

# Kiszámíthatóság elmélet – Történeti áttekintés



**David Hilbert** azt gondolta, hogy **minden probléma megoldható**

- Hilbert 23 problémája (1900, Párizs)

Mit jelent az, hogy egy probléma megoldható?

- Azt hogy van egy olyan algoritmus, ami a probléma **minden bemenetére** kiszámolja a helyes választ

A kiszámíthatóság- és bonyolultságelméletben jellemzően **eldöntési problémákkal** foglalkozunk:

- A bemenetre a válasz **bináris**: igen/nem, true/false, 1/0

Eldöntési probléma például az **ELÉRHETŐSÉG**:

- Bemenet: egy  $G = (V, E)$  irányított gráf, ahol  $V = \{1, \dots, n\}$
- Kérdés: Vajon van-e út  $G$ -ben 1-ből  $n$ -be?

# Kiszámíthatóság elmélet – Eldönthető problémák

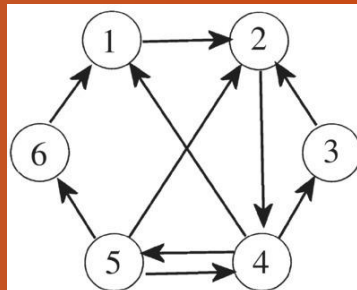
A megoldható eldöntési problémákat **eldönthetőnek** nevezzük

**ELÉRHETŐSÉG** **eldönthető**:

Bemenet:  $G = (V, E)$  irányított gráf, ahol  $V = \{1, \dots, n\}$

Ez egy polinom  
időigényű algoritmus

- Legyen  $S = \{1\}$
- Amíg  $n \notin S$  és  $S \neq \emptyset$ 
  - Vegyünk ki egy  $u$  csúcsot  $S$ -ből;
  - Vegyük be  $S$ -be azon  $u$ -val szomszédos csúcsokat, melyek még nem szerepeltek  $S$ -ben
- Ha  $n \in S$ , akkor kimenet: *igen*, egyébként kimenet: *nem*



$S = \{1\};$   
 $S = \{ \quad \}, u = 1, S = \{2\};$   
 $S = \{ \quad \}, u = 2, S = \{4\};$   
 $S = \{ \quad \}, u = 4, S = \{3,5\};$   
 $S = \{5\}, u = 3, S = \{5\};$   
 $S = \{ \quad \}, u = 5, S = \{6\}$   
Kimenet: *igen*

# Kiszámíthatóság elmélet – Hilbert 10-ik problémája

## Hilbert 10-ik problémája (H10):

- Bemenet: egy  $p$  egész együtthatós polinom,  $x_1, \dots, x_n$  változókkal
- Kérdés: van-e az  $x_1, \dots, x_n$  változóknak olyan egész értékei, melyekre

$$p(x_1, \dots, x_n) = 0?$$

- Például ha  $p = 2x_1 - 3x_1x_2 + 2$ , akkor *igen* a válasz a kérdésre
  - Mert pl.  $p(2,1) = 0$
- De például ha  $p = 2x - 1$ , akkor *nem* a válasz a kérdésre
  - Mert nincs olyan  $i$  egész szám, melyre  $2i - 1 = 0$

## Vajon eldönthető-e H10?

Tudjuk, hogy a  $\mathbb{Z}$ -feletti szám  $n$ -esek felsorolhatók, mondjuk a  $(0, \dots, 0)$ -val kezdve (hogyan?)

# Kiszámíthatóság elmélet – Hilbert 10-es

Tetszőleges  $(i, j, k) \in \mathbb{Z}^3$  esetén legyen  $(i', j', k')$  a számhármaskok felsorolásában az  $(i, j, k)$ -t közvetlenül követő

Tekintsünk a következő **algoritmust**:

Bemenet: egy tetszőleges  $p(x_1, x_2, x_3)$  polinom

- Legyen  $(i, j, k) = (0, 0, 0)$
- Amíg  $p(i, j, k) \neq 0$ 
  - Legyen  $(i, j, k) = (i', j', k')$
- Kimenet: *igen*

**Eldönti ez az algoritmus H10-et?**

- Egyrészt minden „jó” bemeneten **helyes válasszal megáll**
- Másrészt a „rossz” bemeneteken **nem áll meg**
  - Például a  $2x_1 + 0x_2 + 0x_3 - 1$  polinom nem áll meg

Tehát **nem dönti el** a problémát (ahhoz az kellene, hogy minden bemeneten megálljon helyes válasszal)

# Kiszámíthatóság elmélet – Bevezetés

Van **másik algoritmus** ami eldönti a H10 problémát?

