Formális nyelvek és a fordítóprogramok alapjai

2023/2024/2. félév, B szakirány

Nagy Sára gyakorlatai alapján

Utolsó módosítás: 2024. március 28.

Előszó

Ez a gyakorlati jegyzet a 2023/2024/2. félévben készült Nagy Sára tanárnő gyakorlatain, a B szakirányos *Formális nyelvek és a fordítóprogramok alapjai* c. tantárgy keretein belül.

Bizonyos esetekben a feladatokat részletesebben vezettem le, adott esetben saját jelölésrendszert alkalmazva (amik nem szerepeltek a gyakorlatokon), hogy szemléletesebben be tudjam mutatni az egyes feladattípusokhoz tartozó algoritmusokat. Ha nem vagy benne biztos, hogy milyen részletességgel kér(het)ik számon ezeket a zárthelyiken, fordulj a gyakorlatvezetődhöz.

Igyekeztem a legjobb tudásom szerint összeállítani a jegyzetet, ennek ellenére előfordulhatnak benne elgépelések, hibák, stb. Ha találsz ilyet, kérlek értesíts e-mailben a(z) ap3558@inf.elte.hu címen.

Sikeres felkészülést kívánok!

Kiss-Bartha Nimród

Tartalomjegyzék

1. Formális nyelvek

1.1. Nyelv lezártjának meghatározása (nem hivatalos módszer)

Feladat: Határozzuk meg egy nyelv lezártját.

Előfeltétel: (L véges nyelv legyen)

Gyakorlatokon előforduló feladat, hogy határozzuk meg egy adott nyelv iteráltját. Általában nem egy olyan feladatról van szó, amit ránézésre könnyen megállapíthatunk, mindemellett koncentrációt igényel, emiatt könnyű elveszni a hasonló sztringek halmazában. Mindemellett a felsorolás is számít, ugyanis lexikografikusan várják el a zárthelyin. A programozó meg lusta, nem szeret sok időt tölteni ilyen feladatokkal. Erre javaslom a következő megoldást!

A szemléltetés kedvéért vegyük az alábbi nyelvet: $L := \{\mathtt{a}, \mathtt{ab}, \mathtt{cba}\}$.

1. lépés: $L^0 = L_\varepsilon = \{\varepsilon\}$ és $L^1 = L$. Idáig elég egyszerű.
2. lépés: határozzuk meg L^2-t. Ilyen kis nyelvnél még akár fejben is ki lehet találni, nagyobbaknál azonban gondban leszünk. Így ezen a kicsin mutatom be a módszert.

   Írjuk fel az L szavait egy $(n+1)\times(n+1)$-es mátrixba ($n = |L|$), ahol a sorokba és oszlopokba helyezzük el a szavakat *lexikografikusan*! Ezután a szavakat soronként és oszloponként konkatenálva töltsük ki a táblázatot!

LL	a	ab	cba
a	aa	aab	acba
ab	aba	abab	abcba
cba	cbaa	cbaab	cbacba

   Innen kiolvashatjuk, hogy

$$L^2 = \{\mathtt{aa}, \mathtt{aab}, \mathtt{aba}, \mathtt{abab}, \mathtt{acba}, \mathtt{cbaa}, \mathtt{abcba}, \mathtt{cbaab}, \mathtt{cbacba}\}.$$

3. lépés: elvégezzük a hatványraemelést a hatványozás definíciója szerint (azaz ha $n > 1$, akkor $L^n = L^{n-1}L$). Azt megállapítottuk, hogy nem számít, hogy balrekurzívan vagy jobbrekurzívan határozzuk meg a műveletet, mindkét módon helyes eredményeket kapunk. Ezt kihasználva úgy döntöttem, az L^2 elemeit a 1. sorban, az L elemeit meg a 1. oszlopban sorakoztatom fel – ami jobbrekurzív hatványozást jelent.

