

# Bonyolultságelmélet

Ésik Zoltán

SZTE Informatikai Tanszékcsoport  
Számítástudomány Alapjai Tanszék

# A kiszámíthatóság elméletének kialakulása

1900: Hilbert 10. problémája

Adott  $f(x_1, \dots, x_n) = g(x_1, \dots, x_n)$  **diofantikus egyenlet**, ahol  $f$  és  $g$  egész együtthatós polinomok, létezik-e (egész értékű) megoldása.

1971: Matijasevič

Hilbert 10. problémája algoritmikusan **megoldhatatlan**.

1928: Hilbert

Találjunk olyan algoritmust, amellyel a **predikátumkalkulus** (függvénykalkulus) tetszőleges kijelentéséről eldönthetjük, hogy **érvényes**-e.

1930-as évek közepe: Church, Turing

Ilyen algoritmus **nem létezik**.

# A kiszámíthatóság elméletének kialakulása

- ▶ 1931, Gödel: Primitív rekurzív függvények
- ▶ 1934, Gödel: Általános rekurzív függvények
- ▶ 1930-as évek eleje, Church, Kleene, Rosser (Princeton):  
 $\lambda$ -definiálhatóság

1935: AMS New York-i összejövedele

**Church tézise:** Az általános rekurzív függvények (matematikai fogalom) megfelelnek az algoritmikusan kiszámítható függvény fogalmának (intuitív fogalom).

1936: Kleene

Az általános rekurzív függvények megegyeznek a  $\lambda$ -definiálható függvényekkel.

# A kiszámíthatóság elméletének kialakulása

1936: Turing

Bebizonyítja, hogy a Turing-gép ekvivalens a  $\lambda$ -definiálhatósággal, és megfogalmazza azt a tézist, hogy a **Turing-gép megfelel az algoritmussal kiszámítható függvényeknek.**

A **Church-Turing tézist** tapasztalati tények és matematikai eredmények támasztják alá:

- ▶ Minden intuitív értelemben megoldható problémáról sikerült kimutatni, hogy azok megoldhatók a matematikai modellekben is.
- ▶ A matematikai modellek ekvivalensek.

# Kiszámíthatóság és bonyolultság

## Kiszámíthatóságelmélet

Melyek az **algoritmikusan** megoldható problémák?

## Bonyolultságelmélet

Melyek a **gyakorlatilag** megoldható problémák, erőforrásigény.

1971: Cook

**P** és **NP** osztályok, **NP**-teljesség, SAT **NP**-teljes

1972: Karp

Rámutat az **NP**-teljes problémák változatosságára.

1973: Levin

Több kombinatorikus probléma „univerzális a kimerítő keresésre”.

# További bonyolultsági osztályok és számítási módok

- ▶ **L, NL, PSPACE, EXP, ...**
- ▶ Valószínűségi modellek
- ▶ Párhuzamos számítás

stb.

# Polinom időben megoldható problémák

## ELÉRHETŐSÉG

- ▶ Adott: egy  $G = (V, E)$  irányított gráf.  
Feltehetjük, hogy  $V = \{1, 2, \dots, n\}$ .
- ▶ Kérdés: létezik-e 1-ből  $n$ -be vezető irányított út?
- ▶ Módszer:
  1.  $S := \{1\}$ .
  2. Jelöljük meg az 1 csúcsot.
  3. Amíg  $S$  nem üres:
    - 3.1 Választunk egy  $i \in S$  csúcsot.
    - 3.2  $S := S - \{i\}$ .
    - 3.3  $j = 1..n$ :  
Ha  $(i, j) \in E$  és  $j$  nem megjelölt,  
akkor  $S := S \cup \{j\}$  és jelöljük meg  $j$ -t.
- ▶ Ha  $n$  megjelölt csúcs, adjunk igen választ,  
különben adjunk nem választ.

# Polinom időben megoldható problémák

## Néhány kimaradt részlet

- ▶ A bemenet megadása – pl. szomszédsági mátrix (Függ a modelltől, de lényeges szerepet nem játszik.)
- ▶  $S$  reprezentálása
  - ▶ sor: szélességi keresés
  - ▶ verem: egyfajta mélységi keresés

## Hatékonyság

- ▶ A mátrix minden elemét legfeljebb egyszer használjuk fel.
- ▶ Ha az egyszerű műveletek ( $S$  egy elemének kiválasztása, megjelölése, stb.) konstans időigényűek, akkor  $\mathcal{O}(n^2)$ .

Az ELÉRHETŐSÉG egy **eldöntési probléma**.



# Polinom időben megoldható problémák

## MAXIMÁLIS FOLYAM

- ▶ Adott: egy  $(V, E, s, t, c)$  **hálózat**, ahol
  - ▶  $(V, E)$  egy irányított gráf,
  - ▶  $s \in V$  **forrás**,  $t \in V$  **nyelő**,  $t$  elérhető  $s$ -ből
  - ▶  $(u, v) \in E \Rightarrow c(u, v) \in \mathbb{N}^+$  **kapacitás**
- ▶ Kérdés: mekkora a **legnagyobb értékű folyam**?

**Folyam:**  $f : E \rightarrow \mathbb{N}_0$  leképezés, amire:

- ▶  $(u, v) \in E \Rightarrow 0 \leq f(u, v) \leq c(u, v)$ ;
- ▶  $v \notin \{s, t\} \Rightarrow \sum_{(u,v) \in E} f(u, v) = \sum_{(v,u) \in E} f(v, u)$ .

A folyam **értéke:**

$$\sum_{(s,u) \in E} f(s, u).$$

# Segédhálózat

## Állítás

Akkor és csak akkor létezik az  $f$  folyamnál nagyobb értékű  $f'$  folyam, ha az alábbi  $N(f)$  hálózatban  $t$  elérhető  $s$ -ből:

$$\begin{aligned}N(f) &= (V, E', s, t, c') \\ E' &= (E - \{(u, v) : f(u, v) = c(u, v)\}) \cup \\ &\quad \{(v, u) : (u, v) \in E, f(u, v) > 0\} \\ c'(u, v) &= \begin{cases} c(u, v) - f(u, v) & \text{ha } (u, v) \in E \cap E'; \\ f(v, u) & \text{különben.} \end{cases}\end{aligned}$$

( $N(f)$  tartalmazhat ellentétes élpárokat.)

# Javító út

## Az $f$ folyam javítása

Tegyük fel, hogy  $N(f)$ -ben adott egy  $s$ -ből  $t$ -be vezető út.

Legyen  $d$  az úton előforduló minimális élkapacitás.

Minden  $(u, v)$  élre legyen

$$f'(u, v) = \begin{cases} f(u, v) + d & \text{ha } (u, v) \in E \cap E' \text{ előfordul az úton} \\ f(u, v) - d & \text{ha } (v, u) \notin E \text{ előfordul az úton} \\ f(u, v) & \text{különben.} \end{cases}$$

Ekkor az  $f'$  az  $f$ -nél  $d$ -vel nagyobb értékű folyam.

## Módszer

1. Kezdetben  $f$  legyen az azonosan nulla folyam.
2. Készítsük el az  $N(f)$  hálózatot.
3. Ha  $N(f)$ -ben  $t$  nem érhető el  $s$ -ből, akkor  $f$  maximális.
4. Különben egy  $s$ -ből  $t$ -be vezető úthoz és annak minimális kapacitású éléhez készítsük el az  $f'$  folyamot, legyen  $f := f'$  és menjünk vissza a 2. pontra.

## Hatékonyság

- ▶  $C := \max\{c(u, v) : (u, v) \in E\}$ .
- ▶  $\mathcal{O}(nC)$  lépés, lépésenként  $\mathcal{O}(n^2)$  idő:  $\mathcal{O}(n^3C)$ .  
Ez **NEM POLINOMIÁLIS**. ( $C$  miatt.)
- ▶ Az útkeresés szélességi változatával:  $\mathcal{O}(n^5)$ .

A MAXIMÁLIS FOLYAM optimalizálási probléma.  
Eldöntési változata: **MAXIMÁLIS FOLYAM(E)**.

# Az utazóügynök probléma

## TSP

- ▶ **Adott:** az  $1, 2, \dots, n$  városok és bármely két  $i, j$  város távolsága:  $d_{i,j}$ .
- ▶ **Kérdés:** találjunk egy, az összes várost pontosan egyszer érintő legrövidebb körutat.  
Eldöntési változata: **TSP(E)**.
- ▶ **Triviális módszer:** az összes  $\frac{(n-1)!}{2}$  lehetséges körút megvizsgálásával.