- **Nincs** (Matiyasevich, 1970)

Egy tetszőleges  $E$  eldöntési probléma  $B$  bemenetét szokás **pozitívnak** nevezni ha  $B$ -re a válasz igen  $E$ -ben, és **negatívnak**, ha nem a válasz rá

**Félig eldönthetőnek** (vagy **rekurzívan felsorolhatónak**) nevezzük azokat a problémákat, melyekhez van olyan algoritmus, ami **pozitív bemenetekre helyes válasszal megáll**, negatív bemenetekre pedig helyes válasszal megáll vagy nem áll meg

**Eldönthetők** (vagy **rekurzívak**) azok a félig eldönthető problémák, melyekre van olyan algoritmus, ami **minden bemeneten megáll**

- H10 tehát félig eldönthető, de nem eldönthető
- Logikán már láthattunk illet: az **elsőrendű logikai formulák kielégíthetlensége** is félig egy eldönthető, de nem eldönthető probléma

# Kiszámíthatóság elmélet – Eldönthetetlenség

Az, hogy **vannak eldönthetetlen problémák** számossági megfontolások alapján is könnyen belátható:

- Legyen  $\Sigma$  egy ábécé
- $\Sigma$ -feletti formális nyelv **megszámlálhatatlanul végtelen sok** van
- Tetszőleges  $L \subseteq \Sigma^*$  nyelvre a **szóprobléma** is egy eldöntési probléma:
  - Adott egy tetszőleges  $w \in \Sigma^*$  szó; kérdés: vajon  $w \in L$  teljesül-e?
- Tehát az eldöntési problémák **megszámlálhatatlanul végtelen sokan** vannak
- Algoritmus viszont **megszámlálhatóan végtelen sok** van:
  - Minden algoritmus tekinthető egy véges hosszú szónak
  - Tehát az algoritmusok sorba rendezhetők, például lexikografikusan

Ezért ha minden eldöntési problémához hozzárendelünk egy őt eldöntő algoritmust, akkor kell legyen olyan, amelyikhez **nem tudunk** algoritmust rendelni

# Kiszámíthatóság elmélet – Eldönthetetlenség

Hogyan lehet bebizonyítani, hogy egy probléma **nem eldönthető**?

Kell egy **matematikai eszköz**

- Amiről mindenki „elfogadja” hogy amit algoritmikusan el lehet dönteni, azt ezzel az eszközzel is el lehet
- Ilyen eszköz több is van:
  - **Turing-gép** (a véges automaták, veremautomaták általánosítása), Alan Turing 1936
  - **$\lambda$ -kalkulus** (függvények absztrakcióján és alkalmazásán alapuló formális rendszer), A. Church 1936
  - **RAM gép** (egyszerű programozási nyelv, kb. mint a gépi kód)
  - **Általános nyelvtan** (a Chomsky-hierarchia legerősebb nyelvtana), Chomsky 1956

**Church-Turing tézis:** a kiszámíthatóság fenti modelljei mind az effektíven (algoritmikusan) kiszámítható problémák osztályát definiálják (tézis és nem tétel, mert az, hogy „effektíven kiszámítható” egy intuitív fogalom)

# Kiszámíthatóság elmélet – Eldönthetetlenség

A Church-Turing tézis alapján ha egy probléma **nem dönthető el** a fenti kiszámítási modellek egyikével, akkor nem dönthető el semmilyen algoritmussal sem

Az **első problémák**, melyek eldönthetetlensége kiderült:

- A Turing-gépek **megállási problémája**
- $\lambda$ -kalkulusbeli kifejezések ekvivalenciája

Az eldönthetetlenség bizonyítható **visszavezetéssel** is (ezt később tárgyaljuk majd részletesen is):