LL^2	aa	aab	aba	abab	acba	cbaa	abcba	cbaab	cbacba
a	$\mathtt{a}^3$	aaab	aaba	aabab	aacba	acbaa	aabcba	acbaab	acbacba
ab	abaa	abaab	ababa	$\mathtt{ab}^3$	abacba	abcbaa	ababcba	abcbaab	abcbacba
cba	cbaaa	cbaaab	cbaaba	cbaabab	cbaacba	cbacbaa	cbaabcba	cbacbaab	$\mathtt{cba}^3$

A legnagyobb problémát ezúttal csak a lexikografikus felsorolás jelentheti, ugyanis eltérő hosszú szavak vannak az egyes L^i halmazokban, mégha a legrövidebb hossz az halmazról halmazra nő. De még így is megfigyelhető egy olyan tendencia, hogy a táblázat bal felső sarkából elindulva egy olyan útvonalon járhatjuk be a cellákat, hogy azokból megkapjuk a szükséges sorrendet.

1.2. Nyelvtan típusának meghatározása

Legyen $G := (N, T, P, S)$, ahol $T := \{\mathtt{a}, \mathtt{b}\}$ és P szabályai:

$$S \longrightarrow \varepsilon,$$
$$S \longrightarrow \mathtt{a}S\mathtt{b},$$
$$S \longrightarrow SS.$$

Első lépésként azonosítsuk be az egyes szabályokról, hogy alakilak milyen típusú nyelvtanok szabályaihoz tartozhatnak! Ezeket felsorolhatjuk a szabályok mellé.

Vegyük figyelembe, hogy egy szabály több nyelvtani típusnak is része lehet. Gondoljunk csak bele, a 0-ás típusú nyelvtant úgy definiáltuk, hogy nem vonatkozik rá semmilyen megkötés, így minden produkciós szabály automatikusan 0-ás típusú nyelvtannak része.

$S \longrightarrow \varepsilon$	0, 1, **2**, 3
$S \longrightarrow \mathtt{a}S\mathtt{b}$	0, **2**
$S \longrightarrow SS$	0, **2**

Az első szabály mindegyik típusú nyelvtanban szerepelhet.

A második az 1-esben már nem szerepelhet, ugyanis megkötöttük, hogy az üres szót előállító szabályunk nem szerepelhet más szabály jobb oldalán. Márpedig az S előállítha ε-t és jobb oldalon szerepel. 3-as típusú azért nem lehet, mert az $\boxed{\mathtt{a}S\mathtt{b}}$ jobb oldal nem illeszkedik bele egyik alakba sem:

- az S nemterminális után rögtön következik egy terminális szimbólum (ezt az $A \longrightarrow uB$ $(A, B \in N, u \in T^*)$ forma nem engedi),
- valamint a terminálisok közé közbeékelt S nemterminális sem megengedett az $A \longrightarrow u$ $(u \in T^*)$ alaki megszorítás szerint.

Hasonló megfontolásokból következnek a 3. szabályra vonatkozó megállapítások.

Jelöljük ki azt a legmagasabb számot, amely az összes szabálynál szerepel. Ez fogja meghatározni, hogy milyen típusba esik bele a grammatikánk. Ebből megállapíthatjuk, hogy a G egy 2-es típusú vagy környezetfüggetlen nyelvtan, azaz

$$G \in \mathcal{G}_2. \ \checkmark$$

Korábbról tudjuk, hogy

$$\mathcal{G}_3 \subseteq \mathcal{G}_2.$$

Kérdés, hogy **tudunk-e 3-as típusú grammatikát adni ehhez a nyelvhez**. Ez azért egy fontos felvetés, ugyanis a célunk az, hogy *a lehető legbelsőbb részhalmazba tudjunk „átlépni”* (ez esetben a 3-as típusú grammatikák halmazába), ami a fordítóprogramok szempontjából egy óriási előnyt jelent.

Sajnos, meg kell előlegeznünk, hogy ehhez a nyelvhez *nem tudunk $\mathcal{G}_3$-beli grammatikát adni.* Ezt csak később fogjuk tudni belátni.

