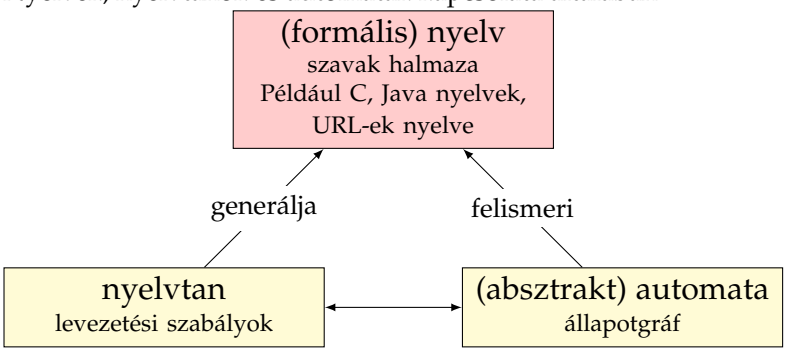
Formális nyelvek és automaták

Horváth Árpád

2017. november 20.

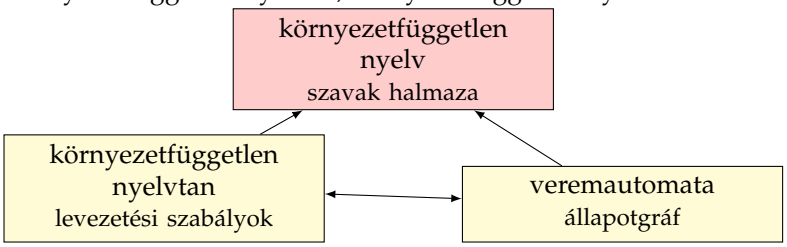
Nézzük először vázlatosan a félév fontosabb fogalmait!

Nyelvek, nyelvtanok és automaták kapcsolata általában



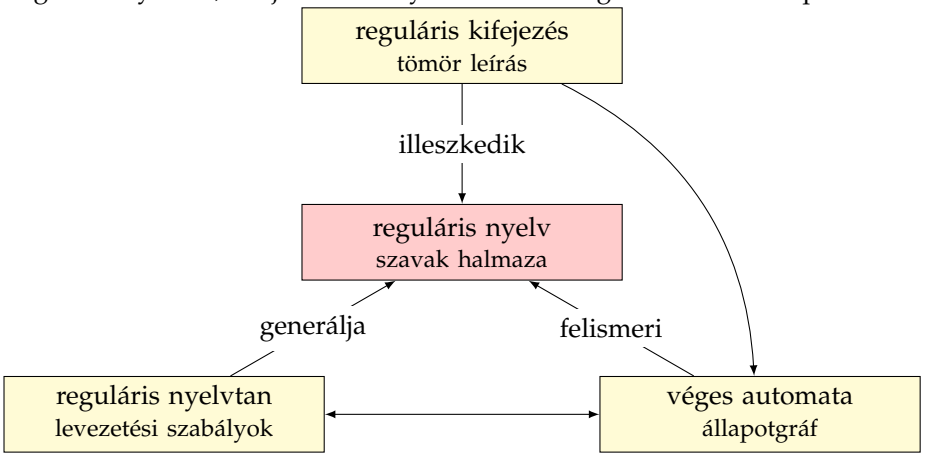
Ezeket a nyelveket majd osztályokba fogjuk sorolni. Például a környezetfüggetlen nyelvekhez tartoznak a programozási nyelvek.

Környezetfüggetlen nyelvek, környezetfüggetlen nyelvtanok ill. veremautomaták kapcsolata



A környezetfüggetlen nyelvekhez hasonló diagram rajzolható fel a környezetérzékeny és az általános nyelvtanok esetén is, csak az automata típusa lesz más és más. Sőt a reguláris nyelvtanok esetén is, de ott a nyelvnek még egy hasznos megadási módot fogunk tanulni: az úgynevezett reguláris kifejezésre illeszkedő szavak is reguláris nyelvet alkotnak.

Reguláris nyelvek, kifejezések és nyelvtanok ill. véges automaták kapcsolata



1. Formális nyelvek

Ebben a segédletben néha használom a *szó* kifejezést, mely a Bach-könyvben *mondatként** szerepel.
Pár azonos értelemben használt kifejezés a szakirodalomból:
ábécé = alfabeta* (veremábécé = verem alfabetája*)
nyelvtan*=grammatika* de nyelvtan≠nyelv!
DFA = FDA = determinisztikus véges automata*
reguláris kifejezés = reguláris halmaz*
végállapot = elfogadó állapot*
A *-gal jelölteket használja a könyv.
De kezdjük az elején.

Formális nyelvek alapfogalmai és jelölései

1.1. definíció. *A formális nyelvek alapvető definíciói.*

- *Ábécé: szimbólumok véges halmaza (Σ) Ezek egyesítése, különbsége a halmazokéhoz hasonlóan definiált.*
- *Ábécé betűi: a szimbólumok (a, b, c_k) $a \in \Sigma$*
- *Szó: szimbólumok véges sorozata (A szavakat később gyakran az ábécé végéről vett jelekkel fogjuk jelölni (v,w,x,y,z), ebben az esetben azt fogja jelenteni, hogy csak terminális szimbólumok vannak benne; hasonló nagybetűkkel (V,W,X,Y,Z) pedig olyanokat, amelyben csak nemterminálisok; görög betűkkel (α, β, ω) pedig olyanokat, amelyikben bármelyik.)*
- *Szó hossza: a sorozat hossza ($len(\alpha)$) (Az angol length=hossz szóból.)*
- *Üres szó (ε): melyre $len(\varepsilon) = 0$ (Ha a szavakra úgy gondolunk, mint a programozási nyelvek sztringjeire, akkor a ε az üres sztringnek, felel meg. Az ε talán könnyebben megjegyezhető úgy, hogy az az angol empty szó kezdőbetűjének görög megfelelője (kis epszilon).)*
- *Szavak konkatenációja (összefűzése):*
  $yw = a_1a_2a_3\cdots a_nb_1b_2\cdots b_m$, ahol $y = a_1a_2a_3\cdots a_n$, $w = b_1b_2\cdots b_m$ (Asszociatív, kommutatív, egységelemes-e? $len(yw) = ?$)
- *Tükörkép: $y^{-1} = a_n\cdots a_2a_1$ a fenti ipszilonnal, azaz a betűit fordított sorrendben írom le.*
- *Hatvány: $y^0 = \varepsilon, y^1 = y, y^n = y^{n-1}y$. (Pl. $(ab)^3 = ababab$)*
- *Σ^* a Σ ábécéből alkotott összes szó, beleértve az üres szót is.*
  Például $\{a,b\}^* = \{\varepsilon,a,b,aa,ab,ba,bb,aaa,\ldots abbaaab,\ldots\}$
- *Σ ábécéből alkotott formális nyelv ($\mathcal{L}(\Sigma)$ vagy $\mathcal{L}$): bármely $\mathcal{L}(\Sigma) \subseteq \Sigma^*$*

1.1. példa. *Példák nyelvekre.*

- $\Sigma = \{0,1\}$,
  $\mathcal{L}_1(\Sigma) = \{01,0,\varepsilon\}$,
  $\mathcal{L}_2(\Sigma) = \{0,1\}^* = \{$ $\}$,
- $\Sigma_2 = \{a,b,c\}$,
  $\mathcal{L}_3(\Sigma_2) = \{a^ib^ic^i \mid i \geq 0\} = \{$ $\}$,
  $\mathcal{L}_4(\Sigma_2) = \{ycy^{-1} \mid y \in \{a;b\}^*\} = \{$ $\}$,
- $\mathcal{L}_0 = \emptyset$, $\mathcal{L}_e = \{\varepsilon\}$,
- $\Sigma_3 = \{if; then; a; b; c\}$ $\mathcal{L}(\Sigma_3) = \{c;$ *if a then b; if a then if b then c*$\}$

Nyelvtanok megadásához 4 dolgot kell megadni: $G(\Sigma;N;S;P)$
Σ: terminális szimbólumok, N: nemterminális szimbólumok, S: kezdőszimbólum, P: helyettesítési szabályok. Általában a jelölésrendszer egyértelműsége miatt a P helyettesítési szabályok megadása elegendő. Általában ugyanis a nemterminálisokat nagybetűvel jelöljük, ezek közül a kezdőszimbólumot S-el, a többi szimbólum terminális szimbólum.

A programozási nyelvek környezetfüggetlen nyelvtanának leírásában általában nyíl helyett ::= jelölést használjuk, és gyakran a terminálisokat idézőjelbe rakják. Erről az úgynevezett Backus–Naur jelölésről a segédletben később esik szó.

1.2. példa. *Határozzunk meg az alábbi helyettesítési szabályok esetén mik lesznek a nem terminális és terminális szimbólumok! Vezessünk le az értékadó_kifejezés szimbólumból egy szót!*

1. *értékadó_kifejezés ::= változó értékadó_operátor értékadó_kifejezés*
2. *értékadó_kifejezés ::= változó*
3. *értékadó_operátor ::=* "="

4. *értékadó_operátor* ::= `"*="`

5. *értékadó_operátor* ::= `"/="`

6. *értékadó_operátor* ::= `"+="`

7. *értékadó_operátor* ::= `"-="`

8. *változó* ::= `"a"`

9. *változó* ::= `"b"`

(A Kernighan–Ritchie: A C programozási nyelv *(1996) című könyv 258. oldalán található* A C nyelv szintaktikájának összefoglalása. *Egy részének jelentősen egyszerűsített változata szerepel itt.)*

A helyettesítési szabályok első sorának jobboldalán három tag szerepel, mindhárom nemterminális, a második sor jobb oldalán egy nemterminális, a következő sorok jobb oldalán terminális szimbólumok szerepelnek. A nemterminális szimbólumok:
N = {*értékadó_kifejezés, változó, értékadó_operátor*}
A terminális szimbólumok: Σ = {*a, b,* =, *=, /=, +=, -=}

Kiindulva az értékadó_kifejezés nemterminális szimbólumból, az alábbi lépésekben megkaphatunk egy csupán terminális szimbólumból álló kifejezést:
értékadó_kifejezés $\Rightarrow$ *(1)*
változó értékadó_operátor értékadó_kifejezés $\Rightarrow$ *(3)*
változó = értékadó_kifejezés $\Rightarrow$ *(8)*
a = értékadó_kifejezés $\Rightarrow$ *(1)*
a = változó értékadó_operátor értékadó_kifejezés $\Rightarrow$ *(4)*
*a = változó *= értékadó_kifejezés* $\Rightarrow$ *(9)*
*a = b *= értékadó_kifejezés* $\Rightarrow$ *(8)*
*a = b *= a*

*(Ez a kifejezés amúgy C-ben érvényes kifejezés, először a *= kifejezés hajtódik végre, megszorozva a b értékét a-val, ez kerül először b-be, majd ez az eredmény kerül a-ba is.)*

A terminális és nemterminális szimbólumok (pl. változó) egy ábécét alkotnak a korábbi elnevezéseink szerint, a kapott kifejezés pedig egy terminális szimbólumokból álló szó.

Az P-ben található helyettesítési szabályok segítségével az S kezdőszimbólumból kiindulva egy vagy több lépésben levezethetünk bizonyos szavakat, amelyek csak nemterminális szimbólumokból állnak. Az így levezethető szavakat nevezzük a G nyelvtan által generált nyelvnek. Jele: $\mathcal{L}(G)$.

A továbbiakban azt vizsgáljuk, hogy milyen lehetőségeink vannak annak ellenőrzésére, hogy egy szó benne van-e a nyelvtan által generált nyelvben.

A feladatmegoldás során a *problémát az általános megoldása felől közelítjük* meg ahelyett, hogy elköteleznénk magunkat a technikai részletek mellett, hogy például a feladatot hardveresen vagy szoftverrel oldjuk-e meg.

A nyelv szavait felismerő elméleti konstrukciókat *automatáknak* fogjuk nevezni. Ezeknek több típusa van, és hogy melyik az a legegyszerűbb típus, amit alkalmazni tudunk, a nyelv típusától függ. Vizsgáljuk meg tehát, milyen típusok vannak.

Az 1.2. példában láttunk egy példát egy C-szerű nyelv egy részletének leírásából. Ott a nemterminális szimbólumok esetén olyan leírásokat találhatunk, mint a változó, értékadó_kifejezés, vagy értékadó_operátor. A továbbiakban mi általában ezeknél rövidebb jelölésmódot fogunk alkalmazni. Latin nagybetűkkel (S, A, B) fogjuk jelölni a nemterminális szimbólumokat, és kisbetűvel (a, b, c) a terminális szimbólumokat.

Nyelvtanok és nyelvek típusai (Chomsky-hierarchia)

A nyelvek típusait a nyelvtanok típusaiból fogjuk származtatni. Először tehát ezt vizsgáljuk meg. Először egy táblázatban összefoglalom a nyelvtanok típusait, majd részletesebben kifejtem.