- Legyen  $K$  és  $L$  két eldöntési probléma, és tegyük fel, hogy  $K$  eldönthetetlen
- Ha van olyan  $A$  algoritmus, ami a  $K$  bemenetéből kiszámolja  $L$  bemenetét úgy, hogy pozitív bemenethez pozitívat, negatívhoz negatívát rendel, akkor  $L$  is eldönthetetlen
- Például vegyünk a következő problémát: adott egy  $F$  elsőrendű formula, vajon **tautológia-e**  $F$  (másképp: **érvényes-e**  $F$ )?
  - Erre a problémára visszavezethető a **kielégíthetlenség** eldöntése (hogyan?), ami eldönthetetlen
  - Így tehát az érvényesség sem lehet eldönthető

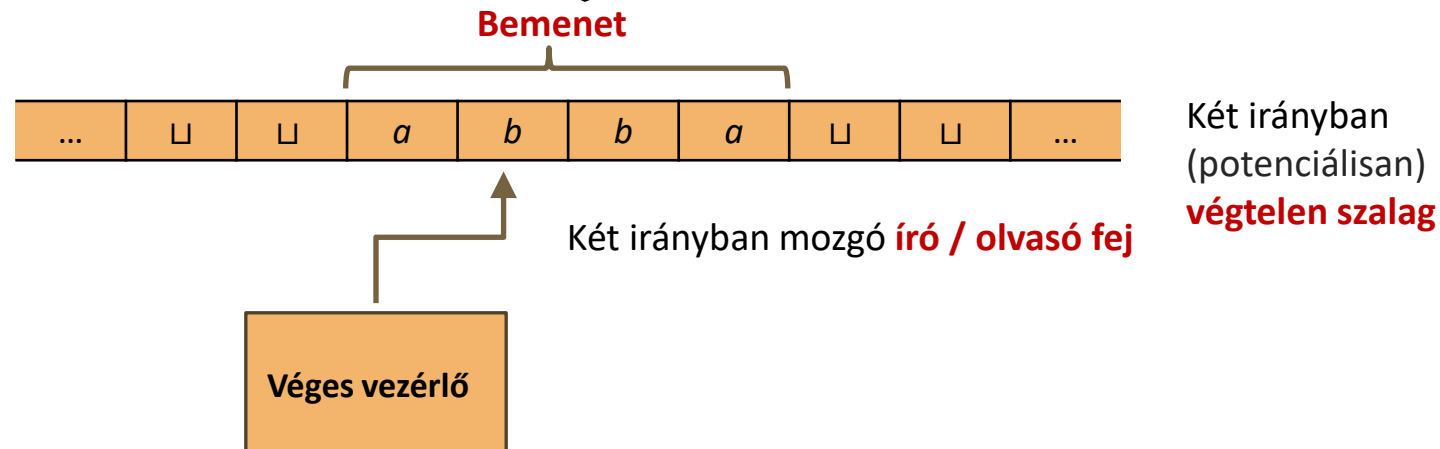


# A Turing-gép

Egy **algorithmus modell** formális nyelvek eldöntésére

Egy  $M = (Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n)$  rendszer, ami egy szalagon lévő szót olvas és ír:

- $\Sigma$  a bemenő jelek ábécéje; a bemenet mindig egy  $\Sigma$  feletti szó
- $\Gamma$  a szalagszimbólumok ábécéje; a szalagon  $\Gamma$ -beli szimbólumok fordulhatnak csak elő,  $\Sigma \subset \Gamma$ ,  $\sqcup \in \Gamma - \Sigma$ ,  
 $\sqcup$ : üres szimbólum, kezdetben ezek vannak a szalagon a bemenet körül



- $Q$  az  $M$  állapothalmaza, véges és nem üres
- Speciális állapotok  $Q$ -ban:  $q_0$  - kezdőállapot,  $q_i$  - elfogadó állapot,  $q_n$  - elutasító állapot
- $\delta$  az  $M$  átmenetfüggvénye, azaz a gép „programja”, ami  $Q' \times \Gamma$ -ból  $Q \times \Gamma \times \{L, R\}$ -be képez, ahol  $Q' = Q - \{q_i, q_n\}$ ,  $L, R$  pedig a fej irányai (Left, Right)

# A Turing-gép

Legyen  $M = (Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n)$  egy Turing-gép

Ha  $(p, b, D) \in \delta(q, a)$ , ahol  $p, q \in Q, a, b \in \Gamma, D \in \{L, R\}$ , akkor