**Nem ismert**, hogy TSP megoldható-e polinom idejű algoritmussal.  
(**NP**-teljes probléma.)

## Definíció

Legyenek  $f, g : \mathbb{N} \rightarrow \mathbb{N}$  függvények.

- ▶  $f(n) = \mathcal{O}(g(n))$ , ha létezik olyan  $c > 0$ , hogy  $f(n) \leq c \cdot g(n)$  majdnem minden  $n$ -re.
- ▶  $f(n) = \Omega(g(n))$ , ha  $g(n) = \mathcal{O}(f(n))$ .
- ▶  $f(n) = \Theta(g(n))$ , ha  $f(n) = \mathcal{O}(g(n))$  és  $g(n) = \mathcal{O}(f(n))$ .

## Példa

$$p(n) = a_0 n^k + a_1 n^{k-1} + \dots + a_k$$
$$q(n) = b_0 n^\ell + b_1 n^{\ell-1} + \dots + b_\ell$$

ahol  $a_i, b_j \in \mathbb{N}$ ,  $a_0, b_0 > 0$ . Ekkor:

- ▶  $p(n) = \mathcal{O}(q(n)) \Leftrightarrow k \leq \ell$ ;
- ▶  $p(n) = \Theta(q(n)) \Leftrightarrow k = \ell$ .

Legyen  $a \in \mathbb{N}$ ,  $a > 1$ . Ekkor  $p(n) = \mathcal{O}(a^n)$ , de  $a^n \neq \mathcal{O}(p(n))$ .

# A **P** osztály

## Definíció

Egy probléma a **P osztályba** esik, ha megoldható polinom időigényű (vagy  $\mathcal{O}(n^k)$  időigényű) algoritmussal.

## A polinomiális tézis

- ▶ Egy algoritmus akkor ad gyakorlatilag kielégítő megoldást egy problémára, ha polinom időigényű.
- ▶ Egy problémát akkor tekintünk **gyakorlatilag megoldhatónak**, ha a **P** osztályba esik.

## Néhány ellenvetés

- ▶ Elfogadható-e a gyakorlatban egy  $\mathcal{O}(n^{100})$  időigényű algoritmus?
- ▶ A konstansok nagysága.
- ▶ Kisméretű adatokon egy exponenciális algoritmus is jobban viselkedhet, mint egy polinomiális (de csak véges sok adaton).
- ▶ Legrosszabb eset – várható viselkedés.
- ▶ Hatékony párhuzamosítás.

## Ugyanakkor

- ▶ A gyakorlatban előforduló **P**-beli problémák rendje kicsi.
- ▶ A **P** osztály robusztus, és elegáns matematikai elmélethez vezet.
- ▶ Fontos információkat szolgáltat a hatékonyan megoldható problémák szerkezetéről.



# Turing-gépek definíciója

## Definíció

**Turing-gép:** egy  $(K, \Sigma, \delta, s)$  négyes, ahol

- ▶  $K$  az állapotok véges, nemüres halmaza;
- ▶  $\Sigma$  a betűk (szimbólumok) véges, nemüres halmaza;
- ▶  $\delta : K \times \Sigma \rightarrow (K \cup \{„igen”, „nem”, h\}) \times \Sigma \times \{\leftarrow, -, \rightarrow\}$  az átmenetfüggvény;
- ▶  $s \in K$  a kezdőállapot.

$\triangleright, \sqcup \in \Sigma, \Sigma - \{\triangleright, \sqcup\} \neq \emptyset,$

$\delta(q, \triangleright) = (p, \sigma, d) \Rightarrow \sigma = \triangleright, d = \rightarrow$  (legalábbis  $d \neq \leftarrow$ ).

# Példa: jobbra tolás

| $p \in K$ | $\sigma \in \Sigma$ | $\delta(p, \sigma)$                | Példa futás  |
|-----------|---------------------|------------------------------------|--|
| $s$       | $\triangleright$    | $(s, \triangleright, \rightarrow)$ | $s \quad \underline{\triangleright}010$                |
| $s$       | $0$                 | $(s, 0, \rightarrow)$              | $s \quad \triangleright\underline{0}10$                |
| $s$       | $1$                 | $(s, 1, \rightarrow)$              | $s \quad \triangleright0\underline{1}0$                |
| $s$       | $\sqcup$            | $(q, \sqcup, \leftarrow)$          | $s \quad \triangleright01\underline{0}$                |
| $q$       | $\triangleright$    | $(h, \triangleright, \rightarrow)$ | $s \quad \triangleright010\underline{\sqcup}$          |
| $q$       | $\sqcup$            | $(q, \sqcup, -)$                   | $q \quad \triangleright01\underline{0}\sqcup$          |
| $q$       | $0$                 | $(q_0, \sqcup, \rightarrow)$       | $q_0 \quad \triangleright01 \sqcup \underline{\sqcup}$ |
| $q$       | $1$                 | $(q_1, \sqcup, \rightarrow)$       | $s \quad \triangleright01\underline{\sqcup}0$          |
| $q_0$     | $\triangleright$    | $(h, \triangleright, \rightarrow)$ | $q \quad \triangleright0\underline{1} \sqcup 0$        |
| $q_0$     | $\sqcup$            | $(s, 0, \leftarrow)$               | $q_1 \quad \triangleright0 \sqcup \underline{\sqcup}0$ |
| $q_0$     | $0$                 | $(s, 0, \leftarrow)$               | $s \quad \triangleright0\underline{\sqcup}10$          |
| $q_0$     | $1$                 | $(s, 0, \leftarrow)$               | $q \quad \triangleright\underline{0} \sqcup 10$        |
| $q_1$     | $\triangleright$    | $(h, \triangleright, \rightarrow)$ | $q_0 \quad \triangleright \sqcup \underline{\sqcup}10$ |
| $q_1$     | $\sqcup$            | $(s, 1, \leftarrow)$               | $s \quad \triangleright\underline{\sqcup}010$          |
| $q_1$     | $0$                 | $(s, 1, \leftarrow)$               | $q \quad \underline{\triangleright} \sqcup 010$        |
| $q_1$     | $1$                 | $(s, 1, \leftarrow)$               | $h \quad \triangleright\underline{\sqcup}010$          |

# Konfigurációk

## Definíció

A számítási folyamat adott pillanatának teljes leírását **konfigurációnak** nevezzük.

Formálisan: egy  $(q, w, u)$  hármas, ahol

- ▶  $q \in K \cup \{„igen”, „nem”, h\}$ ;
- ▶  $w \in \Sigma^+$ , a mutatótól balra eső szó, beleértve azt a betűt is, amelyet a mutató kijelöl;
- ▶  $u \in \Sigma^*$ , a mutatótól jobbra lévő, esetleg üres szó.

# Átmenetreláció

**Definíció:**  $(q, w\sigma, u) \xrightarrow{M}(q', w', u')$

$q \in K, \delta(q, \sigma) = (q', \rho, D)$  és

- ▶  $D = \leftarrow$  esetén  $w' = w$  és  $u' = \rho u$ ;
- ▶  $D = -$  esetén  $w' = w\rho$  és  $u' = u$ ;
- ▶  $D = \rightarrow$  esetén

$$w' = \begin{cases} w\rho\tau & \text{ha } u = \tau u_1 \\ w\rho\sqcup & \text{ha } u = \varepsilon \end{cases} \quad u' = \begin{cases} u_1 & \text{ha } u = \tau u_1 \\ \varepsilon & \text{ha } u = \varepsilon \end{cases}$$

**Definíció**

$(q, w\sigma, u) \xrightarrow{M^k}(q', w', u')$ ,  $(q, w\sigma, u) \xrightarrow{M^*}(q', w', u')$  a szokásos módon.

# A Turing-gép által kiszámított függvény

## Definíció

Legyen  $x \in (\Sigma - \{\triangleright, \sqcup\})^*$ .  $M(x) =$

„igen” ha  $(s, \triangleright, x) \xrightarrow{M^*} (, , \text{igen} , w, u)$  valamely  $w, u$  szavakra;

„nem” ha  $(s, \triangleright, x) \xrightarrow{M^*} (, , \text{nem} , w, u)$  valamely  $w, u$  szavakra;

$y$  ha  $(s, \triangleright, x) \xrightarrow{M^*} (h, w, u)$  valamely  $w, u$  szavakra úgy, hogy  $wu = \triangleright y \sqcup^k$  alakú és  $y$  nem  $\sqcup$ -ra végződik;

↗ különben, azaz ha  $M$  nem áll meg az  $(s, \triangleright, x)$  kezdőkonfigurációból indítva.