Legyen $\alpha, \beta, \omega \in (\Sigma \cup N)^*$ $A, B \in N$ $a \in \Sigma$

szabályok típusa	nyelvtan típusa	automata
$\alpha A \beta \to \omega$	0-típusú, általános	Turing-gép
$\alpha A \beta \to \alpha \omega \beta$	1-típusú, környezetérzékeny	
$A \to \omega$	2-típusú, környezetfüggetlen (CF)	PDA
$A \to aB$ vagy $A \to a$	3-típusú, reguláris	DFA

Ezek a nyelv- és nyelvtanosztályok felülről lefelé tartalmazzák egymást.

A környezetfüggetlen nyelvnél nem szoktuk megengedni, hogy a jobboldalt üres szó szerepeljen. Így a környzetfüggetlen nyelv nemnövelő lesz, ami azt jelenti, hogy a levezetés egyik lépésében sem csökken a szimbólumok száma.

Megengedjük viszont az $S \to \varepsilon$ szabályt a környzetfüggetlen nyelv és a reguláris nyelvek esetén, ha sehol sem szerepel a $\to$ jobboldalán S. Ha ezt hozzávesszük, akkor egy nyelv típusa nem fog függeni attól, ha az üres szót ε hozzáadjuk, vagy elvesszük belőle. (A Python programozási nyelvben például egy 0 bájtos fájl érvényes program, csak nem csinál semmit.)

Az általános nyelvtan tényleg a legáltalánosabb. A szabályoktól csak azt követeljük meg, hogy a baloldalán legyen nem terminális szimbólum, és egy ilyen részt bármilyen szóvá átalakíthatunk.

A környezetérzékeny (Context Sensitive, CS) nyelvtannál már megköveteljük, hogy ha a szabály bal oldalán ha a nemterminális szimbólum mellett annak jobb és baloldalán valamilyen szó áll, akkor az változatlanul maradjon meg az átalakított szó elején.

A környezetfüggetlen (Context Free, CF) nyelvtannál már csak egy nemterminális szimbólumot alakíthatunk át tetszőleges szóvá. A környezetfüggetlen tényleg részhalmaza az előzőnek. A környezetérzékenyből csak azokat a speciális átalakítási szabályokat engedjük itt meg, ahol $\alpha = \beta = \varepsilon$, azaz mindkettő üres szó.

1.2. definíció. *A nyelvet n-típusúnak nevezünk, ha van olyan n-típusú nyelvtan, amely a nyelvet generálja. (Pl: környezetfüggő nyelvet generál a környezetfüggő nyelvtan.)*

Eszerint például a környezetfüggetlen, nyelvtanok által generált nyelveket környezetfüggetlen nyelveknek nevezzük. Ezekhez a nyelvekhez tartozik sok programozási nyelv.

A reguláris nyelvtanok csak egész speciális formájú átalakításokat engednek meg, ahogy a táblázatban látjuk, egy nemterminális szimbólumot alakíthatunk egy terminálissá vagy egy terminális utáni nemterminálissá.

A reguláris nyelv szoros kapcsolatban van a programozási nyelvek nagy részében alapból vagy függvénykönyvtárak segítségével elérhető reguláris kifejezésekkel.

A 9. fejezetben található feladat a Chomsky-féle normálformákba történő besorolásra és szavak levezetésére.

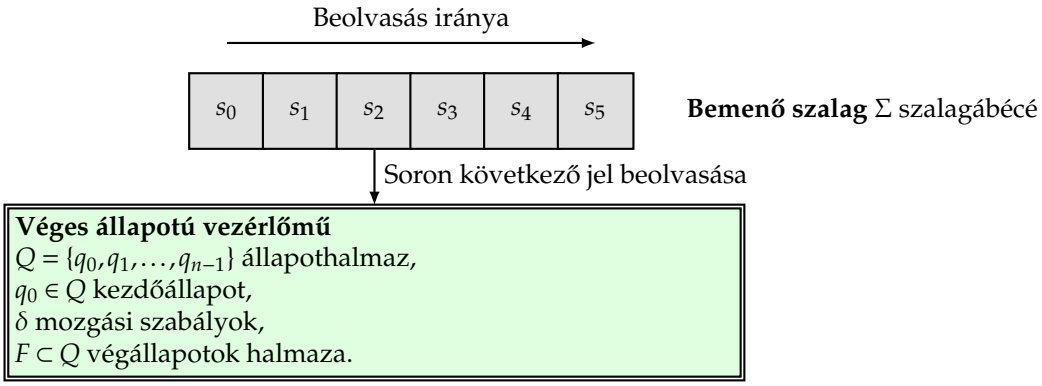
2. Véges automaták, mint nyelvek felismerői

A véges automaták, az alábbi ábrán látható módon, egy szalagból, és egy véges vezérlőműből áll. A szalagon található egy szó, amely egy adott ábécé szimbólumaiból áll. A véges vezérlőmű minden lépésben beolvas egy szimbólumot, és a mozgási szabályok figyelembevételével új állapotba kerülhet. Előfordulhat, hogy az adott állapot esetén adott szalagábécébeli szimbólum esetén nincs szabály arra, milyen állapotba kerüljön. Ilyenkor a felismerés elakad.

A szó felismerésére először azt az egyszerűbb változatot vizsgáljuk meg, amikor minden állapotban minden szalagábécébeli szimbólum esetén egyetlen mozgási szabály van. Ilyenkor azt mondjuk, hogy a véges automata determinisztikus. Determinisztikus véges automata (DFA) esetén akkor mondjuk, hogy egy szalagon szereplő szót felismert, ha a szót végig tudta olvasni, és a végigolvasás végén végállapotba került.

Elvileg megállapítható azoknak a szavaknak a (véges vagy végtelen) halmaza, amelyet az automata felismer. Az ilyen szavak halmaza nyelvet alkot. Ezt a nyelvet nevezzük az automata által felismert nyelvnek. A véges automaták esetén be fogjuk látni, hogy a véges automaták által felismert nyelv csak reguláris nyelv lehet, és bármely reguláris nyelv esetén van olyan véges automata, amely az adott nyelvet ismeri fel.

A véges automata szalagábécéje lesz a felismert nyelv ábécéje, ezért jelöljük Σ-val mindkettőt.



Egy véges automata vezérlőművét legszemléletesebben állapotgráffal adhatjuk meg, ami egy irányított gráf: nyíl mutat a kezdeti (általában q_0-lal vagy S-el jelölt) állapotra. A nyilakon szereplő szimbólumok (a Σ szalagábécéből) jelzik, hogy egy adott állapotból azt olvasván a szalagról hova jut tovább. A kettős körök jelzik a végállapotokat.

A véges automatákat mi állapotgráffal fogjuk ábrázolni általában. Tudni kell azonban, hogy minden állapotgráfhoz 5 dolgot kell tudni, amiknek a jelei $(Q, \Sigma, \delta, q_0, F)$. Q az automata állapotainak a halmaza, Σ a felismerendő nyelv ábécéje, δ függvény a mozgási szabályokat tartalmazza, egy-egy q_i állapotra és ábécébeli a szóra $\delta(q_i, a_j)$ visszaadja, hogy milyen következő állapotba kerül az automata. Tudnunk kell, hogy melyik állapot a kezdőállapot, de ez a jelöléséből (q_0 vagy S) is nyilvánvaló szokott lenni (Állapotgráfon egy rövid nyilacska az állapothoz). És végül tudnunk kell, melyek a végállapotok (F: végállapotok halmaza; állapotgráfon dupla karika jelöli). Gyakran az egész automatának nevet is szoktunk adni (pl. M, M_1, M_2, M'), ilyenkor az alábbi jelölést használjuk $M(Q, \Sigma, \delta, q_0, F)$:

Nemdeterminisztikus a véges automata (NFA), ha van olyan állapot, amelyből egy terminális szimbólum hatására legalább kétfelé mehetünk tovább. NFA esetén akkor mondjuk, hogy egy szót az automata felismer, ha (az általában több lehetséges útvonal közül) van olyan állapotsorozat, amelynek során végig tudjuk olvasni a szót, és végállapotba jutunk.

Tudni kell:

- megállapítani, hogy egy szót felismer-e egy DFA, NFA,
- egy mozgássorozatot konfigurációsorozattal leírni,
- egyszerűbb automatákból megállapítani, milyen nyelvet ismer fel, és azt halmazjelölésekkel felírni,

- megállapítani, hogy egy véges automata DFA vagy NFA-e,
- hogyan alakíthatjuk át a nemdeterminisztikus automatákat determinisztikussá.

Bach 2.2. szakasz, Determinisztikus és nemdeterminisztikus véges automaták. A 37. oldalon van egy NFA→DFA átalakítás, én ennél egyszerűbbeket kérdezek, de azt érdemes végigolvasni és megérteni.

3. Reguláris nyelvek és véges automaták

Az automaták és a nyelvek között a táblázatban látható kapcsolat van. Bármely véges automata által felismert összes szóból alkothatunk egy nyelvet, és ez mindig reguláris nyelv lesz. Fordítva: minden reguláris nyelvhez hozhatunk létre olyan véges automatát, amely azt a nyelvet ismeri fel. Hasonló a kapcsolat a többi, a táblázatban feltüntetett automata és nyelvcsalád között. A véges automata helyett azért írtunk DFA-t, mert belátható, hogy minden véges automata átalakítható *determinisztikus* véges automatává, úgy, hogy ugyanazokat a szavakat ismerje fel illetve utasítsa el mint az eredeti. Tehát a determinisztikus véges automaták mindarra képesek, mintha még hozzájuk vennénk a nem determinisztikusakat (NFA-kat) is.

Az alábbiakban megadjuk, hogyan lehet átalakítani egy reguláris nyelvtant véges automatává úgy, hogy a véges automata ugyanazt a nyelvet ismerje fel, mintha amit a reguláris nyelvtan generál. Mivel ez az átalakítás mindig lehetséges, ebből következik, hogy a reguláris nyelvekhez mindig tartozik azt felismerő véges automata.

Reguláris nyelvtan → véges automata átalakítási szabályok

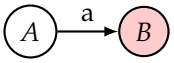
Kell kezdetben egy kezdőállapot 

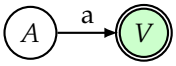
,

és egy végállapot 

ahová a $A \to a$ alakú szabályokat vezetjük.

A további szabályok:

$A \to aB$ 

$A \to a$ 

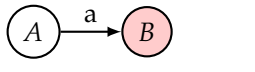
$S \to \varepsilon$ 

Az átalakítás során minden nyelvtanbeli nemterminálishoz létrehozunk egy állapotot, és a szabályok szerint megrajzoljuk a nyilakat. Ha van $S \to \varepsilon$ szabály, akkor a kezdőállapotot egyben végállapottá tesszük, azaz duplán karikázzuk.

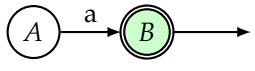
(Lehet-e vajon a fenti átalakításokkal „szabálytalan" epszilon szabályt létrehozni? Bizony lehet, ha a kezdő állapotból, amely végállapot is, önmagába nyíl mutat, akkor jobboldalon is elő fog fordulni az S kezdőszimbólum. Akit érdekel végiggondolhatja, hogyan lehetne átalakítani a „hibás" nyelvtant, hogy ez ne forduljon elő, de ugyanazt a nyelvet ismerje fel. Megnyugtatásul közlöm, hogy ez megtehető a reguláris nyelveket magában foglaló környezetfüggetlen nyelvek körében is, és ezt a Bach-könyv ott tárgyalja.)

Az alábbi szabályok megmutatják, hogyan alakíthatunk át egy véges automatát reguláris nyelvtanná olyan módon, hogy a reguláris nyelvtan ugyanazt a nyelvet generálja, mint amit a véges automata felismer. Ez az átalakítás minden esetben lehetséges, aminek az a következménye, hogy minden véges automata reguláris nyelvet ismer fel.

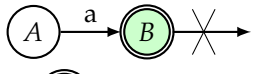
Véges automata → reguláris nyelvtan átalakítási szabályok



$A \to aB$



$A \to aB, \quad A \to a$



$A \to a$

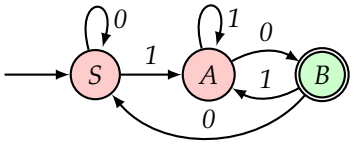
$S \to \varepsilon$

Azaz ha tetszőleges állapotból nem végállapotba megyünk, vagy a végállapotból nem vezet ki nyíl, akkor egyetlen szabállyal helyettesíthetjük a nyilat, ha a végállapotból tovább vezet nyíl (akár önmagába is) akkor viszont két szabállyal helyettesíthetem csak. Ha az S kezdőállapot végállapot is, akkor kell egy epszilon-szabály.