$q \xrightarrow{a/b, D} p$  az  $M$  egy **átmenete**

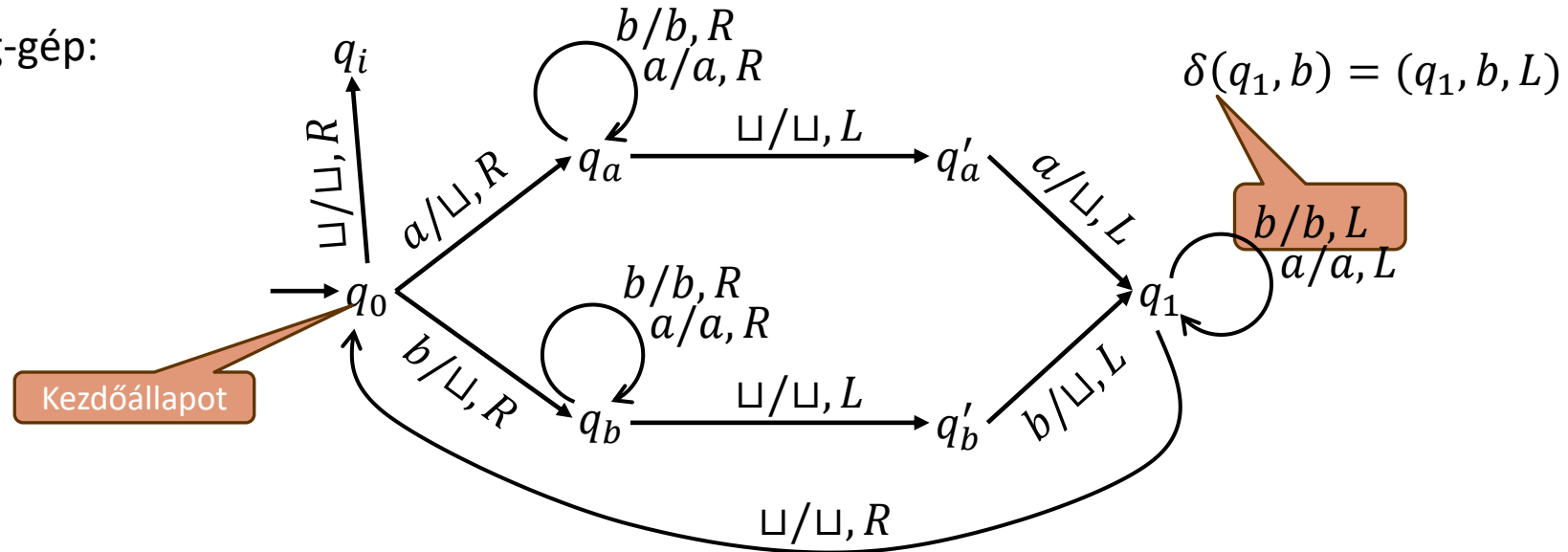
A  $\delta$  most is egyértelműen leírható az **átmeneti diagrammal**

- A csúcsok az állapotok, és két csúcs közte egy  $a/b, D$  hármassal címkézett éllel megfelel egy átmenetnek

$M$  **megadása** átmeneti diagrammal: megjelöljük a kezdő- és végállapotokat

# A Turing-gép – példa

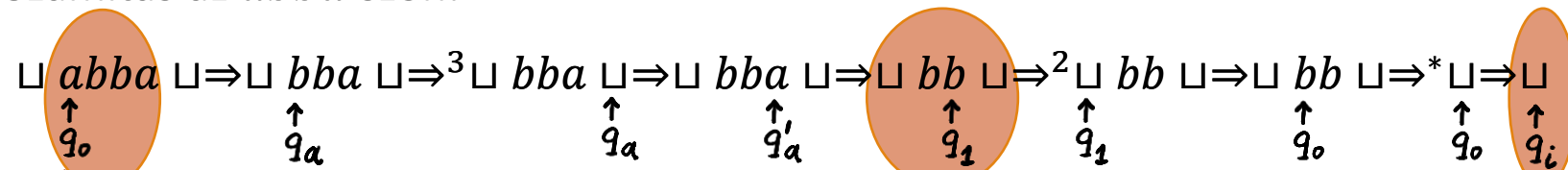
Az  $M_{pal}$  Turing-gép:



A be nem rajzolt átmenetek  $q_n$ -be vezetnek

Bemenő jelek:  $a, b$ , szalagszimbólumok:  $a, b, \square$ , állapotok:  $q_0, q_a, q'_a, q_b, q'_b, q_1, q_i, q_n$