A G által generált nyelv 2-es típusú nyelv lesz, ezért

$$L(G) \in \mathcal{L}_2$$

teljesül. Ha netán mégis létezne hozzá $\mathcal{G}_3$-beli nyelvtan, akkor a **generált nyelv szigorú típusa** $L(G) \in \mathcal{L}_3$ lenne. De mivel nem tudunk ilyet adni, így a szigorú típusa $\mathcal{L}_2$.

1.3. Szó levezetése

Legyen a levezetendő szó $u := \texttt{aabaabbb}$. Idézzük fel a G nyelvtan produkciós szabályait:

$$S \longrightarrow \varepsilon,$$
$$S \longrightarrow \texttt{a}S\texttt{b},$$
$$S \longrightarrow SS.$$

Ha ezt a szót le tudjuk vezetni a P-beli szabályok véges szokszori alkalmazásával, akkor ez a szó eleme lesz a G által generált nyelvnek.

Ismerjük a nyelvtani szabályokat, ismerjük a startszimbólumot, valamint ismerjük a legyártandó szót is. Tehát valamilyen lépéssorozatból az alábbi összefüggést, levezetést kell megkapjuk:

$$S \overset{*}{\underset{G}{\Longrightarrow}} \texttt{aabaabbb}.$$

Egy kis ismétlés: a $\Longrightarrow$ alatt a G azt jelenti, hogy a G nyelvtan generálja a szót, ami egyértelmű kontextus esetén elhagyható. A $*$ meg azt, hogy véges sok lépésből vezethetjük le a szót.

Induljunk ki az S-ből! Vizsgáljuk meg, melyik szabályokat tudjuk alkalmazni – szakszóval ezt úgy mondjuk, hogy **mely szabályok *tüzelőképesek***. A mi esetünkben mindhárom tüzelőképes, így sok választási lehetőségünk van. Kezdetnek alkalmazzuk a 2. szabályt!

$$S \xRightarrow[G]{\text{2. szabály}} \texttt{a}S\texttt{b}.$$

Ezután alkalmazzuk a 3. szabályt!

$$S \xRightarrow[G]{\text{2. szabály}} \texttt{a}S\texttt{b} \xRightarrow[G]{\text{3. szabály}} \texttt{a}SS\texttt{b}.$$

Itt már két nemterminálisunk van, úgyhogy bármelyikkel folytathatjuk, nincs megszabva semmilyen sorrend. A lényeg, hogy létezzen egy véges lépéssorozata a szabályoknak, mellyel az S-ből a kívánt szóba el tudunk jutni – és ez a szabálysorozat lehet tetszőlegesen egyszerű vagy feleslegesen hosszú, kacifántos és bonyolult. Ha van ilyen, akkor más létezik a szó a nyelvben.

Itt az első S-re fogom alkalmazni a 2., majd az 1. szabályt.

$$S \xRightarrow[G]{2.} \texttt{a}S\texttt{b} \xRightarrow[G]{3.} \texttt{a}SS\texttt{b}. \xRightarrow[G]{\text{2. (első } S\text{-re)}} \texttt{aa}S\texttt{b}S\texttt{b} \xRightarrow[G]{\text{1. (első } S\text{-re)}} \texttt{aa}\varepsilon\texttt{b}S\texttt{b} = \texttt{aab}S\texttt{b}.$$

Ismét egyetlen S nemterminálisunk maradt. Ezekre a 2. szabályt fogom alkalmazni kétszer, majd végül az elsőt, hogy lezárjam a szót.

$$\texttt{aab}S\texttt{b} \xRightarrow[G]{2.} \texttt{aaba}S\texttt{bb} \xRightarrow[G]{2.} \texttt{aabaa}S\texttt{bbb} \xRightarrow[G]{\text{1. (üres szavas szabály)}} \texttt{aabaabbb}. \quad \checkmark$$

Nem kell megijedni, ha más megoldást is találunk. Egy adott szót többféleképpen is levezethetünk. A P-beli produkciós szabályok valójában *relációk*, nem függvények, ezért nondeterminisztikusak.

