ELMÉLETI INFORMATIKA

II. rész

Algoritmus- és kiszámíthatóságelmélet

Számítási modellek 1
Turing-gép

10. előadás

Számítási modellek

ALGORITMUS: *Egy meghatározott cél elérésére irányuló, egymástól elkülönített, mechanikusan elvégezhető műveletek sorozata, amelyek segítségével bizonyos kiindulási állapotból véges számú közbenső állapoton keresztül egy előírt feltételeknek eleget tevő végállapotba jutunk*.

Az algoritmusokat tehát valamilyen **probléma** megoldásának céljából tervezzük, majd valamilyen programozási nyelv segítségével kódoljuk, hogy számítógépen végre tudjuk hajtani őket.

A továbbiakban olyan problémákkal foglalkozunk, amelyeket meg lehet adni matematikai fogalmak segítségével, és amelyek megoldása **számítási modellek** segítségével megvalósítható.

PROBLÉMA:

NÉV: *XY*

INPUT: a probléma megengedett bemenete

OUTPUT: itt van leírva, hogy egy adott bemenetre milyen kimenet várható

10.1 példa:

NÉV: *Két természetes szám összege*

INPUT: x és y természetes számok

OUTPUT: z természetes szám, melyre $z = x + y$

10.1 definíció: (számítási probléma)

A P **számítási probléma** egy $(\mathrm{IN}, \mathrm{OUT}, p)$ elemhármas, ahol IN a megengedett bemenetek halmaza, OUT a kimenetek halmaza, a $p\colon \mathrm{IN} \longrightarrow \mathrm{OUT}$ pedig olyan függvény, amely minden egyes bemenethez a megfelelő kimenetet rendeli.

10.2 definíció: (algoritmussal megoldható probléma)

Az A algoritmus **megoldja** a P =(IN, OUT, p) számítási problémát, ha képes elfogadni az IN halmaz tetszőleges x elemének kódját, s ehhez véges számú lépés után megadni az OUT halmaz egy olyan y elemének kódját, amelyre $y = p(x)$ teljesül.

Léteznek problémák, amelyek algoritmussal nem oldhatók meg!

10.2 példa:

NÉV: *Környezetfüggetlen nyelvek ekvivalenciája*

INPUT: G_1 és G_2 környezetfüggetlen nyelvtanok

OUTPUT: IGEN, ha $L(G_1) = L(G_2)$, ill. NEM, ha $L(G_1) \neq L(G_2)$

10.2 definíció: (algoritmussal megoldható probléma)

Az A algoritmus **megoldja** a P =(IN, OUT, p) számítási problémát, ha képes elfogadni az IN halmaz tetszőleges x elemének kódját, s ehhez véges számú lépés után megadni az OUT halmaz egy olyan y elemének kódját, amelyre $y = p(x)$ teljesül.

Léteznek problémák, amelyek algoritmussal nem oldhatók meg!

10.2 példa:

NÉV: *Környezetfüggetlen nyelvek ekvivalenciája*
INPUT: G_1 és G_2 környezetfüggetlen nyelvtanok
KÉRDÉS: Teljesül, hogy $L(G_1) = L(G_2)$?

10.3 definíció: (algoritmussal eldönthető probléma)

Az A algoritmus **eldönti** a P =(IN, OUT, p) eldönthető problémát, ha képes elfogadni az IN halmaz tetszőleges x elemének kódját, s az ehhez tartozó kérdésre véges számú lépés után megadni az IGEN vagy NEM választ.

Az 1930-as évekig a matematikusok úgy vélték, hogy minden matematikai állítás *igazolható* vagy *cáfolható*. Ez azt jelenti, hogy az adott axiómarendszer axiómáiból vagy az állítást, vagy annak negációját le lehet vezetni.

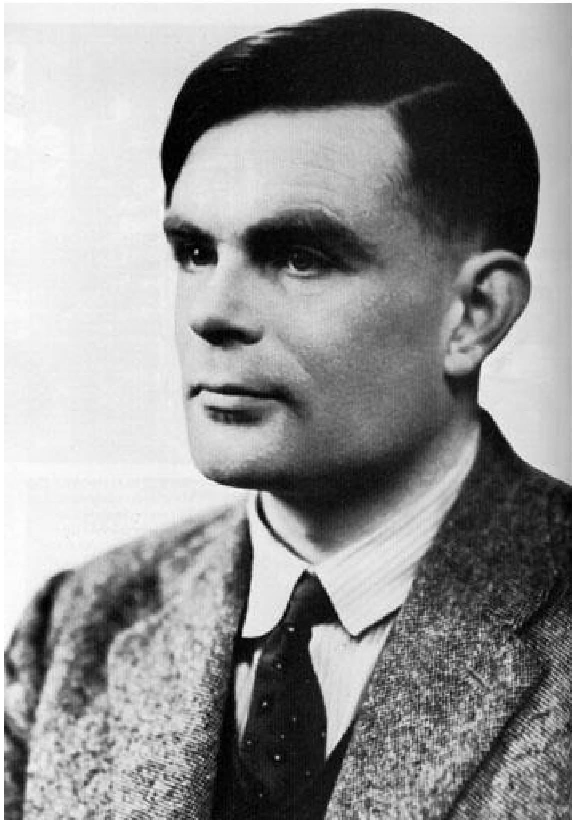
KURT GÖDEL 1931-ben bebizonyította, hogy minden ellentmondás-mentes axiómarendszer, amely tartalmazza a természetes számok Peano-féle axiómarendszerét, **nem teljes** (azaz vannak benne eldönthetetlen problémák).

Rögtön adódik a kérdés: eldönthető-e valami módon egy állításról, hogy érdemes-e vele foglalkoznunk? A kérdés „*Entscheidungs-problem*"-ként (döntési probléma) vonult be a matematika történetébe, s először DAVID HILBERT vetette föl.

ALAN M. TURING 1936-ban nemcsak a fenti problémát oldotta meg, de lefektette az elektronikus számítógépek készítésének elméleti alapjait is.

Turing-gép

A számítások matematikai szempontból „legtisztább” modellje a **Turing-gép**. A fogalmat ALAN M. TURING angol matematikus vezette be 1936-ban, tehát még a program-vezérlésű számítógépek megjelenése előtt.