Számítás az *abba* szón:



Kezdőkonfiguráció

Konfiguráció (leírja, hogy mi van a szalagon, hol van a fej, mi az állapot, ld. köv. fólia)

Elfogadó konfiguráció

# A Turing-gép - Konfiguráció

Az  $M = (Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n)$  Turing-gép egy **konfigurációja** egy hármas, ami leírja, hogy

- mi van a szalag nem csak  $\sqcup$ -ot tartalmazó részén,
- hol van a szalagon a fej,
- milyen állapotban van  $M$

Egy **konfiguráció** megadható

- szemléletesen, mint az előző példában
- vagy  **$uqv$  alakban**, ahol  $u \in \Gamma^*$ ,  $v \in \Gamma^+$ ,  $uv$  a szalagon lévő szó (tőle balra és jobbra csak  $\sqcup$  van),  $q$  az  $M$  aktuális állapota, a fej pedig  $v$  első betűjére mutat

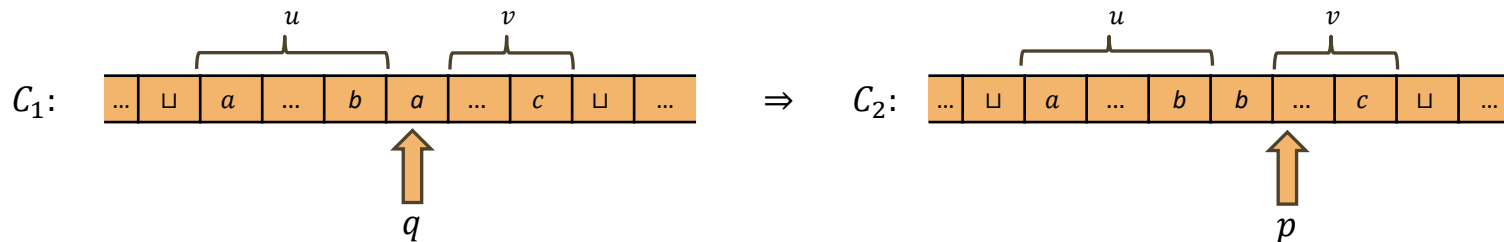
Egy  $uqv$  konfiguráció

- **Kezdőkonfiguráció**, ha  $v \in \Sigma^+ \cup \{\sqcup\}$ ,  $q = q_0$  és  $u = \varepsilon$
- **Elfogadó konfiguráció**, ha  $q = q_i$
- **Elutasító konfiguráció**, ha  $q = q_n$
- Az elfogadó és elutasító konfigurációkat **megállási** konfigurációknak is nevezzük

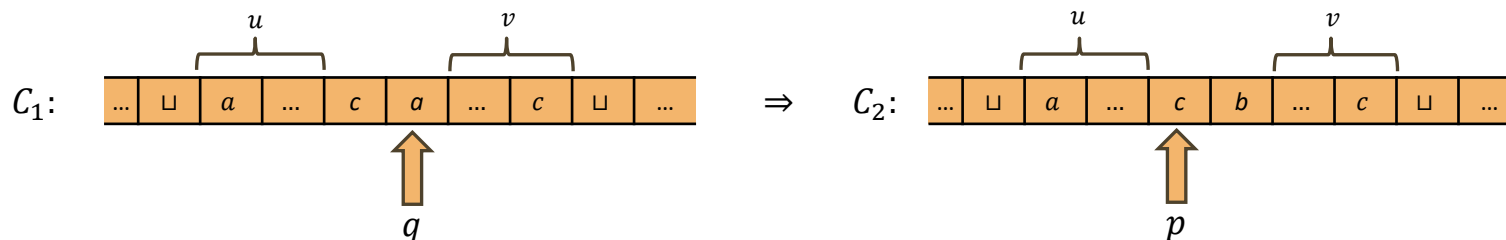
# A Turing-gép – Konfigurációátmenet

Legyen  $C_1$  és  $C_2$  az  $M = (Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n)$  Turing-gép két konfigurációja

- $C_2$  **közvetlenül elérhető**  $C_1$ -ből, jele  $C_1 \Rightarrow C_2$ , ha az alábbiak egyike teljesül:
  - $C_1 = uqav, C_2 = ubpv$  és  $\delta(q, a) = (p, b, R)$  ahol  $a, b \in \Gamma, u \in \Gamma^*, v \in \Gamma^+$



