin L(G')$. Be kell még látnunk, hogy $L(G)$ és $L(G')$ ugyanazokat a nemüres szavakat tartalmazza. Legyen $u \in \Sigma^+$. Ha $u \in L(G)$, akkor u-nak van olyan S-ből induló derivációs fája G-ben, melynek határa u. Tekintsük az összes olyan maximális részfát, melynek határa ε. Ha minden ilyen részfát elhagyunk, akkor az u-nak egy S-ből induló derivációs fáját kapjuk G'-ben. Tehát $L(G) \subseteq L(G')$.

Megfordítva, ha $u \in L(G')$, akkor tekintsük az u egy S-ből induló derivációs fáját a G' nyelvtanban. A gyökértől a levelek felé haladva, minden elágazáshoz beilleszthetünk egy vagy több olyan G feletti derivációs fát (amennyiben ez szükséges) melynek határa ε, úgy, hogy minden elágazásnál R-beli szabály kerüljön alkalmazásra. Az előálló fa az u egy derivációs fája a G nyelvtanban. Tehát $u \in L(G)$, és mivel $u \in L(G')$ tetszőleges volt, $L(G') \subseteq L(G)$. □

Az előző állítás bizonyításában elkészített nyelvtant ε-*mentes*nek nevezzük.

2.5. Állítás. *Minden ε-mentes környezetfüggetlen nyelvtanhoz létezik olyan ekvivalens ε-mentes környezetfüggetlen nyelvtan, melynek nincs láncszabálya.*

Proof. Legyen $G = (V, \Sigma, R, S)$ ε-mentes környezetfüggetlen nyelvtan. Minden A nemterminálisra határozzuk meg az összes olyan B nemterminális V_A halmazát, hogy $A \Rightarrow^* B$. Ezután vegyük az összes olyan $A \to u$ alakú szabály R' halmazát, hogy $u \notin V$, és valamely $B \in V_A$-ra $B \to u$ az R-ben van. Az előálló $G' = (V, \Sigma, R', S)$ olyan ε-mentes és láncszabály-mentes környezetfüggetlen nyelvtan, melyre $L(G) = L(G')$. □

2.6. Állítás. *Minden ε-mentes és láncszabály-mentes környezetfüggetlen nyelvtanhoz létezik vele ekvivalens Chomsky-féle normálformában lévő környezetfüggetlen nyelvtan.*

Bizonyítás. Legyen $G = (V, \Sigma, R, S)$ ε-mentes és láncszabály-mentes környezetfüggetlen nyelvtan. Feltehető, hogy minden szabály jobboldala vagy egy nemterminálisokból álló legalább 2 hosszú szó, vagy egyetlen terminális betű, esetleg az üres szó, de akkor a szabály $S \to \varepsilon$ és nincs más ε jobboldalú szabály. Ezt azért tehetjük fel, mert valahányszor az $A \to u$ szabályban $|u| \geq 2$ de u tartalmazza mondjuk az a terminális betűt, azt helyettesíthetjük az X_a új nemterminálissal, ahol bevesszük a szabályok közé az $X_a \to a$ szabályt is.

Ezek után minden $A \to A_1 \dots A_k$, $k \geq 3$ szabályt helyettesítünk az

$$A \to A_1 Y_1,\ Y_1 \to A_2 Y_2,\ \dots,\ Y_{k-3} \to A_{k-2} Y_{k-2},\ Y_{k-2} \to A_{k-1} A_k$$

szabályokkal, ahol $Y_1, \dots, Y_{k-2}$ csak az $A \to A_1 \dots A_k$ szabálytól függő új nemterminálosok. □

Ezzel beláttuk, hogy minden környezetfüggetlen nyelvtan Chomsky normálformára hozható.

A következőkben egy példát mutatunk be. Tekintsük az alábbi nyelvtant:

$$\begin{array}{lcl} S & \to & ASA \,|\, aB \\ A & \to & B \,|\, S \\ B & \to & b \,|\, \varepsilon \end{array}$$

Ekkor a megjelölt nemterminálisok az A és B lesznek. Így az első lépés után a következő ε-mentes nyelvtanhoz jutunk:

$$\begin{array}{lcl} S & \to & ASA \,|\, AS \,|\, SA \,|\, S \,|\, aB \,|\, a \\ A & \to & B \,|\, S \\ B & \to & b \end{array}$$

Ezek után $V_S = \{S\}$, $V_A = \{S, A, B\}$, $V_B = \{B\}$. Ezt felhasználva a második lépés után az alábbi nyelvtant kapjuk:

$$\begin{array}{lcl} S & \to & ASA \,|\, AS \,|\, SA \,|\, aB \,|\, a \\ A & \to & b \,|\, ASA \,|\, AS \,|\, SA \,|\, aB \,|\, a \\ B & \to & b \end{array}$$

Végül az utolsó lépés után (némi egyszerűsítéssel) kapjuk az alábbi két nyelvtant, melyek közül a második Chomsky-féle normálformában van:

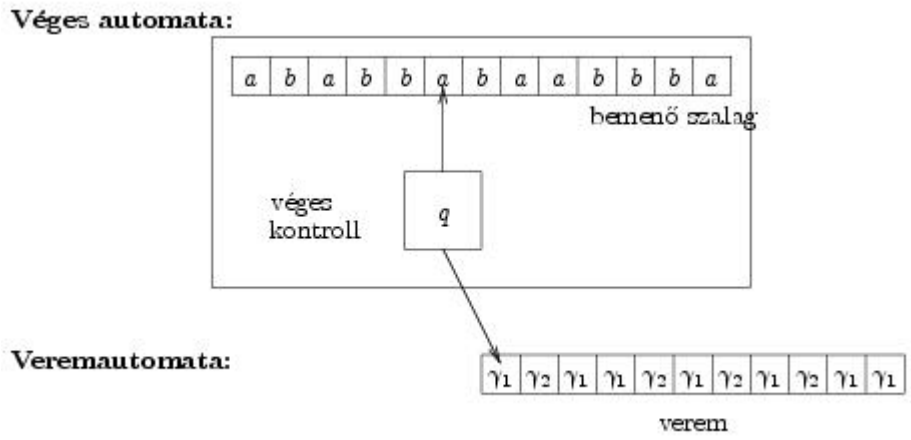
$$\begin{array}{lcl} S & \to & ASA \,|\, AS \,|\, SA \,|\, X_a B \,|\, a \\ A & \to & b \,|\, ASA \,|\, AS \,|\, SA \,|\, X_a B \,|\, a \\ X_a & \to & a \\ B & \to & b \end{array}$$

$$
\begin{array}{lcl}
S & \to & AY \,|\, AS \,|\, SA \,|\, X_aB \,|\, a \\
Y & \to & SA \\
A & \to & b \,|\, AY \,|\, AS \,|\, SA \,|\, X_aB \,|\, a \\
X_a & \to & a \\
B & \to & b
\end{array}
$$

2.5. Veremautomaták

Egy véges automata felfogható olyan diszkrét lépésekben működő gépként, amelynek van egy véges kontrollja és egy bemenőszalagja. A véges kontroll minden pillanatban véges sok állapot valamelyikében van. A szalagra kezdetben egy bemenő szó van felírva, és annak tartlamát a gép balról jobbra egy olvasófej segítségével olvassa. A következő lépést mindig az adott állapot és az olvasott jel határozza meg, determinisztikus automata esetén egyértelműen. Egy lépésben a gép állapotot válthat és az olvasófejet egy mezővel jobbra léptetheti. Ha spontán átmeneteket is megengedünk, akkor a gép attól függetlenül is állapotot válthat, hogy mely betűt olvassa a bemenő szalagon, és ekkor az olvasófej sem lép tovább. Kezdetben a gép a kezdőállapotában van. Akkor fogadja el a szalagra felírt szót, ha annak elolvasása után végállapotba kerül.

A *veremautomata* a véges automata számítási erejét egy potenciálisan végtelen, de korlátozott hozzáférésű verem szerkezetű memóriával növeli. Működése egy adott pillanatban annak is függvénye, hogy a verem tetején milyen szimbólum van, és egy átmenet során a verem tetején lévő szimbólumot törölheti, megváltoztathatja, vagy a verem tetejére egy újabb szimbólumot tárolhat. Kezdetben a verem üres.



Formálisan definiálva, veremautomata egy $(Q, \Sigma, \Gamma, \delta, q_0, F)$ rendszer, ahol Q, Σ, q_0, F ugyanazok, mint véges automata esetén,

- Γ véges, nemüres halmaz, a *verem abc*,
- $\delta : Q \times \Sigma_\varepsilon \times \Gamma_\varepsilon \to P(Q \times \Gamma_\varepsilon)$ az *átmenetfüggvény*.

Amennyiben $(q', \gamma') \in \delta(q, a, \gamma)$, ahol $q, q' \in Q$, $a \in \Sigma_\varepsilon$ és $\gamma, \gamma' \in \Gamma_\varepsilon$, a $((q, a, \gamma), (q', \gamma'))$ rendezett párt a veremautomata átmeneti szabályának nevezzük.

Az $M = (Q, \Sigma, \Gamma, \delta, q_0, F)$ veremautomata a következő módon működik. Akkor fogadja el a $w \in \Sigma^*$ szót, ha léteznek olyan $w_1, \ldots, w_m \in \Sigma_\varepsilon$, $r_0, \ldots, r_m \in Q$, $\alpha_0, \ldots, \alpha_m \in \Gamma^*$, amelyekre

1. $w = w_1 \ldots w_m$,
2. $r_0 = q_0$, $\alpha_0 = \varepsilon$,
3. Bármely $i < m$-re, $(r_{i+1}, \gamma') \in \delta(r_i, w_{i+1}, \gamma)$, ahol $\alpha_i = \gamma\beta$ és $\alpha_{i+1} = \gamma'\beta$ valamely $\gamma, \gamma' \in \Gamma_\varepsilon$, $\beta \in \Gamma^*$ esetén,
4. $r_m \in F$.

Jelölje $L(M)$ az M által elfogadott $w \in \Sigma^*$ szavak halmazát. $L(M)$-et az *M által felismert nyelv*nek nevezzük.

A veremautomata működése másként is megfogalmazható. Ehhez nevezzünk *konfigurációnak* egy $(q, \alpha, u) \in Q \times \Gamma^* \times \Sigma^*$ rendezett hármast. Azt mondjuk, hogy egy (q, α, u) konfigurációból közvetlenül el lehet jutni a (q', α', u') konfigurációba,

$$(q, \alpha, u) \vdash (q', \alpha', u'),$$

ha létezik olyan $((q, a, \gamma), (q', \gamma'))$ szabály és $\beta \in \Gamma^*$, hogy

$$u = au', \quad \alpha = \gamma\beta, \quad \alpha' = \gamma'\beta.$$

Továbbá azt mondjuk, hogy (q, α, u) konfigurációból el lehet jutni a (q', α', u') konfigurációba,

$$(q, \alpha, u) \vdash^* (q', \alpha', u'),$$

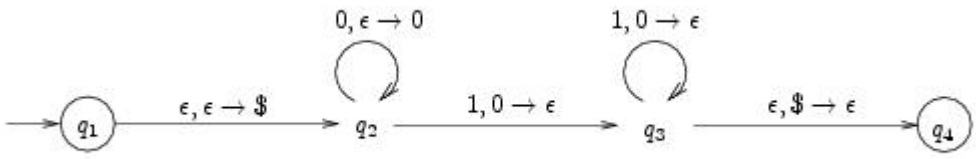
ha valamely (r_i, α_i, u_i), $i = 0, \ldots, n$ konfigurációkra $(r_0, \alpha_0, u_0) = (q, \alpha, u)$, $(r_n, \alpha_n, u_n) = (q', \alpha', u')$, és $(r_i, \alpha_i, u_i) \vdash (r_{i+1}, \alpha_{i+1}, u_{i+1})$ minden $i = 0, \ldots, n-1$ esetén. Tehát a $\vdash^*$ reláció a $\vdash$ reflexív-tranzitív burka. A $\vdash^*$ átmeneti reláció egy fontos tulajdonsága az, hogy amennyiben $(q, \alpha, u) \vdash^* (q', \alpha', u')$, akkor tetszőleges $\beta \in \Gamma^*$ és $v \in \Sigma^*$ szavakra fennáll a $(q, \alpha\beta, uv) \vdash^* (q', \alpha'\beta, u'v)$ reláció is. Végül

$$L(M) \quad = \quad \{u \in \Sigma^* : \exists q \in F \; \exists \alpha \in \Gamma^* \; (q_0, \varepsilon, u) \vdash^* (q, \alpha, \varepsilon)\}.$$

Példaként tekintsük az $L = \{0^n 1^n : n \geq 0\}$ nyelvet. Legyen $Q = \{q_1, q_2, q_3, q_4\}$, $\Sigma = \{0, 1\}$, $\Gamma = \{0, \$\}$, $F = \{q_1, q_4\}$. Az átmenetfüggvény az alábbi táblázattal adott (ahol az üresen hagyott helyek az üres halmaznak felelnek meg):

Bemenet	0			1			ϵ		
Verem	0	$	ϵ	0	$	ϵ	0	$	ϵ
q_1									$\{(q_2, \$)\}$
q_2			$\{(q_2, 0)\}$	$\{(q_3, \epsilon)\}$					
q_3				$\{(q_3, \epsilon)\}$				$\{(q_4, \epsilon)\}$	
q_4									

A veremautomatát az alábbi átmeneti diagrammal is megadhatjuk:



Itt a q_1-ből q_2-be vezető nyíl címkéje azt jelzi, hogy az a $((q_1,\varepsilon,\varepsilon),(q_2,\$))$ szabálynak felel meg. A q_2-ből q_2-be vezető $(0,\varepsilon\to 0)$-val címkézett él a $((q_2,0,\varepsilon),(q_2,0))$ szabálynak felel meg, stb. Átmenetekre példa:

$$(q_1,\varepsilon,0011)\vdash(q_2,\$,0011)\vdash(q_2,0\$,011)\vdash(q_2,00\$,11)\vdash(q_3,0\$,1)\vdash(q_3,\$,\varepsilon)\vdash(q_4,\varepsilon,\varepsilon)$$

Néhány megjegyzés.

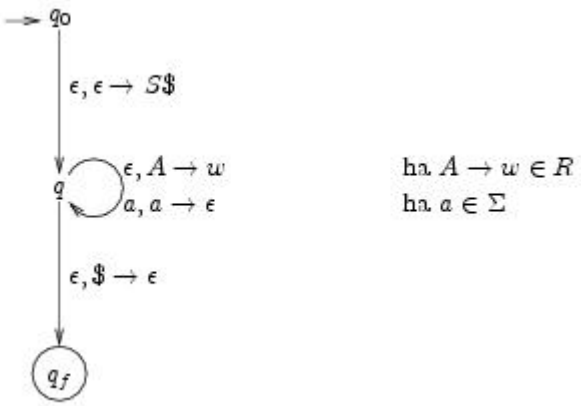
1. A veremautomata fogalmát módosíthatjuk úgy, hogy kezdetben a veremben egy rögzített Γ-beli betű legyen, és úgy is, hogy
2. a verembe az egyes átmenetek esetén 1-nél hosszabb szó is kerülhessen. Ekkor

   $\delta : Q\times\Sigma_\varepsilon\times\Gamma_\varepsilon\to P(Q\times\Gamma^*)$, és

   $\delta(q,a,\gamma)$ véges minden $q\in Q$, $a\in\Sigma_\varepsilon$, $\gamma\in\Gamma_\varepsilon$ esetén.
3. Megkövetelhetjük azt is, hogy egy szó elfogadásakor azon kívül, hogy végállapotban legyen a veremautomata, a verem kiürüljön.
4. A veremautomata *nemdeterminisztikus* modell.

2.10. Tétel. *Minden környezetfüggetlen nyelv felismerhető veremautomatával.*

Bizonyítás. Legyen $G=(V,\Sigma,R,S)$ tetszőleges környezetfüggetlen nyelvtan. Készítsük el az ábrán látható veremautomatát.



Először belátjuk, hogy ha $A \in V$, $u \in \Sigma^*$ és $A \Rightarrow^* u$, akkor $(q,A,u) \vdash^* (q,\varepsilon,\varepsilon)$. Tekintsünk az u-nak egy A-ból való n hosszú levezetését. Ha $n=1$, akkor $A \to u$ R-beli szabály. Legyen mondjuk $u = a_1 \dots a_m$, ahol mindegyik a_i a Σ-ban van. Ekkor:

$$(q,A,u) = (q,A,a_1 \dots a_m) \vdash (q,a_1 \dots a_m, a_1 \dots a_m) \vdash (q,a_2 \dots a_m, a_2 \dots a_m) \vdash \dots \vdash (q,a_m,a_m) \vdash (q,\varepsilon,\varepsilon).$$

Ha $n > 1$, akkor legyen $A \to u_0 B_1 \dots B_m u_m$ a levezetésben elsőként alkalmazott szabály. Ekkor u-t felírhatjuk $u = u_0 v_1 \dots v_m u_m$ alakban úgy, hogy minden i-re $B_i \Rightarrow^* v_i$ n-nél rövidebb levezetéssel. Az indukciós feltevés szerint így $(q,B_i,v_i) \vdash^* (q,\varepsilon,\varepsilon)$ minden i-re. Ezt felhasználva,

$$(q,A,u) \vdash (q,u_0 B_1 \dots B_m u_m, u_0 v_1 \dots v_m u_m) \vdash^* (q,B_1 u_1 \dots B_m u_m, v_1 u_1 \dots v_m u_m) \vdash^* (q,u_1 B_2 \dots B_m u_m, u_1 v_2 \dots v_m u_m) \vdash^* (q,B_2 u_2 \dots B_m u_m, v_2 u_2 \dots v_m u_m) \vdash^* \dots \vdash^* (q,B_m u_m, v_m u_m) \vdash^* (q,u_m,u_m) \vdash^* (q,\varepsilon,\varepsilon).$$

Így $u \in L(G)$ esetén

$$(q_0,\varepsilon,u) \vdash (q,S\$,u) \vdash^* (q,\$,\varepsilon) \vdash (q_f,\varepsilon,\varepsilon)$$

és $u \in L(M)$. Tehát $L(G) \subseteq L(M)$.

Most tegyük fel, hogy ha $(q,A,u) \vdash^* (q,\varepsilon,\varepsilon)$ n lépésben. Belátjuk, hogy $A \Rightarrow^* u$. Ha $n=1$, akkor $u = \varepsilon$ és $A \to \varepsilon$ R-beli szabály, tehát $A \Rightarrow^* \varepsilon = u$.