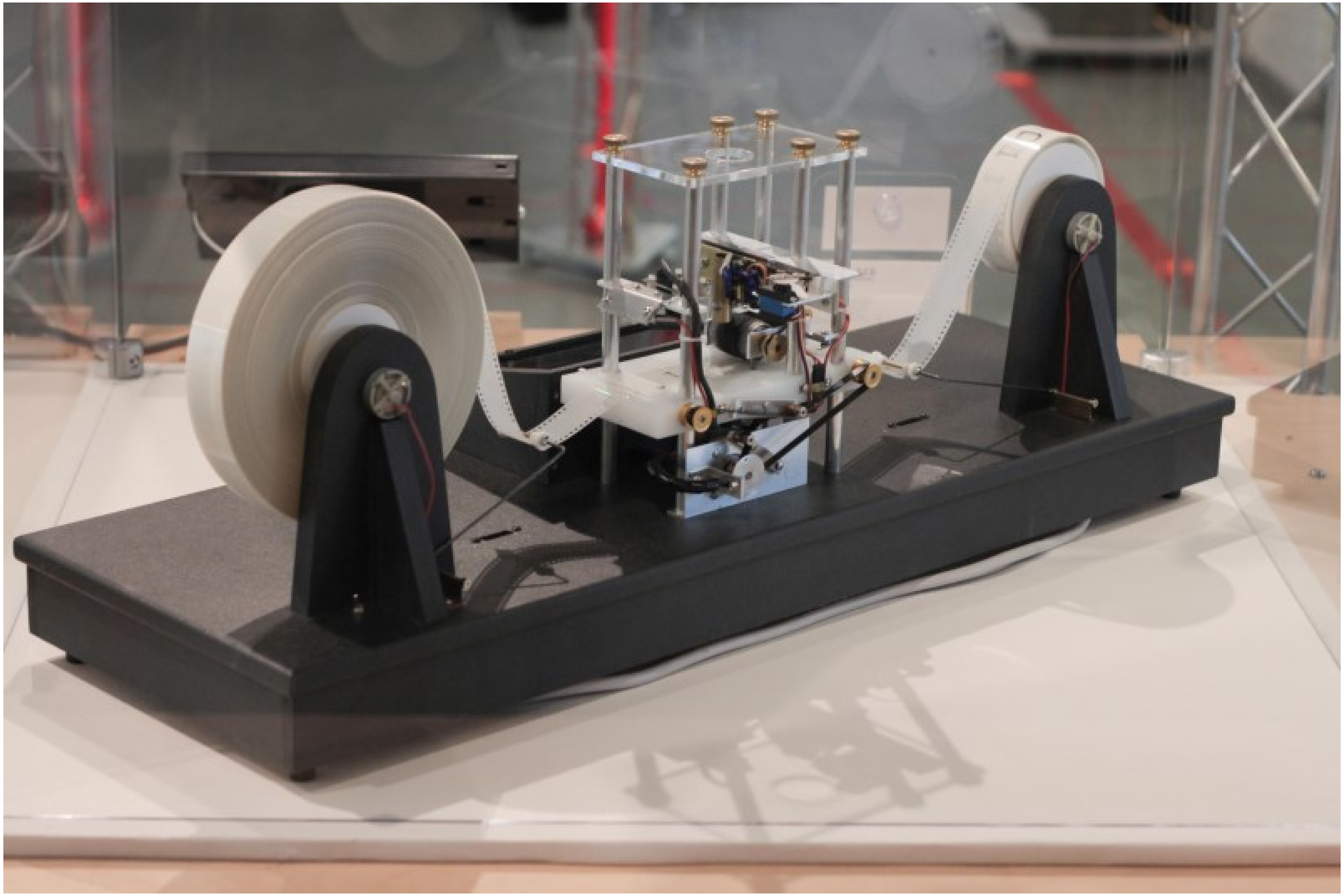
A Turing-gépeken minden olyan számítás elvégezhető, amelyet bármilyen más számítási modellen el lehet végezni.

A Turing-gépek főleg elméleti vizsgálatokban használatosak.

Egy **Turing-gép** a következő részekből áll:

- k darab ($k \geq 1$) **szalag**ból, amelyek mindkét irányban végtelen sok cellára vannak osztva. Minden szalagnak van egy kitüntetett *kezdőcellája*. Minden szalag minden cellájára egy adott véges Γ ábécéből lehet jelet írni. Véges sok cella kivételével ez a jel az ábécé egy speciális „$\square$” jele kell hogy legyen, amely az üres cellát jelöli.
- minden szalaghoz tartozik egy **író-olvasó fej**, amely minden lépésben az adott szalag pontosan egy celláján áll. A fejek mindkét irányban mozgathatók.
- **vezérlőegység**ből, amelynek lehetséges állapotai egy véges halmazt alkotnak. Az állapotok között található egy *kezdőállapot* és egy *végállapot*.

Egy **Turing-gép** a következő részekből áll:



Kiindulási helyzetben a vezérlőegység kezdőállapotban van, az író-olvasó fejek pedig a szalagok kezdőcelláján állnak. Egy lépés során minden fej leolvassa az alatta lévő cellában található jelet, s a vezérlőegység saját állapotától és a beolvasott jelektől függően 3 dolgot csinálhat:

- átmegy egy új állapotba vagy maradhat az aktuális állapotban,
- minden fejnek utasítást ad, hogy az alatta lévő cellában található jelet írja át más jelre,
- minden fejnek utasítást ad, hogy, lépjen egyet jobbra vagy balra, vagy maradjon helyben.

A Turing-gép **megáll**, ha a vezérlőegység végállapotba jut. Előfordulhat az is, hogy az adott bemenetre a gép sohasem áll meg. Ez a jelenség az igazi számítógépeknél előforduló végtelen ciklusnak felel meg.

A Turing-gép **input**ja a szalagokra kiindulási helyzetben írt szavak, miközben feltételezzük, hogy ezek a 0-dik celláktól kezdődően vannak a szalagokra írva.