Megjegyzés. Ha kicseréljük az a-t a $\boxed{(}$ karakterre és a b-t a $\boxed{)}$ karakterre, akkor megkapjuk pontosan a *helyes zárójelezések nyelvét*. Tehát, ha a levezetett szavunkban kicseréljük a szimbólumokat, akkor az alábbi eredményt kapjuk:

$$\Big(()\big(()\big)\Big).$$

1.4. Epszilon-mentesítés (ε-mentesítés)

Feladat: Transzformáljuk át a $G \in \mathcal{G}_2$ (vagy $G \in \mathcal{G}_3$) nyelvtant úgy, hogy

- ha az S-ből közvetetten levezethető az ε, akkor azt a G egyetlen módon legyen képes előállítani,
- ha nem levezehthető az S-ből közvetetten az ε, akkor szabaduljunk meg az ε-szabályoktól.

Előfeltétel: A nyelvtan 2-es vagy 3-as típusú legyen.

A feladathoz kapcsolódik egy tétel, így a megoldás maga ezen tétel bizonyításának az algoritmusa.

Szemléltessük az alábbi nyelvtanon!

$$\begin{aligned} G: \quad & S \longrightarrow BA \mid \mathtt{aa} \\ & A \longrightarrow BB \mid \mathtt{a}A\mathtt{b} \\ & B \longrightarrow \varepsilon \mid S\mathtt{b}A \end{aligned}$$

Legyen H_1 halmaz, ami azon nemterminálisokat tartalmazza, melyekből közvetlenül levezethető az ε szó (más szóval, $A \longrightarrow \varepsilon$ alakúakat tartalmaz).

$$H_1 := \{B\}, \qquad \text{ugyanis } B \longrightarrow \varepsilon.$$

Ezt bővítjük iteratívan.

$$\begin{aligned} H_2 &:= H_1 \cup \{A\} = \{A, B\}, && \text{ugyanis } A\text{-ból } B\text{-be eljuthatunk.} \\ H_3 &:= H_2 \cup \{S\} = \{A, B, S\}, && \text{ugyanis } S\text{-ből } A\text{-ba (ahonnal } B\text{-be) eljuthatunk.} \end{aligned}$$

Mivel $S \in H_3$, ez azt jelenti, hogy $S \overset{*}{\underset{G}{\Longrightarrow}} \varepsilon$ (a startszimbólumból levezethető az üres szó).

Alakítsuk át a G nyelvtant (ezt jelöljük G'-vel) úgy, hogy NE lehessen előállítani az üres szót.[1]

Az átalakítás **alapelve**, hogy fogjuk azokat a szabályokat, amelynek *jobb oldalán szerepelnek nemterminális jelek*. Ezeket a jobb oldalakat úgy szabdaljuk fel, hogy 0 vagy 1 darab szerepeljen belőlük. Végül az így kapott kombinációkat „összeolvasztjuk”, így megkapjuk az új szabályt G'-ben.

I. $\boxed{S \longrightarrow BA \mid \mathtt{aa}}$. A kritikus szabály(ok): I/I.

I/I. felbontása:

$$\left.\begin{aligned} S &\longrightarrow B \\ S &\longrightarrow A \\ S &\longrightarrow \underline{BA} \end{aligned}\right\} \Longrightarrow \boxed{S \longrightarrow A \mid B \mid BA \mid \mathtt{aa}}$$

Ne feledjük hozzácsatolni azokat a jobb oldalakat, amelyeken nem módosítottunk.

II. $\boxed{A \longrightarrow BB \mid \mathtt{a}A\mathtt{b}}$. A kritikus szabály(ok): II/I. és II/II.

II/I. felbontása:

$$\left.\begin{aligned} A &\longrightarrow B \\ A &\longrightarrow \underline{BB} \end{aligned}\right\} \Longrightarrow \boxed{A \longrightarrow B \mid BB}$$

II/II. felbontása:

$$\left.\begin{aligned} A &\longrightarrow \mathtt{ab} \\ A &\longrightarrow \underline{\mathtt{a}A\mathtt{b}} \end{aligned}\right\} \Longrightarrow \boxed{A \longrightarrow \mathtt{ab} \mid \mathtt{a}A\mathtt{b}}$$

Új szabály G'-ben: $\boxed{A \longrightarrow B \mid BB \mid \mathtt{ab} \mid \mathtt{a}A\mathtt{b}}$.