Legyen most $n > 0$, és tegyük fel, hogy állításunkat n-nél rövidebb átmenetekre igazoltuk. Az első lépésben valamely $A \to u_0 B_1 \dots B_m u_m$ szabályhoz tartozó $(q,A,u) \vdash (q,u_0 B_1 \dots B_m u_m, u)$ átmenetet alkalmaztunk és így

$$(q,u_0 B_1 \dots B_m u_m, u) \vdash^* (q,\varepsilon,\varepsilon)$$

n-nél kevesebb lépésben. Ez csak úgy lehet, ha u felírható $u = u_0 v_1 \dots v_m u_m$ alakban úgy, hogy minden $i = 0,\dots,m-1$ esetén

$$(q,u_i B_{i+1} \dots B_m u_m, u_i v_{i+1} \dots v_m u_m) \vdash^* (q,B_{i+1} u_{i+1} \dots B_m u_m, v_{i+1} u_{i+1} \dots v_m u_m) \vdash^* (q,u_{i+1} B_{i+2} \dots B_m u_m, u_{i+1} v_{i+2} \dots v_m u_m) \vdash^* (q,\varepsilon,\varepsilon),$$

továbbá

$$(q,B_{i+1},v_{i+1}) \vdash^* (q,\varepsilon,\varepsilon)$$

kevesebb, mint n lépésben. Az indukciós feltevésből adódik, hogy $B_j \Rightarrow^* v_j$, $j = 1,\dots m$. Így

$$A \Rightarrow u_0 B_1 \dots B_m u_m \Rightarrow^* u_0 v_1 \dots v_m u_m = u.$$

Most tegyük fel, hogy $u \in L(M)$, azaz $(q_0,\varepsilon,u) \vdash^* (q_f,\alpha,\varepsilon)$ valamely α-ra. Mivel az első lépésben a $\$$ jel a verem aljára kerül és az utolsó lépésig ott is marad, $\alpha = \varepsilon$ és részletesebben írhatjuk, hogy

$$(q_0,\varepsilon,u) \vdash (q,S\$,u) \vdash^* (q,\$,\varepsilon) \vdash (q_f,\varepsilon,\varepsilon),$$

továbbá

$$(q,S,u) \vdash^* (q,\varepsilon,\varepsilon).$$

Így $S \Rightarrow^* u$, tehát $u \in L(G)$. □

*Determinisztikus*nak nevezzük az $M = (Q,\Sigma,\Gamma,\delta,q_0,F)$ veremautomatát, ha

1. $|\delta(q,a,\gamma)| \leq 1, \quad q \in Q,\ a \in \Sigma_\varepsilon,\ \gamma \in \Gamma_\varepsilon,$

2. $\delta(q,\varepsilon,\gamma) \neq \emptyset \Rightarrow \delta(q,a,\gamma) = \emptyset. \quad q \in Q,\ a \in \Sigma,\ \gamma \in \Gamma_\varepsilon,$

3. $\delta(q,a,\varepsilon) \neq \emptyset \Rightarrow \delta(q,a,\gamma) = \emptyset, \quad q \in Q, a \in \Sigma_\varepsilon,\ \gamma \in \Gamma.$

Nem igaz az, hogy minden környezetfüggetlen nyelv felismerhető determinisztikus veremautomatával. Az $\{a^n b^n c^m, a^n b^m c^m : n,m \geq 0\}$ nyelv környezetfüggetlen, de nem determinisztikus.

2.11. Tétel. *Minden veremautomatával felismerhető nyelv környezetfüggetlen.*

Bizonyítás. Röviden vázoljuk a tétel bizonyítását. Először is könnyen megmutatható, hogy egy veremautomatával felismerhető nyelv felismerhető valamely veremautomatával úgy is, hogy egy bemenő szó elfogadásakor a verem mindig kiürül. Ezek után legyen $(Q,\Sigma,\Gamma,\delta,q_0,F)$ olyan veremautomata, amely eleget tesz ennek a feltételnek, és amely felismeri az $L \subseteq \Sigma^*$ nyelvet. Azt is feltehetjük, hogy valahányszor

$$((q,a,\gamma),(q',\gamma'))$$

szabály, akkor $\gamma = \varepsilon$ és $\gamma' \in \Gamma$, vagy $\gamma \in \Gamma$ és $\gamma' = \varepsilon$.

Egy olyan G környezetfüggetlen nyelvtant készítünk el, amely nemterminálisai az $X_{q,q'}$ szimbólumok, ahol $q,q' \in Q$, valamint az X_0 szimbólum. Célunk az, hogy az $X_{q,q'}$-ből levezethető terminális szavak azt az $L_{q,q'} \subseteq \Sigma^*$ nyelvet alkossák, amelyre $u \in L_{q,q'}$ akkor és csakis akkor, ha

$$(q,u,\varepsilon) \vdash^* (q',\varepsilon,\varepsilon).$$

Az $L_{q,q'}$ nyelvekre az alábbi rekurziós szabályok érvényesek.

1. $\varepsilon \in L_{q,q}$.

2. Ha $u \in L_{q,q'}$ és $v \in L_{q',q''}$, akkor $uv \in L_{q,q''}$.

3. Ha $(q_1,\gamma) \in \delta(q,a,\varepsilon)$, $q' \in \delta(q_2,b,\gamma)$ és $u \in L_{q_1,q_2}$, akkor $aub \in L_{q,q'}$.

Ennek megfelelően vesszük fel az $X_{q,q} \to \varepsilon$, $X_{q,q''} \to X_{q,q'}X_{q',q''}$ és az $X_{q,q'} \to aX_{q_1.q_2}b$ szabályokat, ahol persze az előzőekben adott feltételek érvényesek. Végül az X_0 baloldalú szabályok: $X_0 \to X_{q_0,q}$, ahol $q \in F$. □

A 2.10 és 2.11 tételek Chomsky klasszikus eredményei az 1960-as évek elejéről.

2.6. Környezetfüggetlen nyelvek pumpáló lemmája

Ebben a fejezetben a környezetfüggetlen nyelvek egy kombinatorikus tulajdonságát ismertetjük és annak felhasználásával belátjuk, hogy bizonyos nyelvek nem környezetfüggetlenek. Az alábbi eredmény Bar-Hillel lemmaként ismert az irodalomban.

2.2. Lemma. *Tetszőleges L környezetfüggetlen nyelvhez van olyan $p \geq 1$ szám, hogy valahányszor $w \in L$, $|w| \geq p$, a w szó mindig felbontható*

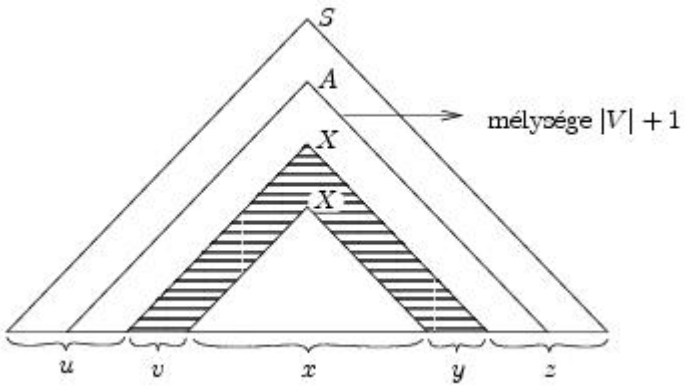
$$w = uvxyz$$

alakban úgy, hogy

1. $\forall i \geq 0 \quad uv^i xy^i z \in L$,
2. $|vy| > 0$,
3. $|vxy| < p$.

Bizonyítás. Legyen $L = L(G)$, ahol $G = (V, \Sigma, R, S)$ Chomsky normálformában adott. Vegyük észre, hogy amennyiben T egy nemterminálisból induló derivációs fa, melynek határa terminális szó és T mélysége legfeljebb $n+1$, akkor a T határán lévő szó hossza legfeljebb 2^n.

Legyen $p = 2^{|V|} + 1$. Ha $w \in L$, $|w| \geq p$, akkor a w szó egy S-ből induló T derivációs fájának mélysége legalább $|V|+1$. Tekintsünk T-ben egy maximális hosszú utat a gyökértől valamelyik levélig. Mivel ezen legalább $|V|+1$ nemterminálissal címkézett csúcs van, található két olyan azonos X nemterminálissal címkézett c_1 és c_2 csúcs úgy, hogy a c_1-hez tartozó T_1 részfa tartalmazza a c_2 csúcsot és mélysége legfeljebb $|V|+1$. Legyen T_2 a c_2 csúcshoz tartozó részfa és x a T_2 határa. Ekkor T_1 határa vxy alakban írható, ahol v és y a T_1 azon leveleinek címkéiből állnak, melyek nem a T_2 levelei. Ehhez hasonlóan w felírható $w = uvxyz$ alakban, ahol u és y betűi a T azon leveleinek címkéi, amelyek nem a T_2 csúcsai. Mivel $S \Rightarrow^* uXz$, $X \Rightarrow^* vXy$ és $X \Rightarrow^* x$, minden i-re fennáll, hogy $S \Rightarrow^* uv^i xy^i z$. Továbbá $|vxy| < p$ mivel T_1 mélysége legfeljebb $|V|+1$, és $vy \neq \varepsilon$ mivel G ε-mentes. □



A pumpáló lemma segítségével megmutathatjuk, hogy bizonyos nyelvek nem környezetfüggetlenek. A lemma két alkalamazával zárjuk ezt a fejezetet.

2.7. Állítás. *Az $L = \{a^n b^n c^n : n \geq 0\}$ nyelv nem környezetfüggetlen.*

Bizonyítás. Belátjuk, hogy bármely $p > 0$-hoz van olyan $w \in L$, $|w| \geq p$, hogy w tetszőleges olyan $w = uvxyz$ felbontására, amelyre $|vy| > 0, |vxy| < p$, létezik olyan i, hogy $uv^i xy^i z \notin L$.

Adott p-hez legyen $w = a^p b^p c^p$. Akárhogyan is bontjuk fel w-ét $w = uvxyz$ alakban úgy, hogy $|vy| > 0$, $|vxy| < p$, lesz olyan betű, amely nem fordul elő vy-ban. Így $uxz \notin L$.

2.8. Állítás. *Az* $L = \{w\#w : w \in \{0,1\}^*\}$ *nyelv nem környezetfüggetlen.*

Bizonyítás. Adott p-re legyen $w = 0^p1^p\#0^p1^p$, és tekintsük a w tetszőleges olyan $w = uvxyz$ felbontását, amelyre $vxy < p$. Ha # nem fordul elő x-ben, akkor $uv^2xy^2z \notin L$. Ha előfordul, akkor u-ban csak az 1, y-ban csak a 0 szerepelhet. Mivel $vy \neq \varepsilon$, ezért $uxz \notin L$ (és $uv^2xy^2z \notin L$). □

Ezzel szemben belátható, hogy az $L' = \{w\#w' : w,w' \in \{0,1\}^*,\ w \neq w'\}$ nyelv környezetfüggetlen.

2.4. Következmény. *A környezetfüggetlen nyelvek nem zártak a metszet és komplementerképzésre.*

Bizonyítás. Az

$$\begin{aligned} \{a^nb^nc^m : n,m \geq 0\} &= \{a^nb^n : n \geq 0\}\{c\}^* \\ \{a^nb^mc^m : n,m \geq 0\} &= \{a\}^*\{b^mc^m : m \geq 0\} \end{aligned}$$

nyelvek környezetfüggetlenek, de metszetük nem az. Mivel a környezetfüggetlen nyelvek zártak az egyesítésre de nem zártak a metszetre, ezért nem zártak a komplementerképzésre.□

Be lehet látni, hogy egy környezetfüggetlen és egy reguláris nyelv metszete mindig környezetfüggetlen.

3. fejezet

A Chomsky-féle hierarchia

Ebben a rövid fejezetben a környezetfüggetlen nyelvtan egy általánosítását ismerjük meg. Definiáljuk az általános (generatív) nyelvtan fogalmát, ahol a szabályok baloldala tetszőleges olyan szó, amely legalább egy nemterminálist tartalmaz. Az ilyen általános nyelvtanok által generált nyelvek alkotják a Chomsky-féle hierarchia legbővebb $\mathcal{L}_0$ nyelvosztályát.

A reguláris és a környezetfüggetlen nyelvek a $\mathcal{L}_0$ részosztályait alkotják, amelyeket $\mathcal{L}_3$-mal és $\mathcal{L}_2$-vel is jelölünk. A Chomsky-féle hierarchia az $\mathcal{L}_1$ nyelvosztállyal, a *környezetfüggő nyelv*ek osztályával válik teljessé, amely a $\mathcal{L}_2$ és $\mathcal{L}_0$ közé esik.

3.1. Általános nyelvtanok

(Általános) nyelvtan egy $G = (V, \Sigma, R, S)$ rendszer, ahol V, Σ, S ugyanazok, mint környezetfüggetlen nyelvtan esetén, R pedig $u \to v$ alakú szabályok véges halmaza, ahol $u, v \in (V \cup \Sigma)^*$ és u tartalmaz legalább egy nemterminálist. Legyen $G = (V, \Sigma, R, S)$ nyelvtan, $u, v \in (V \cup \Sigma)^*$. Azt mondjuk, hogy u-ból *közvetlenül levezethető* (vagy *deriválható*) a v szó, $u \Rightarrow v$, ha létezik olyan $x, y, y', z \in (V \cup \Sigma)^*$, hogy $u = xyz,\ v = xy'z,\ y \to y' \in R$. Mint korábban is, jelölje $\Rightarrow^*$ a $\Rightarrow$ reláció reflexív-tranzitív burkát, tehát $u \Rightarrow^* v$ akkor és csak akkor, ha létezik olyan $n \geq 0$ és $w_0, w_1, \ldots, w_n \in (V \cup \Sigma)^*$, hogy $u = w_0,\ w_0 \Rightarrow w_1, \ldots, w_{n-1} \Rightarrow w_n,\ w_n = v$. A G *által generált nyelv*: $L(G) = \{u \in \Sigma^* : S \Rightarrow^* u\}$.

Példaként tekintsük az alábbi G nyelvtant, mely az $L(G) = \{a^n b^n c^n : n \geq 0\}$ nyelvet generálja.

$$\begin{array}{lcl} S & \to & aSBc \mid \varepsilon \\ cB & \to & Bc \\ aB & \to & ab \\ bB & \to & bb \end{array}$$

Egy G-vel ekvivalens (azaz ugyanazt a nyelvet generáló nyelvtan) az alábbi G':

$$\begin{array}{lcl} S_0 & \to & S \mid \varepsilon \\ S & \to & aSBC \mid aBC \\ CB & \to & XB \\ XB & \to & XC \\ XC & \to & BC \\ aB & \to & ab \\ bB & \to & bb \\ C & \to & c \end{array}$$

*Környezetfüggő*nek nevezünk egy $G = (V, \Sigma, R, S)$ nyelvtant, ha R minden szabálya $uXv \to uwv$ alakú, ahol u, v, w $(V \cup \Sigma)^*$-beli szavak, $X \in V$, $w \neq \varepsilon$. Ez alól csak egyetlen kivétel lehet, nevezetesen az $S \to \varepsilon$ szabály. Ha ez R-ben van, akkor kikötjük azt, hogy S nem fordul elő egyetlen szabály jobboldalán sem. Az előző példában szereplő G' nyelvtan környezetfüggő. Egy $L \subseteq \Sigma^*$ nyelvet környezetfüggőnek nevezünk, ha létezik olyan környezetfüggő $G = (V, \Sigma, R, S)$ nyelvtan, melyre $L = L(G)$. Az $\{a^n b^n c^n : n \geq 0\}$ nyelv tehát környezetfüggő. Mivel minden környezetfüggetlen nyelvtan Chomsky-féle normálformára hozható, ezért minden környezetfüggetlen nyelv környezetfüggő nyelv is. A

$$\{0^{n^2} : n \geq 0\},\ \{0^{2^n} : n \geq 0\},\ \{0^p : p \text{ prímszám}\},$$
$$\{w\#w : w \in \{0,1\}^*\},\ \{u \in \{a,b\}^+ : \forall v \forall n [v^n = u \Rightarrow v = u]\}$$

nyelvek mindegyike környezetfüggő, és az utolsó kivételével egyik sem környezetfüggetlen. Nem ismert, hogy az utolsó *primitív* szavak nyelve környezetfüggetlen-e.

Egy környezetfüggő nyelvtanban az esetleges $S \to \varepsilon$ szabályon kívül, ahol S a kezdőszimbólum, minden szabály jobboldala legalább olyan hosszú, mint baloldala. Ismert, hogy minden olyan általános nyelvtan, melyben minden szabály jobboldala legalább olyan hosszú, mint a baloldala, környezetfüggő nyelvet generál.

Korábban jobblineárisnak neveztünk egy $G = (V, \Sigma, R, S)$ nyelvtant, ha R minden szabálya $A \to uB$ vagy $A \to u$ alakú, ahol $A, B \in V, u \in \Sigma^*$. Egy nyelvet *jobblineárisnak nevezünk*, ha generálható jobblineáris nyelvtannal.

3.9. Állítás. *Egy nyelv akkor és csak akkor jobblineáris, ha reguláris.*

Bizonyítás. Azt már korábban beláttuk, hogy minden reguláris nyelv jobblineáris, megmutattuk ugyanis, hogy minden véges nemdeterminisztikus (spontán átmenetekkel rendelkező) M automata felfogható egy G_M jobblineáris nyelvtanként, amelyre $L(M) = L(G_M)$.

Legyen most $L \subseteq \Sigma^*$ jobblineáris nyelv és $G = (V, \Sigma, R, S)$ olyan jobblineáris nyelvtan, melyre $L = L(G)$. Mivel egy $X \to a_1 \ldots a_n Y$, $n > 1$ alakú szabályt helyettesíthetünk az $X \to a_1 X_1,\ X_1 \to a_2 X_2,\ \ldots, X_{n-2} \to a_{n-1} X_{n-1},\ X_{n-1} \to a_n Y$ szabályokkal, ahol $a_1, \ldots, a_n \in \Sigma$, $X, Y \in V$ és $X_1, \ldots, X_{n-1}$ új nemterminálisok, és mivel minden $X \to b_1 \ldots b_m$, $m \geq 1$ alakú szabályt helyettesíthetünk az $X \to b_1 Y_1,\ Y_1 \to b_2 Y_2,\ \ldots,\ Y_{m-1} \to b_m Y_m,\ Y_m \to \varepsilon$ szabályokkal, ahol $b_1, \ldots, b_m \in \Sigma$, $Y \in V$ és $Y_1, \ldots, Y_m$ új nemterminálisok, ezért feltehető, hogy R minden szabálya $X \to aY$ vagy $X \to \varepsilon$ alakú, ahol $X, Y \in V$ és $a \in \Sigma_\varepsilon$.

Tekintsük az $M = (V, \Sigma, \delta, S, V')$ véges nemdeterminisztikus spontán átmenetekkel rendelkező automatát, melyben teszőleges $X, Y \in V$ és $a \in \Sigma_\varepsilon$ esetén $\delta(X, a) = \{Y : X \to aY \in R\}$ és $X \in V'$ akkor és csak akkor, ha $X \to \varepsilon \in R$. Ekkor $G = G_M$, tehát $L = L(M)$. Mivel L tetszőleges jobblineáris nyelv volt, így minden jobblineáris nyelv reguláris. □

Megjegyezzük, hogy a jobblináris nyelvtanhoz hasonló módon definiálhattuk volna a ballineáris nyelvtanokat és az általuk generált ballineáris nyelveket is. Ha egy G jobblineáris nyelvtan minden $X \to u$ szabályát az $X \to u^{-1}$ szabállyal helyettesítjük, akkor az előálló G^{-1} ballineáris nyelvtan az $L(G)$ tükörképét generálja. Mivel egy nyelv akkor és csak akkor reguláris, ha tükörképe az, így kapjuk azt, hogy egy nyelv akkor és csakis akkor reguláris, ha ballineáris.