# Bináris növelés

| $p \in K$ | $\sigma \in \Sigma$ | $\delta(p, \sigma)$                | Példa futás 1.                                  | Példa futás 2.                                |
|-----------|---------------------|------------------------------------|---|---|
| $s$       | $\triangleright$    | $(s, \triangleright, \rightarrow)$ | $(s, \triangleright, 11011)$                    | $(s, \triangleright, 111)$                    |
| $s$       | $0$                 | $(s, 0, \rightarrow)$              | $(s, \triangleright 1, 1011)$                   | $(s, \triangleright 1, 11)$                   |
| $s$       | $1$                 | $(s, 1, \rightarrow)$              | $(s, \triangleright 11, 011)$                   | $(s, \triangleright 11, 1)$                   |
| $s$       | $\sqcup$            | $(q, \sqcup, \leftarrow)$          | $(s, \triangleright 110, 11)$                   | $(s, \triangleright 111, \varepsilon)$        |
| $q$       | $\triangleright$    | $(h, \triangleright, \rightarrow)$ | $(s, \triangleright 1101, 1)$                   | $(s, \triangleright 111 \sqcup, \varepsilon)$ |
| $q$       | $0$                 | $(h, 1, -)$                        | $(s, \triangleright 11011, \varepsilon)$        | $(q, \triangleright 111, \sqcup)$             |
| $q$       | $1$                 | $(q, 0, \leftarrow)$               | $(s, \triangleright 11011 \sqcup, \varepsilon)$ | $(q, \triangleright 11, 0 \sqcup)$            |
|           |                     |                                    | $(q, \triangleright 11011, \sqcup)$             | $(q, \triangleright 1, 00 \sqcup)$            |
|           |                     |                                    | $(q, \triangleright 1101, 0 \sqcup)$            | $(q, \triangleright, 000 \sqcup)$             |
|           |                     |                                    | $(q, \triangleright 110, 00 \sqcup)$            | $(h, \triangleright 0, 00 \sqcup)$            |
|           |                     |                                    | $(h, \triangleright 111, 00 \sqcup)$            |   |

# A palindromák eldöntése

| $p \in K$ | $\sigma \in \Sigma$ | $\delta(p, \sigma)$                  | $p \in K$ | $\sigma \in \Sigma$ | $\delta(p, \sigma)$                            |
|-----------|---------------------|--------------------------------------|-----------|---------------------|--|
| $s$       | $\triangleright$    | $(s, \triangleright, \rightarrow)$   | $q'_0$    | $0$                 | $(q, \sqcup, \leftarrow)$                      |
| $s$       | $0$                 | $(q_0, \triangleright, \rightarrow)$ | $q'_0$    | $1$                 | $(\text{„nem”}, 1, -)$                         |
| $s$       | $1$                 | $(q_1, \triangleright, \rightarrow)$ | $q'_0$    | $\triangleright$    | $(\text{„igen”}, \triangleright, \rightarrow)$ |
| $s$       | $\sqcup$            | $(\text{„igen”}, \sqcup, -)$         | $q'_1$    | $0$                 | $(\text{„nem”}, 0, -)$                         |
| $q_0$     | $0$                 | $(q_0, 0, \rightarrow)$              | $q'_1$    | $1$                 | $(q, \sqcup, \leftarrow)$                      |
| $q_0$     | $1$                 | $(q_0, 1, \rightarrow)$              | $q'_1$    | $\triangleright$    | $(\text{„igen”}, \triangleright, \rightarrow)$ |
| $q_0$     | $\sqcup$            | $(q'_0, \sqcup, \leftarrow)$         | $q$       | $0$                 | $(q, 0, \leftarrow)$                           |
| $q_1$     | $0$                 | $(q_1, 0, \rightarrow)$              | $q$       | $1$                 | $(q, 1, \leftarrow)$                           |
| $q_1$     | $1$                 | $(q_1, 1, \rightarrow)$              | $q$       | $\triangleright$    | $(s, \triangleright, \rightarrow)$             |
| $q_1$     | $\sqcup$            | $(q'_1, \sqcup, \leftarrow)$         |           |                     |  |

# Turing-gépek mint algoritmusok

Nyelvek eldöntése

Szófüggvények kiszámítása

## Definíció

Azt mondjuk, hogy az  $M$  Turing-gép **eldönti** az  $L \subseteq (\Sigma - \{\triangleright, \sqcup\})^*$  nyelvet, ha bármely  $x \in (\Sigma - \{\triangleright, \sqcup\})^*$  szóra:

$$x \in L \Rightarrow M(x) = \text{„igen”}$$

$$x \notin L \Rightarrow M(x) = \text{„nem”}$$

## Definíció

Azt mondjuk, hogy az  $M$  Turing-gép **felismeri** az  $L \subseteq (\Sigma - \{\triangleright, \sqcup\})^*$  nyelvet, ha bármely  $x \in (\Sigma - \{\triangleright, \sqcup\})^*$  szóra:

$$x \in L \Rightarrow M(x) = \text{„igen”}$$

$$x \notin L \Rightarrow M(x) = \nearrow$$

$$[x \notin L \Rightarrow M(x) \neq \text{„igen”}]$$



## Definíció

Egy  $L \subseteq (\Sigma - \{\triangleright, \sqcup\})^*$  nyelvet **rekurzív**nak vagy **eldönthetőnek** nevezünk, ha létezik olyan  $M$  Turing-gép, mely eldönti az  $L$  nyelvet.

**Rekurzívan felsorolható**nak nevezük az  $L$  nyelvet, ha létezik olyan Turing-gép, amely felismeri  $L$ -et.

## Állítás

Minden rekurzív nyelv rekurzívan felsorolható.  
(A fordított irányú implikáció nem igaz.)

## Definíció

Legyen  $f : (\Sigma - \{\triangleright, \sqcup\})^* \rightarrow \Sigma^*$ . Azt mondjuk, hogy  $f$  **rekurzív függvény**, ha létezik olyan  $M$  Turing-gép, hogy bármely  $x \in (\Sigma - \{\triangleright, \sqcup\})^*$  szóra  $M(x) = f(x)$ .

# Példák

## Példák eldönthető nyelvekre

- ▶ palindromák a  $\{0, 1\}$  ábécé fölött;
- ▶  $\{0^n c 0^m c 0^{n+m} : n, m \geq 0\}$ ;
- ▶  $\{0^n c 0^m c 0^{nm} : n, m \geq 0\}$ ;
- ▶  $\{0^n c 0^{2^n} : n \geq 0\}$ ;
- ▶  $\{\overline{x_1 c x_2 c x_1 + x_2} : x_1, x_2 \geq 0\}$ , ahol  $\bar{x}$  jelöli az  $x$  szám bináris alakját;
- ▶  $\{\bar{x} : x \text{ prímszám}\}$ .

## Példák rekurzív függvényekre

- ▶  $x \mapsto \sqcup x$        $x \in \{0, 1\}^*$ ;
- ▶  $\bar{x} \mapsto \overline{x + 1}$        $x \geq 0$ ;
- ▶  $\overline{xcy} \mapsto \overline{x + y}$        $x, y \geq 0$ ;
- ▶  $\bar{n} \mapsto \overline{p_n}$        $n \geq 0, p_n := \text{az } n. \text{ prímszám.}$

- ▶ Nyelvek – eldöntési problémák
- ▶ Rekurzív nyelvek – algoritmikusan megoldható eldöntési problémák
- ▶ Szófüggvények – optimalizálási problémák
- ▶ Rekurzív függvények – algoritmikusan megoldható optimalizálási problémák

Bármely véges matematikai objektum (algoritmikusan) **kódolható** szavakkal.

Bármely két „elfogadható” kódolás **polinomiális kapcsolatban** áll egymással.

A kiszámíthatóság és bonyolultság elmélete **független a kódolástól**.

# Többszavas Turing-gépek

## Definíció

Egy  $k$  szavas Turing-gépen egy  $M = (K, \Sigma, \delta, s)$  rendszert értünk, ahol  $K, \Sigma, s$  ugyanazok, mint közönséges Turing-gép esetén,  $\delta$  pedig

$$K \times \Sigma^k \rightarrow (K \cup \{h, \text{„igen”}, \text{„nem”}\}) \times (\Sigma \times \{\leftarrow, -, \rightarrow\})^k$$

leképezés.

Kikötés:  $\triangleright$  nem írható át és „nem léphető át”.

## Definíció

**Konfiguráció:**  $(q, w_1, u_1, \dots, w_k, u_k)$ ,  $q \in K \cup \{h, \text{„igen”}, \text{„nem”}\}$ ,  $w_i \in \Sigma^+$ ,  $u_i \in \Sigma^*$ .

