7. CYK, Turing-gépek

1. A Cocke-Younger-Kasami algoritmussal elemezzük az `aaab` szót a következő nyelvtan alapján.

$$S \to XY \mid YX \qquad X \to AZ \mid \texttt{a} \qquad Z \to XA \qquad Y \to AT \mid AA \mid \texttt{b} \qquad T \to AY \qquad A \to \texttt{a}$$

Az alábbi táblázatban már kitöltöttük a 2. és 3. sorokat.

(a) Töltse ki az első sort!

(b) Mit jelent az, hogy két S szimbólum került a 3. sor elsõ mezőjébe?

(c) Egészítse ki a megfelelő indexekkel a táblázatban szereplő változókat!

(d) Mi lesz a legfelső mező tartalma?

(e) A táblázat alapján állapítsa meg, hogy a megadott szó levezethető-e a nyelvtanból!

4.				
3.	S S X T	Y		
2.	Z Y	Z Y	S T	
1.				
	a	a	a	b

Megoldás: A nyelvtan Chomsky-normálformában van, a CYK algoritmust átalakítás nélkül lehet rá alkalmazni,

(a) Ide azok a változók jönnek amiknek van az adott betűből álló jobb oldala, lásd a táblázat.

(b) Azt, hogy S-ből az adott szó 3 hosszú kezdőszelete kétféleképpen is levezethető.

(c-d) Lásd a táblázat. A nem egy betűs szabályokat számoztuk meg:

$$S \to \overset{1}{XY} \mid \overset{2}{YX} \qquad X \to \overset{3}{AZ} \mid \texttt{a} \qquad Z \to \overset{4}{XA} \qquad Y \to \overset{5}{AT} \mid \overset{6}{AA} \mid \texttt{b} \qquad T \to \overset{7}{AY} \qquad A \to \texttt{a}$$

4.	$S_{1,1}$ $T_{7,1}$ $S_{1,3}$			
3.	$S_{1,1}$ $S_{2,2}$ $X_{3,1}$ $T_{7,1}$	$Y_{5,1}$		
2.	$Z_{4,1}$ $Y_{6,1}$	$Z_{4,1}$ $Y_{6,1}$	$S_{1,1}$ $T_{7,1}$	
1.	X A	X A	X A	Y
	a	a	a	b

(e) Mivel S szerepel a legfelső mezőben, a szó levezethető. (Sőt, mivel kétszer is szerepel, kétféle levezetési fa is van. Az indexek mutatják, hogy az egyik az 1., a másik a 2. szabállyal indul.)

2. A CYK-algoritmussal elemezze az alábbi nyelvtant használva az `abbbba` és az `abbba` szavakat! Rajzolja fel a kapott levezetési fákat is!

$$S \to AX \mid BY \mid AA \mid BB \qquad X \to SA \qquad Y \to SB \qquad A \to \mathtt{a} \qquad B \to \mathtt{b}$$

Megoldás: A nyelvtan Chomsky-normálformában van, tehát az algoritmus a nyelvtan további átalakítása nélkül alkalmazható.

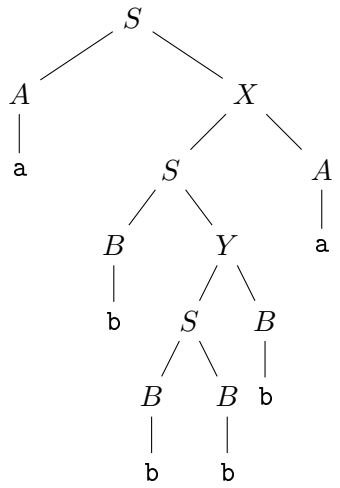
Beszámozzuk a szabályokat:

$$S \to \overset{1}{A}X \mid \overset{2}{B}Y \mid \overset{3}{A}A \mid \overset{4}{B}B \qquad X \to \overset{5}{S}A \qquad Y \to \overset{6}{S}B \qquad A \to \mathtt{a} \qquad B \to \mathtt{b},$$

majd kitöltjük a táblázatot.

6.	$S_{1,1}$					
5.	—	$X_{5,4}$				
4.	—	$S_{2,1}$	—			
3.	—	$Y_{6,2}$	$Y_{6,2}$	$X_{5,2}$		
2.	—	$S_{4,1}$	$S_{4,1}$	$S_{4,1}$	—	
1.	A	B	B	B	B	A
	a	b	b	b	b	a

Látszik, hogy egyetlen levezetési fa van a szóhoz:



A másik, rövidebb szóra:

5.	—				
4.	—	—			
3.	—	$Y_{6,2}$	$X_{5,2}$		
2.	—	$S_{4,1}$	$S_{4,1}$	—	
1.	A	B	B	B	A
	a	b	b	b	a

Mivel a felső mezőbe nem került be az S, ezért a szó nem vezethető le, nincs levezetési fája.