Egy jobblineáris nyelvtant 3-as típusúnak, egy környezetfüggetlen nyelvtant 2-es típusúnak, egy környezetfüggő nyelvtant 1-es típusúnak, általános nyelvtant pedig 0-ás típusúnak is nevezünk. Legyen $i \in \{0, 1, 2, 3\}$. Egy $L \subseteq \Sigma^*$ nyelvet *i típusú*nak nevezünk, ha generálható i típusú nyelvtannal. Jelölje $\mathcal{L}_i$ az i-típusú nyelvek osztályát. Amennyiben $i = 3, 2, 1$, $\mathcal{L}_i$ rendre a jobblineáris, környezetfüggetlen és környezetfüggő nyelvek osztálya. Az $\mathcal{L}_0$ nyelvosztályt a *kifejezés struktúrájú nyelv*ek osztályának is nevezik. Az $\mathcal{L}_i$, $i \in \{0, 1, 2, 3\}$ nyelvosztályok alkotják a *Chomsky-féle hierarchiát*:

3.12. Tétel. $\mathcal{L}_3 \subseteq \mathcal{L}_2 \subseteq \mathcal{L}_1 \subseteq \mathcal{L}_0$. *Továbbá minden tartalmazás valódi.*

Bizonyítás. Az $\mathcal{L}_1 \subseteq \mathcal{L}_0$ és $\mathcal{L}_3 \subseteq \mathcal{L}_2$ tartalmazási relációk nyilvánvalóak. Már beláttuk, hogy $\mathcal{L}_2 \subseteq \mathcal{L}_1$. A $\{0^n 1^n : n \geq 0\}$ nyelv $\mathcal{L}_2 \setminus \mathcal{L}_3$-ban, az $\{a^n b^n c^n : n \geq 0\}$ nyelv pedig $\mathcal{L}_1 \setminus \mathcal{L}_2$-ben van. Később belátjuk, hogy $\mathcal{L}_0 \setminus \mathcal{L}_1$ nem üres. □

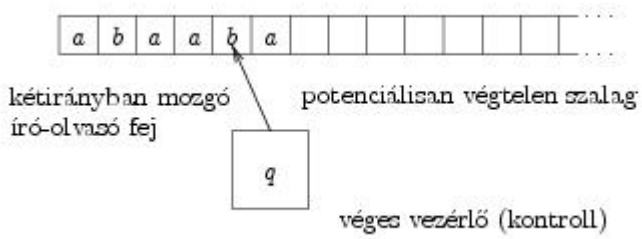
4. fejezet

Kiszámíthatóságelmélet

Ebben a fejezetben bevezetjük a Turing-gépet, melyet nemcsak nyelv felismerő eszközként használunk, hanem segítségével formalizáljuk az algoritmus fogalmát. Egy problémát algoritmikusan megoldhatónak nevezünk, ha Turing-géppel megoldható.

4.1. Turing-gépek

A Turing-gépet Alan Turing vezette be 1936-ban. A Turing-gép a véges automatának egy potenciálisan végtelen, korlátlan hozzáférésű tárral való bővítése.



Mielőtt pontosan definiálnánk a Turing-gépet, példaként megmutatjuk, hogy az $L = \{w\#w : w \in \{a,b\}^*\}$ nyelv felismerhető Turing-géppel.

1. A bemenet egyszeri elolvasásával ellenőrizzük, hogy az a, b betűkön kívül egyetlen #-et tartalmaz.

2. A szalagot oda-vissza olvasva ellenőrizzük (a vizsgált betűk áthúzásával), hogy a # két oldalán ugyanaz a szó helyezkedik-e el.

3. Amennyiben végig egyezést találunk (az összes betű áthúzásra került), akkor a gép elfogadja, különben elutasítja a bemenetet.

Turing-gép egy $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ rendszer, ahol:

1. Q az *állapotok* véges, nemüres halmaza,

2. $q_0 \in Q$ a *kezdőállapot*,

3. $q_{accept} \in Q$ az *elfogadó állapot*,

4. $q_{reject} \in Q$ az *elutasító állapot*,

5. Σ véges, nemüres halmaz, a *bemenő jelek* (szimbólumok) abc-je,

6. Γ a Σ-át tartalmazó véges halmaz, a *szalag abc*; a $\Gamma \setminus \Sigma$ halmaz nemüres, tartalmaz egy speciális $\sqcup$ karaktert, továbbá $\Gamma \cap Q = \emptyset$,

7. $\delta : (Q \setminus \{q_{accept}, q_{reject}\}) \times \Gamma \to Q \times \Gamma \times \{L, R\}$ az *átmenetfüggvény*.

A Turing-gép működését *konfiguráció*k segítségével írjuk le. Konfiguráció egy uqv alakú szó, ahol $q \in Q$, $u, v \in \Gamma^*$, $v \neq \varepsilon$. Az uqv konfiguráció a Turing-gép működésének egy olyan pillanatát adja meg, amelyben az egyirányban végtelen szalag tartalma $uv\sqcup\sqcup\ldots$, tehát uv után a szalag minden mezője a $\sqcup$ üres betűt tartalmazza, a véges kontroll a q állapotban van, az író-olvasó fej pedig v első betűjét jelöli ki.

Legyen uqv konfiguráció, ahol v első betűje a. Tegyük fel, hogy $q \notin \{q_{accept}, q_{reject}\}$ és $\delta(q,a) = (r,b,R)$. Ekkor az uqv konfigurációból a Turing-gép *közvetlenül átmehet* az $ubrw$ konfigurációba, $uqv \vdash ubrw$, ahol w a v-ből az a betű elhagyásával kapott szó, ha ez nemüres, különben $w = \sqcup$. Ha $\delta(q,a) = (r,b,L)$, akkor $uqv \vdash wrcbv'$, ahol $u = wc$, $v = av'$, $c \in \Gamma$, illetve $uqv \vdash urbv'$, ha $u = \varepsilon$ és $v = av'$. Ha $q = q_{accept}$ vagy $q = q_{reject}$, akkor az uqv konfiguráció *megállási konfiguráció*, az első esetben *elfogadó*, a második esetben *elutasító* konfiguráció. Világos, hogy minden nem megállási konfigurációból pontosan egy konfigurációba mehet át a gép.

Legyen a $\vdash^*$ *átmeneti reláció* a $\vdash$ reflexív-tranzitív burka. Egy konfigurációkból álló $u_1q_1v_1 \vdash \ldots \vdash u_nq_nv_n$ sorozatot *számítási sorozat*nak nevezünk.

Legyen M a fenti Turing-gép. Az M *által felismert* $L(M) \subseteq \Sigma^*$ *nyelv* az összes olyan $u \in \Sigma^*$ szóból áll, amelyre $q_0u\sqcup \vdash^* xq_{accept}y$ valamely $x, y \in \Gamma^*$, $y \neq \varepsilon$ szavakra, azaz a Turing-gép az *u-hoz tartozó kezdő konfiguráció*ból eljuthat egy elfogadási konfigurációba. Egy $L \in \Sigma^*$ nyelv *Turing felismerhető*, vagy *rekurzívan felsorolható*, ha $L = L(M)$ valamely M Turing-gépre. Két Turing-gépet *ekvivalens*nek nevezünk, ha ugyanazt a nyelvet ismerik fel.

A Turing-gép nem feltétlenül áll meg egy u szón, azaz nem biztos, hogy a $q_0u\sqcup \vdash^* xq_{accept}y$ vagy $q_0u\sqcup \vdash^* xq_{reject}y$ relációk valamelyike teljesül valamely x, y esetén. (Persze csak az egyik teljesülhet.) Egy $L \subseteq \Sigma^*$ nyelv *(Turing-)eldönthető*, vagy *rekurzív*, ha létezik olyan M Turing-gép, mely minden bemeneten megáll és felismeri az L nyelvet.

Röviden megemlítjük a Turing-gép néhány variánsát. Megengedhettük volna azt is, hogy a Turing-gép egy lépésben az író-olvasó fejet ne mozgassa a szalagon, hiszen a fej helyben maradása megvalósítható úgy, hogy először jobbra lép, majd balra. Azt is megengedhettük volna egy újabb állapot bevezetésével, hogy δ parciális függvény legyen. *A többszalagos Turing-gép* több potenciálisan végtelen szalaggal rendelkezik, melyek közül az első tartalmazza a bemenő szót és a többit részeredmények tárolására használja. Minden munkaszalaghoz egy külön író-olvasó fej tartozik és az átmenetek a munkaszalagokon olvasott mezők tartalmától is függnek. Egy lépésben a Turing-gép átírhatja minden szalagon az olvasott mezőt és a fejet mindkét irányban mozgathatja (kivéve, amikor a fej az első pozíción van). Kezdetben a

munkaszalagok mindegyik mezőjén $\sqcup$ van. Amennyiben a szalagok száma k, δ egy

$$\delta : (Q \setminus \{q_{accept}, q_{reject}\}) \times \Gamma^k \quad \rightarrow \quad Q \times \Gamma^k \times \{L,R\}^k$$

leképezés.

Nem nehéz belátni, hogy minden többszalagos Turing-gép szimulálható egyszalagos Turing-géppel. Ha M egy k-szalagos Turing-gép és $k \geq 2$, akkor egy M-et szimuláló N Turing-gép egyetlen szalagját k sávra osztjuk. Az M szalagjainak tartalmát sávonként tároljuk. Formálisan ez azt jelenti, hogy N szalag abc-je rendezett k-asokat is tartalmaz. Mivel N-nek csak egy író-olvasó feje van, ezert minden sávban meg kell jelölni azt a pozíciót, ahol az M író-olvasó feje áll az adott pillanatban. A szimuláció úgy történik, hogy először N végigolvassa a szalagot és közben az állapotában tárolja azt, hogy az egyes sávokon milyen betűk vannak megjelölve, majd ellentétes irányban is végigolvassa a szalagot és elvégzi a megfelelő átalakítást a szalagon.

A *két irányban végtelen szalag*gal rendelkező Turing-gép szalagjának mezői az egész számokkal indexelhetők. Kezdetben az író-olvasó fej a 0 sorszámú mezőn van és negatív tartományba is elmozdulhat. Egy két irányban végtelen szalaggal rendelkező Turing-gépet szimulálhatunk kétszalagos Turing-géppel, ahol a munkaszalagon a negatív tartományban lévő mezőket tároljuk.

Az eddigi Turing-gép modell *determinisztikus* volt, és a továbbiakban, ha ezt hangsúlyozni szeretnénk, ezt a modellt determinisztikus Turing-gépnek hívjuk. *Nemdeterminisztikus Turing-gép* egy olyan $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$, rendszer, ahol

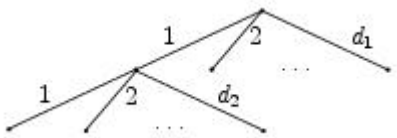
$$\delta : (Q \setminus \{q_{accept}, q_{reject}\}) \times \Gamma \quad \rightarrow \quad P(Q \times \Gamma \times \{L,R\}).$$

Működését ugyanúgy definiálhatjuk a konfigurációkon értelmezett $\vdash$ és $\vdash^*$ relácók felhasználásával, mint determinisztikus esetben, de egy konfigurációból több konfigurációba is átmehet közvetlenül a gép. A nemdeterminisztikus Turing-gép működését egy u szón egy véges vagy végtelen számítási fával ábrázolhatjuk, ahol minden egyes csúcs egy konfigurációval címkézett. A gyökér címkéje az u szóhoz tartozó kezdőkonfiguráció, és ha egy csúcs címkéje a c konfiguráció, akkor közvetlen leszármazottainak címkéi azon c' konfigurációk, amelyekre $c \vdash c'$. Akkor fogadja el a Turing-gép az u szót, ha a fa valamely levele elfogadási konfiguráció, vagyis ha $qu\sqcup \vdash^* xq_{accept}y$ valamely x, y szavakra. Az M által felismert nyelv

$$L(M) = \{u \in \Sigma^* : \exists x, y \quad q_0 u \sqcup \vdash^* x q_{accept} y\}.$$

4.13. Tétel. *Ha egy L nyelv felismerhető nemdeterminisztikus Turing-géppel, akkor L Turing felismerhető.*

Bizonyítás. Legyen adott az $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$ nemdeterminisztikus Turing-gép. Elkészítünk egy olyan D többszalagos determinisztikus Turing-gépet, melyre $L(D) = L(M)$. D szélességi kereséssel megvizsgálja adott u bemenő szón az M számítási fáját. Ha olyan csúcsot talál, melynek címkéje elfogadási konfiguráció, akkor elfogadja az u szót. Különben D nem áll meg az u szón. D-nek 3 szalagja van: a bemenő szalag, a szimulációs szalag, és a cím szalag. Az utóbbin a számítási fa éppen vizsgált csúcsának címe van, amely egy az $1, \ldots, d$ számok feletti szó, ahol d a legnagyobb $\delta(q, a, Z)$ alakú halmaz számossága. A címzést ábrázolja az alábbi ábra:



A kezdő cím a gyökér címe, azaz az üres szó. Amint meghatározásra kerül a következő cím, D a szimulációs szalagján előállítja a megfelelő konfigurációt. Ha ez elfogadó konfiguráció, akkor elfogadja a bemenetet. Különben előállítja a számítási fa következő konfigurációját, vagy ha ilyen nincs, elutasítja a bemenetet. □

Azt mondjuk, hogy egy determinisztikus vagy nemdeterminisztikus Turing-gép *eldönti* az L nyelvet, ha felismeri azt, és ha minden számítási sorozata véges és elfogadáshoz vagy elutasításhoz vezet. Kőnig lemmáját felhasználva kapjuk azt, hogy az a feltétel, hogy minden számítási sorozat véges, ekvivalens azzal, hogy minden számítási fa véges.

4.5. Következmény. *Ha egy nyelv eldönthető nemdeterminisztikus Turing-géppel, akkor eldönthető determinisztikus Turing-géppel.*

Bizonyítás nélkül közöljük az alábbi tételt, ami azon az egyszerű tényen alapszik, hogy nyelvtant szimulálhatunk nemdeterminisztikus Turing-géppel és fordítva.

4.14. Tétel. *Egy L nyelv akkor és csakis akkor Turing-felismerhető, ha az $\mathcal{L}_0$ nyelvosztályba esik.*

A Turing-gépet felhasználhatjuk egy nyelv szavainak felsorolására is. Azt mondjuk, hogy az M 2-szalagos (vagy többszalagos) determinisztikus Turing-gép *felsorolja az L nyelvet*, ha üres bemeneti szalaggal indítva egy másik szalagján, amelyre csak balról jobbra írhat, esetleg ismétlésekkel felsorolja az L elemeit.

4.15. Tétel. *Egy L nyelv akkor és csakis akkor sorolható fel Turing-géppel, ha Turing felismerhető.*

Bizonyítás. Legyen E az $L \subseteq \Sigma^*$ nyelvet felsoroló Turing-gép. Ehhez elkészítünk egy, az L-et felismerő M (3 szalagos) determinisztikus Turing-gépet.

Adott w bemenő szón M szimulálja munkaszalagján az E működését az üres szón. Ha E kimenő szalagjára kiír egy szót, akkor azt M összehasonlítja bemenő szavával. Ha a két szó megegyezik, elfogadással megáll.

Most legyen M az L-et felismerő determinisztikus Turing-gép. Működjön az L-et felsoroló E Turing-gép az alábbiak szerint. Rendezzük Σ^* szavait hosszuk szerint, és azon belül lexikografikusan, mondjuk $\Sigma^* = \{w_1, w_2, \ldots\}$. E $i = 1, 2, 3, \ldots$ esetén végtelen ciklusban futtatja M-et legfeljebb i lépésig az első i szó mindegyikén. Ha valamelyik w_j szót legfeljebb i lépésben M elfogadja, akkor E kiírja w_j-t a kimenő szalagra. □

Végül a Turing-gépet felhasználhatjuk (parciális) függvények kiszámítására is. Tegyük fel, hogy az M többszalagos determinisztikus Turing-gép egyik munkaszalagja kimenő szalag. Azt mondjuk, hogy M kiszámítja az $f : \Sigma^* \to \Delta^*$ parciális függvényt (ahol a szalag abc tartalmazza a Σ mellett a Δ-t is), ha adott w szón akkor és csak akkor áll meg a q_{accept} állapotban, ha $f(w)$ értelmezett, és megálláskor a kimenő szalag tartalma $f(w)$. Egy $f : \Sigma^* \to \Delta^*$

parciális függvényt *parciális rekurzív függvény*nek nevezünk, ha létezik olyan Turing-gép, mely kiszámítja. Egy *rekurzív* vagy *kiszámítható* függvény az egész Σ^*-on értelmezett parciális rekurzív $\Sigma^* \to \Delta^*$ függvény.

4.2. Eldönthető problémák nyelvekre

Bár a Turing-gépet mint nyelveket felismerő és eldöntő eszközt vezettük be, ez ne tévessze meg az Olvasót, a fogalom ennél sokkal érdekesebb.

Egy *eldöntési probléma* egy véges, vagy végesen megadható objektumokra megfogalmazott tulajdonság, és a feladat annak eldöntése, hogy egy adott objektum rendelkezik-e a megadott tulajdonsággal. Amennyiben rendelkezik, az a probléma *igaz* példánya, ellenkező esetben *hamis* példánya. Minden véges, vagy végesen megadható objektumot, természtes számot, véges halmazt vagy gráfot, véges áramkört stb. valamely abc feletti szóval kódolhatunk. Így egy nyelv egy eldöntési probléma *igaz* példányait reprezentálja, és a nyelvet eldöntő Turing-gép programja (szabályai) algoritmust adnak a probléma megoldására. A kódolás lényegében tetszőleges algoritmikusan megvalósítható módszer az objektumok szavakkal való reprezentálására. Az, hogy egy probléma *igaz* példányai eldönthető nyelvet alkotnak azt jelenti, hogy a probléma megoldható Turing-géppel. Ebben a fejezetben több Turing-géppel megoldható problémát és megoldásukat ismertetjük.

Első példaként egy algoritmust mutatunk be gráfok öszefüggőségének eldöntésére. Pontosabban, a feladat az elérhetőség, ahol adott agy G véges irányított gráf és annak két csúcsa, s és t. Az algoritmus arra a kérdésre válaszol, hogy el lehet-e s-ből jutni valamely irányított úton a t-be. A módszer a következő:

- Jelöljük meg az s csúcsot.
- Mindaddig, amíg létezik olyan $x \to y$ él, hogy x megjelölt de y nem, jelöljük meg az y csúcsot.
- Ha t megjelölésre kerül, akkor a válasz *igen*. Ha már nem jelölhető meg újabb csúcs és t nem került korábban megjelölésre, akkor a válasz *nem*.

Az algoritmus megvalósítható egy többszalagos determinisztikus Turing-géppel. Ennek bemenő szava egy (G,s,t) hármas kódja, amelyet mondjuk $\langle G,s,t\rangle$-vel jelölünk. Több lehetőség adódik a kódolás megválasztására. Megadható a G szomszédsági mátrixa soronként, ahol a sorok egy elválasztó jellel különíthetők el, amelyet az s majd a t sorszáma követ binárisan. Egy másik lehetőség az, hogy megadjuk a csúcsok számát binárisan, majd felsoroljuk az éleket. Minden élt egy számpárral reprezentálunk. Végül megadjuk binárisan az s és t sorszámát. A Turing-gép egy munkaszalagon tartja nyilván a megjelölt csúcsokat. Egy lehetőség az, hogy erre egy n-hosszú szót használ a $\{0,1\}$ halmaz felett, ahol n a csúcsok száma és az 1 bejegyzés azt jelzi, hogy a megfelelő csúcs meg van jelölve. Kezdetben az s csúcshoz tartozó bit 1, a többi 0. Minden egyes menetben végigmegy a gráf leírásán, és újabb csúcsokat jelöl meg. Közben nyilvántartja, hogy az adott menetben jelölt-e meg újabb csúcsot. Ha a t csúcs negjelölésre kerül, akkor a bemenetet elfogadja. Ha egy menetben nem kerül újabb csúcs megjelölésre, akkor elutasítja.