Ha egyszer oda-vissza alakítunk, akkor nem feltétlenül ugyanazt a véges automatát kapjuk, de vele egyenértékűt. Az automatává alakításkor csak a V lesz végállapot, és esetleg az S, míg kezdetben akár kettőnél több végállapot is lehet.

A környezetfüggetlen nyelvek felismeréséhez már nem elegendő az DFA. Ahhoz már az DFA-t ki kell egészíteni egy veremmel, egy olyan tárral, aminek minden lépés során a tetejére rakhat egy elemet, vagy levehet onnan egyet, és az állapotváltozásai során figyelembe veszi a verem felső elemét is, ez a veremautomata (Push Down Automaton, PDA), amiről a környezetfüggetlen nyelvek esetén lesz szó, mivel a veremautomaták a környezetfüggetlen nyelvekkel van olyan kapcsolatban, mint a véges automaták a reguláris nyelvekkel.

3.1. példa. *Határozzuk meg, hogy az alábbi DFA felismeri-e a következő szavakat: ε, 010, 0101, $(10)^5$? Vezesse le a 010 szót konfigurációsorozattal! Írjuk le szavakkal és halmazjelölésekkel, hogy milyen szavakat ismer fel! Mi a A és B állapotok jelentése? Milyen típusú nyelv a felismert nyelv? Honnan tudom, hogy determinisztikus ez a véges automata?*



M

Az üres szónál S-ben marad, ami nem végállapot: az üres szót nem ismeri fel. A 010 esetén a következő állapotokon megy végig SSAB. Mivel a B végállapot, ezért a szót felismerte. A 0101 szó esetén ugyanazokon az állapotokon megy keresztül, csak még visszalép a A-ba, ami nem végállapot, tehát a szót nem ismeri fel. $(10)^5 = 1010101010$ *esetén az első 1-esnél átjut A-ba majd A és B között lépked oda-vissza. Végül a B-be jut, tehát felismeri az automata.*

A véges automatáknál a további lépések megállapításához elég a pillanatnyi állapot és a bemenő szó maradék része, tehát egy konfiguráció azt tartalmazza. A 010 levelezése konfigurációsorozattal:

$$(S,010) \mapsto (S,10) \mapsto (A,0) \mapsto (B,\varepsilon)$$

A 0101 levelezése konfigurációsorozattal:

$$(S,0101) \mapsto (S,101) \mapsto (A,01) \mapsto (B,1) \mapsto (A,\varepsilon)$$

Ha a levezetés végén elfogy a szó, és elfogadó állapotba kerülök, akkor a szót felismerte a DFA, az automata által elfogadott nyelvhez tartozik, ha nincs ilyen levezetés, akkor nem. Ha a fenti automatát M-el jelöljük, akkor halmazjelölésekkel:

$$010 \in \mathcal{L}(M), \quad 0101 \notin \mathcal{L}(M), \quad \{010,(10)^5,1110\} \subset \mathcal{L}(M).$$

A A állapotba kerül az automata bárhonnan, ha 1-est olvas. Tehát A jelentése az, hogy utoljára 1-es volt a szóban. A B-be csak a A-ból juthat az automata, ha 0-át olvas. Tehát B jelentése, hogy a szóban utoljára 10 volt. Az automata tehát azokat a szavakat ismeri fel, amelynek a végén 10 van, azaz halmazjelöléssel:

$$\mathcal{L}(G) = \{y10 \mid y \in \{0,1\}^*\}.$$

Az DFA-k pontosan a reguláris nyelveket ismerik fel, tehát $\mathcal{L}(M)$ *reguláris nyelv.*

A determinisztikusság igazolásához ellenőrizni kell minden egyes állapotnál, hogy egyféle szimbólum hatására csak egyfelé tudok-e továbbmenni. Jelen esetben nincs egyetlen olyan állapot sem, amelyből két 1-es cimkéjű, vagy két 0-ás cimkéjű nyíl vezetne ki, tehát tényleg DFA.

3.1. Műveletek nyelvekkel

Ismerni kell a Bach-könyv 2.5. fejezetében szereplő műveleteket: komplemens, unió, metszet, konkatenált, hatvány, tranzitív lezárt.

- Nyelvek konkatenációja: $\mathcal{L}_1\mathcal{L}_2 = \{xy \mid x \in \mathcal{L}_1,\ y \in \mathcal{L}_2\}$
- Nyelvek hatványa: $\mathcal{L}^0 = \{\varepsilon\}, \quad \mathcal{L}^1 = \mathcal{L}, \quad \mathcal{L}^2 = \mathcal{L}\mathcal{L}, \quad \mathcal{L}^n = \mathcal{L}^{n-1}\mathcal{L}$
- Nyelvek tranzitív lezártja: $\mathcal{L}^* = \mathcal{L}^0 \cup \mathcal{L}^1 \cup \mathcal{L}^2 \cup \mathcal{L}^3 \ldots$

Milyen nyelvek kapunk, ha ezekben a műveletekben reguláris nyelvek a kiinduló nyelvek?
Hogyan igazoljuk ezt komplemensképzésre?
Példák. Legyen $\mathcal{L}_1 = \{01,0,101,\varepsilon\}, \quad \mathcal{L}_2 = \{y0y^{-1} \mid y \in \{0;1\}^*\}$ ekkor:

- $\mathcal{L}_1 \cap \mathcal{L}_2 = \{101\}$
- $01'01010 \in \mathcal{L}_1\mathcal{L}_2$ az ' csak a határt jelöli
- $01'0'01'01'0 \in \mathcal{L}_1^5$
- $01'101'01'0 \in \mathcal{L}_1^*$

3.2. Reguláris kifejezések

A reguláris kifejezéseket és reguláris halmazokat rokon értelmű szóként használjuk. A Bach-könyv 2.6. fejezete az utóbbit, mi az előbbit használjuk és kitérünk gyakorlati használatukra is.

Az elméleti és a gyakorlati jelölés némileg eltér egymástól, és az utóbbi jelentősen bővebb.

A legfontosabb eltérés a jelölésben, hogy az elméletben (pl. a könyvben) + jelet használnak ahol a gyakorlatban függőleges vonalat (|).

Gyakorlati alkalmazást nappalisokkal laborban nézzük (linux/vim/regexp_vim.txt) ebből a levelezős, távos hallgatóknak csak a * | és () valamint a karakterek levédése (\) kell.

A gyakorlati alkalmazásban is kétféle formátuma van a reguláris kifejezéseknek, például Java Scriptben, Pythonban a csoportosítás zárójelét, az 1 vagy több ismétlődést jelentő +-ot és az adott számú előfordulást jelentő `{3}` zárójeleket nem kell levédeni \-sel (backslash-sel). Ott viszont az a nehezebb ha kerek vagy kapcsos zárójelre kell illeszteni. Olyankor kell levédeni.

4. Feladatok a nyelvcsaládokhoz és reguláris nyelvekhez, véges automatákhoz

A *távos, levelezős* Számítástudomány tárgyhoz ezek beadandó feladatok voltak. Újabban teszt van, amelyhez felkészüléskor érdemes ezeket átnézni. A tesztnél lehet, hogy papíron végig kell ilyen feladatokat vezetni, hogy válaszolni tudjunk a tesztkérdésekre.

4.1. feladat. *Határozzuk meg, hogy az $\mathcal{L}(G)$ nyelvben benne van-e: ε, abb, aabb? Írjuk le szavakkal és halmazjelölésekkel, milyen szavakat tartalmaz! Soroljuk be a Chomsky-féle hierarchiába a nyelvtant!*

$$\Sigma = \{a,b\}, \quad N = \{S\}$$

$$P = \{S \to ab; S \to aSb\}$$

4.2. feladat. *Határozzuk meg, hogy az $\mathcal{L}(G)$ nyelvben benne van-e:* 01, 111, 1111*? Írjuk le szavakkal milyen szavakat tartalmaz! Soroljuk be a Chomsky-féle hierarchiába a nyelvtant!*

$$\Sigma = \{0,1\}, \quad N = \{S,A\}$$

$$P = \{S \to 1; S \to 1A; S \to 0S; A \to 1S; A \to 0A\}$$

4.3. feladat. *Legyen adott az alábbi nyelv:*

$$G = (\Sigma = \{0,1\}, \quad N = \{S,A\}, \quad P = \{S \to 1; S \to 1A; S \to 0S; A \to 1S; A \to 0A\})$$

Ha lehetséges, hozzunk létre olyan M véges automatát, melyre

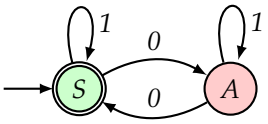
$$\mathcal{L}(G) = \mathcal{L}(M),$$

azaz a nyelvtan által generált nyelv és az automata által felismert nyelv megegyezik.

4.4. feladat. *Adjunk meg olyan reguláris nyelvtant, amely pontosan azokat az 0 és 1 szimbólumokból álló szavakat generálja:*

1. *melyek két 0-ra kezdődnek;*
2. *melyekben pontosan egy 0 van.*

4.5. feladat. *Felismeri-e az alábbi automata a csupa egyesekből álló szavakat? Adjunk meg három olyan szót, amit az alábbi véges determinisztikus automata felismer! Adjuk meg, milyen szavakat ismer fel (szóban vagy halmazjelöléssel)!*



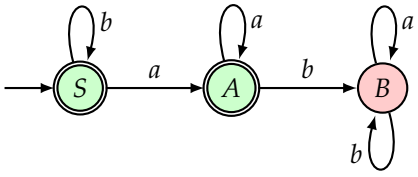
4.6. feladat. *Adjunk meg DFA-t, amely az alábbi nyelvet ismeri fel, illetve reguláris nyelvtant, amely ezt generálja:*

$$\mathcal{L} = \{x01 \mid x \in \{0,1\}^*\}$$

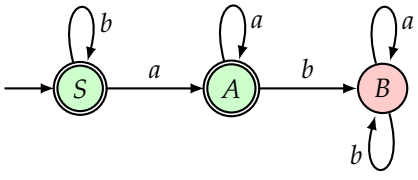
4.7. feladat. *(A második típusút nem csináltunk még, az nem lesz.) Döntsük el, hogy igazak-e az alábbi állítások. Válaszunkat indokoljuk! (Nemleges válasz esetén általában ellenpéldával indokolhatunk.)*

1. *Egy környezetfüggetlen nyelvhez létezhet olyan általános nyelvtan, amely azt állítja elő.*
2. *A szavak konkatenációja egységelemes félcsoport.*

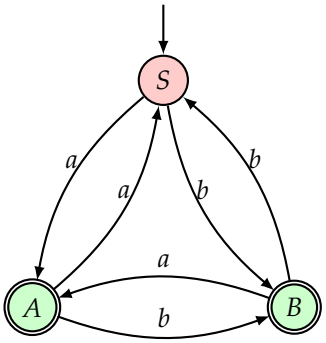
4.8. feladat. *Adjuk meg halmazleírással vagy írjuk le szavakkal milyen nyelvet ad meg az alábbi ábrán látható véges determinisztikus automata.*



4.9. feladat. *Hozzunk létre (lehetőleg reguláris) nyelvtant ahhoz a nyelvhez, melyet az alábbi véges determinisztikus automata ismer fel.*



4.10. feladat. *(Nehezebb, ilyen nehéz nem lesz.) Adjuk meg halmazleírással vagy írjuk le szavakkal milyen nyelvet ad meg az alábbi ábrán látható véges determinisztikus automata.*



4.11. feladat. *Adjunk meg nyelvtant, amely ezt generálja:*

$$\mathcal{L} = \{ww^{-1} \mid w \in \{0,1\}^*\}$$

4.12. feladat. *Vajon van-e olyan DFA, amely az előző feladatban szereplő nyelvet ismeri fel?*

5. Tudnivalók és kiegészítések a környezetfüggetlen nyelvekhez

A Bach-könyvben egy külön fejezet foglalkozik velük.
Környezetfüggetlen nyelv.
A Backus–Naur jelölés.

5.1. Backus–Naur jelölés

A programozási nyelvek magadásánál a nemterminálisokat és a terminálisokat nem egyetlen betűvel szokták jelölni, mivel átláthatatlan lenne melyik betű mit jelöl. A nyelvtan megadására a Backus–Naur jelölést szokták használni. Ennek több változata elterjedt, először egy változatával fogunk megismerkedni, majd az eltéréseket is megbeszéljük.

A nyelvtan megadásának ebben a formájában a nyíl helyett a `::=` szimbólumot használjuk, a terminális kifejezéseket idézőjelek határolják a nemterminális szimbólumok azok, amelyeket nem idézőjelek határolnak, és egybe írottak (az esetleges szóközt aláhúzással helyettesítik).

Kiegészítő anyag. Az alábbi túlmegy a törzsanyagon, és nem fogom kérdezni, de remélem azért lesz aki elolvassa és hasznosnak találja.