[1] Ez elsőre furának tűnhet, hiszen az imént pont azt állapítottuk meg, hogy levezethető a nyelvtanból. Ezt a „*speciális esetet*” később lekezeljük.

III. $\boxed{B \longrightarrow \varepsilon \mid S\texttt{b}A}$. A kritikus szabály(ok): III/II.

Figyelem: Az ε-szabálytól megszabadulunk! Később lekezeljük a hiányát.

III/II. felbontása:

$$\left.\begin{aligned} B &\longrightarrow S\texttt{b} \\ B &\longrightarrow \texttt{b}A \\ B &\longrightarrow \texttt{b} \\ B &\longrightarrow \underline{S\texttt{b}A} \end{aligned}\right\} \Longrightarrow \boxed{B \longrightarrow \texttt{b} \mid \texttt{b}A \mid S\texttt{b} \mid S\texttt{b}A}$$

Végső lépés: hogy ne veszítük e az ε-szó legenerálhatóságát, bevezetünk egy **új startszimbólum**ot:

$$S' \longrightarrow \varepsilon \mid S.$$

Ezzel epszilon-mentesítettük a G grammatikát!

$$G: \quad \begin{aligned} S &\longrightarrow BA \mid \texttt{aa} \\ A &\longrightarrow BB \mid \texttt{a}A\texttt{b} \\ B &\longrightarrow \varepsilon \mid S\texttt{b}A \end{aligned}$$

$$G': \quad \begin{aligned} S &\longrightarrow A \mid B \mid BA \mid \texttt{aa} \\ A &\longrightarrow B \mid BB \mid \texttt{ab} \mid \texttt{a}A\texttt{b} \\ B &\longrightarrow \texttt{b} \mid \texttt{b}A \mid S\texttt{b} \mid S\texttt{b}A \\ \rightarrow S' &\longrightarrow \varepsilon \mid S \end{aligned}$$

1.5. 3-as normálformára hozás algoritmusa és automata előállítása

1.5.1. 3-as normálformára hozás algoritmusa

I. Hosszredukció

II. Befejező szabályok átalakítása

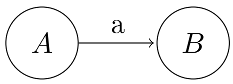
III. Láncmentesítés

1.5.2. Automata előállítása normálformából

A 3-as normálforma az alábbi kétféle alakú szabályt engedi meg:

1. $\boxed{A \longrightarrow aB}$, ahol $A, B \in N$ és $a \in T$ (egyetlen szimbólum).

   Erre gondolhatunk úgy, mintha a *nemterminálisok* a **gráf csúcsai** lennének és a *terminális szimbólum* a szabály jobb oldalán (ne feledjük, hogy kizárólag egyetlen szerepelhet belőle) meg az **él címkéje** lenne. Vizualizálva:

2. $\boxed{A \longrightarrow \varepsilon}$, ahol $A \in N$.

   Az ε azt jelenti, hogy nem tud sehova sem tovább lépni. Ilyenkor **elfogadóállapot**ba érkezünk (vagy végállapotba). Vizualizálva:

1.6. Véges nemdeterminisztikus automata determinisztikussá alakítása (VNDA $\rightarrow$ VDA)

Feladat: Véges nemdeterminisztikus automatát (VNDA vagy NDA) alakítsuk át úgy, hogy determinisztikus legyen (VDA).

Előfeltétel: Véges automata ($\mathcal{G}_3$-beli nyelvtanhoz).