Most tekintsük azt a problémát, amely adott $L \subseteq \Sigma^*$ reguláris nyelvről és $w \in \Sigma^*$ szóról azt kérdezi, hogy $w \in L$ teljesül-e. Azt feltehetjük, hogy Σ az $\{a_1, \ldots, a_n\}$ betűkből áll valemely n-re, és minden a_i-t kódolhatunk az $a\bar{i}$ szóval, ahol $\bar{i}$ az i bináris reprezentációja. Így w megadható egy szóval az $\{a, 0, 1\}$ abc felett, amelyet jelöljünk $\langle w \rangle$-vel. Az L megadására több lehetőségünk van. Használhatunk véges determinisztikus vagy nemdeterminisztikus automatát, vagy reguláris kifejezést is. Amennyiben véges determinisztikus automatát használunk, ahhoz a problémához jutunk, hogy adott $M = (Q, \Sigma, \delta, q_0, F)$ véges determinisztikus automatára és $w \in \Sigma^*$ szóra döntsük el, hogy $w \in L(M)$ teljesül-e. Mindig feltehetjük, hogy az automata állapothalmaza a $q_0, \ldots, q_m$ állapotokat tartalmazza valamely $m \geq 0$ számra, ahol q_0 a kezdőállapot. Így az állapotok halmaza megadható az m szám bináris reprezentációjával. Hasonlóan, Σ megadható a betűk n számának bináris reprezentációjával. Végül ha az állapotok száma $m+1$ és $|\Sigma| = n$, akkor δ megadható n darab szóval. Amennyiben az i-dik szó $v_i = (q\overline{j_0} \ldots q\overline{j_m})$, akkor ez azt jelenti, hogy $\delta(q_0, a_i) = q_{j_0}, \ldots, \delta(q_m, a_i) = q_{j_m}$. (Itt $\overline{j_0}$ a j_0 szám bináris alakja, stb.) Végül az (M, w) pár kódja, amit $\langle M, w \rangle$ jelöl, az $(\overline{m}; \overline{n}; v_1; \ldots; v_n); \langle w \rangle)$ szó az $\{a, q, 0, 1,), (, ;\}$ abc felett (amely betűit persze tovább kódolhatjuk a bináris abc felett). Így annak a problémának, hogy adott M véges determinisztikus automatára és annak w bemenő szavára döntsük el, hogy $w \in L(M)$ teljesül-e, az

$$L_{DFA} = \{\langle M, w \rangle : M \text{ véges det. automata, } w \in L(M)\}$$

nyelv felel meg. Természetesen választhattunk volna más kódolást is, pld. az átmeneti függvényt megadhattuk volna úgy is, hogy felsoroljuk az összes olyan $(q\bar{i}, a\bar{j}, q\bar{k})$ rendezett hármast, amelyre $\delta(q_i, a_j) = q_k$. Ez azonban nem befolyásolná a következő állításunk érvényességét:

4.10. Állítás. *L_{DFA} eldönthető nyelv.*

Bizonyítás. Egy T Turing-gép először ellenőrzi, hogy a bemenő szó egy $\langle M, w \rangle$ alakú szó-e, majd:

1. átmásolja M kódját egy munkaszalagjára,
2. egy másik munkaszalagján szimulálja M működését a w szón. □

Az előző problémában determinisztikus automata helyett nemdeterminisztikus automatát véve azt a problémát kapjuk, amelyet az

$$L_{NFA} = \{\langle M, w \rangle : M \text{ nemdet. véges automata, } w \in L(M)\}$$

nyelv reprezentál. (Itt és a továbbiakban eltekintünk a kódolás részleteinek megadásától, mert az nem befolyásolja eredményeinket.)

4.11. Állítás. *L_{NFA} eldönthető nyelv.*

Bizonyítás. Visszavezetjük ezt a problémát az előzőre. Egy T Turing-gép először ellenőrzi, hogy a bemenő szó $\langle M, w \rangle$ alakú-e, ahol M véges nemdeterminisztikus automata, w az M bemenő szava, majd

- a hatványhalmaz konstrukcióval elkészít egy olyan M' determinisztikus véges automatát, ill. annak kódját, mely M-el ekvivalens,
- majd szimulálja az L_{DFA} nyelvet eldöntő Turing-gépet az $\langle M', w\rangle$ bemeneten. □

Egy harmadik változatban reguláris kifejezést használunk. A problémát az alábbi nyelv reprezentálja:

$$L_{REX} = \{\langle R, w\rangle : R \text{ reguláris kifejezés, } w \in |R|\}.$$

4.12. Állítás. *Az L_{REX} nyelv eldönthető.*

Bizonyítás. A T Turing-gép először ellenőrzi, hogy a bemenet $\langle R, w\rangle$ alakú-e, ahol R reguláris kifejezés és w szó, majd

- elkészít egy olyan M véges, nemdeterminisztikus automatát, melyre $|R| = L(M)$, és
- az L_{NFA} nyelvet eldöntő Turing-gépet szimulálva eldönti, hogy $w \in L(M)$ (azaz $\langle M, w\rangle \in L_{NFA}$) teljesül-e. □

Most tekintsük azt a problémát, hogy adott nyelvtanra és szóra döntsük el, hogy a szó a nyelvtan által generált nyelvbe esik-e. Később belátjuk, hogy általános nyelvtan esetén ez a probléma nem dönthető el Turing-géppel. Speciális esetekben viszont igen. Legyen

$$L_{CFG} = \{\langle G, w\rangle : G \text{ környezetfüggetlen nyelvtan, } w \in L(G)\}$$

$$L_{CSG} = \{\langle G, w\rangle : G \text{ környezetfüggő nyelvtan, } w \in L(G)\}.$$

4.13. Állítás. *L_{CSG} eldönthető.*

Bizonyítás. Egy T Turing-gép először eldönti, hogy bemenete $\langle G, w\rangle$ alakú-e, ahol G egy (V, Σ, R, S) környezetfüggő nyelvtan és $w \in \Sigma^*$, majd ha ez teljesül, akkor két eset lehetséges.

- Ha $w = \varepsilon$, ellenőrzi, hogy $S \to \varepsilon$ szabály-e.
- Ha $w \neq \varepsilon$, előállítja az összes olyan $u_0, u_1, \ldots, u_k \in (V \cup \Sigma)^*$ ismétlődés nélküli sorozatot, melyre $|u_i| \leq |w|$ valamint $|u_j| \leq |u_{j+1}|$ minden $i = 0, \ldots, k$ és $j < k$ esetén, majd ellenőrzi, ezek közt van-e olyan, mely a w szó S-ből induló derivációja. (A sorozatokat lexikografikus sorrendben állítja elő.) □

4.6. Következmény. *L_{CFG} eldönthető.*

Bizonyítás. Egy Turing-gép először eldönti, hogy a bemenő szó $\langle G, w\rangle$ alakú-e, ahol G egy környezetfüggetlen nyelvtan és w a G terminális betűiből álló szó, majd előállít egy olyan G' Chomsky normálformájú nyelvtant, melyre $L(G) = L(G')$. Majd az L_{CSG} nyelvet eldöntő Turing gépet szimulálva eldönti, hogy $\langle G', w\rangle \in L_{CSG}$ teljesül-e. □

Most néhány további eldöntési problémát említünk. Legyen

$$E_{DFA} = \{\langle M\rangle : M \text{ véges, det. automata, melyre } L(M) = \emptyset\}$$

$$E_{NFA} = \{\langle M\rangle : M \text{ véges, nemdet. automata, melyre } L(M) = \emptyset\}$$

4.14. Állítás. *E_{DFA} és E_{NFA} eldönthetőek.*

Bizonyítás. Csak az E_{NFA} nyelvre adjuk meg az egyszerű bizonyítást. Egy Turing-gép először eldönti, hogy a bemenet $\langle M \rangle$ alakú-e, ahol M egy $(Q, \Sigma, \delta, q_0, F)$ véges nemdeterminisztikus automata, majd eldönti, hogy M átmeneti gráfjában van-e olyan út, mely a q_0 kezdőállapotból végállapotba vezet. □

Végül legyen

$$EQ_{DFA} = \{\langle M_1, M_2 \rangle : M_1 \text{ és } M_2 \text{ ekvivalens véges det. automaták}\}$$

$$EQ_{NFA} = \{\langle M_1, M_2 \rangle : M_1 \text{ és } M_2 \text{ ekvivalens véges nemdet. automaták}\}$$

$$E_{CFG} = \{\langle G \rangle : G \text{ környezetfüggetlen, } L(G) = \emptyset\}$$

4.15. Állítás. *EQ_{DFA} és EQ_{NFA} eldönthetőek.*

Bizonyítás.

$$L(M_1) = L(M_2) \Leftrightarrow (L(M_1) \setminus L(M_2)) \cup (L(M_2) \setminus L(M_1)) = \emptyset \qquad \square$$

4.16. Állítás. *E_{CFG} eldönthető.*

Bizonyítás. Adott $G = (V, \Sigma, R, S)$ környezetfüggetlen nyelvtanra legyen $V_1 = \{A \in V : \exists u \in \Sigma^* \quad A \to u \in R\}$ és $V_{n+1} = V_n \cup \{A : \exists u \in (\Sigma \cup V_n)^* \quad A \to u \in R\}$. Ekkor $L(G) \neq \emptyset \Leftrightarrow S \in V_k$, ahol $|V| = k$. □

4.3. A Church-Turing-féle tézis

Bár az első algoritmusok az ókorból származnak, pld. Euklidesz algoritmusa a legnagyobb közös osztó meghatározására, az algoritmus fogalom formalizálására csak a 20. században került sor. 1900-ban Hilbert számos érdekes kérdést vetett fel az AMS kongresszusán. A 10. problémában olyan algoritmus létezését kérdezte, mely tetszőleges egész együtthatós $p(x_1, \ldots, x_n)$ többváltozós polinomról eldönti, van-e a $p(x_1, \ldots, x_n) = 0$ egyenletnek egész számokból álló megoldása. Vagy kérdezhetjük azt is, van-e olyan általános módszer, amellyel eldönthető az, hogy az elsőrendű nyelv egy kijelentése igaz-e a természetes számokra. A matematikai logikában fontos kérdés az, hogy az elsőrendű nyelv következmény fogalma algoritmikusan eldönthető-e.

A fenti kérdések mindegyikére negatív válasz született. Ennek bizonyításához szükség volt az algoritmus intuitív fogalmának formalizálására, matematikai megfeleőjének megalkotására. Az 1930-as évektől kezdve számos ilyen fogalom született. Ezek közül mindmáig az egyik legegyszerűbb és egyben legmeggyőzőbb a Turing-gép. Ma már általánosan elfogadjuk azt a tézist, amely szerint egy feladat akkor oldható meg algoritmikusan, ha Turing-géppel megoldható. Az algoritmus fogalmának a Turing-géppel való azonosítása a *Church-Turing tézis*, melynek legfőbb bizonyítéka amellett, hogy a Turing-gép magába foglalja az intuitív algoritmus fogalom minden jellemzőjét az, hogy a számtalan más javasolt fogalom mindegyikéről kiderült, hogy a Turing-géppel ekvivalens, és az a tapasztalati tény, hogy minden konkrét algoritmus megvalósítható (programozható) Turing-géppel. A Church-Turing tézis értelmében elegendő algoritmusainkat magas szinten megfogalmazni. Biztosak lehetünk abban, hogy amennyiben szükségessé válna, azt át tudnánk ültetni Turing-gépekre.

4.4. Eldönthetetlen problémák

Már említettük, hogy eldöntési probléma egy olyan feladat, amelyre *igen-nem* válasz adható. Egy ilyen problémát eldönthetetlennek (vagy megoldhatatlannak) nevezünk, ha Turing-géppel nem oldható meg. Pontosabban ez azt jelenti, hogy a probléma *igaz* példányait kódoló nyelv nem rekurzív. A kódolást általában nem adjuk meg pontosan, lényegében minden intuitíven algoritmikus kódolás megfelelő. A Church-Turing tézis szerint az, hogy egy probléma eldönthetetelen azt jelenti, hogy algoritmikusan megoldhatatlan.

Az első olyan problémák, amelyek eldönthetetlennek, Turing-gépekre feltett kérdések.

Legyen $L_{TM} = \{\langle M, w\rangle : M \text{ Turing-gép}, \ w \in L(M)\}$. Tehát L_{TM} azt a problémát reprezentálja, hogy adott M Turing-gépre és annak w bemenő szavára döntsük el, hogy M elfogadja-e w-ét. Az, hogy L_{TM} nyelv nem rekurzív azt jelenti, hogy nem létezik olyan algoritmus, amely teszőleges M Turing-gépre és w bemenő szóra eldönti, hogy $w \in L(M)$ teljesül-e.

4.16. Tétel. *Az L_{TM} nyelv nem rekurzív.*

Bizonyítás. A bizonyításban az általánosság megszorítása nélkül feltesszük, hogy a $\{0,1\}$ abc-ét használjuk a Turing-gépek és bemenő szavuk kódolására, tehát $\langle M, w\rangle \in \{0,1\}^*$ minden (M,w) párra. Tegyük fel, hogy L_{TM} rekurzív. Ekkor létezik olyan H Turing-gép, mely eldönti az L_{TM} nyelvet. Így erre a H Turing-gépre teljesül, hogy

$$H(\langle M, w\rangle) = \begin{cases} \mathit{accept} & \text{ha } w \in L(M) \\ \mathit{reject} & \text{ha } w \notin L(M). \end{cases}$$

Amennyiben H bemenete nem $\langle M, w\rangle$ alakú, akkor is elutasítja a bemenetet. Ezek után legyen D olyan Turing-gép, amelynek ha bemenetként adott egy M Turing-gép $\langle M\rangle$ kódja, akkor

$$D(\langle M\rangle) = \begin{cases} \mathit{accept} & \text{ha } \langle M\rangle \notin L(M) \\ \mathit{reject} & \text{ha } \langle M\rangle \in L(M). \end{cases}$$

Amennyiben a bemenet nem Turing-gép kódja, akkor is elutasítja a bemenetet. Ilyen D Turing-gép létezése H létezéséből következik: D egy adott $\langle M\rangle$ bemenetre szimulálja H-t az $\langle M, \langle M\rangle\rangle$ szón. Így

$$D(\langle D\rangle) = \begin{cases} \mathit{accept} & \text{ha } \langle D\rangle \notin L(D) \\ \mathit{reject} & \text{ha } \langle D\rangle \in L(D), \end{cases}$$

ami nyilvánvaló ellentmondás. Az a feltevés, hogy L_{TM} rekurzív, ellentmondáshoz vezetett. Tehát L_{TM} nem rekurzív. □

4.17. Állítás. *Az L_{TM} nyelv rekurzívan felsorolható.*

Bizonyítás. Az *U univerzális Turing-gép* működjön az $\langle M, w\rangle$ bemenő szón úgy, hogy szimulálja M működését a w szón. □

4.7. Következmény. *Létezik olyan rekurzívan felsorolható nyelv, mely nem rekurzív.*

4.8. Következmény. $\mathcal{L}_1 \subset \mathcal{L}_0$.

Bizonyítás. Minden környezetfüggő nyelv rekurzív. □

Most az eldönthetetlen problémák sorát megszakítva, néhány alapvető állítást igazolunk rekurzív és rekurzívan felsorolható nyelvekre.

4.18. Állítás. *Ha $L \in \Sigma^*$ rekurzív, akkor $\overline{L} = \Sigma^* \setminus L$ is rekurzív.*

Bizonyítás. Egy L-et eldöntő Turing-gépben cseréljük fel a q_{accept} és q_{reject} állapotokat. □

4.17. Tétel. *Egy $L \subseteq \Sigma^*$ nyelv akkor és csak akkor rekurzív, ha L és $\overline{L}$ is rekurzívan felsorolható.*

Bizonyítás. Tegyük fel, hogy léteznek az L-et és $\overline{L}$-et felismerő M és $\overline{M}$ Turing-gépek. Adott w szón az N 3-szalagos Turing-gép lépésenként felváltva szimulálja munkaszalagjain az M és $\overline{M}$ működését. Valamelyik gép megáll, és ekkor N megtudja a helyes választ arra a kérdésre, hogy $w \in L$ teljesül-e. □

4.9. Következmény. *$\overline{L}_{TM}$ nem rekurzívan felsorolható.*

Most belátjuk, hogy a Turing-gépek megállási problémája megoldhatatlan. Legyen

$$H_{TM} = \{\langle M, w\rangle : M \text{ Turing-gép mely megáll } w\text{-én}\}.$$

4.18. Tétel. *A H_{TM} nyelv nem rekurzív.*

Bizonyítás. A bizonyításban a visszavezetés módszerét használjuk. Az L_{TM} eldöntésének problémáját visszavezetjük a H_{TM} eldöntésére. Így ha az utóbbi eldönthető lenne, akkor L_{TM} is az lenne, ellentétben korábbi tételünkkel.

Tegyük fel, hogy H_{TM} rekurzív. Ekkor működjön a T Turing-gép az $\langle M, w\rangle$ bemeneten a következő módon:

1. Szimulálja a H_{TM}-et eldöntő Turing-gépet az $\langle M, w\rangle$ szón. Ha H_{TM} nem fogadja el az $\langle M, w\rangle$ szót, akkor T sem fogadja el.

2. Ha H_{TM} elfogadja az $\langle M, w\rangle$ szót, akkor T szimulálja az M gépet a w szón.

A T Turing-gép így eldönti az L_{TM} nyelvet, ami ellentmondás. Tehát H_{TM} nem rekurzív. □

Tekintsük most az $E_{TM} = \{\langle M\rangle : L(M) = \emptyset\}$ nyelvet, ami azt a problémát reprezentálja, hogy adott M Turing-gépről döntsük el, hogy az általa felismert nyelv üres-e.

4.19. Tétel. *E_{TM} nem rekurzív.*

Bizonyítás. Tegyük fel, hogy E_{TM} rekurzív. Adott $\langle M, w\rangle$ bemeneten a T Turing-gép működjön a következő módon:

1. Elkészít egy olyan M_w Turing-gépet, ill. annak kódját, mely a w-én kívül elutasít minden bemenő szót, w-én pedig úgy működik, mint M.

2. Az E_{TM}-et eldöntő Turing-gépet munkaszalagján szimulálva megállapítja, hogy $\langle M_w\rangle \in E_{TM}$ teljesül-e.

3. Amennyiben $\langle M_w \rangle \in E_{TM}$, akkor elutasítja az $\langle M, w \rangle$ bemenetet, különben elfogadja.

Ez a T Turing-gép eldönti az L_{TM} nyelvet, ami ellentmond korábbi tételünknek. □

Az előző tétel szerint nem létezik olyan algoritmus, mely egy Turing-gépről eldöntené, hogy az általa felismert nyelv üres-e. A tétel messzemenően általánosítható. Bizonyítás nélkül közöljük Rice tételét:

4.20. Tétel. *A rekurzívan felsorolható nyelvek egyetlen* nemtriviális *tulajdonsága sem dönthető el.*

Így nem dönthető el, hogy adott M Turing-gép

- az üres nyelvet ismeri-e fel,
- véges nyelvet ismer-e fel,
- reguláris nyelvet ismer-e fel, stb.

4.10. Következmény. *Nem dönthető el adott M_1, M_2 Turing-gépekre, hogy $L(M_1) = L(M_2)$ teljesül-e.*

4.5. Néhány további megoldhatatlan probléma

A *Post megfelelkezési probléma* egy példánya egy

$$P = \{(u_1, v_1), \ldots, (u_k, v_k) : u_i, v_i \in \Sigma^*\}$$

nemüres sorozat, amelyet *dominó készletnek* nevezünk. Azt mondjuk, hogy P-nek van megoldása, ha létezik olyan $i_1 \ldots i_n \in \{1, \ldots, k\}^*$ nemüres szó, hogy

$$u_{i_1} \ldots u_{i_n} = v_{i_1} \ldots v_{i_n}.$$

A továbbiakban eltekintünk attól, hogy problémákat kódolt alakban adunk meg.