Célszerű feltételezni, hogy az input szavak a „□” jelet nem tartalmazzák. Ugyancsak feltételezhetjük, hogy a gép működése során valamennyi inputot elolvassa.

A Turing-gép **output**ja megállás után a *vezérlőegység végső állapota* vagy a szalagokon található szavakból álló rendezett k-s. Ez utóbbi esetben gyakran csak egyetlen szóra vagyunk kíváncsiak, ilyenkor output alatt az utolsó szalagon lévő szót értjük (*output szalag*).

10.4 definíció: (k-szalagos Turing-gép)

A k **-szalagos** $M = (Q, \Gamma, \Sigma, \delta, q_0, F)$ **Turing-gép** egy rendezett elemhatos, ahol

Q – a belső **állapotok halmaza**; nem üres véges halmaz

Γ – a **szalagjelek halmaza**; nem üres véges halmaz

Σ – az **input jelek halmaza**; $\Sigma \subseteq \Gamma \setminus \{\square\}$

δ – az **átmenetfüggvény**; $\delta : (Q \setminus F) \times \Gamma^k \longrightarrow Q \times (\Gamma \times \{-, 0, +\})^k$

q_0 – a **kezdőállapot**, $q_0 \in Q$

F – a **végállapotok halmaza**, $F \subseteq Q$

Bizonyos számításoknál csak egy kétértékű döntést várunk a géptől (pl. eldönthető problémák esetén). Ekkor csak azt nézzük, hogy a gép *milyen állapotban állt meg*. Amennyiben a gép F-beli állapotban állt meg, a válasz IGEN, ha pedig nem F-beli állapotban, akkor a válasz NEM.

Ennek alapján az F-beli állapotokat **elfogadó állapotok**nak, a nem F-beli állapotokat pedig **elutasító állapotok**nak nevezzük.

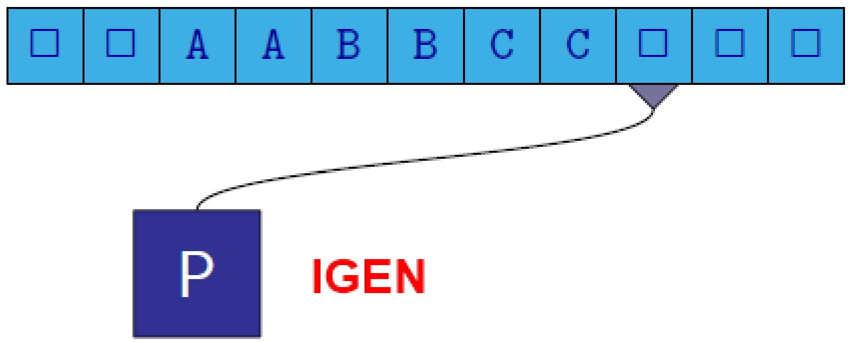
A δ **átmenetfüggvény** a Turing-gép programja, amely megadja, hogy ha a gép q állapotban van, és az i-edik $(1 \leq i \leq k)$ író-olvasó fej alatt lévő szalagjel a_i, akkor mit kell lépni. Ennek megfelelően a $\delta(q, a_1, a_2, \dots, a_k)$ érték $2k+1$ db összetevőből áll: a vezérlőegység lépés utáni q' állapota, az i-edik $(1 \leq i \leq k)$ szalagra írt b_i szalagjel, és az i-edik $(1 \leq i \leq k)$ író-olvasó fej elmozdulása, legfeljebb egy szalagcellával.

A továbbiakban egyszalagos Turing-gépekkel fogunk foglalkozni, ekkor az input szalag megegyezik az output szalaggal.

A Turing-gép egy **konfigurációját** a gép vezérlőegységének aktuális állapota, a szalagon lévő szó és az író-olvasó fej helyzete határozza meg.

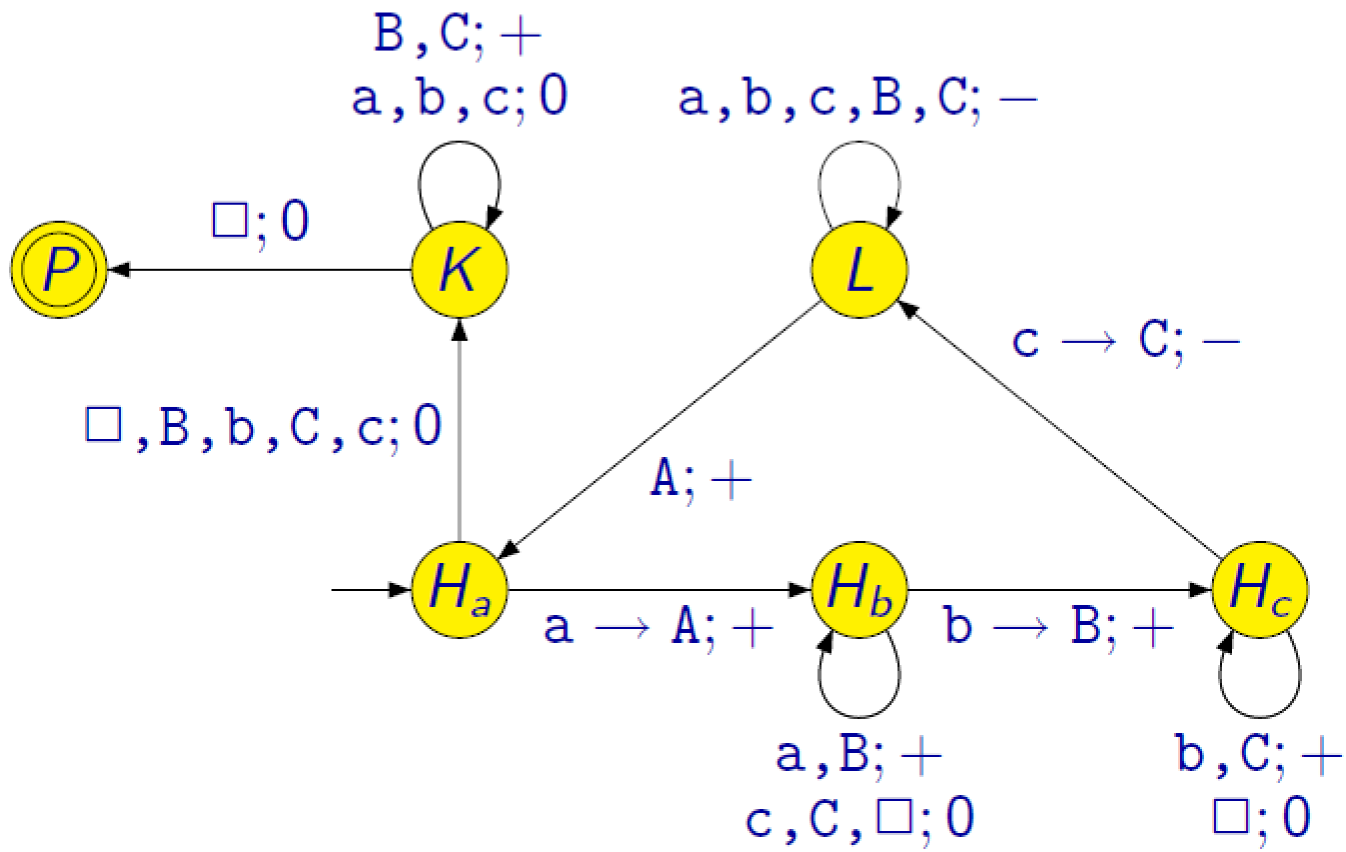
Pl. a $\delta(q_1, a) = (q_2, b, +)$ átmenet jelentése: az egyszalagos Turing-gép vezérlőegysége q_1 állapotban van, a fej a jelet olvas, s ennek hatására a vezérlőegység átmegy q_2 állapotba, a fej az a jelet átírja b jelre, majd a fej egy cellányit jobbra mozog.