`http://docs.python.org/library/string.html#formatstrings` Az itt találhatóak szerint a Pythonban egy érték kiírási formája a következő lehet egy sztring objektum format metódusa esetén:

```
format_spec ::=  [[fill]align][sign][#][0][width][,][.precision][type]
fill        ::=  <a character other than '}'>
align       ::=  "<" | ">" | "=" | "^"
sign        ::=  "+" | "-" | " "
width       ::=  integer
precision   ::=  integer
type        ::=  "b" | "c" | "d" | "e" | "E" | "f" | "F"
               | "g" | "G" | "n" | "o" | "s" | "x" | "X" | "%"
```

Az első sorban látjuk, hogy minden nemterminális (például sign, type) szögletes zárójelben (`[]`) található, ami azt jelenti, hogy elhagyhatóak. Ez a jelölésmód nem csak nyelvek leírásában szerepel, hanem Unix és Linux rendszerek esetén is így jelölik az elhagyható argumentumokat a kézikönyvek SYNOPSIS sorában. Csak írjuk be a `man ls` parancsot. A manból q-val lehet kilépni.

Ennek megfelel például a `"^ 8f"` típusjelölés amely egy számot lebegőpontos (type="f") alakban ír ki, középre rendezve (align="^"), ha nyolc karakter szélességű helyre (width="8") két tizedesjegy pontossággal (precision="2") úgy, hogy az előjelnek pozitív számoknál kihagy egy helyet (sign=" ", szóköz) és a 8-as szélességből megmaradó helyet + karakterrel tölti ki (fill="+")

A következő sorban egy Python kifejezés és az értéke látható:

```
"{0:+^ 8.2f}".format(1.2)
"+ 1.20++"
```

5.1. példa. *Írjuk tömörebb formában az alábbi Backus–Naur jelöléssel megadott nyelvtant, majd alakítsuk át az formális nyelvek elméletében használt tömörebb jelölésre. Az* `értékadó_kifejezés`*-t tekintsük kezdőszimbólumnak.*

```
értékadó_kifejezés ::= változó értékadó_operátor értékadó_kifejezés
értékadó_kifejezés ::= változó
változó ::= "a"
változó ::= "b"
```

```
értékadó_operátor ::= "="
értékadó_operátor ::= "+="
értékadó_operátor ::= "-="
értékadó_operátor ::= "*="
értékadó_operátor ::= "/="
```

Megoldás: *A rövid forma:*

```
értékadó_kifejezés ::= változó értékadó_operátor értékadó_kifejezés
                     | változó
változó ::= "a"
          | "b"
értékadó_operátor ::= "+="
                    | "-="
                    | "*="
                    | "/="
```

Az `értékadó_kifejezés`*-t jelöljük S-sel, a többi nemterminálist az ABC első nagybetűivel, a terminálisokat pedig az első kisbetűivel.*

eredeti	*új*
értékadó_kifejezés	*S*
változó	*A*
értékadó_operátor	*B*
"a"	*a*
"b"	*b*
"="	*c*
"="*	*d*
"/="	*e*
"+="	*f*
"− ="	*g*

Így a fenti szabályok a következővé alakulnak:

$$P = \{S \to ABS, S \to A, A \to a \mid b, B \to c \mid d \mid e \mid f \mid g\}$$

Itt felhasználtuk, hogy az $A \to a$ és $A \to b$ szabályt rövidíthetjük a $A \to a|b$ formában. Az utolsó szabálycsoport is 5 szabályt foglal magában.

Az 1.2. példában megnéztük, hogyan lehet levezetni belőle az alábbi C nyelvben érvényes értékadó kifejezést.

```
a = b *= a
```

5.1.1. Más változatok

A Backus–Naur jelölésnek, mint jeleztük, ettől eltérő formái is léteznek. Elterjedt még, hogy nem a terminális szimbólumokat teszik idézőjelbe, hanem a nem terminális szimbólumokat teszik „kacsacsőrös" zárójelek közé: <`értékadó-operátor`>. Vagy a kacsacsőrt és az idézőjeleket is használják. Ilyen szerepel a könyv 1.3 fejezetében (Miért grammatika a grammatika – Egy illusztratív példa) vagy a C nyelvtanának leírásában az alábbi oldalon

`https://cs.wmich.edu/~gupta/teaching/cs4850/sumII06/The syntax of C in Backus-Naur form.htm`

(rövidebben: `https://goo.gl/d71mbi`).

Előfordul, hogy a `::=` helyett egyszerűen csak kettőspontot használnak.

5.2. Levezetési fa

Bal- és jobboldali levezetés. Levezetési fa. Szintaktikai egység. Egyértelmű, többértelmű nyelvtan.

Egyértelmű, többértelmű nyelv. Van-e algoritmus az eldöntésére?

Nyelvtanok ekvivalenciája. Van-e algoritmus két reguláris illetve két környezetfüggetlen nyelvtan ekvivalenciájának megállapítására? Ha van, hogyan lehet?

Ne keverjük a nyelvet és a nyelvtant!

A Bach-könyv bizonyítja, hogy az első nyelvtan (amiben E, T, F van) egyértelmű, a bizonyítás nem kell. Ezt a nyelvtant érdemes megjegyezni.

A Bach-könyvben példa van olyan nyelvre, ami nem egyértelmű, a példát ismerni kell, és tudni milyen szavaihoz nincs egyértelmű levezetés semelyik hozzá tartozó CF nyelvtanban.

Reguláris nyelv lehet-e többértelmű. Milyen alakú a levezetési fája?

A könyvben szereplő nyelvtan levezetési szabályai:

$$E \to E + T \mid T, \qquad T \to T * F \mid F, \qquad F \to (E) \mid a \tag{1}$$

Itt kivételesen nem S-sel jelöltük a kezdőszimbólumot, hanem az angol kifejezése kezdőbetűjével, E-vel.

Ezt a nyelvtant a Backus-Naur jelöléssel így írhatnánk.

```
expression ::= expression "+" term
expression ::= term
term ::= term "*" factor
term ::= factor
factor ::= "(" expression ")"
factor ::= "a"
```

Vagy tömörebben:

```
expression ::= expression "+" term
             | term

term ::= term "*" factor
       | factor

factor ::= "(" expression ")"
         | "a"
```

Ez egyes elemek jelentése magyarul: expression=kifejezés, term=tag, factor=szorzótényező.

5.2.1. A kifejezés nyelvének szűrése

A nyelvtan a következő levezetési szabályokat tartalmazza:

$$E \to E+T|T; \quad T \to T*F|F; \quad F \to (E)|a$$

Az E (expression) a kezdőszimbólum, és a T (term) és F (factor) is nemterminálisok, a többi terminális.

A nyelvtant könnyen kiegészíthetnénk úgy, hogy a kivonás és osztás műveletét is tartalmazza, de a további vizsgálataink ez csak bonyolítaná.

Ebben a nyelvben nincs felesleges szimbólum. Erről meggyőződhetünk, ha a kétféle szűrést elvégezzük. Kötelezően alulról kell kezdeni a szűrést (bottom up), a terminális szimbólumoktól:

$$\begin{aligned} B_0 &= \{+,*,(,),a\} \\ B_1 &= \{+,*,(,),a,F\} \\ B_2 &= \{+,*,(,),a,F,T\} \\ B_3 &= \{+,*,(,),a,F,T,E\} \end{aligned}$$

Mivel minden szimbólum szerepel a halmazban, tehát minden nemterminálisból levezethető valamilyen terminálisok sorozata. folytassuk a felülről lefelé (top down) szűréssel. Itt a kezdőszimbólummal kezdünk, amely jelen esetben az E.

$$\begin{aligned} T_0 &= \{E\} \\ T_1 &= \{E,+,T\} \\ T_2 &= \{E,+,T,*,F\} \\ T_3 &= \{E,+,T,*,F,(,),a\} \end{aligned}$$

Ezzel megmutattuk, hogy minden szimbólum esetén van olyan szó, amely levezethető a kezdőszimbólumból, és az adott szimbólumot tartalmazza.

5.2.2. Reguláris nyelv levezetési fája

A reguláris nyelvekben kétféle szabály lehet, az utolsó lépést kivéve az $A \to aB$ alakút használjuk, ekkor mindig megmarad egy darab nemterminális, amely a szó legvégén található. A levezetés végén egy $A \to a$ alakú szabály használatával tudjuk a nemterminálist eltüntetni.

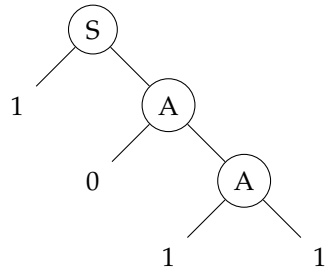
Egy korábbi feladatban szerepelt az alábbi levezetési szabályokkal megadott nyelvtan. (A nagybetűk a nemterminálisok.)

$$P = \{S \to 1; S \to 1A; S \to 0S; A \to 1S; A \to 0A\}$$

Ebben érvényes az alábbi levezetés.

$$S \Rightarrow 1A \Rightarrow 10A \Rightarrow 101S \Rightarrow 1011$$

A levezetés minden lépésében a jobboldalon szereplő nemterminálist bontjuk tovább, a levezetési fa egy jobbra lefelé növő fa lesz.



Ebben a nyelvtanban minden egyes szó levezetése egyetlen sorrendben lehetséges, ami biztosítja azt, hogy egyetlen fa tartozzon a levezetéshez. Ez a nyelvtan és így az általa meghatározott nyelv is egyértelmű tehát. Általában bizonyítható, hogy minden reguláris nyelvhez található egyértelmű nyelvtan, azaz *a reguláris nyelvek egyértelműek*.

5.2.3. Eldöntő algoritmusok

5.1. definíció. Két nyelvtant ekvivalensnek *nevezünk, ha ugyanazt a nyelvet generálják.*

Van-e olyan algoritmus, amellyel eldönthető,

1. hogy két reguláris nyelvtan ekvivalens-e?

   Igen. Mindegyikhez elkészítem a véges automatát, ha nem determinisztikus, akkor azzá alakítom, elkészítem a minimálautomatát. Ha a két nyelvtanból képzett minimálautomata megegyezik az állapotok jelölésétől eltekintve, akkor a két nyelv azonos nyelvet generál.

   Az állapotok esetén lehetséges, hogy amit az egyik esetén q_1-gyel jelöltünk, a másik esetben q_2-vel; ettől még a két automata lényegében megegyezik.

2. hogy két környezetfüggetlen nyelvtan ekvivalens-e?

   Nem. Nemcsak hogy eddig nem tudtunk létrehozni ilyet, de bizonyítható, hogy nem hozható létre ilyen algoritmus.

3. hogy egy környezetfüggetlen nyelvtan egyértelmű-e?

   Nem. Itt is bizonyítható, hogy nem hozható létre olyan algoritmus, amellyel ez eldönthető.

5.3. Környezetfüggetlen nyelvtanok átalakítása

Mikor felesleges egy terminális ill. nemterminális szimbólum?
Alulról felfelé és felülről lefelé átfésülés. Melyiknél milyen halmazzal kezdünk? Lényeges-e a sorrend?
A Bach-könyv nyomtatott (könyvként 2001-ben kiadott) változatában a nyelvtani szabályok egyikét elírták, a következő szabályokkal igaz, hogy a fordított sorrendű szűrés során a $C \to b$ szabály feleslegesen bent marad, csak az a jó, ha először szűrök alulról (a terminálisoktól) és utána felülről (a kezdőszimbólumtól):

$$S \to a, \quad S \to B, \quad B \to BC, \quad C \to b$$

Ki kell tudni szűrni felesleges szimbólumokat egy nyelvtanból.
Rekurzív nemterminális.
Álrekurzivitás, programozási példa.
Mikor rekurzív egy nyelv? Hasznos-e vagy káros a rekurzivitás?

5.2. definíció. *Egy nyelvtanban egy A nemterminálist rekurzívnak nevezünk, ha belőle kiindulva levezethető olyan szó, amelyben szintén szerepel A.*

$$A \Rightarrow^* \alpha A \beta, \qquad \alpha, \beta \in (\Sigma \cup N)^*$$

A fenti definicióban $\Rightarrow^*$ azt jelöli, hogy a jobboldal egy vagy több lépésben levezethető a baloldalból. (Logikusabb lenne erre a $\Rightarrow^+$ jelölés, de nem írjuk felül a szokásos jelölést.)

5.2. példa. *Adjunk példát egy programbeli álrekurzióra és egy nyelvtanbelire.*
Megoldás: *A Python nyelvben a* `def` *a függvénydefiniálás utasítása, a zárójelezés helyett a sorok behúzásával tagolja a kódot. Mi a hiba az alábbi rekurzív függvényben?*

```
def factorial(n):
    return n*factorial(n-1)
```

A függvény nem tud befejeződni, mert mindig újból önmagát hívja meg. Kell tenni bele egérutat, amikor befejeződik, és nem újból önmagát hívja:

```
def factorial(n):
    if n == 0:
        return 1
    return n*factorial(n-1)
```

Ekkor, legalábbis pozitív egészekre, helyesen működik. Az első változat álrekurzív, a második valódi rekurzív függvény.