Az  $\xrightarrow{M}$ ,  $\xrightarrow{M^t}$ ,  $\xrightarrow{M^*}$  relációk értelem szerint definiáltak.

# Palindromák eldöntése két szalaggal

| $p$ | $\sigma_1$       | $\sigma_2$       | $\delta(p, \sigma_1, \sigma_2)$                                 | Példa futás  |
|-----|------------------|------------------|---|--|
| $s$ | $\triangleright$ | $\triangleright$ | $(s, \triangleright, \rightarrow, \triangleright, \rightarrow)$ | $(s, \triangleright, 010, \triangleright, \varepsilon)$                                |
| $s$ | $0$              | $\sqcup$         | $(s, 0, \rightarrow, 0, \rightarrow)$                           | $(s, \triangleright 010 \sqcup, \varepsilon, \triangleright 010 \sqcup, \varepsilon)$  |
| $s$ | $1$              | $\sqcup$         | $(s, 1, \rightarrow, 1, \rightarrow)$                           | $(q, \triangleright 010, \sqcup, \triangleright 010 \sqcup, \varepsilon)$              |
| $s$ | $\sqcup$         | $\sqcup$         | $(q, \sqcup, \leftarrow, \sqcup, -)$                            | $(q, \triangleright, 010 \sqcup, \triangleright 010 \sqcup, \varepsilon)$              |
| $q$ | $0$              | $\sqcup$         | $(q, 0, \leftarrow, \sqcup, -)$                                 | $(p, \triangleright 0, 10 \sqcup, \triangleright 010, \sqcup)$                         |
| $q$ | $1$              | $\sqcup$         | $(q, 1, \leftarrow, \sqcup, -)$                                 | $(p, \triangleright 01, 0 \sqcup, \triangleright 01, 0 \sqcup)$                        |
| $q$ | $\triangleright$ | $\sqcup$         | $(p, \triangleright, \rightarrow, \sqcup, \leftarrow)$          | $(p, \triangleright 010, \sqcup, \triangleright 0, 10 \sqcup)$                         |
| $p$ | $0$              | $0$              | $(p, 0, \rightarrow, 0, \leftarrow)$                            | $(p, \triangleright 010 \sqcup, \varepsilon, \triangleright, 010 \sqcup)$              |
| $p$ | $1$              | $1$              | $(p, 1, \rightarrow, 1, \leftarrow)$                            | $(\text{„igen”}, \triangleright 010 \sqcup, \varepsilon, \triangleright 0, 10 \sqcup)$ |
| $p$ | $0$              | $1$              | $(\text{„nem”}, 0, -, 1, -)$                                    |  |
| $p$ | $1$              | $0$              | $(\text{„nem”}, 1, -, 0, -)$                                    |  |
| $p$ | $\sqcup$         | $\triangleright$ | $(\text{„igen”}, \sqcup, -, \triangleright, \rightarrow)$       |  |

## Definíció

Legyen  $x \in (\Sigma - \{\sqcup, \triangleright\})^*$ .

- ▶  $M(x) = \text{„igen”}$ , ha  
 $(s, \triangleright, x, \triangleright, \varepsilon, \dots, \triangleright, \varepsilon) \xrightarrow{M^*} (\text{„igen”}, w_1, u_1, \dots, w_k, u_k)$ ;
- ▶  $M(x) = \text{„nem”}$ , ha  
 $(s, \triangleright, x, \triangleright, \varepsilon, \dots, \triangleright, \varepsilon) \xrightarrow{M^*} (\text{„nem”}, w_1, u_1, \dots, w_k, u_k)$ ;
- ▶  $M(x) = y$ , ha  $(s, \triangleright, x, \triangleright, \varepsilon, \dots, \triangleright, \varepsilon) \xrightarrow{M^*} (h, w_1, u_1, \dots, w_k, u_k)$   
és  $y$  a  $w_k u_k$  szó elején álló  $\triangleright$  és a végén esetleg előforduló  
 $\sqcup$  jelek elhagyásával kapott szó;
- ▶  $M(x) = \nearrow$ , különben.

Eldöntés, felismerés stb. közönséges Turing-gépekhez hasonlóan.

# Időkorlát

## Definíció

Legyen  $f : \mathbb{N} \rightarrow \mathbb{N}$ ,  $M$  pedig többszavas Turing-gép.  $M$   $f(n)$  időkorlátos Turing-gép, ha  $M$  minden  $x$  bemeneten legfeljebb  $f(|x|)$  lépésben megáll. ( $|x|$  az  $x$  szó hosszát jelöli.)

Más elnevezés:  $M$  időigénye (legfeljebb)  $f(n)$ .

## Definíció

$\text{TIME}(f(n)) = \{L : \exists M f(n) \text{ időkorlátos } L\text{-et eldöntő Turing-gép}\}$ .

## Példa

- ▶  $L \subseteq \{0, 1\}^*$ , palindromák. Ekkor:
- ▶  $L \in \text{TIME}\left(\frac{(n+1)(n+2)}{2}\right)$ ;
- ▶  $L \in \text{TIME}(3n + 3)$ .

Időkorlát esetén mindig kikötjük, hogy  $f(n) > n$ .

# $k$ -szavas gép szimulálása egyszavassal

## Tétel

Bármely  $k$ -szavas  $f(n) > n$  időkorlátos  $M$  Turing-géphez megadható vele ekvivalens  $\mathcal{O}((f(n))^2)$  időkorlátos **egyszavas**  $M'$  Turing-gép.

**Ekvivalens:**  $M'(x) = M(x)$  minden  $x$ -re.

## Bizonyítás

- ▶  $M'$  egyetlen szóban tárolja  $M$   $k$  szavának konkatenációját.
- ▶ Minden egyes szóban egy betű „alá van húzva”.
- ▶ Egy lépés szimulálásához  $M'$  végigolvassa az egész szót.
- ▶ Esetleg jobbra lépteti a szót.
- ▶ Időigény lépésenként  $\mathcal{O}(f(n))$ .
- ▶ Teljes időigény:  $\mathcal{O}((f(n))^2)$ .



# Lineáris felgyorsítás

## Tétel

Tegyük fel, hogy  $L \in \text{TIME}(f(n))$ . Ekkor minden  $\epsilon > 0$  valós számra  $L \in \text{TIME}(f'(n))$  is teljesül, ahol  $f'(n) = \epsilon f(n) + n + 2$ .

## Bizonyítás

Legyen  $M$   $k$ -szavas,  $f(n)$  időkorlátos Turing-gép.

$$k' = \begin{cases} k & \text{ha } k > 1 \\ 2 & \text{ha } k = 1 \end{cases}$$

Belátjuk, hogy  $M$  szimulálható  $k'$ -szavas,  $f'(n)$  időkorlátos Turing-géppel ( $M'$ -vel).

# Lineáris felgyorsítás

- ▶  $M'$  az  $M$  szavait  $m$  hosszú blokkokra bontja, és minden blokkot **egyetlen betűként** tárol.
- ▶ Első menet: **tömörítés**.  $M'$  második szavában elkészíti az első szó (bemenet) tömörített változatát:  $n + 2$  lépés.
- ▶ A második szó **elejére mozgatja** a mutatót:  $\lceil \frac{n}{m} \rceil$  lépés.
- ▶  $M'$  állapotai:  $(q, i_1, \dots, i_k)$ ,  $1 \leq i_j \leq m$ .
- ▶  $M'$  az  $M$  gép  $m$  egymást követő lépését **6 lépésben** szimulálja:  $6 \lceil \frac{f(n)}{m} \rceil$  lépés.
- ▶ Teljes időigény:  $n + 2 + \lceil \frac{n}{m} \rceil + 6 \lceil \frac{f(n)}{m} \rceil \leq n + 2 + 7 \lceil \frac{f(n)}{m} \rceil$ .
- ▶ Ha  $m$  elég nagy, akkor ez  $\leq \epsilon f(n) + n + 2$ .

# A **P** osztály

## Következmény

- ▶ Ha  $L \in \text{TIME}(f(n))$ , ahol  $f(n) = an + b$ ,  $a, b \in \mathbb{N}$ ,  $a > 0$ , akkor  $n$  együtthatóját tetszőlegesen közel vihetjük 1-hez:  $L \in \text{TIME}(f'(n))$ , ahol  $f'(n) = (1 + \epsilon)n + c$ , valamely tetszőlegesen kicsi pozitív  $\epsilon$ -ra és valamely  $c$ -re.
- ▶ Ha  $L \in \text{TIME}(f(n))$ , ahol  $f(n)$  másodfokú polinom, akkor a másodfokú tag együtthatóját tetszőlegesen közel vihetjük 0-hoz.