10.3 példa: Az alábbi Turing-gép az input szóról eldönti, hogy beletartozik-e az $L = \{a^n b^n c^n \mid n \geq 0\}$ nyelvbe.

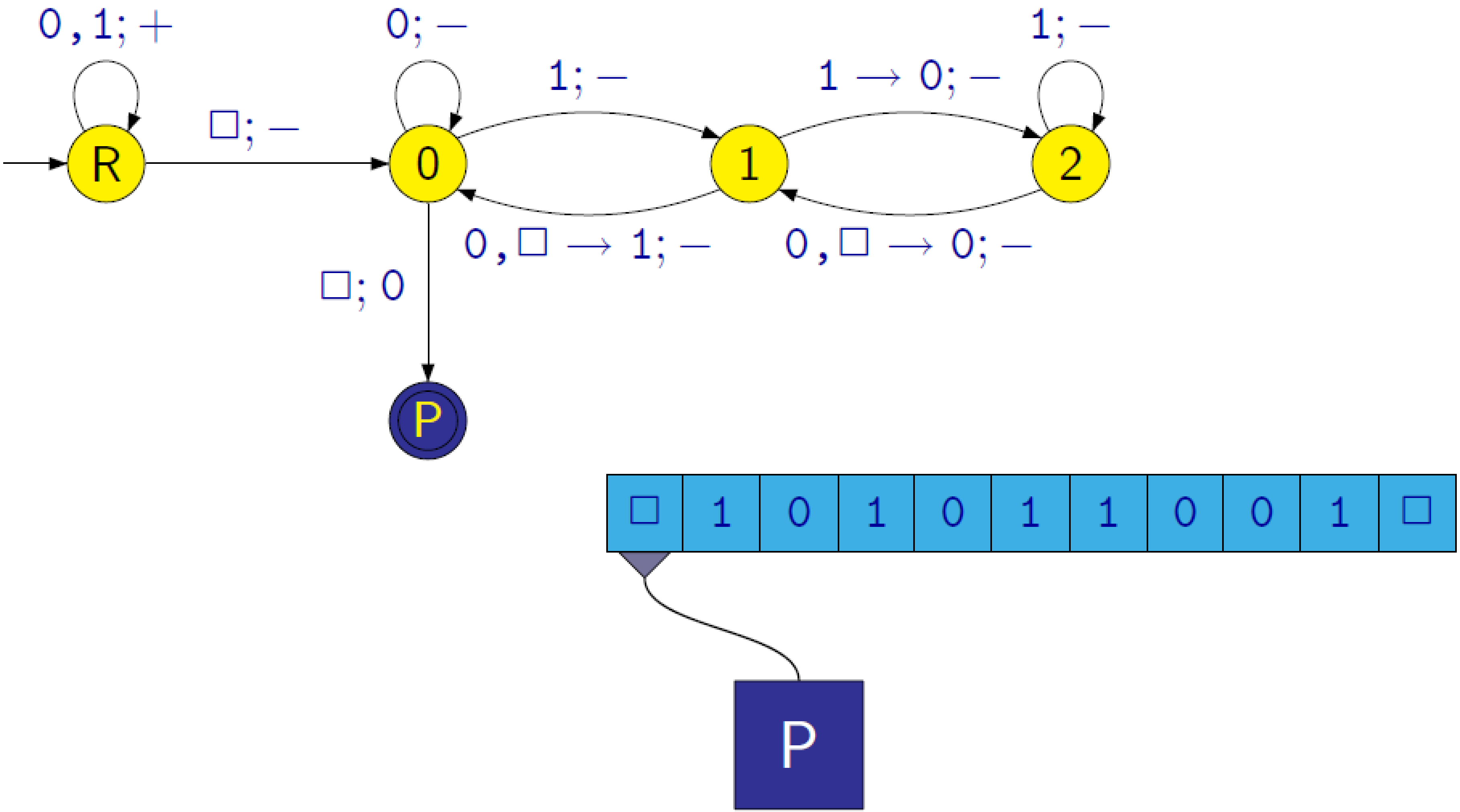


1) Olvas az első **a** jelig, majd átírja **A**-ra. Ha **b**-t vagy **c**-t olvas, akkor végtelen ciklusba kerül. Ha már nincs több **a** jel, akkor megvizsgálja, hogy csak nagybetűk szerepelnek-e.
2) Olvas az első **b** jelig, majd átírja **B**-re. Ha **c**-t vagy □-t olvas, akkor végtelen ciklusba kerül.
3) Olvas az első **c** jelig, majd átírja **C**-re. Ha □-t olvas, akkor végtelen ciklusba kerül.
4) A fejet balra mozgatja, egészen az első **A** jelig.
5) Ugrás az 1) pontra.