A fentihez hasonlóan működik a $B \to bB$ szabály, ha a B-től sohasem tudok megszabadulni. Az alábbi esetben C-nél ott ez egérút, B-től viszont nem lehet megszabadulni.

$$P = \{S \to B, B \to bB, S \to C, C \to cC, C \to c\}$$

Ha hozzáadnék egy $B \to b$ szabályt, akkor igazi rekurzió lenne a $B \to bB$ és nem lenne benne felesleges szimbólum.

Figyelem! A fenti Pythonos példát csak hasonlatnak szántuk. Az, hogy egy nyelv tartalmazhat-e rekurziót, és az, hogy a nyelvet leíró nyelvtan tartalmaz-e rekurziót, egymástól független dolgok. Majdnem minden valamirevaló programnyelv nyelvtanában van rekurzió, másképpen csak véges sokféle programot tudnánk létrehozni. A nyelvbeli rekurzió – hogy egy függvény hívhatja-e önmagát – az inkább attól függ, hogy a fordító képes-e megküzdeni a rekurzív függvényhívással.

Mikor nem küszöbölhető ki mindegyik ε-szabály? Milyen formában engedjük meg az ε-szabályt olyan nyelv nyelvtanában, amely üres szót (ε) tartalmaz? Az ε-szabályok kiszűrési algoritmusa nem kell.

5.1. tétel. *Egy nyelvtanban minden ε-szabálytól megszabadulhatunk, ha az általa generált nyelvben nincsen benne az üres szó, azaz a kezdőszimbólum nem enyészhet el.*

Ellenkező esetben a nyelvtant átalakíthatjuk úgy, hogy egyetlen ε-szabályt tartalmazzon, az $S \to \varepsilon$ szabályt, és az S kezdőszimbólum nem szerepel sehol sem a levezetési szabályok jobboldalán.

Kiküszöbölhetőek-e az egyszeres szabályok? Milyen esetben érdemes kiszűrni? A kiszűrés algoritmusa nem kell.

Mit jelent a jólfésült nyelvtan (proper grammar)? Minden nyelvtan átalakítható-e ilyenné?

5.2. tétel. *Minden környezetfüggetlen nyelvtannak van jólfésült egyenértékese.*

A felesleges szimbólumok kiszűrése mindenképpen hasznos, de az ε-szabályok és az egyszeres szabályok kiszűrése ronthat az áttekinthetőségen.

5.1. feladat. *Szűrjük ki az alábbi nyelvtanból a felesleges szimbólumokat! Határozzuk meg a nyelvtan által generált nyelvet!*

$$P = \{S \to AC, A \to aA, C \to c, S \to cC, C \to aB, B \to bE, F \to b\}$$

5.4. Környezetfüggetlen nyelvtanok normálalakjai

Milyen alakú átalakításokat tartalmazhat a Chomsky-féle normálalak (CNF)? Hogyan alakíthatjuk ilyenné a nyelvtant?

Milyen alakú átalakításokat tartalmazhat a Greibach-féle normálalak (GNF)?

Minden környezetfüggetlen nyelv átalakítható-e CNF illetve GNF formára?

Benne lehet-e az $S \to \varepsilon$ szabály a normálalakokban?

Gyakran a normálalakok kevésbé áttekinthetőek, mint az eredeti nyelvtan, de gyakran az áttekinthető nyelvtan alapján nehezebb szintaktikai elemzőprogramot készíteni, mint a normálalakokból. Mivel a fordítóprogramokat a számítógép futtatja, ezért az elemezhetőség fontosabb, mint hogy az ember számára áttekinthető-e.

Mit jelent, hogy egy szimbólum balrekurzív?

5.3. definíció. *Egy nyelvtan A* nemterminálisát balrekurzívnak *nevezünk, ha egy vagy több lépésben olyan kivejezést tudunk levezetni belőle, amelynek a baloldalán található az A nemterminális.*

$$A \Rightarrow^* A\beta, \qquad \beta \in (\Sigma \cup N)^*$$

Ha egy környezetfüggetlen nyelv rekurzív, meg lehet-e szüntetni a rekurzivitását?

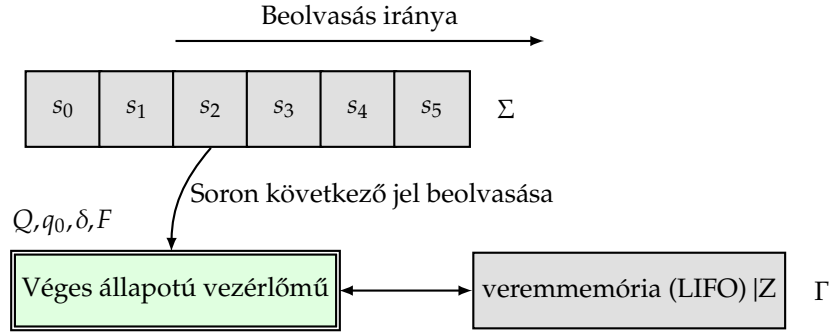
Ha egy nyelv balrekurzív, meg lehet-e szüntetni a balrekurzivitását?

Miért kell a balrekurzivitást megszüntetni?

Lehet-e a Chomsky-féle normálalak balrekurzív? Indokoljuk!

Lehet-e a Greibach-féle normálalak balrekurzív? Indokoljuk!

5.5. Veremautomaták



Σ szalagábécé	q_0 kezdőállapot ($q_0 \in Q$)
Γ veremábécé	Q állapotok
Z veremalja jel ($Z \in \Gamma$)	F végállapotok ($F \subset Q$)
δ mozgási szabályok	

A veremautomata kezdetben a kezdőállapotban található, és az „olvasófej" a szalag elején található, ugyanúgy, mint a véges automatánál. A lényeges különbség abban van, hogy egy verem csatlakozik a vezérlőműhöz, így a következő állapot attól is függhet, hogy mi van a verem tetején. A veremautomata működésekor mi feltételezzük, hogy minden lépésben csak a verem legtetejének a tartalmát tudjuk megvizsgálni, és esetleg nem is vagyunk kíváncsiak erre a legfelső elemre.

A veremben kezdetben sem üres: egy veremalja jel található benne, amit Z-vel fogunk jelölni (a Bach-könyv Z_0-lal). Ez hasznos lesz majd, ha meg kell vizsgálnunk, hogy leszedtük-e az összes szimbólumot, amit a veremre raktunk. Azokat a szimbólumokat, amelyek a veremre kerülhetnek, veremábécének nevezzük.

A következő állapot a veremautomatánál már nem csak az eredeti állapottól és a szalagról olvasott jeltől függ**het**, hanem a verem felső elemétől is függ**het**. A mozgási szabály „normál esetben" valahogy így néz ki a veremautomatánál:

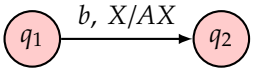
$$\delta(q_1, b, X) = (q_2, AX)$$

Ez azt jelenti, hogyha q_1 állapotban van a veremautomata, a szalagon a következő szimbólum a b, és a verem tetején X található, akkor átmehetünk a q_2 állapotba, miközben AX-et írunk a veremre.

A továbbiakban, amikor konfigurációsorozattal fogjuk levezetni egy szó felismerését, a verem tartalmát egymás mellé fogjuk írni. A verem-tartalom értelmezéséhez szükség van egy megállapodásra: mi úgy fogjuk leírni a veremtartalmat, hogy a végén szereplő szimbólum van a verem alján, az elején szereplő a tetején, ahogy a veremautomata felépítéséről szóló ábrán is látható.

Tehát a fenti szabály szerint a verem tetején található szimbólum vizsgálatához a verem tetején levő szimbólumot be kell olvasni (más tantárgyban talán találkoztak a POP utasítással), így a fenti mozgási szabály alkalmazásakor a verem tetejéről eltűnik az X, viszont helyettel AX-et írok. Tehát összességében a verem tartalma annyiban változott a két veremművelet után, hogy az eredetileg ott lévő X „fölé" egy A került.

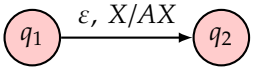
A mozgási szabályokat majd ritkán fogjuk a fenti formában írni, leggyakrabban állapotgráfon ábrázoljuk. A fenti átmeneti szabályt a következő formában ábrázoljuk.



A korábbi szövegben két helyen kiemeltük a feltételes módot. A mozgási szabályok esetén megengedjük, hogy egy mozgás során ne kelljen mindenképpen vizsgálni a szó következő szimbólumát. Ha nem vizsgáljuk, akkor az olvasófej sem lép tovább a következő szimbólumra. Ilyenkor a mozgási szabályban a szimbólum helyére az üres szó jelét, ε-t írjuk. Például a

$$\delta(q_1, \varepsilon, X) = (q_2, AX)$$

mozgási szabály esetén, ha q_1-ben vagyok, és X van a verem tetején, akkor a szalagon következő betűtől függetlenül tovább mehetek a következő állapotba. Ezt a fentihez hasonlóan ábrázoljuk.



A másik feltételes mód arra utal, hogy egy mozgási szabályban elmaradhat a verem tetejének vizsgálata is. Ilyenkor nem is vesszük le a verem tetejéről az ott található szimbólumot. Az alábbi első példában, az, hogy a mozgás létrejöhet-e – a pillanatnyi állapoton kívül – csak a szalagon következő szimbólumtól fog függeni, az másodikban pedig csak a pillanatnyi állapottól.

$$\begin{aligned} \delta(q_2, b, \varepsilon) &= (q_2, B) \\ \delta(q_1, \varepsilon, \varepsilon) &= (q_1, AB) \end{aligned}$$

Fontos észrevenni, hogy a fenti szabályok esetén a ε jel nem eleme sem a veremábécének, sem a szalagábécének. A ε nem egy szimbólumot jelöl, hanem egy szót, amelynek történetesen nulla a hosszúsága.

A mozgási szabályok tehát a függvények szokásos jelölésével leírva következő típusúak:

$$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \longrightarrow Q \times \Gamma^*.$$

A függvényérékként fellépő pár második tagja akárhány (nulla vagy több) veremábécébeli szimbólumból állhat. Az alábbi szabályt alkalmazva például a mozgáskor a veremről törlődik egy X.

$$\delta(q_2, b, X) = (q_0, \varepsilon)$$

A mozgási szabályok tisztázása után egy nagyon fontos lépés van hátra: annak a tisztázása, hogy mikor mondjuk, hogy egy szalagra írt szót felismert a veremautomata. A veremautomatának *kétféle működési módja* van, azaz tulajdonképpen a veremautomatának kétféle definíciója, amelyekben ez a feltétel különböző.

1. A *végállapottal felismerő* veremautomata (a véges automatákhoz hasonlóan) akkor ismeri fel a szót, ha a szó végigolvasásakor végállapotba kerül, vagy további lépésekkel végállapotba tud jutni.

2. Az *üres veremmel felismerő* veremautomata akkor ismeri fel a szót, ha a szó végigolvasásakor üres lesz a verem, vagy további lépésekkel ki tudja üríteni a vermet.

Az üres veremmel felismerő veremautomata definiásához nem is szükséges végállapotokat megadni. A végállapottal felismerő veremautomatát tehát a

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

hatos határozza meg, az üres veremmel felismerő veremautomatát pedig az

$$M(Q, q_0, \Sigma, \Gamma, \delta)$$

Azt, hogy az egyik automata sem nagyobb tudású, az alábbi tétel mondja ki.

5.3. tétel. *Egy nyelvhez akkor és csakis akkor létezik olyan veremautomata amely azt végállapottal ismeri fel, ha létezik olyan veremautomata amely üres veremmel ismeri fel.*

Ez a tétel teszi lehetővé, hogy a következő tételben (ami a környezetfüggetlen nyelvek és a veremautomaták közötti hasonló kapcsolatról szól, mint ami a reguláris nyelvek és a véges automaták között van) ne kelljen pontosítani, hogy a veremautomata melyik működési módjáról beszélünk.

5.4. tétel. *Egy $\mathcal{L}$ nyelvet pontosan akkor ismer fel (legalább) egy veremautomata, ha $\mathcal{L}$ környezetfüggetlen nyelv.*

A fenti két tételt mi nem bizonyítjuk. Ha valakit érdekel, a Bach-könyvben megtalálja.

Korábban volt szó róla, hogy a véges automaták működését konfigurációsorozattal lehet leírni. A konfiguráció tartalmazza az összes tudást, ami ahhoz kell, hogy az automata további működése meghatározható legyen. A véges automatánál elegendő volt a pillanatnyi állapot és a szalagon levő szó még nem olvasott részének ismerete. A veremautomatánál egy harmadik dolgot, a verem teljes tartalmát is ismerni kell. A konfigurációsorozattal történő levezetésre egy alábbi példában és a Bach-könyvben is találhatunk példát.