3. Adjon meg egy 2 szalagos, determinisztikus Turing-gépet (az átmeneti függvény leírásával) az $\{\mathtt{a}^n\mathtt{b}^n\mathtt{c}^n : n \geq 0\}$ nyelvhez!

Megoldás: A működés elve: a 2. szalagra egy, a szalag elejét mutató X karakter kiírása után q_a állapotban lemásoljuk az `a`-kat. Amikor az 1. szalagon a `b`-khez érünk, egy új q_b állapotban összehasonlítjuk a `b`-k számát a 2. szalagon levő `a`-kkal. Ha egyszerre érünk az 1. szalagon az első `c`-hez és a 2-on az X-hez, akkor a q_c állapotban az első szalag `c`-inek számát hasonlítjuk a 2. szalag `a`-ihoz. Elfogadunk (q_+), ha egyszerre érünk az első $*$-hoz mindkét szalagon.

	(∗,∗)	(a, ∗)	(b,∗)	(b,a)	(c,X)	(c,a)
q_0	(q_+, ∗ , ∗ ,H,H)	(q_a,a, X, H, J)				
q_a		(q_a,a, a, J, J)	(q_b,b, ∗ ,H,B)			
q_b				(q_b,b,a,J,B)	(q_c, c,X,H,J)	
q_c	(q_+, ∗ , ∗ ,H,H)					(q_c,c,a,J,J)

Elfogadó állapot: q_+ – itt a számítás véget ér. Vagy az üres bemenetnél léphetünk ide q_0-ból, vagy ha sikeresen feldolgoztuk az egész szót.

4. Legyen M egy véges automata, aminek ábécéje $\Sigma = \{0,1,2,3\}$. Hogyan lehet ebből egy olyan M' véges automatát készíteni, aminek ábécéje $\Sigma' = \{0,1\}$ és az $L(M')$ szavait úgy kapjuk $L(M)$ szavaiból, hogy Σ karakterei helyett a megfelelő két-két bitet írjuk egymás után. (Például, ha $0103 \in L(M)$, akkor $00010011 \in L(M')$.)

Megoldás: Ötlet: képzeljük azt, hogy M átmenetei két bitesek. Ez így a biteken nem lenne szabályos VA, hiszen annak átmenetei csak egy-egy karaktertől függhetnek. Úgy kaphatunk egy (szabályos) M' véges automatát, hogy minden eredeti átmenetet két lépéssel helyettesítünk.

Az M minden q állapotához vegyünk fel két további állapotot: q_0, q_1. Az M minden $(q,a) \mapsto q'$ átmenetéhez (ahol $a = b_1b_2 \in \{00,01,10,11\}$) tartozzon az M'-ben a $(q,b_1) \mapsto q_{b_1}$ és a $(q_{b_1}, b_2) \mapsto q'$ átmenet.

Kezdőállapot az M kezdőállaota, elfogadók az M elfogadó állaotai.

Világos, hogy az M-beli számítási utak megfelelnek az M' számítási útjainak (csak az utóbbiban kétszer annyi lépésből állnak), a két automata ugyanazokat a szavakat fogadja el.

5. Vázolja, hogyan kell az előző feladat konstrukcióját módosítani, kiegészíteni, ha véges automaták helyett Turing-gépekről van szó.

Megoldás: A Turing-gép felül is tudja írni a szalag tartalmát, és a fejét is megfelelően mozgatni kell. Itt 1 eredeti lépést nem két lépés fog helyettesíteni, mert két lépés kell ahhoz, hogy a bemeneti információt összegyűjtsük, és ez után lehet csak felülírni a szalagot és a fejet a megfelelő helyre vinni. Pl. egy szalagos esetben egy $\delta(q,a) = (q',c,B)$ átmenetet $a = a_1a_2$, $c = c_1c_2$ esetén a következő lépéssorozattal „helyettesíthetjük":

beolvasás+2. bit felülírása: $(q,a_1) \mapsto (q_{a_1}, a_1, B)$, $(q_{a_1}, a_2) \mapsto (q_{a_1,a_2}, c_2, B)$

1. bit felülírása: $(q_{a_1,a_2}, a_1) \mapsto (q', c_1, B)$

ahol q_{a_1}, q_{a_1,a_2} az eredeti átmenethez tartozó új állapotok.