## Definíció

$$\mathbf{P} = \text{TIME}(n^k), \text{ azaz } \mathbf{P} = \bigcup_{i=1}^{\infty} \text{TIME}(n^i).$$

Tehát  $L \in \mathbf{P} \Leftrightarrow \exists p(n)$  polinom, hogy  $L \in \text{TIME}(p(n))$ .

# Tárkorlátok

## Definíció

Legyen  $k > 2$ . Egy  $k$ -szavas lyukszalagos Turing-gép olyan  $k$ -szavas Turing-gép, mely

- ▶ első szalagját **csak olvassa**,
- ▶ utolsó szalagjára **csak ír**.

(itt a „szalag” a „szó” szinonimája.)

$$\delta(q, \sigma_1, \dots, \sigma_k) = (p, \varrho_1, D_1, \dots, \varrho_k, D_k) \Rightarrow \sigma_1 = \varrho_1 \text{ és } D_k \neq \leftarrow.$$

Továbbá ha  $\sigma_1 = \sqcup$ , akkor  $D_1 = \leftarrow$ .

## Állítás

Bármely  $k$ -szavas  $f(n)$  időkorlátos  $M$  Turing-géphez elkészíthető egy vele ekvivalens,  $\mathcal{O}(f(n))$  időkorlátos  $(k + 2)$ -szavas lyukszalagos  $M'$  Turing-gép.

# Tárigény

## Definíció

Legyen  $M$  egy  $k$ -szavas Turing-gép és  $x$  egy bemenő szó.  
Tegyük fel, hogy

$$(s, \triangleright, x, \triangleright, \varepsilon, \dots, \triangleright, \varepsilon) \xrightarrow{M^*} (p, w_1, u_1, \dots, w_k, u_k),$$

ahol  $p \in \{, , \text{igen} , , \text{nem} , h\}$ .

Ekkor  $M$  **tárigénye**  $x$ -en  $\sum_{i=1}^k |w_i u_i|$ .

**Kivéve**, ha  $M$  (hangsúlyozottan) lyukszalagos.

Ekkor a tárigény:  $\sum_{i=2}^{k-1} |w_i u_i|$ .

## Definíció

Azt mondjuk, hogy  $M$  egy  **$f(n)$  tárigényű Turing-gép**, ha  $M$  minden  $x$  bemeneten megáll és tárigénye minden  $x$  bemeneten legfeljebb  $f(|x|)$ .

# Tárbonyolultsági osztályok

## Definíció

Azt mondjuk, hogy az  $L$  nyelv a  $\text{SPACE}(f(n))$  bonyolultsági osztályban van, ha létezik az  $L$  nyelvet eldöntő,  $f(n)$  tárkorlátos lyukszalagos Turing-gép.

## Jelölés

$$\mathbf{L} = \text{SPACE}(\log n)$$

## Példa

A palindromák nyelve az  $\mathbf{L}$  osztályba esik.

- ▶ Egy lyukszalagos Turing-gép második szavában egy  $1 \leq i \leq n$  szám **bináris alakját** tárolja.
- ▶ Az  $i$ -edik menetben megnézi, hogy a bemenet  $i$ -dik betűje megegyezik-e a hátulról  $i$ -dik betűjével. A számláláshoz egy harmadik munkaszót használ.

# Lineáris tárösszehúzás

## Tétel

$$L \in \text{SPACE}(f(n)) \Rightarrow \forall \epsilon > 0 : L \in \text{SPACE}(\epsilon f(n)).$$

## Bizonyítás

Legyen  $M$  az  $L$ -et eldöntő  $f(n)$  tárkorlátos lyukszalagos Turing-gép. Legyen  $m > 0$ . Készítünk egy  $M$ -et szimuláló  $M'$  lyukszalagos Turing-gépet, mely  $M$  munkaszalagjait  **$m$ -es blokkokban** tárolja egyetlen munkaszalagján.

$M'$  tárigénye:  $\left\lceil \frac{f(n)}{m} \right\rceil$ .

De  $\left\lceil \frac{f(n)}{m} \right\rceil \leq \epsilon f(n)$ , ha  $m \geq \left\lceil \frac{1}{\epsilon} \right\rceil$ .

# Nemdeterminisztikus Turing-gép

## Definíció

$k$ -szavas **nemdeterminisztikus Turing-gép** egy  $N = (K, \Sigma, \Delta, s)$  rendszer, ahol  $K$ ,  $\Sigma$  és  $s$  ugyanazok, mint közönséges  $k$ -szavas Turing-gép esetén, és

$$\Delta \subseteq [K \times \Sigma^k] \times [(K \cup \{„igen”, „nem”, h\}) \times (\Sigma \times \{\leftarrow, -, \rightarrow\})^k].$$

A **konfiguráció**, **közvetlen átmenet** ( $\xrightarrow{N}$ ),  $t$  **lépéses átmenet** ( $\xrightarrow{N^t}$ ), **közvetett átmenet** ( $\xrightarrow{N^*}$ ) relációkat az előzőeknek megfelelően definiáljuk.

$$\begin{array}{ccc} (p, w_1, u_1, \dots, w_k, u_k) & \rightarrow & (p', w'_1, u'_1, \dots, w'_k, u'_k) \\ & \searrow & \\ & & (p'', w''_1, u''_1, \dots, w''_k, u''_k) \end{array}$$



## Definíció

Legyen  $L \subseteq (\Sigma - \{\triangleright, \sqcup\})^*$ . Azt mondjuk, hogy  $N$  **eldönti** az  $L$  nyelvet, ha minden  $x \in (\Sigma - \{\triangleright, \sqcup\})^*$  szóra:

$$x \in L \iff \exists w_1, u_1, \dots, w_k, u_k : \\ (s, \triangleright, x, \triangleright, \varepsilon, \dots, \triangleright, \varepsilon) \xrightarrow{N^*} („igen”, w_1, u_1, \dots, w_k, u_k)$$

Tehát **nem követeljük meg**, hogy  $N$  minden bemeneten megálljon és létezhetnek végtelen számítási sorozatok is!

## Definíció

Legyen  $L \subseteq (\Sigma - \{\triangleright, \sqcup\})^*$ ,  $f : \mathbb{N} \rightarrow \mathbb{N}$ . Azt mondjuk, hogy  $N$   **$f(n)$  időben eldönti  $L$ -t**, ha eldönti, és valahányszor egy  $x$  szóra,  $t \geq 0$  számra és  $(q, w_1, u_1, \dots, w_k, u_k)$  konfigurációra:

$$(s, \triangleright, x, \triangleright, \varepsilon, \dots, \triangleright, \varepsilon) \xrightarrow{N^t} (q, w_1, u_1, \dots, w_k, u_k),$$

mindig fennáll, hogy  $t \leq f(|x|)$ .

Tehát minden számítási sorozat véges.

# Az $\text{NTIME}(f(n))$ osztályok

## Definíció

$\text{NTIME}(f(n)) = \{ L : \text{létezik } L\text{-t } f(n) \text{ időben eldöntő nemdeterminisztikus Turing-gép} \}.$

Itt is feltesszük, hogy  $f(n) > n$ .

## Állítás

$$\text{TIME}(f(n)) \subseteq \text{NTIME}(f(n)).$$

## Tétel

Minden  $\epsilon > 0$  valós számra

$$\text{NTIME}(f(n)) \subseteq \text{NTIME}(\epsilon f(n) + n + 2).$$

# Az NP osztály

## Definíció

$$\mathbf{NP} = \text{NTIME}(n^k) = \bigcup_{i=1}^{\infty} \text{NTIME}(n^i).$$

## Példa

$$\text{TSP}(E) \in \mathbf{NP}.$$

**Nem ismert** (de nagyon valószínűtlen), hogy létezik-e módszer nemdeterminisztikus Turing-gépek „hatékony” szimulálására.

$$\mathbf{P} \stackrel{?}{=} \mathbf{NP}$$

# Nemdeterminisztikus gépek szimulálása

## Tétel

$$\text{NTIME}(f(n)) \subseteq \bigcup_{c>1} \text{TIME}(c^{f(n)}).$$

## Bizonyítás

Legyen  $N = (K, \Sigma, \Delta, s)$  egy  $f(n)$  időkorlátos nemdeterminisztikus Turing-gép.