- $C_1 = ucqav, C_2 = upcbv$  és  $\delta(q, a) = (p, b, L)$ , ahol  $a, b, c \in \Gamma, u, v \in \Gamma^*$



# A Turing-gép – Konfigurációátmenet

## Többlépéses konfiguráció átmenet:

- Legyen  $C, C'$  az  $M$  két konfigurációja
- $C'$  **elérhető**  $C$ -ből, jele  $C \Rightarrow^* C'$ , ha
  - vannak olyan  $C_1, \dots, C_n$  konfigurációk ( $n \geq 1$ ) hogy
    - $C = C_1 \Rightarrow C_2 \Rightarrow \dots C_{n-1} \Rightarrow C_n = C'$

$M$  **felismeri/elfogadja** az  $u \in \Sigma^*$  szót, ha az  $q_0u$  kezdőkonfigurációból elérhető egy elfogadó konfiguráció

**Az  $M$  által felismert nyelv:**

$$L(M) := \{ u \in \Sigma^* \mid M \text{ felismeri az } u\text{-t} \}$$

Tekintsük a korábbi  $M_{pal}$  Turing-gépet:

- Láttuk, hogy  $M_{pal}$  a  $q_0aabb$  kezdőkonfigurációból indulva elér egy elfogadási konfigurációt, tehát elfogadja az  $aabb$  bemenetet
- $L(M_{pal}) = \{uu^{-1} \mid u \in \{a,b\}^*\}$

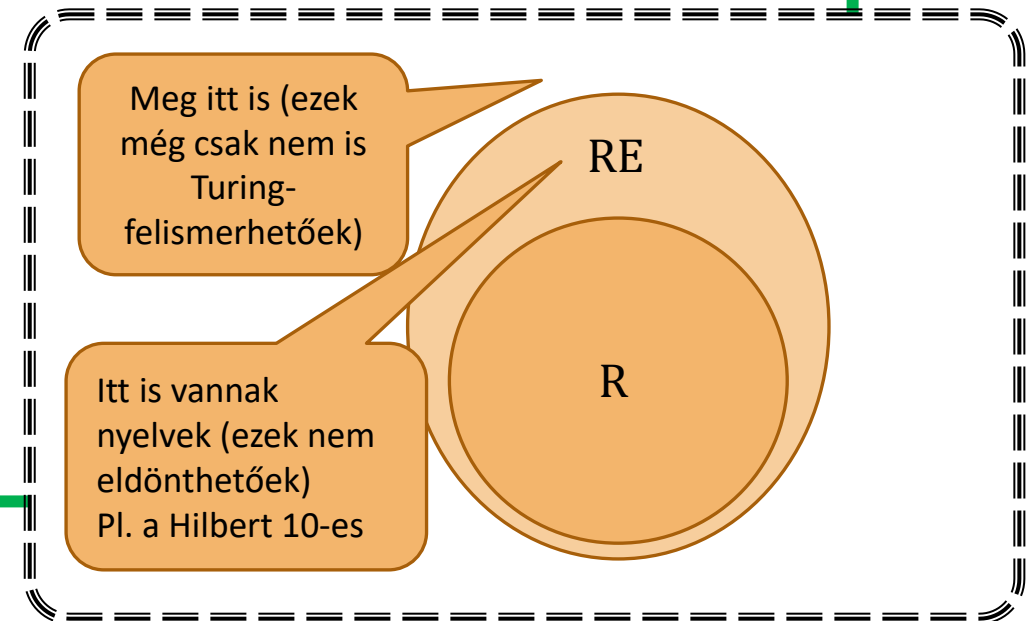
# Turing-felismerhető nyelvek

Egy  $L$  nyelv **Turing-felismerhető**, ha van olyan  $M$  Turing gép, hogy  $L=L(M)$

- Ha  $M$  minden bemeneten meg is áll, akkor  $L$  **eldönthető**

A Turing-felismerhető nyelvek ugyanazok, mint a **rekurzívan felsorolható** nyelveknek

- Ezen nyelvek osztályát RE-vel jelöljük
- Az eldönthető nyelvek osztályát pedig R-rel jelöljük



