7. Turing-gépek, R, RE

1. 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 karaktertól 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 $a = b_1 b_2 \in \{00, 01, 10, 11\}$) 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.

2. 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 $a = a_1 a_2$, $c = c_1 c_2$ 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).

3. 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.

4. 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) $L \in \text{RE}$?

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 $L_d \notin \text{RE}$, tehát $L \notin \text{RE}$.

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

5. 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.

6. Igazolja, hogy ha L rekurzívan felsorolható, akkor L^* is rekurzívan felsorolható!

Megoldás: Az előző, a rekurzív nyelvre vonatkozó megoldással itt az a baj, hogy ha egy szóra M nem áll meg, akkor nem lenne lehetőség a következő felbontást kipróbálni. Ezért a szokásos lépésszámkorlátos futtatást kell bevezetni, például így:

$\ell = 1, 2, \ldots$ esetén nézzük sorban végig az összes lehetséges felbontást (csak véges sok van!), mindegyiknél minden darabon ℓ lépésig futtassuk az M gépet. Ha valamelyik felbontásnál minden darabot elfogad M ezen a lépésszámon belül, akkor M^* álljon meg elfogadó állapotban, különben folytassa az eljárást.

Ekkor, ha $w \in L^*$, akkor van olyan felbontás, és olyan ℓ, hogy az adott felbontás minden darabját az M gép ℓ lépésen belül elfogadja. Ilyen esetben pedig az M^* megáll elfogadó állapotban.

Ha viszont $w \notin L^*$, akkor ez az M^* nem fog megállni, és ezért persze nem is fogad el, ami helyes működés.