Belátjuk, hogy  $N$  szimulálható egy 3-szavas,  $c^{f(n)}$  időkorlátos (determinisztikus)  $M$  Turing-géppel valamely  $c > 1$  számra.

Minden  $(q, \sigma) \in K \times \Sigma$  rendezett párra legyen

$$\begin{aligned} C_{q,\sigma} &= \{(q', \sigma', D) : ((q, \sigma), (q', \sigma', D)) \in \Delta\}, \\ d &= \max_{q,\sigma} |C_{q,\sigma}|. \end{aligned}$$

Feltehető, hogy  $d > 1$ .

- ▶ Minden  $t$  lépésből álló nemdeterminisztikus választási sorozat reprezentálható egy  $c_1 \dots c_t$ ,  $c_i \in \{0, \dots, d-1\}$  sorozattal. A sorozatokat rendezzük **lexikografikusan**.
- ▶  $M$  az egyik munkaszalagján sorra előállítja a  $c_1 \dots c_t$  választási sorozatokat, a másikon az adott sorozatra szimulálja  $N$  működését.
- ▶  $M$  akkor áll meg „igen” állapotban, ha valamely választási sorozatra  $N$  ugyanezt teszi.
- ▶  $M$  akkor áll meg „nem” állapotban, ha valamely  $t$  esetén  $N$  az összes  $t$  hosszú választási sorozatra egy „igen”-től különböző állapotban áll meg.

Időigény:

**Választások száma:**  $\sum_{t=0}^{f(n)} d^t = \frac{d^{f(n)+1} - 1}{d-1} = \mathcal{O}(d^{f(n)})$ .

**Sorozatonként:**  $\mathcal{O}(f(n))$ .

**Összesen:**  $\mathcal{O}(f(n)d^{f(n)}) = \mathcal{O}(d_1^{f(n)})$ .

# Az $\text{NSPACE}(f(n))$ osztályok

## Definíció

Azt mondjuk, hogy a  $k$ -szavas  $N = (K, \Sigma, \Delta, s)$  nemdeterminisztikus lyukszalagos Turing-gép  $f(n)$  tárral (vagy tárkorláttal) **eldönti** az  $L \subseteq (\Sigma - \{\triangleright, \sqcup\})^*$  nyelvet, ha eldönti, és valahányszor

$$(s, \triangleright, x, \triangleright, \varepsilon, \dots, \triangleright, \varepsilon) \xrightarrow{N^*} (q, w_1, u_1, \dots, w_k, u_k),$$

ahol  $x \in (\Sigma - \{\triangleright, \sqcup\})^*$ ,  $q \in K \cup \{,,igen", ,,nem", h\}$ ,  $w_i, u_i \in \Sigma^*$ , fennáll, hogy

$$\sum_{i=2}^{k-1} |w_i u_i| \leq f(|x|).$$

## Definíció

$\text{NSPACE}(f(n)) = \{L : \text{létezik olyan nemdeterminisztikus Turing-gép, mely } f(n) \text{ tárral eldönti } L\text{-et}\}.$

## Definíció

$$\mathbf{NL} = \mathbf{NSPACE}(\log n).$$

A Turing-gépnek létezhet végtelen számítási sorozata is!

## Állítás

$$\mathbf{SPACE}(f(n)) \subseteq \mathbf{NSPACE}(f(n)).$$

# ELÉRHETŐSÉG

Példa

ELÉRHETŐSÉG  $\in$  NL.

Bizonyítás

- ▶ 4-szavas lyukszalagos nemdeterminisztikus Turing-gépet tervezünk.
- ▶ Egyik munkaszalag:  $i$  aktuális csúcs bináris alakja (kezdetben 1).
- ▶ Másik munkaszalag: nemdeterminisztikusan generál a gép egy  $j$  csúcst (binárisan).
- ▶ Ellenőrzi, hogy  $(i, j)$  él-e.
  - ▶ Ha nem, akkor „nem” állapotban megáll.
  - ▶ Ha igen, akkor
    - $j = n$  esetén „igen” állapotban megáll;
    - $j \neq n$  esetén kicseréli  $i$ -t  $j$ -vel és folytatja az eljárást.
- ▶ Az „igen” választ az output szalagra kiírja.



## Megjegyzés

Később azt is belátjuk, hogy

$$\text{ELÉRHETŐSÉG} \in \text{SPACE}(\log^2(n)).$$

## Tétel

$$\forall \epsilon > 0 : \text{NSPACE}(f(n)) \subseteq \text{NSPACE}(\epsilon f(n)).$$

# Közvetlen hozzáférésű gépek (RAM)

A **Church-Turing tézis** legfőbb bizonyítéka: az algoritmusfogalom bármely két matematikai modellje ekvivalens.

## Erősebb tézis

„Az algoritmusok és időigényük matematikai eszközökkel való modellezésére tett bármely ésszerű kísérlet szükségképpen olyan modellhez és időigény-fogalomhoz vezet, mely **polinomiálisan ekvivalens** a Turing-géppel.”

Ezt demonstráljuk RAM esetén.

# RAM

- ▶ Adatszerkezet: **regiszterek**, melyek 0-tól kezdve sorszámozottak. Minden regiszter tetszőleges (pozitív vagy negatív) egész számot tárolhat. A 0. regiszter: **akkumulátor**.
- ▶ **Címzési módok:**
  - direkt:  $j$
  - indirekt:  $\uparrow j$
  - közvetlen:  $= j$
- ▶ **utasításszámláló:**  $\kappa$
- ▶ **bemenet:**  $(i_0, i_1, \dots, i_t)$  egész számok sorozata

A  $j$ . regiszter tartalmát  $r_j$  jelöli.

## Utasítások

READ  $j$

$$r_0 := i_j$$

READ  $\uparrow j$

$$r_0 := i_{r_j}$$

STORE  $j$

$$r_j := r_0$$

STORE  $\uparrow j$

$$r_{r_j} := r_0$$

$x \in \{j, \uparrow j, =j\}$  esetén

LOAD  $x$

$$r_0 := x$$

ADD  $x$

$$r_0 := r_0 + x$$

SUB  $x$

$$r_0 := r_0 - x$$

HALF

$$r_0 := \lfloor r_0/2 \rfloor$$

JUMP  $j$

$$\kappa := j$$

JZERO  $j$

ha  $r_0 = 0$ , akkor  $\kappa := j$

JPOS  $j$

ha  $r_0 > 0$ , akkor  $\kappa := j$

JNEG  $j$

ha  $r_0 < 0$ , akkor  $\kappa := j$

HALT

# RAM időigény

**Program:** Utasítások véges sorozata. A program szemantikáját természetes módon definiáljuk. Hibás utasítás a program működésének terminálását eredményezi.

## Időigény számítása

Minden utasítás egységnyi

Túlzottan liberális – nem

## Bemenet mérete

$I = (i_0, i_1, \dots, i_t)$  mérete:  $\ell(I) = \sum_{j=0}^t \ell(i_j)$ .

$\ell(i_j)$  az  $i_j$  szám bináris reprezentációjának hossza.

Tehát  $\mathcal{O}(n)$  időigény azt jelenti, hogy a lépésszám arányos a bemeneten adott számok **logaritmusával**.

## Tétel

A Turing-gép polinomiálisan ekvivalens a RAM-mal.

Pontosabban: A és B

A

Legyen  $L \subseteq (\Sigma - \{\triangleright, \sqcup\})^*$  a  $T, f(n)$  időkorlátos Turing-géppel eldöntött nyelv. Legyen  $\Sigma = \{\sigma_1, \dots, \sigma_k\}$  és

$$D_\Sigma = \{(i_1, \dots, i_n, 0) : n \geq 0, 1 \leq i_j \leq k\}.$$

(Tehát  $D_\Sigma$  a  $\Sigma^*$ -nak felel meg.)

Legyen

$$\varphi_L : D_\Sigma \rightarrow \{0, 1\} \quad \varphi_L(i_1, \dots, i_n, 0) = 1 \iff \sigma_{i_1} \dots \sigma_{i_n} \in L.$$

Ekkor létezik olyan RAM-program, mely  $\mathcal{O}(f(n))$  időben kiszámítja  $\varphi_L$ -t.

## B

Legyen  $D$  egész számok véges sorozatainak halmaza,  $\varphi$  pedig  $D$ -t az egész számok halmazába képező függvény. Adott  $i$  egészre jelölje  $b(i)$  az  $i$  szám bináris reprezentációját,

$I = (i_0, i_1, \dots, i_t)$  esetén pedig legyen

$$b(I) = b(i_0); b(i_1); \dots; b(i_t).$$

Ha  $\varphi$  „kiszámítható RAM programmal”, akkor létezik olyan  $M$  Turing-gép, mely kiszámítja  $\varphi$ -t a következő értelemben:

tetszőleges  $I \in D$  esetén  $M(b(I)) = b(\varphi(I))$ .