# Aszimptotikus jelölések

Legyen  $f, g: \mathbb{N} \rightarrow \mathbb{N}^+$  függvények

- Az  $f$  **legfeljebb olyan gyorsan nő mint a  $g$**  (jele:  $f(n) = O(g(n))$ ), ha létezik olyan  $c > 0$ , hogy  $f(n) \leq c \cdot g(n)$  majdnem minden  $n$ -re
- $f(n) = \Omega(g(n))$ , ha  $g(n) = O(f(n))$
- $f(n) = \Theta(g(n))$ , ha  $f(n) = O(g(n))$  és  $g(n) = O(f(n))$

Ha  $f(n) = O(g(n))$ , de  $g(n) \neq O(f(n))$ , akkor ennek jele  $f(n) = o(g(n))$

Ha  $p(n)$  és  $q(n)$  polinomok (pozitív főegyütthatóval), akkor

- $p(n) = O(q(n))$  pontosan akkor, ha  $p$  fokszáma legfeljebb akkora, mint  $q$ -é
- $p(n) = \Theta(q(n))$  pontosan akkor, ha fokszámaik megegyeznek

Legyen  $a > 1$  egész szám. Akkor  $p(n) = o(a^n)$

Ha  $c \in \mathbb{N}$ , akkor minden  $p(n)$  pozitív főegyütthatójú polinomra

- $\log n^c = O(\log n)$  és  $\log n = o(p(n))$ ,
- $p(n) = o(a^{\log n}) = o(a^n)$  minden  $a > 1$  egész szám esetén.



# Turing-gép – Időigény

Legyen  $f: \mathbb{N} \rightarrow \mathbb{N}$  egy függvény,  $M$  pedig egy Turing-gép

- $M$  **időigénye**  $f(n)$ , ha  $M$  minden  $n$  hosszú bemeneten legfeljebb  $f(n)$  lépésben megáll

Határozzuk meg  $M_{pal}$  által felismert nyelvet és az időigényét!

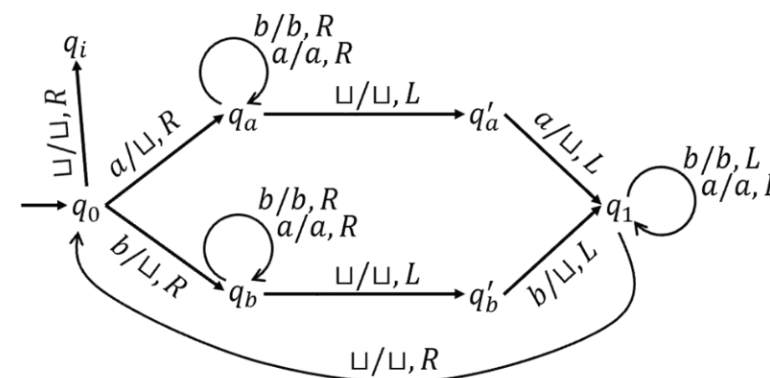
Az  $M_{pal}$  **működése**:

Amíg  $q_0$ -ban nem  $\sqcup$ -ra mutat a fej

- Letöröl egy betűt a szalagon lévő szó elejéről és az új állapotban megjegyzi ezt a betűt
- Elmegy a szó végére (ezt  $\sqcup$  jelzi)
- Visszalép egyet (a fej pozíciója a szó utolsó betűje)
- Ha a fej pozícióján a szó elején letörölt betű van, akkor letörli az utolsó betűt, és az új állapot  $q_1$ , egyébként elutasítja a bemenetet és megáll
- $q_1$ -ben visszamegy a szó elejére (ezt  $\sqcup$  jelzi) és  $q_0$ -ba lép

Elfogadja a bemenetet

A **felismert nyelv**:  $L(M_{pal}) = \{uu^{-1} \mid u \in \{a, b\}^*\}$



- Az **időigény**: Legyen a bemenő szó  $u$ , a hossza  $n$ .
- $M_{pal}$   $u$ -n akkor lép a legtöbbet, ha elfogadja
- Ekkor a ciklusmag egyszeri végrehatása  $O(n)$  lépés
- A ciklusmag legfeljebb  $O(n)$ -szer hajtódik végre
- $M_{pal}$  időigénye  $O(n^2)$