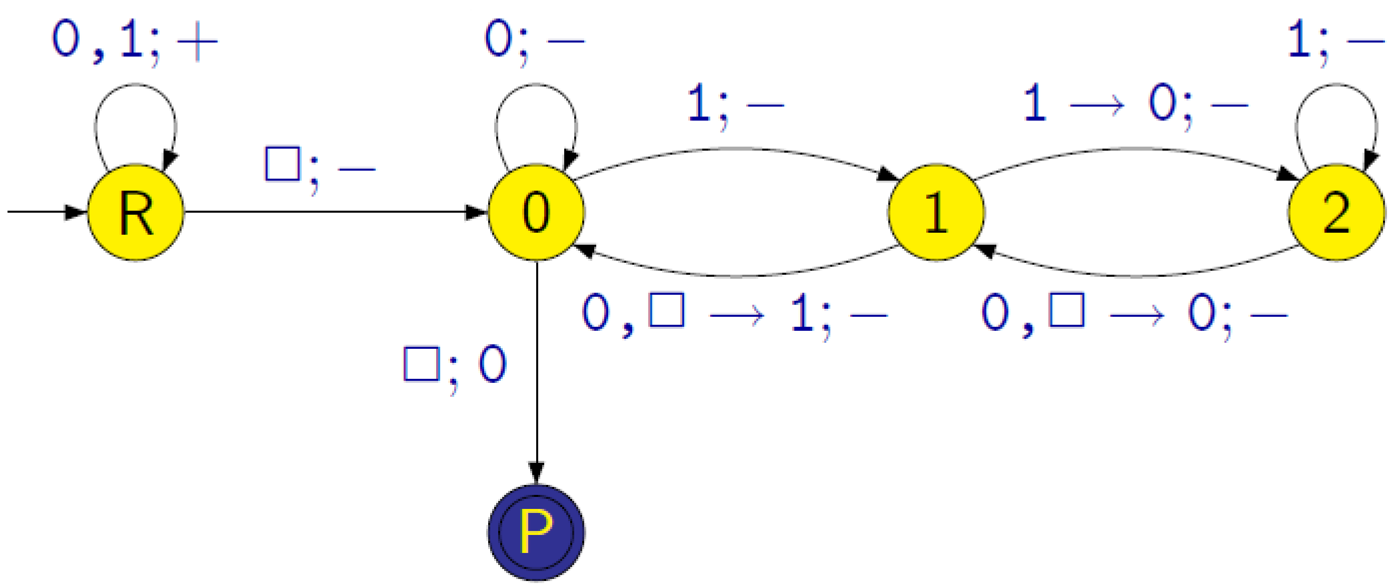
10.3 példa: Az alábbi Turing-gép az input szóról eldönti, hogy beletartozik-e az $L = \{a^n b^n c^n \mid n \geq 0\}$ nyelvbe.



10.4 példa: Az alábbi Turing-gép kiszámítja az input bináris szám háromszorosát.



10.4 példa: Az alábbi Turing-gép kiszámítja az input bináris szám háromszorosát.



$(\mathbf{R}, \square\square\underline{1}110011\square) \vdash (\mathbf{R}, \square\square1\underline{1}10011\square) \vdash (\mathbf{R}, \square\square11\underline{1}0011\square) \vdash (\mathbf{R}, \square\square111\underline{0}011\square) \vdash$
$\vdash (\mathbf{R}, \square\square1110\underline{0}11\square) \vdash (\mathbf{R}, \square\square11100\underline{1}1\square) \vdash (\mathbf{R}, \square\square111001\underline{1}\square) \vdash (\mathbf{R}, \square\square1110011\underline{\square}) \vdash$
$\vdash (\mathbf{0}, \square\square111001\underline{1}\square) \vdash (\mathbf{1}, \square\square11100\underline{1}1\square) \vdash (\mathbf{2}, \square\square1110\underline{0}01\square) \vdash (\mathbf{1}, \square\square111\underline{0}001\square) \vdash$
$\vdash (\mathbf{0}, \square\square11\underline{1}1001\square) \vdash (\mathbf{1}, \square\square1\underline{1}11001\square) \vdash (\mathbf{2}, \square\square\underline{1}011001\square) \vdash (\mathbf{2}, \square\underline{\square}1011001\square) \vdash$
$\vdash (\mathbf{1}, \underline{\square}01011001\square) \vdash (\mathbf{0}, \underline{\square}101011001\square) \vdash (\mathbf{P}, \underline{\square}101011001\square)$

Selye János Egyetem
GIK, Informatika Tanszék

A Turing-gép számításának eredménye

A Turing-gépek *nyelvek felismerésére* és *függvények kiszámítására* használhatók.

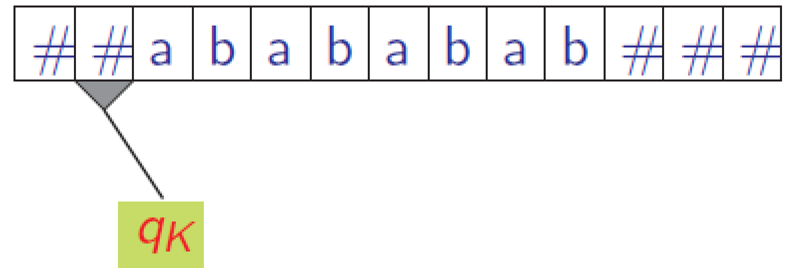
10.5 definíció: (a Turing-gép által felismert nyelv)

Az M **Turing-gép által felismert** $L(M)$ **nyelv** azokból a $w \in \Sigma^*$ szavakból áll, amelyeket inputként megadva a gép végállapotban megáll.