4.21. Tétel. *Nem létezik olyan algoritmus, mely eldöntené adott dominókészletről, hogy van-e megoldása.*

A bizonyítás azon múlik, hogy Turing-gépet szimulálhatunk egy dominó készlettel. Pontosabban, minden M Turing-géphez és annak w bemenő szavához megadható egy olyan $P_{M,w}$ dominó készlet, melynek akkor és csakis akkor van megoldása, ha $w \in L(M)$.

Most a Post megfelelkezési probléma megoldhatatlanságát használjuk fel környezetfüggetlen nyelvtanokra vonatkozó problémák megoldhatatlanságának bizonyítására.

4.22. Tétel. *Nem létezik olyan algoritmus, mely tetszőleges környezetfüggetlen nyelvtanról eldöntené, hogy egyértelmű-e.*

Bizonyítás. Legyen P a fenti dominó készlet. Tekintsük az alábbi G_P környezetfüggetlen nyelvtant.

$$\begin{array}{lcl} S & \to & A \mid B \\ A & \to & iAu_i^{-1} \mid iu_i^{-1} \\ B & \to & iBv_i^{-1} \mid iv_i^{-1} \end{array}$$

P-nek akkor és csak akkor van megoldása, ha G_P nem egyértelmű.

Valóban, ha $i_1 \dots i_n$ a P megoldása, akkor az

$$i_1 \dots i_n u_{i_n}^{-1} \dots u_{i_1}^{-1} = i_1 \dots i_n v_{i_n}^{-1} \dots v_{i_1}^{-1}$$

szónak két különböző baloldali levezetése:

$$S \Rightarrow A \Rightarrow i_1 A u_1^{-1} \Rightarrow \dots \Rightarrow i_1 \dots i_{n-1} A u_{i_{n-1}}^{-1} \dots u_{i_1}^{-1} \Rightarrow i_1 \dots i_n u_{i_n}^{-1} \dots u_{i_1}^{-1}$$

és

$$S \Rightarrow B \Rightarrow i_1 B v_1^{-1} \Rightarrow \dots \Rightarrow i_1 \dots i_{n-1} B v_{i_{n-1}}^{-1} \dots v_{i_1}^{-1} \Rightarrow i_1 \dots i_n v_{i_n}^{-1} \dots v_{i_1}^{-1}.$$

Tegyük most fel, hogy G_P nem egyértelmű. Mivel a 2. és 3. sorban megadott szabályok mindegyike önmagában egyértelmű nyelvtant eredményez, ez csak úgy lehet, ha létezik olyan $w \in \Sigma^*$ szó, amelynek létezik egy olyan levezetése, amelyben először az $S \to A$ szabályt használtuk, és egy olyan is, melyben az első alkalmazott szabály $S \to B$. Mivel A-ból a $j_1 \dots j_m u_{j_m}^{-1} \dots u_{j_1}^{-1}$, B-ből pedig a $j_1 \dots j_m v_{j_m}^{-1} \dots v_{j_1}^{-1}$ alakú terminális szavak vezethetők le, ahol $m \geq 1$, $1 \leq j_1, \dots, j_n \leq k$, ezért van olyan $i_1, \dots, i_n$, $n \geq 1$, melyre $i_1 \dots i_n u_{i_n}^{-1} \dots u_{i_1}^{-1} = i_1 \dots i_n v_{i_n}^{-1} \dots v_{i_1}^{-1}$, és így $u_{i_1} \dots u_{i_n} = v_{i_1} \dots v_{i_n}$. □

4.23. Tétel. *Nem létezik olyan algoritmus, mely adott G_1, G_2 környezetfüggetlen nyelvtanokról eldönti, hogy $L(G_1) = L(G_2)$ teljesül-e.*

Bizonyítás. Legyen adott a fenti P dominó készlet. Legyen G_1 olyan nyelvtan, mely az

$$\{i_1 \dots i_n \# w : w \in \Sigma^*,\ n > 0,\ w \neq (u_{i_1} \dots u_{i_n})^{-1} \text{ vagy } w \neq (v_{i_1} \dots v_{i_n})^{-1}\}$$

nyelvet generálja. Legyen G_2 olyan jobblineáris nyelvtan, mely az $\{1, \dots, k\}^+ \# \Sigma^*$ reguláris nyelvet generálja. Ilyen nyelvtanok elkészíthetőek a P ismeretében. Másrészt $L(G_1) = L(G_2)$ akkor és csak akkor, ha P-nek nincs megoldása. □

5. fejezet

Bonyolultságelmélet

A kiszámíthatóságelmélet két irányban került kiterjesztésre. Egyrészt, ha feltesszük, hogy egy bizonyos eldönthetetlen probléma mégis eldönthető (valamilyen orákulummal), akkor vizsgálhatjuk azt, hogy mely más problémák válnak eldönthetővé. A relatív kiszámíthatóság az eldönthetetlen problémákat osztályozza eldönthetetlenségi fokuk szerint. Ezzel szemben a bonyolultságelmélet fő célkitűzése az algoritmikusan megoldható feladatok osztályozása a megoldásukhoz szükséges erőforrások mennyisége szerint. Ezen belül is két fontos esetet különböztethetünk meg, a legrosszabb eset és az átlagos eset vizsgálatát. Ebben a fejezetben a bonyolultságelmélet néhány alapvető fogalmával és eredményével ismerkedünk meg. Két erőforrással foglalkozunk: idő és tár, és csak a legrosszabb esetben való erőforrásigényt vizsgáljuk.

5.1. Néhány egyszerű probléma megoldásának időigénye

Legyenek $f, g : \mathbb{N} \to \mathbb{R}_+$ függvények, ahol $\mathbb{N} = \{0, 1, \ldots\}$, $\mathbb{R}_+$ pedig a nemnegatív valós számok halmaza.

Azt mondjuk, hogy $f(n) = O(g(n))$, ha létezik olyan $c > 0$ és $n_0 \in \mathbb{N}$, hogy

$$f(n) \leq c \cdot g(n)$$

minden $n \geq n_0$ esetén. Ha $f(n) = O(g(n))$ és $g(n) = O(f(n))$ is teljesül, akkor $f(n) = \Theta(g(n))$.

Néhány példa:

- $5n^3 + 6n^2 + 1 = \Theta(n^3)$
- $n^k = O(2^n)$
- $\log_2 n = \Theta(\log_3 n)$
- $\log \log n = O(\log n)$

Tekintsük a már gyakran szerepelt $L = \{0^k 1^k : k \geq 0\}$ nyelvet. Egy L-et eldöntő M_1 Turing-gép adott $w \in \{0,1\}^*$ szón működjön a következő módon (ld. az előző fejezetet is):

1. Először olvassa végig a bemenő szót, és ellenőrizze, hogy 1-es után nem következik-e 0. Ezt egy véges determinisztikus automata is el tudja végezni.

2. Mindaddig, amíg 0 és 1 is van a szalagon ismételje az alábbiakat:

    3. Olvassa végig a szalagot (balról jobbra, vagy jobbról balra), és húzzon át egy 0-t és egy 1-et.

4. Ha végül nem maradt a szalagon áthúzatlan 0 vagy 1, akkor elfogadja a bemenetet, különben elutasítja.

Feltételezve azt, hogy a Turing-gép minden közvetlen átmenetének időigénye egységnyi, a Turing-gép által megvalósított algoritmus teljes időigénye $O(n+n^2+n) = O(n^2)$.

Annak eldöntésére, hogy egy u szó az L nyelvbe esik-e, adhatunk egy másik algoritmust is, amelyet az M_2 Turing-gép valósít meg. Adott $w \in \{0,1\}^*$ szón M_2 az alábbiakban leírtak szerint működik.

1. Először végigolvassa a szalagot, és ellenőrzi, hogy egyetlen 1-et sem követ 0.

2. Mindaddig, amíg legalább egy 0 és egy 1 marad a szalagon, az alábbiakat ismétli:

    3. Ellenőrzi, hogy a szalagon maradt karakterek száma páros-e. Ha páratlan, elutasítja a bemenetet.
    4. Majd végigolvassa a szalagot, és áthúzza minden második 0-t és 1-et (az első 0-t és 1-et áthúzva).

5. Ha végül nem marad áthúzatlan 0 vagy 1, akkor elfogadja a bemenetet. Különben elutasítja.

Az időigény most $O(n \log n)$.

Végül egy harmadik M_3 kétszalagos Turing-gép működjön az alábbi algoritmus szerint:

1. Ellenőrzi, hogy a szalagon nem követ egyetlen 1-et sem 0.

2. Átmásolja a 0-kat egy munkaszalagra.

3. Végigolvassa az 1-eseket a bemeneten, és minden egyes 1-esre áthúz a munkaszalagon egy 0-t.

4. Ha mindegyik 0-t áthúzta és az 1-eseket is elolvasta, akkor elfogadja a bemenetet. Különben elutasítja.

Az időigény most $O(n)$.

5.2. Időbonyolultsági osztályok, P és NP

Legyen M olyan többszalagos determinisztikus Turing-gép, amely minden bemeneten megáll. Az M *időigénye* az az $f : \mathbb{N} \to \mathbb{N}$ függvény, amelyre $f(n)$ a legfeljebb n-hosszú bemeneten az M által megtett lépések számának maximuma. Azt mondjuk, hogy M időigénye $O(g(n))$, vagy hogy M $O(g(n))$ időkorlátos Turing-gép, ahol $g : \mathbb{N} \to \mathbb{R}_+$, ha $f(n) = O(g(n))$.

Már beláttuk, hogy minden többszalagos Turing-gép szimulálható egyszalagos Turing-géppel. Módszerünk a lépések számában négyzetes romlást eredményezett.

5.24. Tétel. *Legyen $t(n) \geq n$. Minden $O(t(n))$ időigényű többszalagos Turing-géphez létezik ekvivalens, $O(t^2(n))$ időigényű egyszalagos Turing-gép.*

Most definiáljuk a nemdeterminisztikus Turing-gép időigényét. Legyen T olyan többszalagos nemdeterminisztikus Turing-gép, melynek minden számítási sorozata véges és elfogadáshoz vagy elutasításhoz vezet. A T *időigényét* az az $f : \mathbb{N} \to \mathbb{N}$ függvény adja, amelyre tetszőleges n esetén $f(n)$ a T legfeljebb n-hosszú bemeneten való számítási sorozatai hosszának maximuma (= leghosszabb út hossza a T legfeljebb n-hosszú bemeneten való számítási fáiban).

Nem ismert, hogy létezik-e nemdeterminisztikus Turing-gépek determinisztikus géppel való szimulálására hatékony általános módszer. Az könnyen látható, hogy az erre korábban ismertetett módszerünk exponenciális romlást okoz.

5.25. Tétel. *Legyen T olyan többszalagos nemdeterminisztikus Turing-gép, melynek minden számítási sorozata véges és elfogadáshoz vagy elutasításhoz vezet. Tegyük fel, hogy T időigénye $O(t(n))$, ahol $t(n) \geq n$. Ekkor létezik T-vel ekvivalens, $2^{O(t(n))}$ időigényű determinisztikus Turing-gép.*

Tegyük fel, hogy $t : \mathbb{N} \to R_+$ olyan függvény, melyre $t(n) \geq n$. Ekkor legyen $\mathrm{TIME}(t(n))$ az összes olyan nyelv osztálya, amely eldönthető $O(t(n))$ időigényű többszalagos determinisztikus Turing-géppel. Hasonlóan, legyen $\mathrm{NTIME}(t(n))$ az összes olyan nyelv osztálya, amely eldönthető $O(t(n))$ időigényű többszalagos nemdeterminisztikus Turing-géppel. A *polinomidőben eldönthető* nyelvek osztálya:

$$\mathbf{P} = \bigcup_{k \geq 1} \mathrm{TIME}(n^k).$$

A *nemdeterminisztikus polinomidőben eldönthető* nyelvek osztálya:

$$\mathbf{NP} = \bigcup_{k \geq 1} \mathrm{NTIME}(n^k).$$

Nyilván minden **P**-beli nyelv **NP**-ben van.

Már hozzászokott ahhoz az Olvasó, hogy a nyelvekkel problémákat, feladatokat is reprezentálunk. Ebben a vonatkozásban a **P** és **NP** osztály definíciója sokkal érdekesebb, mint ahogy az az első pillanatban tűnik: **P** a polinomidőben *megoldható* problémák osztályát reprezentálja, **NP** pedig az ún. polinomidőben *verifikálható* problémák osztályát (ld. később).

Több indok hozható fel amellett, hogy **P** tekinthető a gyakorlatban hatékonyan megoldható problémák osztályának, és persze több ellenérv is felhozható. Az érvek közt szerepel az,

hogy egy olyan általános fogalmat alkottunk meg, melyhez bár egy konkrét számítási modellt, a Turing-gépet használtuk, de valójában **P** független a felhasznált számítási modelltől, mert a szóba jöhető modellek bármelyike polinomiális romlással szimulálható bármely másikkal. (Ez az ún. *polinomiális tézis*, amelyet illusztrál az az eredmény, hogy többszalagos Turing-gép négyzetes romlással szimulálható egyszalagos géppel.) A fogalom a feladatok kódolásához felhasznált módszertől is független, hiszen az elfogadható kódolások közt is polinomiális kapcsolat van, gondoljunk csak a természetes számok 2-es vagy 10-es számrendszerben való ábrázolására. (Az unáris ábrázolás persze exponenciálisan hosszabb, ezért elfogadhatatlan.) Az ismert hatékony algoritmusokkal megoldható problémák nagy részéről valóban megmutatható az, hogy a **P** osztályba esik, és sok esetben a nagyságrend is kezelhető. Az ellenérvek közt szerepel az, hogy tekinthetünk-e egy $\Theta(n^{100})$ időigényű Turing-gépet vagy algoritmust valójában hatékonynak. Vagy bővíteni kellene a **P** osztályt, mert valószínűségi módszerekkel is kaphatunk gyakorlatban használható hatékony algoritmusokat, és ezekkel a módszerekkel esetleg megoldhatunk olyan problémákat is, melyek nem esnek **P**-be. Vagy mégis szűkíteni kellene a **P** osztályt valamilyen más okból, pld. a hatékony párhuzamosíthatóság szempontjából. Mindezek figyelembevételével is tekinthetjük a **P** osztályt mint a hatékonyan megoldható problémák egy jó, és matematikai szempontból nagyon érdekes nemtriviális közelítését.

Néhány példa polinomidőben megoldható problémára.

Az elérhetőség az a feladat, hogy döntsük el adott véges G irányított gráfra és annak s és t csúcsaira, hogy vezet-e irányított út s-ből t-be. Formálisan a problémához tartozó nyelv az összes olyan (G,s,t) hármas kódjából áll, ahol G egy véges irányított gráf az s és t csúcsokkal, amelyre létezik G-ben s-ből t-be vezető irányított út. Az alábbi egyszerű algoritmus, melyet már bemutattunk és amely könnyen megvalósítható determinisztikus polinomidejű Turing-géppel, megoldja ezt a feladatot:

- Jelöljük meg az s csúcsot.
- Mindaddig, amíg már nem jelölhető meg új csúcs, ismételjük meg az alábbiakat: Olvassuk végig az éleket. Ha olyan (a,b) élet találunk, hogy a meg van jelölve, de b nincs, jelöljük meg a b csúcsot.
- Ha végül t is meg van jelölve, akkor elfogadjuk, különben elutasítjuk a bemenetet.

5.26. Tétel. *Minden környezetfüggetlen nyelv a* **P** *osztályba esik.*

Bizonyítás. Legyen $L \subseteq \Sigma^*$ környezetfüggetlen nyelv. Ekkor létezik L-et generáló $G = (V,\Sigma,R,S)$ Chomsky normálformában adott nyelvtan. Ahhoz, hogy eldöntsük, hogy $\varepsilon \in L$ teljesül-e, csak azt kell ellenőriznünk, hogy $S \to \varepsilon$ szabály-e. Legyen most $w \in \Sigma^*$, $w = w_1,\ldots,w_n$, $n > 0$ nemüres szó, ahol $w_i \in \Sigma$ minden $i = 1,\ldots,n$ esetén. Definiáljuk a $T_{i,j}$, $1 \le i \le j \le n$ halmazokat a következő módon:

$$T_{i,j} = \{X \in V : X \Rightarrow^* w_i \ldots w_j\}$$

A $T_{i,j}$ halmazok mindegyike meghatározható $O(n)$ időben:

$$\begin{aligned} T_{i,i} &= \{X \in V : X \to w_i \in R\} \\ T_{i,j} &= \{X \in V : \exists A,B \exists k \quad X \to AB \in R,\ A \in T_{i,k},\ B \in T_{k+1,j},\ i < k < j\} \end{aligned}$$

Végül $w \in L$ akkor és csak akkor teljesül, ha $S \in T_{1,n}$. Az algoritmus összes időigénye $O(n^3)$. □

Mai ismereteinkkel csak kimerítő kereséssel oldhatók meg az alábbi problémák:

- hamilton-út, melyben adott egy véges irányított G gráf az s és t csúcsokkal, és azt kérdezzük, hogy vezet-e s-ből t-be olyan irányított út, mely minden csúcsot pontosan egyszer érint.

- teljes részgráf, amelyben adott egy G véges gráf és egy k szám, és azt kérdezzük, hogy létezik-e G-nek k csúcsú teljes részgráfja.

- egész értékű programozás, melyben adott egy egész együtthatós lineáris egyenlőtlenség rendszer, és azt kérdezzük, hogy van-e egész értékű megoldás.

- kielégíthetőség (sat), melyben adott az ítéletkalkulus egy konjunktív normálformájú formulája (Boole-formula), és azt kérdezzük, hogy kielégíthető-e.

5.19. Állítás. *A fenti* hamilton-út*,* teljes részgráf*,* egész értékű programozás*,* kielégíthetőség *problémák mindegyike megoldható nemdeterminisztikus polinomidőben, és így az* **NP** *osztályba esik.*

Bizonyítás. Csak a hamilton-út esetére vázoljuk a bizonyítást. Annak eldöntésére, hogy a bemenetként megadott G gráfban vezet-e Hamilton-út s-ből t-be, egy Turing-gép nemdeterminisztikusan generálja a csúcsok egy permutációját munkaszalagján, majd determinisztikusan $O(n)$ időben ellenőrzi, hogy az adott permutáció s-ből t-be vezető Hamilton-utat határoz-e meg. Ha a permutáció Hamilton-utat határoz meg, elfogadja a bemenetet, különben elutasítja. □

Az **NP**-ben lévő feladatok mindegyikére az jellemző, hogy a lehetséges megoldások polinomidőben előállíthatóak, azokat egy nemdeterminisztikus Turing-gép polinomidőben generálhatja munkaszalagján, majd determinisztikusan polinomidőben ellenőrizhető az, hogy az előállított lehetséges megoldás valóban megoldás-e. Azt mondjuk, hogy egy L nyelv *polinomidőben verifikálható*, ha létezik olyan $K \in \mathbf{P}$ nyelv és k szám, hogy

$$L = \{x : \exists y\ (x,y) \in K \wedge |y| = O(|x|^k)\}.$$

A K nyelv persze nemcsak (x,y) alakú rendezett párokból állhat, hanem ezek polinom méretű $\langle x,y\rangle$ kódjaiból is.

5.27. Tétel. *Teszőleges L nyelvre, $L \in$* **NP** *akkor és csakis akkor, ha L polinomidőben verifikálható.*

Ezért az **NP** osztályt a polinomidőben verifikálható problémák osztályának is nevezhetjük.