Adott az alábbi VNDA:

	δ	a	b
$\rightarrow$	S	A, C	
$\leftarrow$	A	A	B, S
	B	A	C, S
$\leftarrow$	C	S	

A $\rightarrow$ nyíl jelöli a kezdőállapotot, a $\leftarrow$ pedig az elfogadóállapotokat (más néven a végállapotokat). Nemdeterminisztikus automatával van dolgunk, ugyanis S-ből A-ba és B-be is eljuthatunk, melyek ugyanazt a terminálist eredményezik.

Írjuk fel halmazosan az **első sor**t. Az egyes halmazok fogják jelölni az állapotok „címkéit” (minta q_1 vagy 1-ről lenne szó). Az új címkéket aláhúzással jelölöm. Minden egyes új címkét feldolgozunk.

	δ	a	b
$\rightarrow$	$\{S\}$	$\underline{\{A,C\}}$	$\underline{\emptyset}$

Dolgozzuk fel az új címkéket. Az eljárás hasonlít a BFS-re (szélességi gráfbejárásra).

Az $\{A, C\}$ halmaz elemei a táblázat szerint elfogadóállapotok, így a halmaz is megjelölhető ezzel a tulajdonsággal.

	δ	a	b
$\rightarrow$	$\{S\}$	$\underline{\{A,C\}}$	$\underline{\emptyset}$
$\leftarrow$	$\{A,C\}$	$\underbrace{\{A\}}_{A} \cup \underbrace{\{S\}}_{C} = \underline{\{A,S\}}$	$\underbrace{\{B,S\}}_{A} \cup \emptyset = \underline{\{B,S\}}$
	$\emptyset$	$\emptyset$ ✓	$\emptyset$ ✓

Az $\{A, C\}$ halmaz két új halmazt, címkét eredményezett, így ezeket fel kell dolgoznunk.

Az üreshalmazból nyilvánvalóan egyik állapotba sem tudunk eljutni. Új halmaz sem jön létre, ezért feldolgozottnak jelöljük (kipipáljuk). Ha a hibát is jelezni akarjuk, akkor a(z) $\emptyset$-t kicserélhetjük egy H hibahalmazra.

	δ	a	b
$\rightarrow$	$\{S\}$	$\underline{\{A,C\}}$	$\underline{\emptyset}$
$\leftarrow$	$\{A,C\}$	$\underline{\{A,S\}}$	$\underline{\{B,S\}}$
	$\emptyset$	$\emptyset$ ✓	$\emptyset$ ✓
$\leftarrow$	$\{A,S\}$	$\{A\} \cup \{C\} = \{A,C\}$ ✓	$\emptyset \cup \{B,S\} = \{B,S\}$

Az $\{A, S\}$-ben van elfogadóállapot, így az egész megjelölhető annak. A halmaz nem hozott létre újabb halmazokat. Mivel az $\{A, C\}$-t korábban feldolgoztuk, ezért kipipálással megjelölöm. A $\{B, S\}$ meg amúgy is fel lett volna dolgozva a soron következő lépésben.

Hasonló lépésekkel végighaladunk az összes címkén. Ha elfogynak a halmazaink, az eljárás megáll.

δ	a	b
$\rightarrow$ $\{S\}$	$\{A, C\}$	$\emptyset$
$\leftarrow$ $\{A, C\}$	$\{A, S\}$	$\{B, S\}$
$\emptyset$	$\emptyset$ ✓	$\emptyset$ ✓
$\leftarrow$ $\{A, S\}$	$\{A\} \cup \{C\} = \{A, C\}$ ✓	$\emptyset \cup \{B, S\} = \{B, S\}$
$\{B, S\}$	$\{A\} \cup \{A, C\} = \{A, C\}$	$\{C, S\} \cup \emptyset = \{C, S\}$
$\leftarrow$ $\{C, S\}$	$\{S\} \cup \{A, C\} = \{A, C, S\}$	$\emptyset \cup \emptyset = \emptyset$
$\leftarrow$ $\{A, C, S\}$	$\{A\} \cup \{S\} \cup \{A, C\} = \{A, C, S\}$ ✓	$\{B, S\} \cup \emptyset \cup \emptyset = \{B, S\}$

A kapott táblázatunk a bal oldalon található. Gyakran átnevezzük, átcímkézzük ezeket a halmazokat a könnyebb olvashatóság kedvéért számjegyekkel. Ez analóg a $q_0, q_1, \ldots$ jelöléssel.