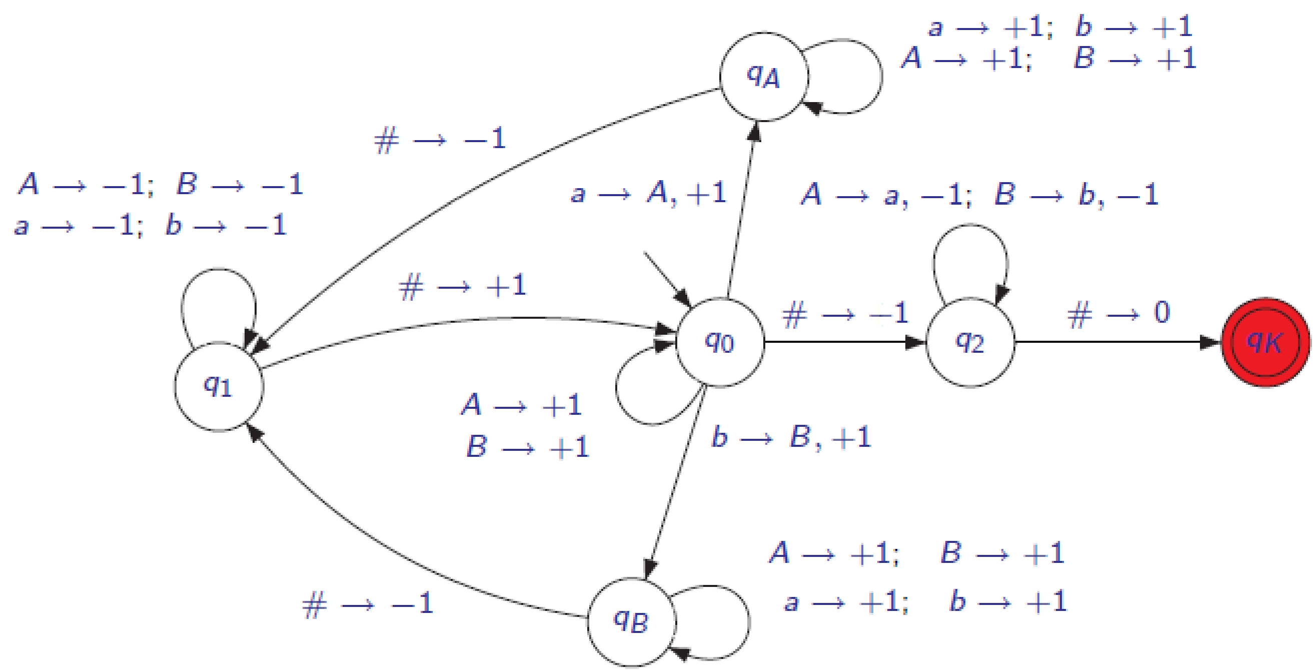
10.6 definíció: (a Turing-gép által kiszámított függvény)

Az M **Turing-gép által kiszámított** $f_M\colon \Sigma^* \longrightarrow \Sigma^*$ **függvény**t így értelmezzük: $f_M(s) = w$, ha az M gép az $s \in \Sigma^*$ input szóval indulva véges számú lépés után megáll, s az output szalagon a $w \in \Sigma^*$ szó fog szerepelni.

10.5 példa: Az alábbi egyszalagos TG megduplázza az input szót.



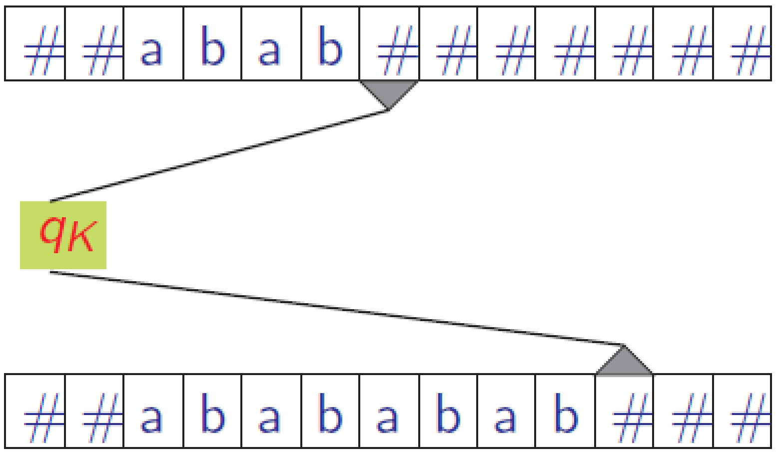
$$\delta(q_0, a) = (q_A, A, +1)$$
$$\delta(q_0, b) = (q_B, B, +1)$$
$$\delta(q_0, A) = (q_0, A, +1)$$
$$\delta(q_0, B) = (q_0, B, +1)$$
$$\delta(q_0, \#) = (q_2, \#, -1)$$
$$\delta(q_1, a) = (q_1, a, -1)$$
$$\delta(q_1, b) = (q_1, b, -1)$$
$$\delta(q_1, A) = (q_1, A, -1)$$
$$\delta(q_1, B) = (q_1, B, -1)$$
$$\delta(q_1, \#) = (q_0, \#, +1)$$
$$\delta(q_A, a) = (q_A, a, +1)$$
$$\delta(q_A, b) = (q_A, b, +1)$$
$$\delta(q_A, A) = (q_A, A, +1)$$
$$\delta(q_A, B) = (q_A, B, +1)$$
$$\delta(q_A, \#) = (q_1, A, -1)$$
$$\delta(q_B, a) = (q_B, a, +1)$$
$$\delta(q_B, b) = (q_B, b, +1)$$
$$\delta(q_B, A) = (q_B, A, +1)$$
$$\delta(q_B, B) = (q_B, B, +1)$$
$$\delta(q_B, \#) = (q_1, B, -1)$$
$$\delta(q_2, A) = (q_2, a, -1)$$
$$\delta(q_2, B) = (q_2, b, -1)$$
$$\delta(q_2, \#) = (q_K, \#, 0)$$

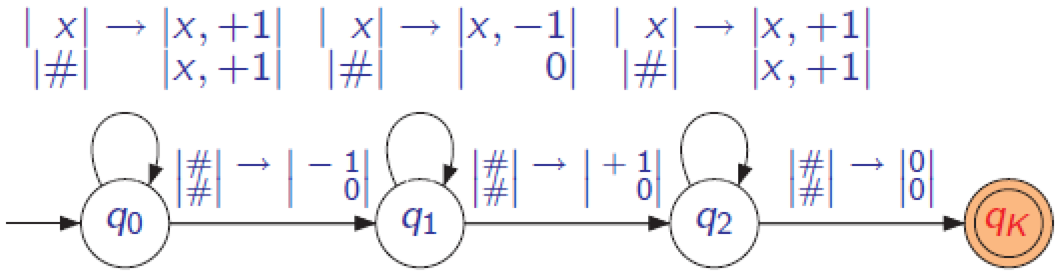


A TG végállapotban van.