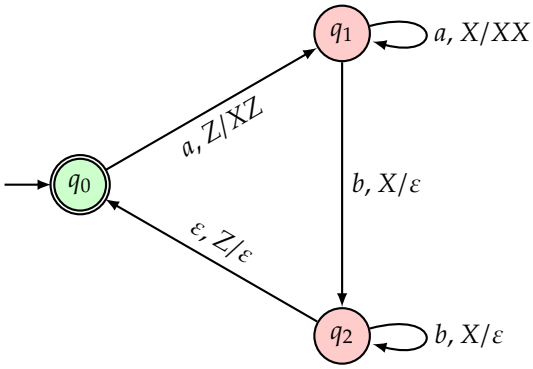
5.5.1. A kérdések, amikre tudni kell válaszolni

Mikor mondjuk, hogy egy végállapottal felismerő veremautomata felismert egy szót?
Mikor mondjuk, hogy egy üres veremmel felismerő veremautomata felismert egy szót?
Miket kell megadni az egyik illetve másik megadásához?
Bővebb-e az egyik által felismert nyelvek halmaza, mint a másik által felismerté?
Több nyelvet lehet-e felismertetni, ha nem csak a legfelső veremértéket tudja olvasni az automata?
Melyik milyen nyelvosztályt ismer fel? (Igazolni nem kell.)
Ismerni kell az általunk órán használt gráfos megjelenítést. A δ mozgási szabályokkal megadott alakot át kell tudni alakítani gráffá, és viszont.
Egy szó felismerését konfigurációsorozattal meg kell tudni adni. (Bach-könyv elektronikus változat 107. oldal.)
A verem alja szimbólumnak én a továbbiakban a Z-t használom Z_0 helyett, igazodva a JFLAP-hez.

5.3. példa. *Alakítsuk át az alábbi mozgási szabályokat + végállapotokat állapotgráffá!*

$$\begin{aligned}
&\delta(q_0, a, Z) = (q_1, XZ) \\
&\delta(q_1, a, X) = (q_1, XX) \qquad\qquad \delta(q_1, b, X) = (q_2, \varepsilon) \\
&\delta(q_2, b, X) = (q_2, \varepsilon) \qquad\qquad \delta(q_2, \varepsilon, Z) = (q_0, \varepsilon) \\
&F = \{q_0\}
\end{aligned}$$

Vezessük le konfigurációsorozattal az aabbb, aab, ab szavakat, ha lehet. Ha nem lehet, vezessük le addig, amíg el nem akad a levezetés.
Határozzuk meg, hogy determinisztikus-e az automata!
Milyen nyelvet fog felismerni végállapottal?
Megoldás:



$$(q_0, aabbb, Z) \to (q_1, abbb, XZ) \to (q_1, bbb, XXZ) \to$$
$$(q_2, bb, XZ) \to (q_2, b, Z) \to (q_0, b, \varepsilon) \not\to$$

Nem tudjuk végigolvasni az aabbb szót, mert az utolsó szituációra nincs szabály, amivel továbbmehetnénk. Az aabbb szó tehát nem szava a fenti veremautomata által felismert nyelvnek.

$$aabbb \notin \mathcal{L}(M)$$

$$(aab, q_0, Z) \to (ab, q_1, XZ) \to (b, q_1, XXZ) \to (\varepsilon, q_2, XZ)$$

Sikerült végigolvasni a szót, de nem végállapotba jutottam, és nem is tudok átmenni a végállapotba, mert ahhoz a verem tetején Z-nek kellene lennie. Tehát aab sem eleme az automata által végállapottal felismert nyelvnek.

$$(q_0, ab, Z) \to (q_1, b, XZ) \to (q_2, \varepsilon, Z) \to (\varepsilon, q_0, \varepsilon)$$

Sikerült végigolvasni a szót, és végigolvasás után lehetőség van arra, hogy a végállapotba átmenjünk. Tehát az automata elfogadta az ab szót végállapotbal (mellesleg üres veremmel is).

$$ab \in \mathcal{L}(M)$$

Az automata determinisztikus. Bár a végén ε hatására megy át q_0-ba, de q_2-ből arra a helyzetre, ha a verem tetején Z van, csak ez az egyetlen átmenet lehetséges. A véges automatáknál egy ε-szabály mindig nemdeterminizmust okoz, itt a verem tetejét is figyelni kell. A többi állapotnál pedig a kimenő nyilakon szereplő szalagábécé-beli szimbólumok különböznek.

A fenti automata végállapotban marad, ha üres szavunk (ε) volt a szó. Máskülönben csak akkor tud elindulni, ha a-val kezdődik. Ilyenkor minden a-nál rak a veremre egy X-et. Ha b jön ezután, akkor átmegy a q_2 állapotba és onnantól b szimbólumok hatására leszedegeti az X-eket. Csak akkor tud a végén a q_0 végállapotba átmenni, ha az összes X-et leszedte. Mivel ilyenkor a veramalja jel eltűnik, a veremautomata működése nem tud folytatódni. Tehát az automata az alábbi nyelvet fogadja el:

$$G = \{a^i b^i \mid i \geq 0\} = \{\varepsilon, ab, aabb, aaabbb, \ldots\}$$

Vegyük észre, hogyha a q_2-ből q_0-ba menő nyílon visszaraknám a Z-t a verembe, akkor akár többször is „körbe lehetne menni" az automatán, felismerné végállapottal az abaabbaaabbb aabbaabb abab abababab szavakat is.

5.6. Feladatok a környezetfüggetlen nyelvekhez és veremautomatákhoz

5.2. feladat. *Először hagyja el azt a 0 vagy több szabályt, amelyek nem férnek bele egy környezetfüggetlen nyelvtanba! (A tanult szűrések csak környezetfüggetlen nyelvtanokra működnek.)*

$$S \to AB;\ C \to aE;\ A \to aA;\ B \to b;\ A \to a;\ S \to bB;\ MiK \to ULaS;\ B \to bC;\ E \to aC$$

Szűrje ki a tanult módszerrel a felesleges szimbólumokat, és írja le a megmaradt levezetési szabályokat! (A szűrés követhető legyen a papíron.)

Szűrés után az alábbi szimbólumok bizonyultak feleslegesnek:

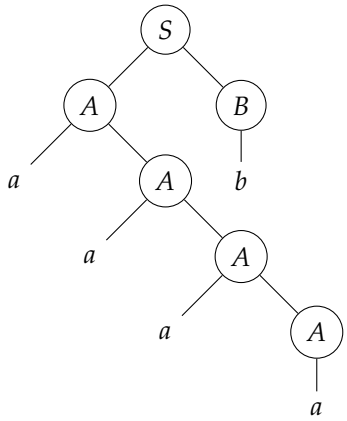
$F = \{$ $\}$.

Vezesse le lépésenként a aaaab *szót, és rajzolja le a levezetési fáját!*

Megoldás: *A Mikulásos általános, az $S \to AB$ környezetfüggetlen, a többi reguláris. Elég annyit tudni ahhoz, hogy csak környezetfüggetlenbe beleférőket tartsuk meg, hogy azok baloldalán egyetlen nemterminális (azaz szokásos jelölésünkkel nagybetű) állhat.*

$B = \{a, b, A, B, S\}$. Alulról szűrés után, és a legvégén is a következőek maradnak meg:

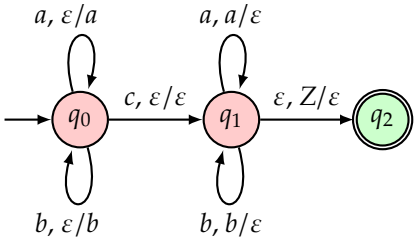
$$S \to AB;\ A \to aA;\ B \to b;\ A \to a;\ S \to bB;$$



5.3. feladat. *Határozzuk meg, hogy az alábbi automata felismeri-e üres veremmel az* `abbcbba` *illetve* `abcab` *szavakat! Ha igen vezessük le konfigurációsorozattal!*

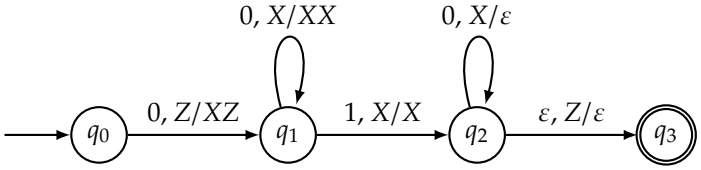
Melyik nyelvet ismeri fel?

Adjuk meg a δ mozgásszabályokat, a Q állapothalmazt, az F végállapot-halmazt, az elemzendő nyelv Σ ábécéjét, a Γ veremábécét.



(Mi megengedjük, hogy a verem tetejének olvasása nélkül mehessen tovább új állapotba. Bach Iván könyvében lévő veremautomata-változat minden lépésben kiolvassa a verem tartalmát, legfeljebb visszaírja azt. Az alábbi példában is ilyen szerepel. Ez az eltérés ugyanúgy nem befojásolja a veremautomatával felismerhető nyelvek halmazát, mint ahogy az sem, hogy üres veremmel vagy végállapottal felismerő automatát használok-e. Érdemes lehet kitalálni, hogyan lehet csökkenteni eggyel a szükséges állapotok számát, és jelentősen a szükséges mozgásszabályok számát, ha kihasználjuk, hogy nem muszáj kiolvasni a veremtartalmat.)

5.4. feladat. *Az alábbi ábrán egy veremautomata állapotdiagrammja látható.*



Vezessük le konfigurációsorozattal (amíg lehet), hogy a 00101 szót végállapottal felismeri-e! Indokoljuk, miért vagy miért nem!

Az automata állapothalmaza Q = _____ ,

a végállapot-halmaza F = _____ ,

az elemzendő nyelv ábécéje Σ = _____ ,

a veremábécéje Γ = _____ . *Két (teszőlegesen választott) δ mozgásszabálya*

_____ .

Határozzuk meg, hogy az alábbi automata felismeri-e végállapottal az ε, 0001000, 10 illetve 01 szavakat!

Melyik nyelvet ismeri fel?

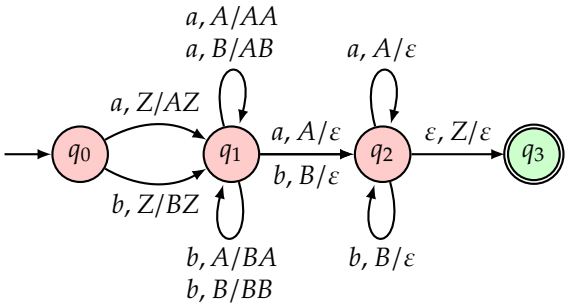
Megoldás: *A 00101, a ε, 10 és a 01 szavakat nem ismeri fel, a 0001000 szót igen.*

A $\{0^i10^i \mid i > 0\} = \{010, 00100, 0001000, \ldots\}$ nyelvet ismeri fel ez a veremautomata végállapottal és üres veremmel is.

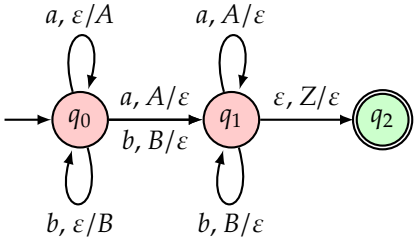
5.5. feladat. *Határozzuk meg, hogy az alábbi automata felismeri-e végállapottal az* `abba` *illetve* `aaa` *szavakat! Ha igen vezessük le konfigurációsorozattal!*

Melyik nyelvet ismeri fel?

Adjuk meg a δ mozgásszabályokat, a Q állapothalmazt, az F végállapot-halmazt, az elemzendő nyelv Σ ábécéjét, a Γ veremábécét.



Megjegyzés Ha megengedjük, hogy a fenti automatának ne kelljen mindig ellenőrizni a verem tetején lévő jelet (a per előtt lehessen ε is), akkor az alábbi egyszerűbb automatát is használhatnánk.



Érdemes észrevenni, hogy ez az automata nem determinisztikus. Érdemes kipróbálni az *aa*, *aaaa*, *abbbba* szavakra. Ahányszor két egymásutáni ugyanolyan betűhöz ér, a konfigurációsorozat elágazhat. Ami érdekes, hogy az általa felismert $\{wcw^{-1} \mid w \in \{a,b\}^*\}$ nyelvhez nem is lehet determinisztikus automatát készíteni. Ellenben a véges automatákkal tehát, nem minden nem determinisztikus veremautomatához lehet vele ekvivalens determinisztikusat találni.

5.6. feladat. *Adjon meg egy automatát, amely a szabályos zárójelezések nyelvét ismeri fel üres veremmel!*

$$\{(),(()),()(),((())),(()()),((((()))),(()()()),((())()), \dots\}$$

Figyelem! Ez nem az órán említett $\{(^i)^i \mid i > 0\}$ nyelv.