Továbbá, ha  $\varphi$   $f(n)$  lépésben kiszámítható RAM programmal, akkor kiszámítható  $\mathcal{O}(f(n)^3)$  időkorlátos Turing-géppel.

# Univerzális Turing-gép

## Tétel

Létezik olyan  $U$  Turing-gép, amelynek ha adott egy  $M$  Turing-gép és annak egy  $x$  bemenete, úgy viselkedik, mint  $M$  az  $x$ -en, vagyis  $U(M;x) = M(x)$ .

Természetesen  $M$  és  $x$  kódolva adottak  $U$  számára, és  $U$  az  $M(x)$ -et ezen kódolás szerint számítja ki.



# A kódolás

$M = (K, \Sigma, \delta, s)$ . Feltehető, hogy

$$\Sigma = \{1, 2, \dots, |\Sigma|\}$$

$$K = \{|\Sigma| + 1, \dots, |\Sigma| + |K|\}$$

$$s = |\Sigma| + 1$$

Így  $M$  megadható a  $|K|$ ,  $|\Sigma|$  számok bináris alakjával, melyet  $\delta$  leírása követ, amely  $((q, \sigma), (p, \varrho, D))$  alakú párok sorozata. A  $q$ ,  $p$ ,  $\sigma$ ,  $\varrho$ ,  $D$  mindegyike bináris számmal megadható. A  $(, )$ ,  $,$  stb. karakterek is kódolhatók binárisan.  $\leftarrow$ ,  $\rightarrow$ ,  $-$ , „igen”, „nem”,  $h$  feleljen meg a  $|\Sigma| + |K| + 1, \dots, |\Sigma| + |K| + 6$  számoknak. Az  $x$  bemenet szintén megadható bináris számok sorozatával.

# $U$ működése

Kétszavas Turing-gép.

1. szó:  $M; x$  bemenet

2. szó:  $M$  aktuális konfigurációjának kódja

$U$  az  $M$  minden lépését egy menetben szimulálja, melyben

- ▶ Végigolvassa a 2. szót
- ▶ Meghatározza a 2. szón elvégzendő transzformációt, és azt elvégzi

Ha  $M$  megáll,  $U$  is azt teszi.

Amennyiben az  $U$  bemenete nem  $M; x$  alakú, akkor (mondjuk) végtelen ciklusba esik.

# A megállási probléma

Definíció

Legyen  $H = \{M; x : M(x) \neq \nearrow\}$ .

Tétel

$H$  rekurzívan felsorolható, de **nem rekurzív**.

Bizonyítás

- ▶ Rekurzívan felsorolhatóság: az univerzális Turing-gép létezéséből.
- ▶  $H$  nem rekurzív: indirekt bizonyítás.  
Tegyük fel, hogy  $H$  eldönthető egy  $M_H$  Turing-géppel.  
Legyen  $D$  olyan Turing-gép, melyre

$$D(M) = \text{if } M_H(M; M) \text{ then } \nearrow \text{ else halt.}$$

Ekkor:

$$D(D) = \nearrow \Leftrightarrow M_H(D; D) = \text{„igen”} \Leftrightarrow D(D) \neq \nearrow,$$

ami ellentmondás.

## MEGÁLLÁS

- ▶ **Adott:**  $M$  és  $x$
- ▶ **Kérdés:** Megáll-e  $M$  az  $x$ -en?

MEGÁLLÁS eldönthetetlen probléma.

### Megjegyzés

$H$  **teljes** a rekurzívan felsorolható nyelvek között:

- ▶  $H$  rekurzívan felsorolható
- ▶ Minden rekurzívan felsorolható nyelv (rekurzívan) **visszavezethető**  $H$ -ra:
  - ▶ Legyen  $L$  az  $M$  Turing-gép által felismert nyelv.
  - ▶  $x \in L \Leftrightarrow M; x \in H$
  - ▶ Tehát az  $x \in L$  kérdés eldöntését visszavezettük az  $M; x \in H$  kérdésre.

## Állítás

Az alábbi probléma algoritmikusan eldönthetetlen.

- ▶ **Adott:**  $M$  Turing-gép
- ▶ **Kérdés:**  $M$  megáll-e minden bemenetén?

## Bizonyítás

Adott  $M$  és  $x$  esetén legyen  $M'$  olyan Turing-gép, hogy az  $M'$  minden  $y$  bemenő szavára:

$$M'(y) = M(x).$$

Így

$$M(x) \neq \nearrow \Leftrightarrow M' \text{ megáll minden bemenetén.}$$

Ha tehát az adott probléma eldönthető lenne, akkor  
MEGÁLLÁS is az lenne.

# Rice tétele

## Rice tétele

A rekurzívan felsorolható nyelvek egyetlen nemtriviális tulajdonsága sem dönthető el algoritmikusan.

## Állítás

Ha  $L$  rekurzív, akkor  $\bar{L}$  is az.

## Bizonyítás

Az „igen” és a „nem” állapotok felcserélésével.

## Állítás

Egy  $L$  nyelv akkor és csak akkor rekurzív, ha  $L$  és  $\bar{L}$  rekurzívan felsorolhatók.

## Bizonyítás

- ▶  $L$  rekurzív  $\Rightarrow L$  rekurzívan felsorolható  
 $L$  rekurzív  $\Rightarrow \bar{L}$  rekurzív, így  $\bar{L}$  rekurzívan felsorolható
- ▶ Tegyük fel, hogy  $L$  és  $\bar{L}$  rekurzívan felsorolhatók. Ekkor  $L$  felismerhető egy  $M$ ,  $\bar{L}$  pedig egy  $\bar{M}$  Turing-géppel.  
Szimuláljuk  $M$ -et és  $\bar{M}$ -et párhuzamosan!

$\text{co}\mathcal{C}$

## Definíció

Legyen  $\mathcal{C}$  nyelvek egy osztálya. Ekkor

$$\text{co}\mathcal{C} = \{\bar{L} : L \in \mathcal{C}\}.$$

## Jelölés

$$\mathbf{R} = \{L : L \text{ rekurzív}\}$$

$$\mathbf{RE} = \{L : L \text{ rekurzívan felsorolható}\}$$



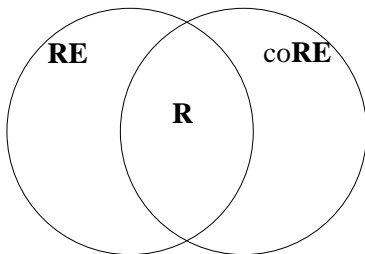
# **R**, **RE** és **coRE** viszonya

Következmény

$$\mathbf{R} = \mathbf{coR}$$

$$\mathbf{RE} \neq \mathbf{coRE}$$

$$\mathbf{R} = \mathbf{RE} \cap \mathbf{coRE}$$



## Hilbert 10. problémája

- ▶ **Adott:**  $f(x_1, \dots, x_n) = g(x_1, \dots, x_n)$  diofantikus egyenlet, ahol  $f$  és  $g$  egész együtthatós polinomok.
- ▶ **Kérdés:** Létezik-e egész értékű megoldás?

## Tétel (Matijasevič)

Hilbert 10. problémája algoritmikusan megoldhatatlan.

## Post megfeleltetési problémája

- ▶ **Adott:**  $u_1, v_1, \dots, u_n, v_n \in \Sigma^*$ .
- ▶ **Kérdés:** Létezik-e olyan  $i_1, \dots, i_k$  ( $k > 0$ ,  $1 \leq i_j \leq n$ ) sorozat, hogy

$$u_{i_1} \dots u_{i_k} = v_{i_1} \dots v_{i_k} ?$$

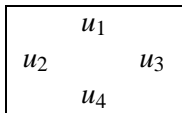
## Tétel (Post)

A fenti probléma algoritmikusan megoldhatatlan.

# Egy dominó probléma

- ▶ **Adott:** Dominó típusok véges halmaza.
- ▶ **Kérdés:** Kirakható-e ezekkel a dominókkal az egész sík hézagmentesen?

Típus:



$$u_i \in \Sigma^*.$$

A dominók nem forgathatóak.

Tétel

A dominó probléma algoritmikusan megoldhatatlan.

# Egy megoldatlan probléma

- ▶ **Adott:**  $m$  pozitív egész szám.
- ▶ **Kérdés:** Kongruens-e  $m$ ?