10.6 példa: Az alábbi kétszalagos TG megduplázza az input szót.

$$
\begin{aligned}
\delta(q_0, a, \#) &= (q_0, a, +1, a, +1)\\
\delta(q_0, b, \#) &= (q_0, b, +1, b, +1)\\
\delta(q_0, \#, \#) &= (q_1, \#, -1, \#, 0)\\
\delta(q_1, a, \#) &= (q_1, a, -1, \#, 0)\\
\delta(q_1, b, \#) &= (q_1, b, -1, \#, 0)\\
\delta(q_1, \#, \#) &= (q_2, \#, +1, \#, 0)\\
\delta(q_2, a, \#) &= (q_2, a, +1, a, +1)\\
\delta(q_2, b, \#) &= (q_2, b, +1, b, +1)\\
\delta(q_2, \#, \#) &= (q_K, \#, 0, \#, 0)
\end{aligned}
$$





A TG végállapotban van.

A Turing-gép idő és tárigénye

A Turing-gép mint modell egyik nagy előnye, hogy segítségével elég egyszerűen vizsgálható a számítások idő és tárigénye.

Az M Turing-gép **számítási ideje** az s inputon a megállásig végrehajtott lépések száma, **tárigénye** pedig felhasznált szalagcellák száma.

Megjegyzés:

Ha a gép input szalagja csak olvasható, akkor előírhatjuk, hogy ez a szalag ne számítson bele a tárigénybe. Hasonló a helyzet a csak írásra szolgáló output szalaggal, amin sohasem léphetünk bal irányba. Mindez mögött az a megfontolás áll, hogy kizárólag az érdemi munka helyigényét mérjük, és ne foglalkozzunk a bemenet olvasásának és az eredmény írásának elkerülhetetlenül fellépő költségével.

$T_M(n)$ – az M Turing-gép **maximális számítási ideje** az n jelből álló inputon,

$S_M(n)$ – az M Turing-gép **maximális tárigénye** az n jelből álló inputon.

A $T_M(n)$ és $S_M(n)$ függvények segítségével lehetőég nyílik arra, hogy az algoritmusok hatékonyságát összehasonlítsuk.

Megjegyzés:

Ha valamely n hosszúságú w input szó esetén az M Turing-gép végtelen ciklusba kerül, akkor a $T_M(n)$ és $S_M(n)$ függvények értéke végtelen.

10.7 definíció: (időkorlátos Turing-gép)

Legyen $t: \mathbb{N} \longrightarrow \mathbb{N}$ függvény. Az M Turing-gép $t(n)$ **időkorlátos**, ha n jelből álló input szavakon legfeljebb $t(n)$ lépést tesz, vagyis teljesül hogy $T_M(n) \leq t(n)$.

Megjegyzés:

Az M Turing-gép (algoritmus) akkor tekinthető gyorsnak, ha $t(n)$ egy lassan növekvő függvény.

10.8 definíció: (tárkorlátos Turing-gép)

Legyen $s: \mathbb{N} \longrightarrow \mathbb{N}$ függvény. Az M Turing-gép $s(n)$ **tárkorlátos**, ha n jelből álló input szavakon legfeljebb $s(n)$ szalagcellát használ, vagyis teljesül hogy $S_M(n) \leq s(n)$.

10.9 definíció: (k-szalagos nemdeterminisztikus Turing-gép)

A k-**szalagos** $M = (Q, \Gamma, \Sigma, \delta, q_0, F)$ **nemdeterminisztikus Turing-gép** egy rendezett elemhatos, ahol

Q – a belső **állapotok halmaza**; nem üres véges halmaz

Γ – a **szalagjelek halmaza**; nem üres véges halmaz

Σ – az **input jelek halmaza**; $\Sigma \subseteq \Gamma \setminus \{\square\}$

δ – az **átmenetfüggvény**;

$$\delta : (Q \setminus F) \times \Gamma^k \longrightarrow P\big(Q \times (\Gamma \times \{-, 0, +\})^k\big)$$

q_0 – a **kezdőállapot**, $q_0 \in Q$

F – a **végállapotok halmaza**, $F \subseteq Q$

Selye János Egyetem
GIK, Informatika Tanszék

Egy nemdeterminisztikus Turing-gép (NTG) tehát csak annyiban különbözik egy determinisztikus Turing-géptől, hogy minden helyzetben a vezérlőegység állapota és a szalagokhoz tartozó fejek által leolvasott jelek több lehetséges lépést is megengedhetnek.

10.10 definíció: (NTG által felismert nyelv)

Az M **NTG által felismert** $L(M)$ **nyelv** azokból a $w \in \Sigma^*$ szavakból áll, melyekkel ha kezdő konfigurációból elindítjuk az M gépet, akkor létezik legalább egy elfogadó konfigurációban véget érő számítási út.

10.7 példa: Tételezzük fel, hogy az M egyszalagos NTG q állapotban van, a fej alatti szalagjel a, és $\delta(q, a) = \{(q, a, +), (p, b, +), (r, c, -)\}$. Ekkor a gép 3 lehetséges lépés közül választhat.

A determinisztikus Turing-gépek mintájára beszélhetünk időkorlátos ill. tárkorlátos nemdeterminisztikus Turing-gépekről.