A számítástudomány és a matematika egy híres megoldatlan kérdése, hogy $\mathbf{P} = \mathbf{NP}$ teljesül-e. Később belátjuk majd, hogy $\mathbf{P} = \mathbf{NP}$ akkor és csakis akkor, ha az 5.19 Állításban szereplő valamelyik probléma (és akkor mindegyikük) a **P** osztályba esik.

5.3. NP-teljes problémák

Egy $f : \Sigma^* \to \Delta^*$ függvény *polinomidőben kiszámítható*, ha létezik olyan M polinomidejű többszalagos determinisztikus Turing-gép, amely kiszámítja f-et abban az értelemben, hogy tetszőleges $w \in \Sigma^*$ szóra mindig megáll és megálláskor egy kitüntett munkaszalagjának tartalma $f(w)$.

Legyenek $L_1 \subseteq \Sigma_1^*$ és $L_2 \subseteq \Sigma_2^*$ nyelvek. Az L_1-nek L_2-re való *polinomidejű visszavezetése* egy olyan polinomidőben kiszámítható

$$f : \Sigma_1^* \to \Sigma_2^*$$

függvény, hogy tetszőleges $w \in \Sigma_1^*$ szóra:

$$w \in L_1 \Leftrightarrow f(w) \in L_2.$$

Vegyük észre, hogy ha f az L_1 polinomidejű visszavezetése L_2-re, akkor ugyanez az f függvény az $\overline{L}_1$ polinomidejű visszavezetése $\overline{L}_2$-re. Azt mondjuk, hogy L_1 *polinomidőben visszavezethető* L_2-re, $L_1 \leq_p L_2$, ha létezik az L_1-nek polinomidejű visszavezetése az L_2 nyelvre.

5.20. Állítás. *$A \leq_p$ reláció előrendezés (reflexív és tranzitív).*

Bizonyítás. Legyen $L_i \subseteq \Sigma_i^*$, $i = 1,2,3$, és tegyük fel, hogy $f : \Sigma_1^* \to \Sigma_2^*$ az L_1 polinomidejű visszavezetése L_2-re, $g : \Sigma_2^* \to \Sigma_3^*$ az L_2 polinomidejű visszavezetése L_3-ra. Belátjuk, hogy a $h = f \circ g$ összetett függvény polinomidőben visszavezeti L_1-et L_3-ra. Léteznek olyan $O(n^k)$ ill. $O(n^m)$ időkorlátos M és N Turing-gépek, amelyek kiszámítják az f és g függvényeket. A T Turing-gép adott $w \in \Sigma_1^*$ szón szimulálja először az M működését, míg az elő nem állítja az $f(w)$ szót, melynek hossza $O(n^k)$. Ezek után szimulálja az N Turing-gépet az $f(w)$ szón, így kiszámítva a $g(f(w))$ szót. A T időigénye $O(n^{k+m})$. □

5.21. Állítás. *Az* **NP** *osztály zárt a polinomidejű visszavezetésre: Ha $L_1 \leq_p L_2$ és $L_2 \in$* **NP***, akkor $L_1 \in$* **NP**.

Bizonyítás. Tegyük fel, hogy $L_2 \subseteq \Sigma_2^*$ az **NP**-beli nyelv, $L_1 \subseteq \Sigma_1^*$ és $f : \Sigma_1^* \to \Sigma_2^*$ az L_1 polinomidejű visszavezetése L_2-re. Legyen M olyan $O(n^k)$ időkorlátos nemdeterminisztikus Turing-gép, amely eldönti az L_2 nyelvet, és legyen D olyan $O(n^m)$ időkorlátos determinisztikus Turing-gép, mely kiszámítja f-et. Működjön a T nemdeterminisztikus Turing-gép a következő módon. Adott $w \in \Sigma_1^*$ szóra először a D szimulálásával kiszámítja egy munkaszalagján az $f(w)$ szót, majd nemdeterminisztikusan szimulálja M lehetséges működéseit az $f(w)$ szón. Ha M elfogadja $f(w)$-t, akkor T elfogadja a bemenetként adott w szót, különben elutasítja.

Világos, hogy T eldönti az L_1 nyelvet és időigénye $O(n^{m+k})$. □

Megjegyezzük, hogy a **P** osztály is nyilvánvalóan zárt a polinomidejű visszavezetésre, de ez nem igazán érdekes, mert a polinomidejű visszavezetés definíciójában polinommal korlátoztuk a Turing-gép időigényét. Így ha $L_1, L_2 \in$ **P** és L_1, L_2 és komplemenseik sem üresek, akkor $L_1 \leq_p L_2$.

Most bevezetünk egy fontos fogalmat, amely lehetőséget ad arra, hogy az egész **NP** osztályt egyetlen problémával reprezentáljuk. Azt mondjuk, hogy egy L nyelv (ill. az általa reprezentált probléma) **NP**-*nehéz*, ha $L' \leq_p L$ teljesül minden $L' \in$ **NP** nyelvre. Egy L nyelv (probléma) **NP**-*teljes*, ha $L \in$ **NP** és L **NP**-nehéz.

5.22. Állítás. *Tegyük fel, hogy $L \subseteq \Sigma^*$* **NP***-teljes. Akkor egy $L' \subseteq \Delta^*$ nyelv akkor és csak akkor van* **NP***-ben, ha $L' \leq_p L$.*

Bizonyítás. Ha $L' \leq_p L$, akkor $L' \in$ **NP** az 5.21 Állítás szerint és mivel $L \in$ **NP**. Ha pedig $L' \in$ **NP**, akkor $L' \leq_p L$, mivel L **NP**-nehéz. □

5.23. Állítás. *Tegyük fel, hogy L* **NP***-teljes. Ekkor* **P** = **NP** *akkor és csak akkor, ha $L \in$* **P**.

Bizonyítás. A szükségesség triviális. Az elegendőség bizonyításához tegyük fel, hogy $L \in$ **P**. Ekkor tetszőleges $L' \in$ **NP** esetén $L' \leq_p L$ miatt $L' \in$ **P**. Tehát **NP** $\subseteq$ **P**, így **P** = **NP**. □

5.24. Állítás. *Tegyük fel, hogy $L \leq_p L'$. Ha L* **NP***-nehéz, akkor L' is az. Továbbá, ha L* **NP***-nehéz, akkor L' akkor és csak akkor* **NP***-teljes, ha $L' \in$* **NP**.

Bizonyítás. Az első állítás adódik a $\leq_p$ tranzitivitásából. A második az elsőből következik. □

A hátralévő részében számos **NP**-teljes problémát mutatunk be. Ezek közül az első a konjunktív normálformájú Boole-formulák kielégíthetősége, a már említett sat probléma és a hozzá tartozó nyelv. Emlékeztetőül, Boole-formulán tetszőleges olyan kifejezést értünk, amely logikai változókból épül fel a $\vee$ (konjunkció), $\wedge$ (diszjunkció) és $^-$ (tagadás) logikai műveletekkel. Konjunktív normálformában azok a Boole-formulák vannak, amelyek klózok (tagok) nemüres konjunkciói, ahol egy klóz literálok nemüres diszjunkciója. Literál minden változó (pozitív literál) és minden változó negáltja (negatív literál). Egy formula kielégíthető, ha a benne szereplő változók értéke megválasztható *igaz*nak (1) vagy *hamis*nak (0) úgy, hogy a formula értéke *igaz* legyen.

Példa: $(x_1 \vee \bar{x}_2 \vee \bar{x}_3) \wedge (\bar{x}_1 \vee x_2 \vee x_3) \wedge (\bar{x}_1 \vee \bar{x}_2 \vee \bar{x}_3)$ kielégíthető.

5.28. Tétel. *(Cook)* sat **NP***-teljes.*

A tétel bizonyítását később adjuk csak meg, miután Cook tételének és a polinomidejű visszavezetés felhasználásával egyéb problémákról is megmutattuk, hogy **NP**-teljesek.

Legyen $k \geq 1$. Ekkor $k$sat a sat azon speciális esete, amelyet akkor kapunk, ha csak olyan konjunktív normálformákat tekintünk, amelyekben minden klóz k (nem feltétlenül különböző) literál konjunkciója. Nem nehéz belátni, hogy 2sat $\in$ **P**.

5.29. Tétel. 3sat **NP***-teljes.*

Bizonyítás. Mivel sat $\in$ **NP**, nyilvánvalóan 3sat $\in$ **NP**. Belátjuk, hogy sat $\leq_p$ 3sat. Ehhez minden φ konjunktív normálformához el kell készítenünk egy „3sat-alakú” $f(\varphi)$ konjunktív normálformát úgy, hogy φ akkor és csak akkor kielégíthető, ha $f(\varphi)$ az. A konstrukciónak egyszerűnek is kell lennie abban az értelemben, hogy φ-ből az $f(\varphi)$ polinomidőben előállítható legyen (egy determinisztikus Turing-géppel).

Először tegyük fel, hogy φ egyetlen c klózból áll. Ha c egy literált tartalmaz csak, akkor ismételjük meg ezt még kétszer, ha pedig két literált tartalmaz, akkor ismételjük meg az egyiket. Ha c három literált tartalmaz, akkor $f(c)$ legyen c. Tegyük most fel, hogy c legalább 4 literált tartalmaz, $c = l_1 \vee \ldots \vee l_n$, $n \geq 4$. Ekkor tekintsünk $n-3$ új változót, mondjuk ezek $y_1, \ldots, y_{n-3}$, és legyen

$$\begin{aligned} f(c) \;=\; & (l_1 \vee l_2 \vee y_1) \wedge (\bar{y}_1 \vee l_3 \vee y_2) \wedge \ldots \wedge \\ & (\bar{y}_{n-4} \vee l_{n-2} \vee y_{n-3}) \wedge (\bar{y}_{n-3} \vee l_{n-1} \vee l_n). \end{aligned}$$

Ha a változók egy τ értékelése kielégíti $f(c)$-t, akkor csak felejtkezzünk el az új változók értékéről. Az értékelés kielégíti c-t, hiszen $f(c)$ nem lehet csak azért *igaz* τ mellett, mert mindegyik klózában valamelyik új literál *igaz*, hiszen egy új változó csak az $f(c)$ egy klózát teheti *igaz*zá, és $f(c)$-nek eggyel több klóza van. Ha pedig az eredetileg c-ben előforduló változók egy τ értékelése c-t kielégíti, akkor az folytatható az új változókra úgy, hogy $f(c)$-t kielégítő értékelést kapjunk. Valóban, ha τ *igaz*zá teszi az l_i literált, akkor tekintsük $f(c)$ azon klózát, amelyben l_i előfordul. Minden ezt megelőző klózban pozitívan előfordul egy új változó, és tőle jobbra negatívan egy olyan új változó, amely nem fordul elő balra eső klózban. Az előzőek értékét válasszuk *igaz*nak, az utóbbiak értékét pedig *hamis*nak.

Ha φ több klózból áll, mondjuk

$$\varphi = c_1 \wedge \ldots \wedge c_m,$$

akkor legyen

$$f(\varphi) = f(c_1) \wedge \ldots \wedge f(c_m)$$

ahol minden egyes $f(c_i)$ elkészítéséhez vadonatúj változókat használunk. Így ha $f(\varphi)$-t kielégíti egy τ értékelés, akkor elhagyva az új változók értékeit, a φ egy kielégítő értékelését kapjuk. Ha pedig adott a φ egy kielégítő értékelése, akkor ezt folytatni tudjuk az új változókra úgy, hogy minden egyes $f(c_i)$ igaz legyen.

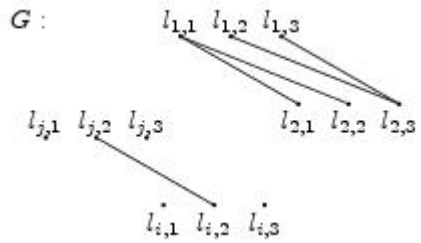
Az $f(\varphi)$ mérete a φ méretének polinom függvényével korlátozható. Valójában könnyű egy olyan determinisztikus többszalagos Turing-gépet tervezni, amely φ-ből egyik munkaszalagján elkészíti az $f(\varphi)$ formulát. Ehhez egy munkaszalagon számolja a már bevezetett új változókat. □

A már említett teljes részgráf probléma azt kérdezi, hogy egy véges gráfnak van-e k csúcsú teljes részgráfja. Már láttuk, hogy teljes részgráf **NP**-ben van.

5.30. Tétel. *A* teljes részgráf *probléma* **NP**-*teljes.*

Bizonyítás. Az **NP**-nehézséget sat-ból való visszavezetéssel igazoljuk.

Legyen $\varphi = c_1 \wedge \ldots \wedge c_k$ egy konjunktív normálforma, ahol $c_i = l_{i,1} \vee l_{i,2} \vee l_{i,3}$ minden i-re. Tekintsük azt a $3k$ csúcsú G gráfot, amelyben a csúcsok hármas csoportokba vannak osztva úgy, hogy minden egyes csoportnak egy c_i klóz felel meg, és a csoportot alkotó csúcsok a c_i literáljainak felelnek meg.



Az összes élet behúzzuk, kivéve

- az egy csoporton belül lévő csúcsok köztieket, és

- az ellentétes literálokkal címkézett csúcsok köztiket.

Azt állítjuk, hogy G-nek akkor és csak akkor van n csúcsú teljes részgráfja, ha φ kielégíthető.

Valóban, ha egy τ értékelés kielégíti a φ formulát, akkor minden egyes c_i klózból válasszunk ki egy olyan l_{i,j_i} literált, amelynek értéke τ mellett *igaz*, és tekintsük az így kiválasztott literálokhoz tartozó csúcsokat. Ezek közül bármely kettő össze van kötve éllel, hiszen a csúcsok páronként különböző csoportba tartoznak és a kiválasztott literálok közt nincs két ellentétes.

Tegyük most fel, hogy G tartalmaz egy n csúcsú teljes részgráfot, melyet H jelöl. Ekkor H minden egyes csoportból pontosan egy csúcsot tartalmaz, és ezek mindegyikének megfelel egy c_i klóz és a c_i egy l_{i,j_i} literálja. Ezek után legyen egy x változó értéke *igaz*, ha az l_{i,j_i} literálok közt x szerepel, különben legyen x értéke *hamis*. Ez az értékelés jól definiált és kielégíti φ-t, mert az l_{i,j_i} literálok közt nincs két ellentétes.

A φ formulából, ill. annak kódjából polinomidejű Turing-géppel előállítható a (G,n) kódja. Tehát 3sat $\leq_p$ teljes részgráf. Mivel 3sat **NP**-teljes és teljes részgráf $\in$ **NP**, ezért teljes részgráf is **NP**-teljes. □

A független csúcshalmaz probléma bemenetét egy G véges gráf és egy $k \geq 1$ szám alkotja, és arra ad választ, hogy G-nek van-e k csúcsú független csúcshalmaza.

5.31. Tétel. független csúcshalmaz **NP**-*teljes.*

Bizonyítás. Egy n csúcsú G gráfnak akkor és csak akkor van k csúcsú teljes részgráfja, ahol $k \leq n$, ha a G komplementerének van k csúcsú független csúcshalmaza. □

5.32. Tétel. hamilton-út **NP** *teljes.*

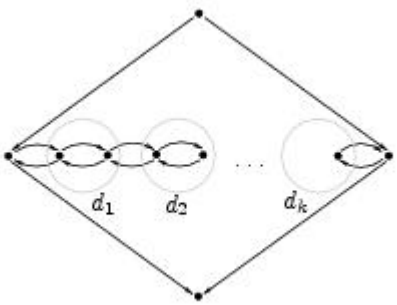
Bizonyítás. Azt már tudjuk, hogy hamilton-út **NP**-ben van. A 3sat **NP**-teljességének felhasználásával bizonyítjuk azt, hogy **NP**-nehéz.

Legyen φ egy 3sat alakú formula:

$$\varphi = d_1 \wedge \ldots \wedge d_k, \quad \text{ahol} \quad d_j = a_j \vee b_j \vee c_j,\ j = 1,\ldots,k.$$

Legyenek $x_1, x_2, \ldots, x_l$ a formulában felhasznált változók. Feladatunk egy G véges irányított gráf elkészítése, és a gráfban két csúcs, s és t megnevezése úgy, hogy akkor és csak akkor vezet s-ből t-be Hamilton út, ha φ kielégíthető. A G gráfot részgráfokból rakjuk össze, amelyeket eszközöknek nevezünk.

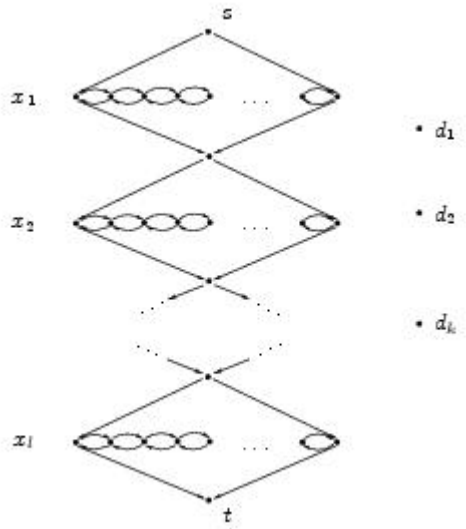
Minden egyes x_i változóra az x_i-hez tartozó eszköz:



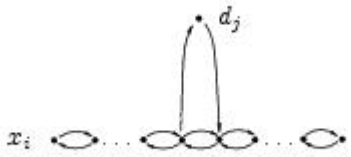
Itt a középső sorban $2k+2$ csúcs szerepel. Ezek a két szélső kivételével kettes csoportokra vannak osztva úgy, hogy minden d_j klózhoz tartozik egy csoport. Minden egyes d_j-re a d_j-hez tartozó eszköz egyetlen csúcs:

•

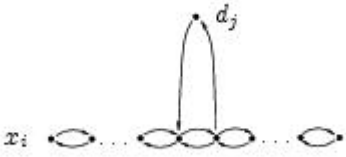
Ezek után a G gráf és annak s és t csúcsai:



A gráf megadását a következő élek teszik teljessé:



ha d_j tartalmazza x_i-t, és



ha d_j tartalmazza $\bar{x}_i$-t. Tehát amennyiben x_i előfordul d_j-ben, akkor az x_i-hez tartozó eszközben a d_j-hez tartozó kettes csoport baloldali csúcsából él vezet a d_j-hez tartozó eszközhöz, és onnan vissza él vezet a csoport jobboldali csúcsába. Az élek fordított irányúak akkor, ha x_i negatívan fordul elő d_j-ben. Azt feltehetjük, hogy egyetlen klózban sem fordul elő pozitívan és negatívan is egy változó.

Belátjuk, hogy φ akkor és csak akkor kielégíthető, ha G-ben vezet s-ből t-be Hamilton-út. Először tegyük fel, hogy φ kielégíthető, és jelölje τ a változók olyan értékelését, amely kielégíti a φ formulát. Minden egyes d_j klózhoz válasszunk ki egy olyan literált az a_j, b_j és c_j közül, jelölje ezt l_j, amelyre $\tau(l_j) = 1$. Ezek után s-ből indulva járjuk be a változókhoz tartozó eszközöket rendre úgy, hogy egy x_i változóhoz tartozó eszköz középső sorát balról jobbra járjuk be, ha $\tau(x_i) = 1$, és jobbról balra, ha $\tau(x_i) = 0$. Így eljutunk s-ből t-be úgy, hogy a klózokhoz tartozó csúcsok kivételével minden csúcsot pontosan egyszer érintettünk. Ezt az utat módosíthatjuk úgy, hogy végül Hamilton-utat kapjunk. Ehhez minden egyes d_j klózra, amennyiben mondjuk l_j az x_i változó és így $\tau(x_i) = 1$, miközben a az x_i-hez tartozó eszköz bejárásában az d_j-hez tartozó kettes csoport baloldali csúcsához érünk, látogassuk meg a d_j-hez tartozó eszközt, ami egyetlen csúcs, és innen térjünk vissza a csoport jobboldali csúcsába, majd folytassuk a bejárást. Amennyiben $l_j = \bar{x}_i$ akkor $\tau(x_i) = 0$, és az x_i-hez tartozó eszközt jobbról balra járjuk be. Ebben az esetben módosítsuk a bejárást úgy, hogy közben „kiugrunk” a d_j-hez tartozó csúcshoz, amikor a d_j-hez tartozó kettes csoport jobboldali csúcsához érünk. Tehát amennyiben φ kielégíthető, létezik s-ből t-be vezető Hamilton-út.