5.7. feladat. *Adjunk példát az alábbi nyelvtanból álrekurzív nemterminálisra.*
Az alábbi nyelvtanból csak azokat a szabályokat hagyjuk meg, amelyek nem tartalmaznak felesleges szimbólumokat.

$$S \to b, \quad S \to B, \quad S \to BC, \quad C \to cC, \quad B \to Bb, \quad B \to b, \quad C \to DC, \quad D \to d$$

A megmaradt nyelvtanból határozzuk meg az általa generált nyelv ábécéjét, és a generált nyelvet.

Végeredmény A C rekurzív (két szabály szerint is: $C \to cC, \quad C \to DC$), ha C egyszer belekerül a levezetésbe, akkor soha nem tudunk tőle megszabadulni, tehát álrekurzív.

Az megmaradó levezetési szabályok:

$$S \to b, \quad S \to B, \quad B \to Bb, \quad B \to b$$

(Zárthelyin és viszgán természetesen le kell írni ilyenkor a szűrés teljes folyamatát.)

A generált nyelv ábécéje $\Sigma = \{b\}$. A generált nyelv:

$$\mathcal{L} = \{b^i \mid i > 0\} = \{b, bb, bbb, bbbb, \dots\}$$

Megjegyzés Az előbbi nyelvhez egy kevesebb szabályt tartalmazó nyelvtan is készíthető:

$$S \to b, \quad S \to bS$$

5.8. feladat. *Hozzuk Chomsky-féle normálformára a következő nyelvtant! Van-e balrekurzív nemterminális az eredeti, illetve a kapott nyelvtanban? Ha igen, hol?*

$$S \to SaSb; \qquad S \to ab;$$

Megoldás: *Az első átalakítás után:*

$$S \to S\hat{A}S\hat{B}; \quad S \to \hat{A}\hat{B}; \quad \hat{A} \to a; \quad \hat{B} \to b$$

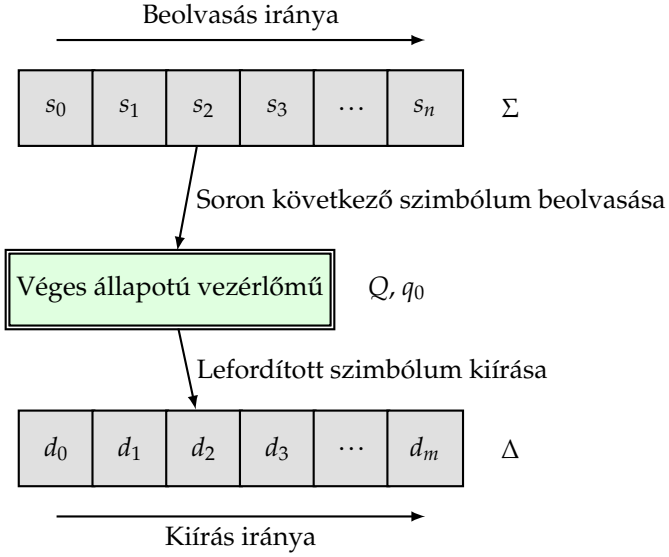
Miután az első szabályt „felszeleteltük":

$$S \to SS_1; \quad S_1 \to \hat{A}S_2; \quad S_2 \to S\hat{B}; \quad S \to \hat{A}\hat{B}; \quad \hat{A} \to a; \quad \hat{B} \to b$$

Mind az eredeti nyelvtan, mind a CNF-ben levőben az első levezetési szabályban van balrekurzív nemterminális. Méghozzá egyetlen lépésben át tudjuk alakítani az S-et tartalmazó szavakat, hogy újabb S legyen benne.

6. Tudnivalók és kiegészítések a fordító automatákhoz

Bach könyv 4. fejezetében van szó róla.
Véges fordító (Mealy-automata változat):

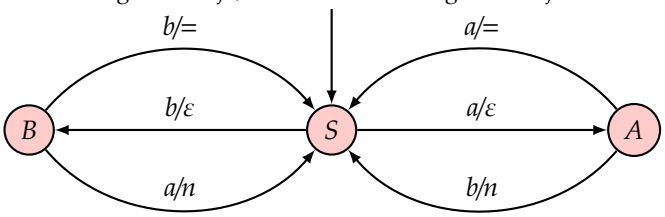


A véges fordítónak kétféle változata van. A Bach-könyv azzal foglalkozik, amikor az átmenetek során történik a szalagra írás. Ezt a szakirodalom Mealy-automatának nevezi. Lehet olyan véges fordítót is készíteni, amely akkor ír a szalagra, amikor egy új állapotba kerül, ezt a szakirodalom Moore-automatának nevezi. Belátható, hogy mindkét fordítócsalád ugyanarra képes: tehát minden Mealy-automatához van olyan Moore-automata, amelyek, ha azonos a bemenő szalagon lévő szó, akkor ugyanazt írják ki a szalagra, és fordítva; minden Moore-automatához van ilyen Mealy-automata.

Milyen változatai vannak a véges fordítóknak? Mi a lényegi különbség a működésükben?

Részletesen magyarázza el a Mealy-automata működését! Adjon meg egy példát rá állapotgráffal!

6.1. feladat. *Határozza meg, mit ad vissza az alábbi ábrán szereplő automata a* `baabbbaabaaabb` *szóra, azaz mire fordítja? Mi lesz a bemeneti szalag Σ ábécéje, és a kimeneti szalag Δ ábécéje?*



Adjon meg olyan szimbólum-sorozatot, amelynél a fenti automata esetén a kimeneti szalagon a =nn==nn== *jelenik meg. Egyértelmű-e ez a szimbólum-sorozat? Ha nem, adjon meg még egyet!*

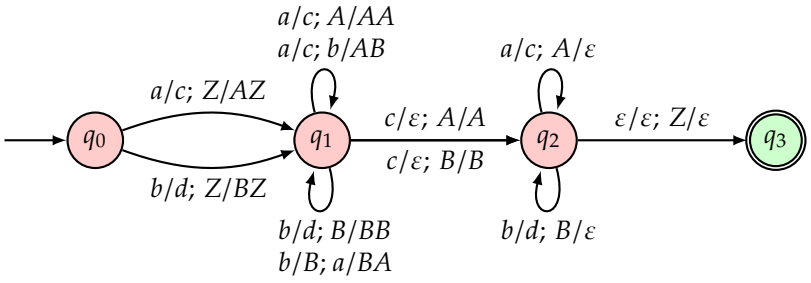
A véges fordítók egy bemeneti nyelvhez egy kimeneti nyelvet rendelnek. Azaz, ha a bemeneti nyelv minden szavára lefuttatjuk az automatát, akkor a kimeneti szalagon kapott szavak halmazát nevezzük kimeneti nyelvnek.

Mit mondhatunk a véges fordító kimeneti nyelvéről, ha a bemeneti nyelve reguláris nyelvet, illetve környezetfüggetlen nyelv?

A szintaxisvezérelt fordítási sémák 2015 tavaszán még nem kellenek.

Magyarázza el a veremfordító működését!

Alább van egy példa veremfordítóra. Az első / előtt van, hogy mit olvas a bemenő szalagról, utána, hogy mit ír ki a kimenő szalagra. A második per előtt, hogy mit olvas a veremről (egy szimbólum vagy semmi), utána, hogy mit ír rá (nulla vagy több szimbólum a veremábécéből).



7. Tudnivalók az informatikus hallgatók Formális nyelvek tantárgyához

A segédletek egy része az `http://elearning.uni-obuda.hu` oldal *Formális nyelvek* kurzusában található.

Kötelező irodalom:

- Bach Iván: *Formális nyelvek*, TypoTEX, 2001, az Interneten legálisan elérhető ingyenesen.
- Az http://elearning.uni-obuda.hu oldal segédletei

Ajánlott irodalom:

- Roger Penrose: *A császár új elméje*, Akadémiai Kiadó, 1993, különösen Turing-géppel foglalkozó rész, de a többit is ajánlom
- Demetrovics, Denev, Pavlov: *A számítástudomány matematikai alapjai*, Tankönyvkiadó, Budapest, 1989, Két fejezete: 4. A formális nyelvek és automaták, 5. A Turing-gép
- Tóth Mihály: *Bevezetés a formális nyelvek és automaták elméletébe* (Handout, 1992.)

8. Tételsor

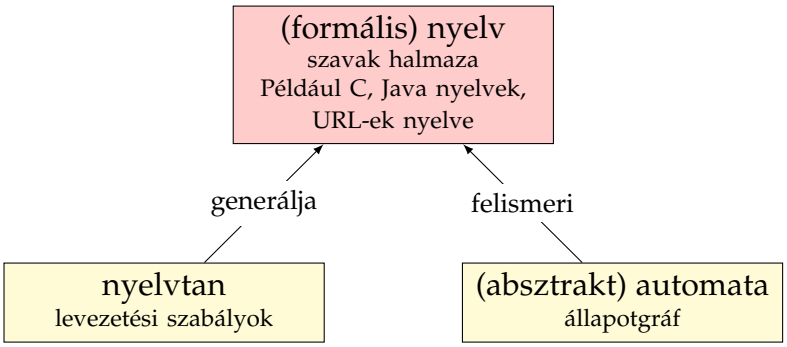
2017. november 20.

1. Nyelvek, nyelvtanok, automaták kapcsolata. Mit jelent, hogy egy nyelvtan generál egy nyelvet, ill. egy automata felismer egy nyelvet? Milyen automaták társíthatóak a két legszűkebb nyelvosztályhoz?
2. Chomsky-féle nyelv- és nyelvtan-osztályok
3. Véges automaták felépítése, működése, megadásuk állapotgráffal, és csoportosításuk (determinisztikusság, teljesség), determinisztikussá alakítható-e a nemdeterminisztikus.
4. Reguláris nyelvek és a véges automaták kapcsolata, egymásba alakítása
5. Műveletek nyelvekkel (komplemens, unio, metszet, konkatenáció, tranzitív lezárt). Milyen nyelv lesz ezek eredménye a reguláris nyelvek esetén?
6. Reguláris kifejezések: szűk nyelv, halmazok és intervallumok, ismétlések, speciális karakterek, karakterek levédése
7. Környezetfüggetlen nyelvtanok és nyelvek egyértelműsége; a két műveletet és zárójelet tartalmazó kifejezések (E, T, F nemterminálisokat tartalmazó) nyelvtana; levezetési fa; példa nem egyértelmű nyelvre
8. Programozási nyelvek nyelvtanának megadása Backus-Naur normálalakban. Honnan ismerhetőek fel ebben a terminálisok, nemterminálisok (2 változat)? Hogyan írhatóak tömörebben az azonos baloldalú szabályok?
9. Környezetfüggetlen nyelvtanok jólfésült nyelvtanná alakítása
10. Környezetfüggetlen nyelvtanok normálalakjai, Chomsky-féle normálalakra alakítás, rekurzió, balrekurzió
11. Veremautomaták felépítése és működése, megadása állapotgráffal, típusai (végállapottal ill. üres veremmel felismerő), levezetés konfigurációsorozattal.

Csak informatikusoknak:

12. (csak informatikusoknak) Milyen algoritmikusan eldönthető és eldönthetetlen problémákat ismerünk a formális nyelvekkel kapcsolatban?

9. Feladatok, nyomtatnivalók órára



9.1. Nyelvtanok és nyelvek típusai (Chomsky-hierarchia)

A nyelvek típusait a nyelvtanok típusaiból fogjuk származtatni. Először tehát ezt vizsgáljuk meg. Először egy táblázatban összefoglalom a nyelvtanok típusait, majd részletesebben kifejtem.

Legyen $\alpha, \beta, \omega \in (\Sigma \cup N)^*$ $A, B \in N$ $a \in \Sigma$

szabályok típusa	nyelvtan típusa	automata
$\alpha A \beta \to \omega$	0-típusú, általános	Turing-gép
$\alpha A \beta \to \alpha \omega \beta$	1-típusú, környezetérzékeny	
$A \to \omega$	2-típusú, környezetfüggetlen (CF)	PDA
$A \to aB$ vagy $A \to a$	3-típusú, reguláris	DFA

9.2. Reguláris nyelvtan → véges automata átalakítási szabályok

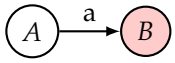
Kell kezdetben egy kezdőállapot 

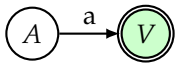
,

és egy végállapot 

ahová a $A \to a$ alakú szabályokat vezetjük.

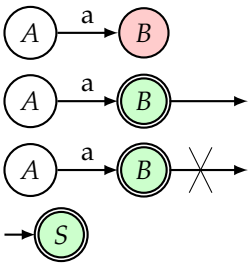
A további szabályok:

$A \to aB$ 

$A \to a$ 

$S \to \varepsilon$ 

Véges automata → reguláris nyelvtan átalakítási szabályok



$A \to aB$

$A \to aB, \quad A \to a$

$A \to a$

$S \to \varepsilon$

9.3. Műveletek nyelvekkel

Ismerni kell a Bach-könyv 2.5. fejezetében szereplő műveleteket: komplemens, unió, metszet, konkatenált, hatvány, tranzitív lezárt.