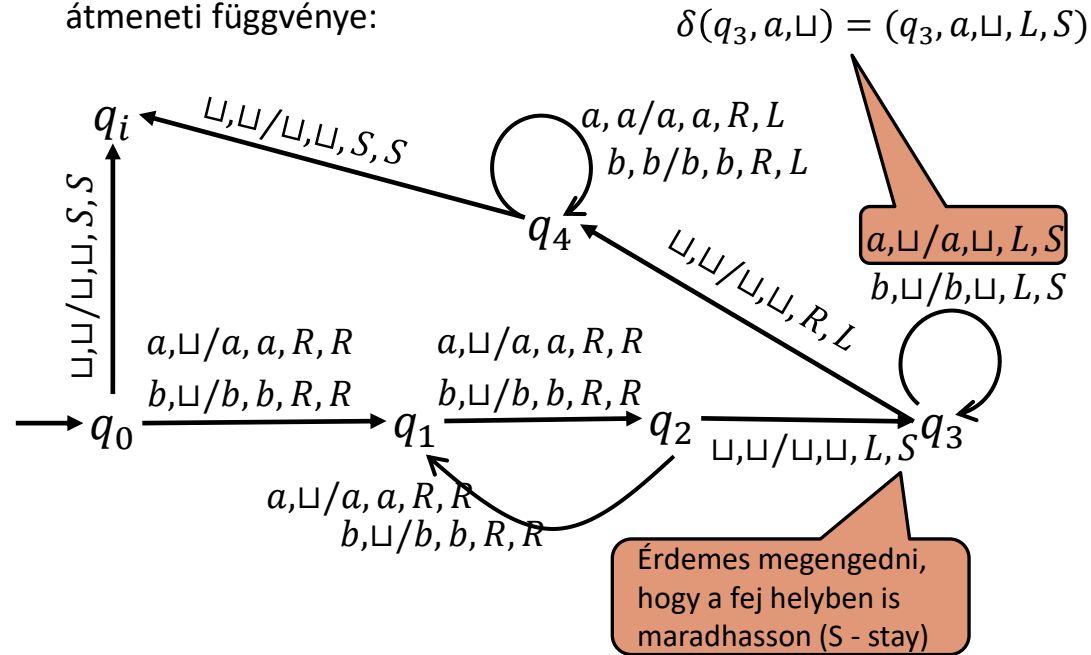
# Többszalagos Turing-gép

## Többszalagos Turing-gép

- Turing-gép  $k$  (konstans) számú szalaggal
- A szalagok mindegyike rendelkezik egy független író / olvasó fejjel
- A bemenet az első szalagra kerül, a többi szalag üres
- Az átmeneti függvény:
$$\delta: (Q - \{q_i, q_n\}) \times \Gamma^k \rightarrow Q \times \Gamma^k \times \{L, R\}^k$$
- A további komponensek mint az egyszalagos esetben
- Az egyszalagosra tanult definíciók kiterjeszthetők a többszalagos esetre (átmenet, konfiguráció-átmenet, felismert nyelv, időigény, stb.)
- Az átmeneti függvény és a gép most is megadható az átmeneti diagrammal

# Többszalagos Turing-gép – Példa

Az  $M_{pal2}$  kétszalagos Turing-gép  
átmeneti függvénye:



- A be nem rajzolt átmenetek  $q_n$ -be vezetnek
- bemenő jelek:  $a, b$
- A **felismert nyelv**:  

$$L(M_{pal2}) = \{uu^{-1} \mid u \in \{a, b\}^*\}$$
- **Időigény**:  $O(n)$

$M_{pal2}$  működése:

- Ha üres a bemenet, akkor elfogad, egyébként átmásol 1 betűt a 2-es szalagra és  $q_1$ -be lép
- Ha a fej  $\sqcup$ -ra mutat, akkor elutasít, egyébként átmásol egy betűt a 2-es szalagra és  $q_2$ -be lép
- Amíg van betű a 2-es szalagon
  - Átmásol két betűt a 2-es szalagra (ha csak egy betű van, akkor elutasít)
- $q_3$ -ban az első szalagon a szó elejére megy, majd  $q_4$ -be lép
- $q_4$ -ben az első szalagon jobbra, a másodikon balra lépve összehasonlítja a két szalagon lévő szót
- Ha megegyeznek, akkor elfogad, egyébként elutasít

$O(n)$   
lépés

$O(n)$   
lépés

$O(n)$   
lépés