Most tegyük fel, hogy a Hamilton-út létezik, és tekintsünk egy s-ből t-be vezető Hamilton-utat. Minden x_i változóhoz tartozó eszközt balról jobbra vagy fordítva járunk be azzal, hogy közben esetleg kiugrunk néhány klózhoz tartozó csúcshoz. Legyen $\tau(x_i) = 1$ akkor és csak akkor, ha az x_i-hez tartozó eszközt balról jobbra járjuk be a tekintett Hamilton-úton. Ekkor τ kielégíti a φ formulát és így φ kielégíthető.

Mivel adott φ formulából (G,s,t) polinomidejű Turing-géppel elkészíthető, ezzel beláttuk, hogy 3sat polimomidőben visszavezethető a hamilton-út problémára. □

5.4. Cook tétele, újra

Most belátjuk Cook tételét, amelyet ismét kimondunk:

5.33. Tétel. sat **NP**-*teljes*.

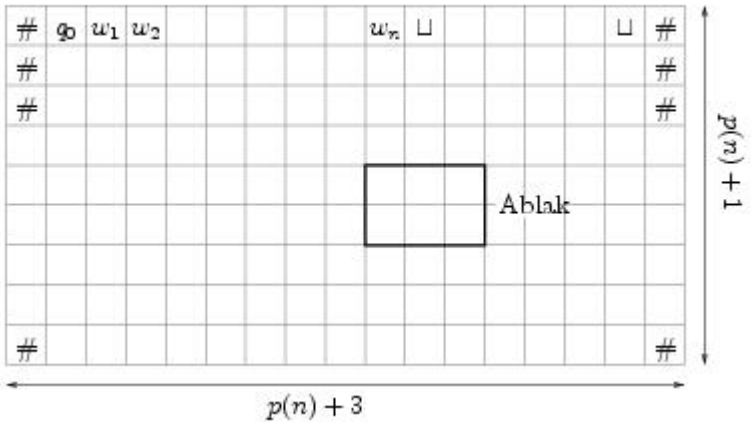
Bizonyítás. Már tudjuk, hogy sat $\in$ **NP**, így csak azt bizonyítjuk, hogy sat **NP**-nehéz. Ehhez legyen $L \subseteq \Sigma^*$ **NP**-ben. Az **NP** definíciója szerint van olyan $p(n) \geq n$ polinom időkorlátos M nemdeterminisztikus Turing-gép, melyre $L = L(M)$. Adott $w = w_1 \dots w_n \in \Sigma^*$, $w_i \in \Sigma$ bemenő szóhoz el kell készítenünk egy φ formulát úgy, hogy:

(1) $w \in L$ akkor és csak akkor, ha φ kielégíthető,

(2) a $w \mapsto \varphi$ hozzárendelés megvalósítható polinom időkorlátos determinisztikus Turing-géppel.

Legyen $M = (Q, \Sigma, \Gamma, \delta, q_0, q_{accept}, q_{reject})$. Az M egy lehetséges működését a w szón egy táblázattal írhatjuk le, melynek sorai a Turing-gép konfigurációinak felelnek meg. Az első

sor a w-hez tartozó kezdőkonfigurációját írja le, és minden egyes újabb sor a Turing-gép működésének egy lehetséges lépését adja meg.



Azért, hogy a táblázat pontosan $p(n)+1$ sorból álljon, megengedjük azt is, hogy a Turing-gép egy lépésben ugyanabban a konfigurációban maradjon. A táblázat első és utolsó oszlopa a speciális # jelet tartalmazza, minden más betű pedig vagy a Γ eleme, vagy egy állapot. Persze minden sorban pontosan egyszer fordul elő állapot.

Minden $1 \leq i \leq p(n)+1$, $1 \leq j \leq p(n)+3$ és $s \in Q \cup \Gamma \cup \{\#\}$ hármasnak feleltessük meg az $x_{i,j,s}$ változót. A φ formulában ezeket a változókat használjuk (amelyek száma $O(p^2(n))$.

Azt, hogy a táblázat a Turing-gép egy lehetséges működését írja le a w szón, kifejezhetjük azzal, hogy az első sorban a w-hez tartozó kezdőkonfiguráció van és azzal, hogy egy lokális feltétel teljesül minden olyan 2×3-as „ablakra”, amely elhelyezhető a táblázatban.

5.25. Állítás. *Megadható csak az M-től függően véges sok olyan*

$$\begin{bmatrix} a_1 & a_2 & a_3 \\ a_4 & a_5 & a_6 \end{bmatrix}$$

„legális” mátrix, hogy egy táblázat akkor és csak akkor adja meg az M egy lehetséges működését a w szón, ha a táblázat első sorában a w-hez tartozó kezdőkonfiguráció van, és minden 2×3-as ablak két sora egy legális mátrixot határoz meg.

A φ formulát részformulák konjunkciójaként állítjuk elő:

$$\varphi = \varphi_0 \wedge \varphi_{start} \wedge \varphi_{move} \wedge \varphi_{accept}$$

Felírásához célszerű szem előtt tartani azt a konvenciót, hogy $x_{i,j,s} = 1$ annak felel meg, hogy a táblázat (i,j) pozíciójában s szerepel.

A φ_0 azt fejezi ki, hogy a táblázat minden egyes mezőjében pontosan egy karakter van, azaz $\forall i,j \exists! s\, x_{i,j,s}$. Másképp felírva:

$$\varphi_0 = \bigwedge_{i,j} \bigvee_{s} x_{i,j,s} \wedge \bigwedge_{i,j} \bigwedge_{s \neq t} (\overline{x}_{i,j,s} \vee \overline{x}_{i,j,t}).$$

Ennek hossza $O(p^2(n))$.

A φ_{start} azt fejezi ki, hogy táblázat első sorában a w-hez tartozó kezdőkonfiguráció van.

$$\varphi_{start} = x_{1,1,\#} \wedge x_{1,2,q_0} \wedge x_{1,3,w_1} \wedge \ldots \wedge x_{1,n+2,w_n} \wedge x_{1,n+3,\sqcup} \wedge \ldots \wedge x_{1,p(n)+2,\sqcup} \wedge x_{1,p(n)+3,\#}.$$

Ennek hossza $O(p(n))$.

A φ_{move} azt fejezi ki, hogy a táblázat minden ablaka legális.

$$\varphi_{move} = \bigwedge_{i,j} \psi_{i,j}$$

ahol $\psi_{i,j}$ egy, a

$$\bigvee_{(a_1,\ldots,a_6) \text{ legális}} x_{i,j-1,a_1} \wedge x_{i,j,a_2} \wedge x_{i,j+1,a_3} \wedge x_{i+1,j-1,a_4} \wedge x_{i+1,j,a_5} \wedge x_{i+1,j+1,a_6}$$

formulával ekvivalens konjunktív normálformájú (konstans hosszú) formula.

Itt $1 \leq i \leq p(n)$, $2 \leq j \leq p(n)+2$. A részformula hossza $O(p^2(n))$.

Végül φ_{accept} azt fejezi ki, hogy a táblázat utolsó sorában elfogadó konfiguráció van.

$$\varphi_{accept} = \bigvee_{j=2}^{p(n)+2} x_{p(n)+1,j,q_{accept}}.$$

Ennek hossza $O(p(n))$.

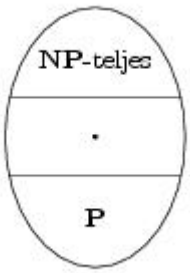
Tegyük most fel, hogy $w \in L$. Töltsük ki a táblázatot az M egy olyan számítási sorozatának megfelelően, amely a w elfogadásához vezet. Tekintsük a változók azon τ értékelését, amelyre $\tau(x_{i,j,s}) = 1$ akkor és csak akkor, ha a táblázat (i,j) pozícióján s van. Ez az értékelés kielégíti az összes részformulát és így φ-t is.

Fordítva, tegyük fel, hogy φ kielégíthető és τ egy kielégítő értékelés. Mivel τ mellett φ_0 *igaz*, minden (i,j)-re pontosan egy olyan s van, hogy $x_{i,j,s}$ *igaz*. Töltsük ki a táblázatot úgy, hogy az (i,j)-dik mezőbe akkor írjunk s-et, ha $\tau(x_{i,j,s}) = 1$. Mivel τ kielégíti a φ_{start} részformulát, ezért az első sorban a w-hez tartozó kezdőkonfiguráció van. Mivel τ kielégíti a φ_{move} formulát, ezért a táblázat az M egy lehetséges számítási sorozatának felel meg. Végül, mivel τ kielégíti a φ_{accept} formulát is, ezért w-t elfogadja M, azaz $w \in L$.

Nem nehéz egy olyan polinomidejű Turing-gépet készíteni, amely w-ből elkészíti a φ formulát. Ehhez a Turing két bináris számlálót használ az i és j tárolására. □

5.5. Az NP térképe és a coNP osztály

Egy $L \in \mathbf{NP}$ nyelvet **NP**-*közbülső* nyelvnek nevezünk, ha nincs **P**-ben, és nem is **NP**-teljes. Ismert, hogy ha $\mathbf{P} \neq \mathbf{NP}$, akkor létezik **NP**-közbülső nyelv. Ezek szerint ha $\mathbf{P} \neq \mathbf{NP}$, akkor az **NP** lehetséges térképe:



Mint már említettük, az, hogy a **P** és **NP** osztályok egybeesnek-e, híres nyitott kérdés (az általános várakozás az, hogy $\mathbf{P} \neq \mathbf{NP}$). Ennek megfelelően senkinek nem sikerült olyan nyelvet (problémát) megadnia, amely **NP**-közbülső. A gráfizomorfizmus probléma bemeneteként adott két gráf, és azt kérdezzük, hogy a két gráf izomorf-e. Ez a probléma **NP**-ben van, de nem ismert, hogy **NP**-teljes-e, és az sem, hogy **P**-ben van-e. Ez a probléma ismereteink szerint egy lehetséges **NP**-közbülső probléma. Sokáig nem volt ismert, hogy vajon **P**-ben van-e az a probléma, hogy adott számról döntsük el, hogy prímszám-e, míg végül kiderült, hogy polinomidőben megoldható.

Most térjünk át a **coNP** osztályra. Amennyiben $\mathcal{C}$ nyelvek egy osztálya, akkor **co**$\mathcal{C}$ az $\mathcal{C}$-beli nyelvek komplemenseiből áll. (Minden L nyelvet azzal a Σ halmazzal együtt tekintünk adottnak, amelyre L-et a Σ^* részhalmazának tekintjük.) A **coNP** definícióját úgy kapjuk, hogy $\mathcal{C}$-nek az **NP** osztályt választjuk.

5.26. Állítás. *Ha $\mathcal{C}$ zárt a polinomidejű visszavezetésre, akkor* **co**$\mathcal{C}$ *is zárt.*

Bizonyítás. Tegyük fel, hogy $\mathcal{C}$ zárt a polinomidejű visszavezetésre, és legyen $L_1 \leq_p L_2$, $L_2 \in \mathbf{co}\mathcal{C}$. Ekkor $\overline{L}_1 \leq_p \overline{L}_2$ és $\overline{L}_2 \in \mathcal{C}$. Mivel $\mathcal{C}$ zárt a polinomidejű visszavezetésre, így $\overline{L}_1 \in \mathcal{C}$. Tehát $L \in \mathbf{co}\mathcal{C}$. □

Mivel **NP** zárt a polinomidejű visszavezetésre, ezért kapjuk azt, hogy **coNP** is zárt a polinomidejű visszavezetésre.

Most általánosítjuk az **NP**-nehéz és **NP**-teljes nyelv fogalmát. Legyen $\mathcal{C}$ nyelvek egy osztálya. Egy L nyelv $\mathcal{C}$-*nehéz*nek nevezünk (a polinomidejű visszavezetésre nézve), ha $L' \leq_p L$ teljesül minden $L' \in \mathcal{C}$ nyelvre. Ha L $\mathcal{C}$-nehéz és $L \in \mathcal{C}$, akkor L $\mathcal{C}$-*teljes* (a polinomidejű visszavezetésre nézve).

5.27. Állítás. *Egy L nyelv akkor és csak akkor $\mathcal{C}$-nehéz ($\mathcal{C}$-teljes), ha $\overline{L}$* **co**$\mathcal{C}$*-nehéz (***co**$\mathcal{C}$*-teljes).*

Bizonyítás. Elég csak az egyik irányt bizonyítanunk, hiszen $\mathbf{co}(\mathbf{co}\mathcal{C}) = \mathcal{C}$.

Tegyük fel, hogy L $\mathcal{C}$-nehéz, és legyen $L_1 \in \mathbf{co}\mathcal{C}$. Ekkor $\overline{L}_1 \in \mathcal{C}$, és mivel L $\mathcal{C}$-nehéz, $\overline{L}_1 \leq_p L$, és így $L_1 \leq_p \overline{L}$. Beláttuk, hogy $\overline{L}$ **co**$\mathcal{C}$-nehéz. Ha még $L \in \mathcal{C}$ is teljesül, akkor $\overline{L} \in \mathbf{co}\mathcal{C}$ és $\overline{L}$ **co**$\mathcal{C}$-teljes. □

Tehát **coNP**-teljes nyelvet kapunk, ha egy **NP**-teljes nyelv komplementerét vesszük, vagy kevésbé formálisan, egy **NP**-teljes probléma ellentettje **coNP**-teljes. Ezt felhasználva könnyen adódik, hogy **coNP**-teljes pld. az a probléma, hogy egy diszjunktív normálformában adott formula tautológia-e.

Nem ismert, hogy $\mathbf{NP} = \mathbf{coNP}$ teljesül-e, az a várakozás, hogy a két osztály különbözik.

5.6. Tárbonyolultság

Az idő és tár mint erőforrás közt alapvető különbséget az jelent, hogy míg az idő nem újra felhasználható, addig a tár igen, sokszor ugyanazt a memóriaterületet használjuk a számítási folyamatban a részeredmények tárolására. Egy másik különbség az, hogy tár esetén szublineáris erőforrásigény is nemcsak érdekes, de fontos is.

A tárbonyolultság modellezésére az „offline" Turing-gépet használjuk. Ez olyan többszalagos Turing-gép, mely bemenő szalagját csak olvassa. A felhasznált munkaterületbe nem számít a bemenő szalag. Egyébként a memóriaigényt determinisztikus és nemdeterminisztikus esetben ugyanúgy számítjuk, mint az időigényt.

Legyen $f : \mathbb{N} \to \mathbb{R}_+$. Ekkor jelölje $\mathrm{SPACE}(f(n))$ ($\mathrm{NSPACE}(f(n))$) az összes olyan nyelv osztályát, amely eldönthető $O(f(n))$ tárkorlátos determinisztikus (nemdeterminisztikus) offline Turing-géppel.

Ezen a ponton visszaidézhetjük a környezetfüggetlen nyelveket. Adott G környezetfüggetlen nyelvtanra és a terminális abc n hosszú w szavára $O(n)$ tárkorlátos (egy vagy többszalagos) nemdeterminisztikus Turing-géppel könnyen ellenőrizhetjük, hogy $w \in L(G)$ teljesül-e. Megfordítva, lineáris tárkorlátos nemdeterminisztikus Turing-gépet szimulálhatunk környezetfüggetlen nyelvtannal. Tehát egy nyelv akkor és csak akkor környezetfüggetlen, ha eldönthető (vagy felismerhető) lineáris tárkorlátos nemdeterminisztikus Turing-géppel. Az ilyen nemdeterminisztikus Turing-gépeket lineárisan korlátos automatáknak is nevezik, amelyeket Myhill vezetett be az 1960-as évek elején.

5.34. Tétel. *(Savitch) Ha* $f(n) \geq \log n$, *akkor* $\mathrm{NSPACE}(f(n)) \subseteq \mathrm{SPACE}(f^2(n))$.

Bizonyítás. Tegyük fel, hogy L-et eldönti az M $O(f(n))$ tárigényű nemdeterminisztikus Turing-gép.

Ha w egy n hosszú bemenő szó, akkor egy konfiguráció leírásához meg kell adni a bemenő szalagon az író-olvasó fej pozícióját, amelyhez $O(\log n)$ bit szükséges, az aktuális állapotot, és a munkaszalagok tartalmát kijelölve az egyes szalagokon olvasott betűt. Mivel $f(n) \geq \log n$, a w-hez tartozó kezdőkonfigurációból elérhető konfigurációk mérete $O(f(n))$, száma $2^{O(f(n))}$. Megadható továbbá egy olyan c konstans, hogy tetszőleges i esetén a legfeljebb i méretű konfigurációk száma legfeljebb 2^{ci}.

Tekintsük a következő eljárást:

TEST(k_1, k_2, t, i): Adott k_1, k_2 konfigurációkra és $t \geq 1$, $i \geq 1$ egész számokra:

1. Ha $t = 1$, ellenőrizzük, hogy $k_1 = k_2$ vagy k_1-ből k_2 legfeljebb 1 lépésben megkapható-e, ennek megfelelően az eljárás *igaz* vagy *hamis* válasszal ér véget.

2. Ha $t > 1$, akkor minden k legfeljebb i méretű konfigurációra:

   3. **TEST**$(k_1, k, \lceil \frac{t}{2} \rceil, i)$,
   4. **TEST**$(k, k_2, \lceil \frac{t}{2} \rceil, i)$.

5. Ha mindkét hívás eredménye *igaz*, akkor **TEST**(k_1, k_2, t, i) is az, ellenkező esetben *hamis*.

6. Ha korábban nem ért véget az eljárás *igaz*-zal, akkor **TEST**(k_1, k_2, t, i) *hamis*.

A fenti eljárás akkor és csak akkor ad *igaz* választ, ha a k_1 konfigurációból az M Turing-gép el tud jutni a k_2 konfigurációba legfeljebb t lépésben úgy, hogy minden közbülső konfiguráció mérete legfeljebb i.

Ezek után az N determinisztikus Turing-gép működjön egy w bemenő szón a következő algoritmus szerint: $i = 1, 2, 3, \ldots$ esetén végtelen ciklusban:

1. Minden egyes lehetséges, legfeljebb i méretű elfogadó konfigurációra a **TEST** eljárás hívásával ellenőrzi, hogy a w-hez tartozó kezdőkonfigurációból elérhető-e az elfogadó konfiguráció úgy, hogy közben csak legfeljebb i méretű konfiguráción haladunk át. (Ehhez a t paraméter értékét 2^{ci}-nek kell választani.) Ha egyszer **TEST** *igaz* válasszal tér vissza, akkor N elfogadja w-ét.

2. Ezután, ha korábban nem fogadta el a w szót, a **TEST** ismételt hívásaival ellenőrzi, hogy egyáltalán elérhető-e i méretű konfiguráció (ehhez t megint 2^{ci}). Ha nem, akkor elutasítja a bemenetet. Különben i-t 1-gyel növeli.