Ha a fej eredetileg helyben marad, akkor hasonló jó, csak az utolsó lépésben nem B, hanem H kell (és ezzel az első biten áll a fej). A jobbra lépéshez itt is meg még egyszer jobbra kell lépni (amihez újabb állapot szükséegs).

6. Igazolja, hogy $\overline{L_d}$

   (a) rekurzívan felsorolható!

   (b) nem rekurzív!

Megoldás: $\overline{L_d} = \{w : w \notin L_d\} = \{w :$ nem TG kód vagy $w \in L(M_w)\}$.

(a) Ehhez azt kell megmutatni, hogy van az $\overline{L_d}$ nyelvet elfogadó TG. Vázolunk egyet.
Az M gép egy w bemeneten

- ellenőrzi, hogy w egy TG kódja-e. Ha nem, akkor M álljon meg elfogadó állapotban.
- Futtatja az M_w gépet a w bemeneten.
  - Ha ez elfogad, akkor M is álljon meg elfogadóban.
  - Ha M_w megáll és elutasít, akkor M is álljon meg egy nem elfogadó állapotban.

(Természetesen, ha M_w nem áll meg, akkor az ezt futtató M sem fog.)

Ekkor $L(M)$ az első pont miatt tartalmazza a nem kódokat, továbbá, a második miatt azokat a w szavakat, melyekre M_w elfogadja w-t – és pont ez amit akartunk.

(b) Ha $\overline{L_d}$ egy rekurzív nyelv, akkor a komplementere is az, de $L_d \notin$ RE és ezért $L_d \notin$ R.

7. Legyen $L = \{w\#s : w \in L_d$ és M_w nem fogadja el az s szót $\}$. Igaz-e, hogy

   (a) $L \in$ R?

   (b) $L \in$ RE?

Megoldás: Egyik sem igaz. Ehhez vegyük észre, hogy $\{w : w\#w \in L\} = L_d$. Nézzük előbb a (b) pontot.

(b) Tegyük fel, hogy igaz, ami azt jelenti, hogy van olyan M TG, amire $L(M) = L$. Ebből készíthetünk egy, az L_d nyelvet felismerő M_d TG-t:
M_d egy tetszőleges w bemeneten azt csinálja, hogy futtatja az M gépet a $w\#w$ szón. Az M_d pontosan akkor fogadjon el, ha M így tesz. Ekkor $L(M_d) = L_d$, ami azt is jelentené, hogy L_d rekurzívan felsorolható. De tudjuk, hogy $L_d \notin$ RE, tehát $L \notin$ RE.

(a) Ha $L \notin$ RE, akkor $L \notin$ R, hiszen R $\subset$ RE.

8. Igazolja, hogy ha L rekurzív, akkor L^* is rekurzív!

Megoldás: L rekurzív, tehát van olyan 1 szalagos, determinisztikus M TG, ami minden bemeneten megáll és $L(M) = L$.

Egy adott w bemeneten M^*-nak azt kell eldöntenie, hogy w felbontható-e néhány részre úgy, hogy mindegyik rész L-ben legyen. Az M^* TG a következő algoritmust valósítja meg: Ha a bemenet az üres szó, akkor megáll, elfogad. Egyébként egy nem üres w bement esetén sorban veszi ennek a különböző felbontásait (1 részre, 2 részre az összes lehetséges módon, stb., egészen a $|w|$ részre, azaz betűkre bontásig). Az M^* az aktuális felbontás minden darabján egymás után futtatja az M gépet. Ha valamelyik felbontásnál mindegyik futtatás elfogadással zárul, akkor M^* álljon meg elfogadó állapotban. Ha ez egyszer sem történik meg, akkor amikor M^* az eljárás végére ért, álljon meg elutasító állapotban.

Így M^* pontosan akkor fogadja el a w szót, ha $w \in L^*$.

Ezt egyszerűen meg lehet valósítani, pl. egy 4 szalagos TG-vel: A másodikra a szalag elejének megjelölése után lemásoljuk a bemenetet (ez csak azért kell, hogy a bemenetnek egyszerűen vissza tudjunk menni az elejére). A 3. szalag az aktuális felbontás és az aktuálisan vizsgált darab nyilvántartására szolgál. A 4. szalagra másolja át M^* a megfelelő darabot, és ezen futtatja M-et.

Minden w szó esetén véges sok lehetőség van a felbontásra, ezek mindegyikénél véges sok darab van, amikre az M-et futtatni kell. Mivel a feltevés szerint M mindig megáll, ez az egész egy véges eljárás lesz.