δ	a	b
$\rightarrow$ $\{S\}$	$\{A, C\}$	$\emptyset$
$\leftarrow$ $\{A, C\}$	$\{A, S\}$	$\{B, S\}$
$\emptyset$	$\emptyset$	$\emptyset$
$\leftarrow$ $\{A, S\}$	$\{A, C\}$	$\{B, S\}$
$\{B, S\}$	$\{A, C\}$	$\{C, S\}$
$\leftarrow$ $\{C, S\}$	$\{A, C, S\}$	$\emptyset$
$\leftarrow$ $\{A, C, S\}$	$\{A, C, S\}$	$\{B, S\}$

δ	a	b
$\rightarrow$ 1	2	3
$\leftarrow$ 2	4	5
3	3	3
$\leftarrow$ 4	2	5
5	2	6
$\leftarrow$ 6	7	3
$\leftarrow$ 7	7	5

1.7. Véges determinisztikus automata (VDA) minimalizálása

Más néven: minimális automata előállítása, VDA összefüggővé tétele.

Feladat: VDA-ban az ekvivalens állapotokat egybeolvasztjuk – azaz, azon állapotokat, melyek azonos szóra azonos eredményt adnak.

Előfeltétel: VDA (feltételezzük, hogy megszabadultunk a nemdeterminisztikus jellegétől).

Vegyük az alábbi VDA-t:[2]

	δ	a	b
$\to$	1	4	5
$\leftarrow$	2	3	4
$\leftarrow$	3	2	8
	4	9	2
	5	2	3
	6	8	7
	7	8	1
	8	9	3
$\leftarrow$	9	9	9

A δ-relációra tekinthetünk úgy, mint egy olyan irányított gráfra, melynek mindegyik csúcsának fokszáma 2 és az élei fel vannak címkézve.

Első lépésként meg kell határoznunk azon állapotokat (azaz a gráf azon csúcssait), amik elérhetők közvetett módon a kezdőállapotból (startcsúcsból). Ez a **BFS**-t vagy **szélességi gráfbejárás**t fogja jelenteni.[3]

$d(u)$ címkék									$u : d(u)$	$Q : Queue$	$\pi(u)$ címkék								
1	2	3	4	5	6	7	8	9			1	2	3	4	5	6	7	8	9
0	∞	∞	∞	∞	∞	∞	∞	∞		$\langle 1 \rangle$	⦸	⦸	⦸	⦸	⦸	⦸	⦸	⦸	⦸
			1	1					$1 : 0$	$\langle 4, 5 \rangle$				1	1				
	2							2	$4 : 1$	$\langle 5, 2, 9 \rangle$		4							4
		3							$5 : 1$	$\langle 2, 9, 3 \rangle$			5						
									$2 : 2$	$\langle 9, 3 \rangle$									
									$9 : 2$	$\langle 3 \rangle$									
							4		$3 : 3$	$\langle 8 \rangle$								3	
									$8 : 4$	$\langle \rangle$									
0	2	3	1	1	∞	∞	4	2	eredmény		⦸	4	5	1	1	⦸	⦸	3	4

Megállapíthatjuk belőle, hogy ahol $d(u) = \infty \land \pi(u) = ⦸$, azon csúcsok nem elérhetők az 1-es startcsúcsból (kezdőállapotból). Tehát, a 6-os és 7-es állapotoktól könnyedén megszabadulhatunk.

Az 1-esből *elérhető állapotokat felosztjuk* két halmazra: **nem-elfogadóállapotok** (azaz $\{1, 4, 5, 8\}$) és **elfogadóállapotok** (azaz $\{2, 3, 9\}$) halmazára. Az, hogy melyek elfogadóállapotok, azokat a táblázatból könnyedén leolvashatjuk („$\leftarrow$” nyíllal vannak megjelölve).