- Nyelvek konkatenációja: $\mathcal{L}_1\mathcal{L}_2 = \{xy \mid x \in \mathcal{L}_1, y \in \mathcal{L}_2\}$
- Nyelvek hatványa: $\mathcal{L}^0 = \{\varepsilon\}, \quad \mathcal{L}^1 = \mathcal{L}, \quad \mathcal{L}^2 = \mathcal{L}\mathcal{L}, \quad \mathcal{L}^n = \mathcal{L}^{n-1}\mathcal{L}$
- Nyelvek tranzitív lezártja: $\mathcal{L}^* = \mathcal{L}^0 \cup \mathcal{L}^1 \cup \mathcal{L}^2 \cup \mathcal{L}^3 \ldots$

9.4. Nyelvek, Chomsky-féle nyelv- és nyelvtanosztályok

9.1. feladat. *Soroljon fel mindegyik nyelvnél 3-3 szót, ami beletartozik a nyelvbe, illetve amely szó – bár a nyelv ábécéjéből tartalmaz csak betűket – mégsem tartozik a nyelvhez!*

$\mathcal{L}_1 = \{a^i b^i c^i \mid i \geq 0\}$

eleme: _____

nem eleme: _____

$\mathcal{L}_2 = \{x2x^{-1} \mid x \in \{0;1\}^*\}$

eleme: _____

nem eleme: _____

9.2. feladat. *Sorolja fel az alábbi nyelvek összes szavát!*

$\mathcal{L}_1 = \left\{x^3 \mid x \in \{a;ab\}\right\} = \{$ $\}$

$\mathcal{L}_2 = \{xx \mid x \in \{a;b\}^* \wedge len(x) \leq 2\} = \{$ $\}$

$\mathcal{L}_3 = \{xyxa \mid x \in \{a;b\} \wedge y \in \{ab;\varepsilon\}\} = \{$ $\}$

9.3. feladat. *Határozza meg egyesével melyik legszűkebb Chomsky-féle nyelvtanosztályba férnek bele az alábbi levezetési szabályok!*

$$^{1.}S \to AB;\ ^{2.}CEB \to CBB;\ ^{3.}A \to cE;\ ^{4.}A \to aaB;\ ^{5.}BB \to Bc;\ ^{6.}C \to cC;\ ^{7.}C \to c;$$

$$^{8.}B \to bB;\ ^{9.}B \to b;\ ^{10.}S \to E;\ ^{11.}E \to eE;\ ^{12.}cEB \to cEe;\ ^{13.}CeB \to ceB;\ ^{14.}aBC \to aaaBC$$

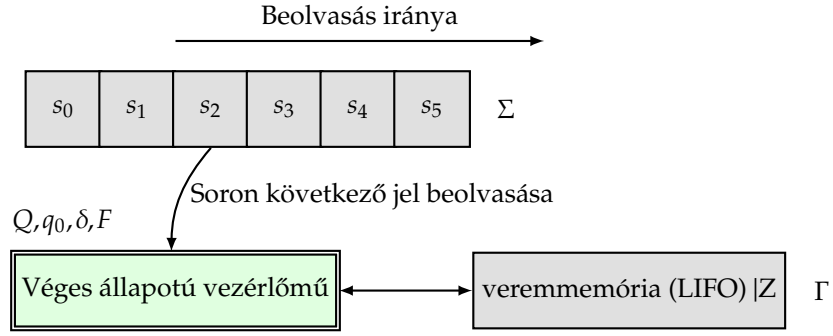
A teljes nyelvtan a _____ *nyelvtanosztályba fér bele.*
Vezessük le a következő szavakat a fenti nyelvtannal:

aabbbb *aabc*

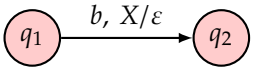
Levezethető-e valamely szó az alábbi levezetés folytatásaként? Melyik az a szó, vagy – ha nincs ilyen – miért nem vezethető le?

$$S \Rightarrow E \Rightarrow \ldots$$

9.5. Veremautomaták

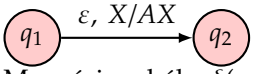


Σ szalagábécé	q_0 kezdőállapot ($q_0 \in Q$)
Γ veremábécé	Q állapotok
Z veremalja jel ($Z \in \Gamma$)	F végállapotok ($F \subset Q$)
δ mozgási szabályok	



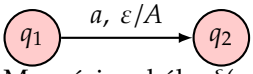
Ilyenkor POP utasítással leszed a veremről egy X-et, és az üres szót írja rá, azaz semmit. Összességében egy X tűnik el a veremről.

Mozgási szabály: $\delta($, , $) =$



Ilyenkor nem olvas a szalagról, és nem is lép jobbra az olvasófej.

Mozgási szabály: $\delta($, , $) =$



Ilyenkor nem olvas a veremről (nincs POP utasítás). A Bach-könyv ilyet nem enged meg.

Mozgási szabály: $\delta($, , $) =$

$$\delta : Q \times (\Sigma \cup \{\varepsilon\}) \times (\Gamma \cup \{\varepsilon\}) \longrightarrow Q \times \Gamma^*.$$

Mikor mondjuk, hogy egy szalagra írt szót felismert a veremautomata? A veremautomatának *kétféle működési módja* van, azaz tulajdonképpen a veremautomatának kétféle definíciója, amelyekben ez a feltétel különböző.

1. A *végállapottal felismerő* veremautomata (a véges automatákhoz hasonlóan) akkor ismeri fel a szót, ha a szó végigolvasásakor végállapotba kerül, vagy további lépésekkel végállapotba tud jutni.
2. Az *üres veremmel felismerő* veremautomata akkor ismeri fel a szót, ha a szó végigolvasásakor üres lesz a verem, vagy további lépésekkel ki tudja üríteni a vermet.

9.1. példa. *Alakítsuk át az alábbi mozgási szabályokat + végállapotokat állapotgráffá!*

$$\delta(q_0, a, Z) = (q_1, XZ)$$
$$\delta(q_1, a, X) = (q_1, XX) \qquad \delta(q_1, b, X) = (q_2, \varepsilon)$$
$$\delta(q_2, b, X) = (q_2, \varepsilon) \qquad \delta(q_2, \varepsilon, Z) = (q_0, \varepsilon)$$
$$F = \{q_0\}$$

Vezessük le konfigurációsorozattal az aabbb, aab, ab szavakat felismeri-e végállapottal. Ha nem lehet, vezessük le addig, amíg el nem akad a levezetés.

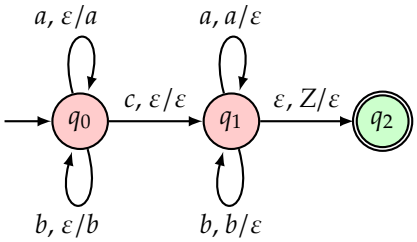
Milyen nyelvet fog felismerni végállapottal?

9.6. Feladatok a környezetfüggetlen nyelvekhez és veremautomatákhoz

9.4. feladat. *Határozzuk meg, hogy az alábbi automata felismeri-e üres veremmel az* `abbcbba` *illetve* `abcab` *szavakat! Ha igen vezessük le konfigurációsorozattal!*

Melyik nyelvet ismeri fel?

Adjuk meg a δ mozgásszabályokat, a Q állapothalmazt, az F végállapot-halmazt, az elemzendő nyelv Σ ábécéjét, a Γ veremábécét.



9.5. feladat. *43 Szűrje ki az alábbi környezetfüggetlen nyelvtanból a felesleges szimbólumokat, és egészítse ki a következő mondatokat!*

$$S \to AB;\ S \to E;\ A \to Ce;\ C \to Cc;\ C \to c;\ F \to f;\ B \to b;\ E \to eE$$

Ha a nyelvtant csak alulról felfele szűrjük, a következő szimbólumok maradnak meg:

$B = \left\{ \right\}.$

Ha ezután a nyelvtant felülről lefele szűrjük, végül a következő szimbólumok maradnak meg:

$T = \left\{ \right\}.$

Tehát ezek a szimbólumok bizonyulnak feleslegesnek:

$F = \left\{ \right\}.$

A szűrés végén megmaradt levezetési szabályok:

$P' = \left\{ \right\}.$

Vezesse le lépésenként a ccceb szót, és adja meg a levezetési fát!

9.6. feladat. *43 Szűrje ki az alábbi környezetfüggetlen nyelvtanból a felesleges szimbólumokat, és egészítse ki a következő mondatokat!*

$$S \to AC;\ S \to aA;\ C \to b;\ S \to bC;\ C \to aS;\ C \to bB;\ B \to aE;\ E \to cB;\ T \to dK$$

Ha a nyelvtant csak alulról felfele szűrjük, a következő szimbólumok maradnak meg:

$B = \left\{ \right\}.$

Ha ezután a nyelvtant felülről lefele szűrjük, végül a következő szimbólumok maradnak meg:

$T = \left\{ \right\}.$

Tehát ezek a szimbólumok bizonyulnak feleslegesnek:

$F = \left\{ \right\}.$

A szűrés végén megmaradt levezetési szabályok:

$P' = \left\{ \right\}.$

Vezesse le lépésenként a ababb szót, és adja meg a levezetési fát!

9.7. Jólfésült nyelvtan (proper grammar) és a normálformák

9.7. feladat. *Alább láthatóak egy nyelvtan levezetési szabályai. Szűrjük ki a nyelvtanból a felesleges szimbólumokat! Határozzuk meg a nyelvtan által generált nyelvet!*

$$P = \{S \to AC, \quad A \to aA, \quad C \to c, \quad S \to cC, \quad C \to aBC, \quad B \to bE, \quad F \to b, \quad D \to cCb, \quad B \to cF\}$$

1. lépés _____ *szűrés:*

$B_0 = \{$_____$\}$

A megmaradt szabályok az 1. lépés után.

_____

Az 1. lépés után feleslegesnek bizonyult szimbólum(ok): _____ .

2. lépés _____ *szűrés:*

$F_0 = \{$____$\}$

Az összes feleslegesnek bizonyult szimbólum: _____ .
Vezesse le a cacbc szót, és rajzolja fel a levezetési fát!

9.8. feladat. *Egészítse ki az alábbi mondatokat, hogy igazak legyenek!*
Egy környezetfüggetlen nyelvtant egyértelműnek nevezünk, ha

_____ .

Egy adott környezetfüggetlen nyelvhez találtam több azt generáló nyelvtant. 3-ról bebizonyítottam, hogy egyértelmű, 2-ről, hogy többértelmű, tehát a nyelvet _____ *nyelvnek nevezzük.*

Példa többértelmű nyelvre $\{$_____$\mid i, j \geq 0\} \cup \{$_____$\mid i, j \geq 0\}$. *Ezekben a* _____ *alakú szavak bármelyik nyelvtana szerint legalább kétféle fával generálhatóak.*

A _____ *nyelvosztályba tartozó nyelvek mindig egyértelműek.*

A felesleges szimbólumok kiszűréséhez a környezetfüggetlen nyelveket először _____ *(irány) kell szűrni.*

Az indulási halmaz (B_0) a levezetési szabályok _____ *oldalán szereplő*_____ *szimbólumokat tartalmazza.*

9.9. feladat. *Pipáljuk ki, melyikben lehet ilyen szabály!*

szabály	$A \to AabB$	$B \to aABB$	$S \to BB$	$A \to a$	$S \to \varepsilon$	$A \to \varepsilon$	$A \to bA$	$B \to A$
Jólfésült nyelvtan								
Chomsky NF								
Greibach NF								

Összefoglaló a szabályok megengedett alakjairól:
Chomsky-*normálformában (CNF)* $A \to a$ *vagy* $A \to BC$,
Greibach-*normálformában (GNF)* $A \to aW$, *ahol* $W \in N^*$ *(azaz W nemterminálisokból álló szó, akár nulla hosszúságú).*
Mindkettőben megengedett az $S \to \varepsilon$ *alak, ekkor nem szerepelhet az S semelyik szabály jobboldalán sem.*

9.10. feladat. *Balrekurzív-e az alábbi a nyelvtanban az A nemterminális?*

Alakítsa át az alábbi nyelvtanrészletet Chomsky-normálformára!
$S \to AB \quad A \to BaAa \quad A \to a \quad B \to AabAb \quad S \to \varepsilon$

1. lépés *jobboldali terminálisok megszüntetése a nem megfelelő alakú szabályokban:*

A 2. lépés *(szeletelés) részletezése:*

Az eredeti nyelvtanrészletet helyettesítő összes szabály, amely Chomsky-normálformában van:

Tartalomjegyzék