Mivel M minden lehetséges működése a w bemeneten véges és az elérhető konfigurációk mérete $O(f(n))$, a legnagyobb i-érték, melyre **TEST** meghívásra kerül $O(f(n))$, és a legnagyobb t érték $2^{O(f(n))}$.

A rekurzióban a legnagyobb mélység így $O(f(n))$. Egy hívás tárigénye $O(f(n))$. Így az összes tárigény $O(f^2(n))$. □

5.7. Polinomiális tár

A polinomiális tárral megoldható problémák osztályát **PSPACE**-szel jelöljük. Formálisan:

$$\mathbf{PSPACE} = \bigcup_{k>0} \mathrm{SPACE}(n^k)$$

Savitch tételének alkalmazásával azonnal adódik az alábbi összefüggés.

5.11. Következmény.

$$\mathbf{PSPACE} = \bigcup_{k>0} \mathrm{NSPACE}(n^k)$$

Világos, hogy $\mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE} \subseteq \mathbf{EXP}$, ahol

$$\mathbf{EXP} = \bigcup_{k>0} \mathrm{TIME}\left(2^{n^k}\right).$$

Ismert, hogy $\mathbf{P} \subset \mathbf{EXP}$. Általánosan elfogadott az a sejtés, hogy mindegyik tartalmazás valódi.

Nevezzük qbf-nek azt a problémát, amely egy adott, zárt, prenex alakú Boole-formuláról azt kérdezi, hogy *igaz*-e. Mint nyelv, qbf az ilyen *igaz* Boole-formulák kódjait tartalmazza.

Példaként

$$\exists x_1 \forall x_2 \exists x_3 [(x_1 \vee x_2) \wedge (x_2 \vee x_3) \wedge (\bar{x}_2 \vee \bar{x}_3)] \in \mathrm{qbf}$$
$$\exists x \forall y [(x \vee y) \wedge (\bar{x} \vee \bar{y})] \notin \mathrm{qbf}$$

5.35. Tétel. *(Stockmeyer és Meyer)* qbf **PSPACE**-*teljes.*

Bizonyítás. Először belátjuk, hogy qbf $\in$ **PSPACE**. Ehhez tekintsük az alábbi algoritmust. Legyen adott a φ zárt, prenex alakú Boole-formula.

1. Ha φ kvantormentes, akkor csak konstansokat tartalmaz, így értékeljük ki.
2. Ha $\varphi = \exists x\psi$ alakú, akkor hívjuk az eljárást ψ-re először úgy, hogy az x helyébe 0-t, majd úgy, hogy x helyébe 1-et helyettesítünk. Ha valamelyik hívás *igaz*-zal tér vissza, fogadjuk el a φ bemenetet, különben utasítsuk el.
3. Ha $\varphi = \forall x\psi$ alakú, akkor hasonlóan járunk el, de abban az esetben fogadjuk el a bemenetet, ha mindkét hívás *igaz*-zal tér vissza.

Az algoritmus tárigénye: $O(n^2)$ (ill. $O(n)$, ha formálisan nem végezzük el a behelyettesítést, csak a változók rögzített értékét tároljuk).

Még be kell látnunk, hogy qbf **PSPACE**-nehéz. Ehhez legyen $L \in$ **PSPACE**, mondjuk L eldönthető az M $O(n^k)$ tárigényű determinisztikus offline Turing-géppel, ahol $k \geq 1$. Feltehető, hogy a Turing-gépnek egy munkaszalagja van.

Adott n hosszú w bemenő szóhoz szeretnénk olyan n-ben polinom méretű formulát készíteni, mely akkor és csak akkor *igaz*, ha $w \in L(M)$. Az egyszerűség kedvéért feltesszük, hogy n hosszú szavakon M elfogadás esetén mindig ugyanabban a konfigurációban áll meg.

A konfigurációk leírásához Boole-változókat használunk, mint Cook tételének bizonyításában. Egy konfiguráció leírásához $O(n^k)$ változó szükséges. Mivel az elérhető konfigurációk hossza $O(n^k)$, ezért az időigény $2^{O(n^k)}$.

Jelöljön $\varphi_{c_1,c_2,t}$ olyan formulát, mely akkor és csakis akkor igaz, ha a c_1 konfigurációból legfeljebb t lépésen elérhető a c_2 konfiguráció. A $\varphi_{c_1,c_2,1}$ könnnyen felírható, hossza $O(n^k)$ (ha minden változó hosszát 1-nek vesszük). Legyen $t > 1$. Ekkor

$$\varphi_{c_1,c_2,t} = \exists c[\varphi_{c_1,c,\lceil \frac{t}{2} \rceil} \wedge \varphi_{c,c_2,\lceil \frac{t}{2} \rceil}]$$

megfelelő lenne, de ez a módszer exponenciálisan hosszú formulákat eredményez! Ezért ezt a formulát az alábbiak szerint adjuk meg:

$$\varphi_{c_1,c_2,t} = \exists c \forall d \forall d' \left[((d = c_1 \wedge d' = c) \vee (d = c \wedge d' = c_2)) \rightarrow \varphi_{d,d',\lceil \frac{t}{2} \rceil} \right]$$

(Itt a $d = c_1$ egy olyan $O(n^{2k})$ hosszú formulának a rövidítése, amely azt fejezi ki, hogy a d által leírt konfiguráció megegyezik a c által leírt konfigurációval.)

Valójában a megadott formula nem prenex alakú, de könnyen prenex alakra hozható. Végül $w \in L$ akkor és csak akkor, ha $\varphi_{c_0,c_f,t}$ *igaz*, ahol c_0 a w-hez tartozó kezdőkonfiguráció, c_f az elfogadó konfiguráció, és t alkalmas $2^{O(n)}$ nagyságrendű szám. □

Megjegyezzük, hogy qbf akkor is **PSPACE**-teljes, ha a formula magja konjunktív normálforma, a kvantorok alternálnak, és az első kvantor $\exists$ (esetleg az utolsó is $\exists$). Ekkor qbf felfogható kétszemélyes játékként. Adott

$$\varphi = \exists x_1 \forall x_2 \exists x_3 \ldots Q x_k \psi$$

formulához tartozó játékban,

- az 1. játékos választja az $x_1, x_3, \ldots$ változók értékét és célja ψ igazzá tétele,
- a 2. játékos választja felváltva az $x_2, x_4, \ldots$ értékét, célja ψ hamissá tétele.

Így $\varphi \in$ qbf akkor és csak akkor, ha az 1. játékosnak nyerő stratégiája van.

A földrajzi játék-ban adott egy G irányított gráf és annak egy p kezdőcsúcsa. Két játékos p-ből indulva felváltva egy olyan út csúcsait nevezi meg, mely nem halad át már meglátogatott csúcson. Az a játékos veszít, aki nem tud újabb csúcsot megnevezni. A feladat annak eldöntése, hogy az első játékosnak van-e nyerő stratégiája.

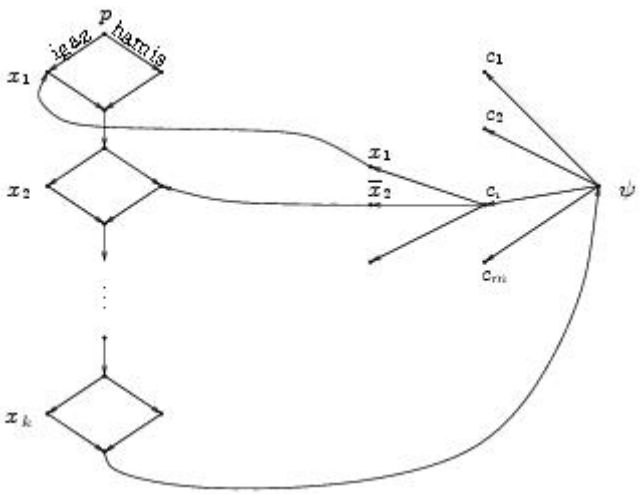
5.36. Tétel. *A* földrajzi játék **PSPACE**-*teljes.*

Bizonyítás. Az, hogy földrajzi játék **PSPACE**-ben van, a qbf-hez hasonlóan igazolható. Azt, hogy **PSPACE**-nehéz, a qbf-ből való polinomidejű visszavezetéssel igazoljuk.

Legyen adva a φ formula:

$$\varphi = \exists x_1 \forall x_2 \exists x_3 \ldots \exists x_k \psi$$

ahol k páratlan, $\psi = c_1 \wedge \ldots \wedge c_m$ egy konjunktív normálforma. Készítsük el az alábbi gráfot:



A gráfban minden változóhoz tartozik egy rombusz, a formula magjához a ψ-vel címkézett csúcs, a ψ minden c_i klózához egy csúcs, végül minden egyes c_i-ben előforduló literálhoz egy-egy csúcs. A kezdő csúcs p, az x_i változóhoz tartozó rombusz felső csúcsa. A változókhoz tartozó rombuszok fel vannak fűzve. Az x_k-koz tartozó rombusz alsó csúcsából él vezet a ψ-hez tartozó csúcsba, és onnan egy-egy él a klózokhoz tartozó csúcsokhoz. Minden klózhoz tartozó csúcsból él vezet a benne lévő literálokhoz tartozó csúcsokhoz. Végül ha egy literál az x_i változó, akkor a literálhoz tartozó csúcsból él vezet az x_i-hez tartozó rombusz baloldali csúcsába, ha pedig a literál $\bar{x}_i$, akkor a jobboldali csúcsába.

A játékosok a p-ből indulva minden rombuszon baloldalon, vagy jobboldalon haladnak végig. Annak, hogy egy x_i változóhoz tartozó rombusz baloldalán haladnak végig, feleljen

meg az x_i *igaz* (1) értéke, ha pedig a jobboldalon haladnak végig, a *hamis* (0) értéke. Miközben a játékosok végighaladnak a rombuszokon, megválasztják a változók egy lehetséges értékelését, mégpedig úgy, hogy az 1. játékos választja meg a páratlan sorszámú, 2. játékos a páros sorszámú változók értékét.

Most tegyük fel, hogy φ kielégíthető. Akkor a a qbf játékban az 1. játékosnak nyerő stratégiája van. A földrajzi játék 1. játékosa játsszon úgy, hogy valahányszor a qbf játékban az 1. játékos az x_i értékét 1-nek választja, akkor az x_i rombusz balodali csúcsát nevezi meg, különben a jobboldalit. Az így meghatározott τ értékelés kielégíti ψ-t.

Mivek k páratlan, az x_k-hoz tartozó rombusz alsó csúcsát az 2. játékos választja, és ezután az 1. játékos a ψ-hez tartozó csúcsot választja (mint egyetlen lehetőséget). Ezek után válassza ki a 2. játékos mondjuk a c_j klózhoz tartozó csúcsot. Mivet τ kielégíti c_j-t, τ mellett a c_j valamelyik literálja *igaz*. Az 1. játékos válassza ki az ennek megfelelő csúcsot! Ekkor a 2. játékos vesztett. Valóban, ha a literál az x_i változó, akkor a 2. játékos egyetlen lehetősége az lenne, hogy az x_i-hez tartozó rombusz baloldali csúcsát választja, de ezen már áthaladtak, hiszen $\tau(x_i) = 1$. Hasonlóan, ha a literál $\bar{x}_i$, akkor a 2. játékos csak az x_i-hez tartozó rombusz jobboldali csúcsát választhatná, de ezen már áthaladtak, hiszen $\tau(x_i)$ ebben az esetben 0.

Fordítva, tegyük most fel, hogy az 1. játékosnak nyerő stratégiája van a földrajzi játékban. Ekkor a qbf első játékosa játsszon úgy, hogy valahányszor a földrajzi játék 1. játékosa az x_i-hez tartozó rombusz baloldali csúcsát választja, akkor legyen $x_i = 1$, különben $x_i = 0$. Ezzel a stratégiával megnyeri a qbf játékot. □

A véges automaták elméletében is számos **PSPACE**-teljes probléma ismert, pld. a véges nemdeterminisztikus automaták ekvivalenciája, vagy az a kérdés, hogy két reguláris kifejezés ugyanazt a nyelvet ismeri-e fel. Véges determinisztikus automaták ekvivalenciája polinomidőben eldönthető. Egy további **PSPACE**-teljes nyelv a korábban bevezetett L_{CSG}, amely annak a problémának felel meg, hogy adott G környezetfüggetlen nyelvtanra és w terminális szóra döntsük el, hogy $w \in L(G)$ teljesül-e.

5.8. Logaritmikus tárral való visszavezetés és az L és NL osztályok

Az **L** és **NL** nyelvosztályok a logaritmikus tárral determinisztikusan ill. nemdeterminisztikusan megoldható problémáknak felelnek meg. Formálisan:

$$\begin{aligned} \mathbf{L} &= \mathrm{SPACE}(\log n) \\ \mathbf{NL} &= \mathrm{NSPACE}(\log n) \end{aligned}$$

Például, az elérhetőség **NL**-ben van. Valóban, ha adott egy G irányított gráf az s és t csúcsokkal, akkor s-ből indulva nemdeterminisztikusan válasszunk egy olyan csúcsot, amelybe az előzőleg meglátogatott csúcsból él vezet. Ehhez mindig csak az utoljára meglátogatott csúcsot kell tárolnunk, amihez logaritmikus tár elég. Amennyiben a t csúcsot kapjuk, fogadjuk el a bemenetet. Azt, hogy a nemdeterminisztikus Turing-gép minden számítási sorozata véges legyen, egy számláló segítségével érjük el, amelyben számoljuk a megtett lépéseket. Ha ez n-en túl megy, és korábban nem látogattuk meg a t csúcsot, elutasítjuk a bemenetet. A számlálónak is elegendő a logaritmikus tár.

Világos, hogy

$$\mathbf{L} \subseteq \mathbf{NL} \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE}.$$

Az is ismert, hogy $\mathbf{NL} \subset \mathbf{PSPACE}$. Az a sejtés, hogy a fenti tartalmazások mindegyike valódi.

Legyen $L_1 \subseteq \Sigma_1^*$, $L_2 \subseteq \Sigma_2^*$. Azt mondjuk, hogy az $f : \Sigma_1^* \to \Sigma_2^*$ függvény az L_1 *logaritmikus tárral való visszavezetése* L_2-re, ha minden $w \in \Sigma_1^*$ szóra $w \in L_1$ akkor és csak akkor, ha $f(w) \in L_2$ és f kiszámítható $O(\log n)$ tárkorlátos determinisztikus Turing-géppel. Itt olyan offline Turing-gépre gondolunk, amelynek egyik munkaszalagja kitüntetett, arra csak balról jobbra ír, és itt állítja elő az $f(w)$ szót. A kimenő szalag nem számít bele a felhasznált munkaterületbe, így $f(w)$ hossza a w hosszának polinomfüggvénye is lehet. Azt mondjuk, hogy $L_1 \subseteq \Sigma_1^*$ *logaritmikus tárral visszavezethető* az $L_2 \in \Sigma_2^*$ nyelvre, $L_1 \leq_\ell L_2$, ha létezik olyan $f : \Sigma_1^* \to \Sigma_2^*$ függvény mely az L_1-nek az L_2-re való logaritmikus tárral való visszavezetése. Mivel minden logaritmikus tárkorlátos Turing-gép polinom időkorlátos, ezért ha $L_1 \leq_\ell L_2$, akkor $L_1 \leq_p L_2$.

5.37. Tétel. *$A \leq_\ell$ reláció előrendezés.*

A bizonyítás, amit itt nem közlünk, erősen kihasználja a tár újrahasznosíthatóságát.

Ismertek az alábbi eredmények is.

5.38. Tétel. *Az* elérhetőség **NL***-teljes a logaritmikus tárral való visszavezetésre.*

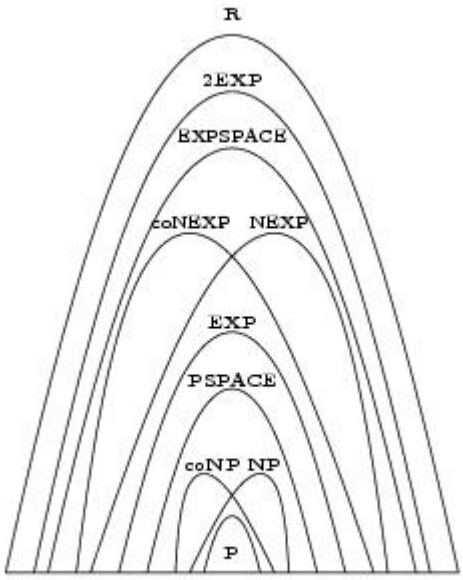
5.39. Tétel. *(Immermann – Szelepcsényi)* $\mathbf{NL} = \mathbf{coNL}$.

A logaritmikus tárral való visszavezetésre nézve definiáhatóak **P**-teljes problémák. Ismert, hogy **P**-teljes annak eldöntése, hogy egy környezetfüggetlen nyelvtan üres nyelvet generál-e. (Ld. a 4.16 Állítást is.)

5.9. Túl a PSPACE osztályon

A **PSPACE** osztály felfelé is kiterjeszthető. Már említettük az **EXP** osztályt, melynek nemdeterminisztikus megfelelője a **NEXP**. Mind **NEXP**, mind **coNEXP** az $\mathbf{EXPSPACE} = \bigcup \mathrm{SPACE}(2^{n^k})$ részosztálya. Ezekben a bonyolultsági osztályokban is léteznek teljes problémák. Ilyeneket kapunk akkor, ha a **P**-teljes, **NP**-teljes vagy **PSPACE**-teljes problémák tömör változatait tekintjük. A tömör Hamilton-út probléma esetén a gráfot binárisan kódoljuk, azaz a csúcsokat és az él relációt Boole-függvényekkel írjuk le, és ezeket hálózatokkal adjuk meg, melyek mérete akár exponenciálisan is kisebb lehet. A tömör Hamilton-út probléma **NEXP**-teljes.

Minden $k \geq 2$-re definiálhatjuk a $\mathbf{kEXP} = \bigcup \mathrm{TIME}(2^{2^{\cdots^{2^n}}})$ osztályt (a toronyban k darab kettes szerepel), melyek egyesítése az *elemi* nyelvek (problémák) osztálya, a rekurzív nyelvek **R** osztályának valódi részosztálya.


R
2EXP
EXPSPACE
coNEXP
NEXP
EXP
PSPACE
coNP
NP
P

Ajánlott irodalom

Magyar nyelven

1. Fülöp Zoltán: *Automaták és formális nyelvek*, SZTE, Számítástudomány Alapjai Tanszék, 2009.

2. Gécseg Ferenc: *Automaták és formális nyelvek*, Polygon, Szeged, 2005.

3. Lovász László: *Algoritmusok bonyolultsága*, Tankönyvkiadó, 1989.

4. Ch. Papadimitriou: *Számítási bonyolultság*, Novadat Kiadó, 1999.

5. Rónyai Lajos, Ivanyos Gábor, Szabó Réka, *Algoritmusok*, Typotex Kiadó, 2008.

Idegen nyelven

1. M. D. Davis, E.J. Weyuker: *Computability, Complexity, and Languages*, Academic Press, 1985.

2. M. A. Harrison, *Introduction to Formal Language Theory*, Addison Wesley, Reading, 1978.

3. J. E. Hopcroft, R. Motwani, J. D. Ullman, *Introduction to Automata Theory, Languages, and Computation*, 2007.

4. D. C. Kozen, *Automata and Computability*, Springer, 1997.

5. H. R. Lewis, Ch. Papadimitriou: *Elements of the Theory of Computation*, 2nd ed., Prentice Hall. 1998.

6. M. Sipser: *Introduction to the Theory of Computation*, PWS Publishing Co., 1997.