Ezeket a halmazokat szeretnénk **finomítani**, más szóval **partícionáljuk** őket. Ez azt jelenti, hogy egyre több halmaz jön létre és az egyes halmazok elemszáma egyre csökken. A partícionálás a *szavak hossza szerint* történik. A partíciók jele: $\boxed{\sim^i}$ vagy $\boxed{\overset{i}{\sim}}$, ahol $i = \ell(u)$ $(u \in L(A))$.

[2]**Vigyázat**: Ez nem ugyanaz, mint amit az előző feladatban kaptunk.

[3]Fontos kiemelni, hogy a mi esetünkben nem egyszerű gráfról van szó, ugyanis hurokéleket tartalmaz. Emiatt szigorú értelemben arra számíthatunk, hogy az az algoritmus, amit *Algoritmusok és adatszerkezetek II.*-n tanultunk, elakadhat emiatt. Ennek ellenére mégis ezt a szemléltetést fogom alkalmazni. Az az érvem mellette, hogy ha olyan csúcsot fedezünk fel, melynek van hurokéle, a π-címkéjének átállítása után másodjára nem fog bekerülni a $Q : Queue$ sorba, ezért mégsem fog komoly gondot jelenteni.

	Nem-elfogadóállapotok	**Elfogadóállapotok**
$\overset{0}{\sim}$:	$\{1,4,5,8\} =: A$	$\{2,3,9\} =: B$

$\overset{1}{\sim}$:

Nem-elfogadóállapotok

Felbontandó partíciók: A.

δ	`a`	`b`
1	$4 \in A$	$5 \in A$
4	$9 \in B$	$2 \in B$
5	$2 \in B$	$3 \in B$
8	$9 \in B$	$3 \in B$

Avagy ugyanez tömörebben.

δ	`a`	`b`
1	**A**	**A**
4	B	B
5	B	B
8	B	B

Új partíciók: $\{1\}$; $\{4,5,8\}$.

Elfogadóállapotok

Felbontandó partíciók: B.

δ	`a`	`b`
2	$3 \in B$	$4 \in A$
3	$2 \in B$	$8 \in A$
9	$9 \in B$	$9 \in B$

Avagy ugyanez tömörebben.

δ	`a`	`b`
2	B	A
3	B	A
9	**B**	**B**

Új partíciók: $\{2,3\}$; $\{9\}$.

$\overset{2}{\sim}$:

Nem-elfogadóállapotok

$\{1\} =: C$; $\{4,5,8\} =: D$

Felbontandó partíciók: D.

δ	`a`	`b`
4	F	E
5	**E**	**E**
8	F	E

Új partíciók: $\{1\}$; $\{4,8\}$; $\{8\}$.

Elfogadóállapotok

$\{2,3\} =: E$; $\{9\} =: F$

Felbontandó partíciók: E.

δ	`a`	`b`
2	E	D
3	E	D

Itt nem történik finomodás, marad minden a régiben.

$\overset{3}{\sim}$:

Nem-elfogadóállapotok

$\{1\} =: C$; $\{4,8\} =: G$; $\{5\} =: H$

Felbontandó partíciók: G.

δ	`a`	`b`
4	G	E
8	G	E

Nem történt finomodás

Elfogadóállapotok

–

Mivel $\overset{3}{\sim}$ -ban ($\ell(u) = 3$ szavakban) nem történt finomodás a nem-elfogadóállapotok esetében az előző lépéshez képest (az elfogadóállapotok meg korábban véget értek), így leáll az algoritmus. Újracímkézzük azon állapotokat, partíciókat, melyek több elemből állnak. Így

$$\{2,3\} \to 23; \quad \{4,8\} \to 48.$$

A minimális automata / minimalizált VDA pedig:

	δ	`a`	`b`
$\to$	1	48	5
	48	9	23
$\leftarrow$	23	23	48
	5	23	23
$\leftarrow$	9	9	9

2. Fordítóprogramok

2.1. title