10.11 definíció: (időkorlátos nemdeterminisztikus Turing-gép)

Legyen $t\colon \mathbb{N} \longrightarrow \mathbb{N}$ függvény. Az M nemdeterminsztikus Turing-gép $t(n)$ **időkorlátos**, ha n jelből álló input szavakon minden számítási út mentén legfeljebb $t(n)$ lépést tesz.

10.12 definíció: (tárkorlátos nemdeterminisztikus Turing-gép)

Legyen $t\colon \mathbb{N} \longrightarrow \mathbb{N}$ függvény. Az M nemdeterminsztikus Turing-gép $s(n)$ **tárkorlátos**, ha n jelből álló input szavakon minden számítási út mentén legfeljebb $s(n)$ szalagcellát használ.

Univerzális Turing-gépek

Az univerzális Turing-gépek a fordítóprogramok családján belül az *interpreter*ek közé tartoznak, és bármely Turing-gépet képesek szimulálni. Egy U univerzális Turing-gép inputja két részből áll:

- **az interpretálandó program**, ami egy M Turing-gép leírása (az egyszerűség kedvéért feltételezhető, hogy az M gépnek egy szalagja és egy végállapota van, input ábécéje pedig $\Sigma = \{0,1\}$),
- az interpretálandó M Turing-gép tetszőleges $s \in \Sigma^*$ **inputja**.

Az U univerzális Turing-gép értelmezi az M gép leírását (kódját), és lépésről-lépésre szimulálja annak működését az s inputon.

Ehhez tehát az interpretálandó M Turing-gépből olyan adatot kell csinálni, amelyet az univerzális Turing-gép olvasni tud. Arra kell csak ügyelni, hogy minden M gép kódja véges bitsorozat legyen, valamint a kódolás/dekódolás algoritmussal elvégezhető legyen.

A Turing-gép egy lehetséges kódolása

Legyen $M = (Q, \Gamma, \Sigma, \delta, q_0, F)$ egyszalagos Turing-gép az alábbi módon megadva (minden adat természetes szám):

$Q = \{0, 1, 2, \dots, r\}$ $\quad q_0 = 0$

$\Gamma = \{0, 1, 2, \dots, t\}$ $\quad \square = t$

$\Sigma = \{0, 1, 2, \dots, z\}$

$F = \{r\}$

fejmozgatás: $\text{balra} = 0, \text{ jobbra} = 1, \text{ helyben} = 2$

Ekkor az M Turing-gép kódolható pl. a következőképpen:

$$\underbrace{\#\#\#}_{\substack{\text{elválasztók} \\ \notin \Sigma}} \quad \underset{\substack{\uparrow \\ \text{szalagszimbólumok} \\ \text{száma}}}{t} \quad \#\# \quad \underset{\substack{\uparrow \\ \text{az utolsó állapot} \\ \text{sorszáma}}}{r} \quad \#\# \underbrace{q\#a\#q'\#b\#d}_{\substack{\delta(q,a)=(q',b,d) \\ \text{átmenet leírása}}} \#\#q'' \dots \#\#\#$$

Minden szóba jöhető M Turing-gépet egy $w \in \Sigma^*$ szóval írunk le. Tetszőleges w szóhoz legfeljebb egy Turing-gép létezik, amelynek a kódja w. Jelöljük ezt a gépet M_w-vel.

Az M_w Turing-gép ismeretében a w kód algoritmussal kiszámítható.
A w kód ismeretében az M_w Turing-gép jellemzői algoritmussal megkaphatók.

Az U univerzális Turing-gép az alábbi módon működik:

$$U(w\#s) = \begin{cases} M_w(s) & \text{ha } w \text{ Turing-gép kód} \\ \text{reject} & \text{ha } w \text{ nem Turing-gép kód} \end{cases}$$

Selye János Egyetem
GIK, Informatika Tanszék

10.1 tétel: Létezik olyan 3-szalagos U UTG, amelyre teljesül, hogy ha $w, s \in \Sigma^*$ és az M_w TG létezik, akkor az U gép a $w\#s$ inputot pontosan akkor fogadja el (utasítja el), ha az M_w gép az s inputot elfogadja (elutasítja).

Bizonyítás:

Legyen az U UTG-nek három szalagja.

Az U gép **első szalag**ja a $w\#s$ inputot tartalmazza, és csupán a w kód formális ellenőrzésére szolgál.

Az U gép **második szalag**ja az M_w TG egyetlen szalagjának felel meg. Tartalma és a hozzá tartozó fej helyzete az U gép működése során mindig azonos az M_w gép aktuális lépéséhez tartozó szalag-tartalommal és fejpozícióval.

Az U gép **harmadik szalag**ja az M_w TG aktuális állapotát tartalmazza.

A szimuláció megkezdése előtt az U gép ellenőrzi, hogy az M_w gép létezik-e. Ha nem létezik, akkor U megáll elutasító állapotban. Ha létezik, akkor U az s input szót átmásolja a 2. szalagra, a 3. szalagra pedig feljegyzi az M_w gép kezdőállapotának kódját.

$w\#s$
s
0

Az U gép egy lépése: az 1. szalagon w kód meghatározza, hogy az M_w gép mit lépne, ha az U 3. szalagján lévő állapotban van, és a 2. szalaghoz tartozó fej alatti jelet olvassa. Ez alapján meglépi az az M_w gép lépését, majd módosítja a 2. és a 3. szalag tartalmát.

$w\#s$
az M_w gép szalagja az i-edik lépés után
az M_w gép állapota az i-edik lépés után

Az U gép akkor áll meg, ha az M_w gép is megáll. Ez esetben az U gép pontosan akkor fogadja el a $w\#s$ input szót, ha megállás után a 3. szalagon az M_w gép végállapotának kódja van.

Elmondható tehát, hogy az így megalkotott U univerzális Turing-gép megfelel a tétel követelményeinek. ■

Megjegyzés:

Az univerzális Turing-gép létezése azt mutatja, hogy elvileg szerkeszthető olyan számítási eszköz, amely programozható és mindent ki tud számítani, ami kiszámítható.

A gyakorlati megvalósulás felé a következő lépés a Neumann-elv megalkotása volt, amely összhangban van a Turing-géppel. Ezért a Turing-gép méltán tekinthető a számítógépek elméleti modelljének.