Tehát azt kérdezzük, hogy létezik-e olyan derékszögű háromszög, mely oldalai racionális hosszúságúak és melynek területe  $m$ .

## Példa

1, 2, 3, 4 **nem** kongruensek. 5 kongruens (Fibonacci):

$$a = \frac{3}{2}, b = \frac{20}{3}, c = \frac{41}{6}$$

**Nem ismert**, hogy a probléma algoritmikusan eldönthető-e.

# Bonyolultsági osztályok

## Bonyolultsági osztály megadása

- ▶ számítási modell (általában nem túl fontos)
- ▶ számítási mód
- ▶ korlátozandó erőforrás
- ▶ korlát:  $f : \mathbb{N} \rightarrow \mathbb{N}$

## Definíció

Az  $f(n)$  függvény **megengedett bonyolultsági függvény**, ha monoton nemcsökkenő, és létezik olyan  $M_f$   $k$ -szavas lyukszalagos Turing-gép, mely  $\mathcal{O}(n + f(n))$  időben és  $\mathcal{O}(f(n))$  tárbán kiszámítja  $f$ -et a következő értelemben:

Minden  $n$  hosszú  $x$  bemenetre

$$(s, \triangleright, x, \triangleright, \varepsilon, \dots, \triangleright, \varepsilon) \xrightarrow{M_f^t} (h, \triangleright, x, \triangleright, \sqcup^{j_1}, \dots, \triangleright, \sqcup^{j_{k-2}}, \triangleright \sqcap^{f(n)}, \varepsilon),$$

ahol a  $t = \mathcal{O}(f(n) + n)$  és  $j_i = \mathcal{O}(f(n))$  csak  $n$ -től függenek.

Idő esetén  $f(n) > n$  is kikötés!

## Példák

- ▶ Minden konstans függvény.
- ▶  $\lceil \log n \rceil, 2^n, n$ .
- ▶  $f, g \Rightarrow f + g, f \cdot g, f^g$ .
- ▶ Tehát minden polinom.

## Definíció

Egy  $M$  (lyukszalagos vagy nem, determinisztikus vagy nondeterminisztikus) Turing-gép **pontos**, ha léteznek olyan  $f$  és  $g$  függvények, hogy  $M$  bármely  $n$  hosszú bemenetéhez tartozó számítási sorozata **pontosan**  $f(n)$  hosszú és a megállás pillanatában  $M$  minden szava (lyukszalagos gép esetében az első és az utolsó kivételével) **pontosan**  $g(n)$  hosszú.

## Állítás

Tegyük fel, hogy az  $M$  (determinisztikus vagy nemdeterminisztikus) Turing-gép  $f(n)$  időben (vagy tárral) eldönti az  $L$  nyelvet, ahol  $f$  megengedett.

Ekkor létezik olyan  $M'$  **pontos** Turing-gép, mely  $\mathcal{O}(f(n))$  időben (vagy tárral) dönti el az  $L$  nyelvet.

(Tehát  $M'$  még nemdeterminisztikus esetben is mindig megáll!)

## Bizonyítás

$M'$  egy adott  $n$  hosszú  $x$  bemeneten az „ $f(n)$ -t kiszámító” Turing-gép szimulálásával kezdi működését, melynek végén egy munkaszalagon előáll az  $\sqcap^{f(n)}$  szó.

Feltehető, hogy minden további munkaszalag hossza is ennyi az első fázis után.

Ezután idő esetén az  $\sqcap^{f(n)}$  szót mint órát használva pontosan  $f(n)$  lépésig szimulálja  $M$  működését (esetleg üres lépésekkel).

Tár esetén valamely alkalmas  $c$  konstansra  $c^{f(n)}$  lépésig szimulálja  $M$ -et.

# Jelölések

$\text{TIME}(f(n))$ ,

$\text{SPACE}(f(n))$ ,

$\mathbf{P} = \text{TIME}(n^k)$ ,

$\mathbf{PSPACE} = \text{SPACE}(n^k)$ ,

$\mathbf{L} = \text{SPACE}(\log n)$ ,

$\mathbf{EXP} = \text{TIME}(2^{n^k})$

$\text{NTIME}(f(n))$

$\text{NSPACE}(f(n))$

$\mathbf{NP} = \text{NTIME}(n^k)$

$\mathbf{NPSPACE} = \text{NSPACE}(n^k)$

$\mathbf{NL} = \text{NSPACE}(\log n)$



# A hierarchia tételek

## Tétel

Ha  $f(n) > n$  megengedett bonyolultsági függvény, akkor

$$\text{TIME}(f(n)) \subsetneq \text{TIME}( (f(2n + 1))^3 ).$$

## Bizonyítás

Legyen

$$H_f = \{M; x : M \text{ legfeljebb } f(|x|) \text{ lépésben elfogadja } x\text{-et}\},$$

ahol  $M$  tetszőleges többszavas Turing-gép.

**Állítás:**  $H_f \in \text{TIME}( (f(n))^3 )$ .

**Állítás:**  $H_f \notin \text{TIME}( f(\lfloor n/2 \rfloor) )$ .

## Tétel

Ha  $f(n) > n$  megengedett bonyolultsági függvény, akkor

$$\text{TIME}(f(n)) \subsetneq \text{TIME}( f(n) \log^2(f(n)) ).$$

# A hierarchia tételek

Példa

$$\mathbf{P} \subseteq \text{TIME}(2^n) \subsetneq \text{TIME}( (2^{2n+1})^3 ) \subseteq \text{TIME}(2^{n^2}) \subseteq \mathbf{EXP}.$$

Tehát  $\mathbf{P} \subsetneq \mathbf{EXP}$ .

Tétel

Ha  $f(n) \geq n$  megengedett bonyolultsági függvény, akkor

$$\text{SPACE}(f(n)) \subsetneq \text{SPACE}(f(n) \log f(n)).$$

Példa

$$\mathbf{L} \subseteq \text{SPACE}(n) \subsetneq \text{SPACE}(n^2) \subseteq \mathbf{PSPACE}.$$

Tehát  $\mathbf{L} \subsetneq \mathbf{PSPACE}$ .

# Néhány alapvető összefüggés bonyolultsági osztályokra

## Tétel

(a)  $\text{TIME}(f(n)) \subseteq \text{NTIME}(f(n))$  és hasonlóan tárra

(b)  $\text{NTIME}(f(n)) \subseteq \text{SPACE}(f(n))$

(c)  $\text{NSPACE}(f(n)) \subseteq \text{TIME}(k^{\log n + f(n)})$

## Bizonyítás

(a) triviális.

$$\text{NTIME}(f(n)) \subseteq \text{SPACE}(f(n))$$

(b) Legyen  $M$   $f(n)$  időkorlátos nemdeterminisztikus Turing-gép. Feltehető, hogy minden számítási sorozat  $f(n)$  lépésből áll  $n$  hosszú  $x$  bemeneten, és minden nem megállási konfigurációnak  $d > 0$  „leszármazottja” van. Így a **nemdeterminisztikus választási sorozatokat reprezentálhatjuk az  $\{1, \dots, d\}^{f(n)}$  halmaz elemeivel.**  $M'$  ezeket generálja lexikografikusan, és a „tárat újra felhasználva” szimulálja minden sorozatra  $M$  működését.

$M'$  megáll „igen” állapotban, ha  $M$  „igen” állapotban áll meg.

$M'$  akkor áll meg „nem” állapotban, ha  $M$  egyetlen választási sorozat esetén sem állt meg „igen” állapotban.

Mivel  $f(n)$  megengedett, az első választási sorozatot generálni tudjuk ( $1^{f(n)}$ ).

Tárigény:  $f(n)$ .

$$\text{NSPACE}(f(n)) \subseteq \text{TIME}(k^{\log n + f(n)})$$

(c) bizonyítása:

$M$  legyen az  $L$  nyelvet  $f(n)$  tárral eldöntő nemdeterminisztikus Turing-gép. Mivel  $M$  lyukszalagos, az utolsó szalag tartalmára nincs szükség.

Adott  $x$ ,  $n$ -hosszú bemenethez tartozó konfiguráció:

$$(q, i, w_2, u_2, \dots, w_{k-1}, u_{k-1})$$

**Ezek száma:**  $nc_1^{f(n)} = c_1^{\log n + f(n)}$ .

**Kérdés:** Elérhető-e az  $x$ -hez tartoz konfigurációból egy („igen”, . . .) alakú konfiguráció?

Ez az ELÉRHETŐSÉG probléma, mely lineáris időben megoldható. Tehát létezik olyan  $c$  konstans, hogy  $L$  eldönthető Turing-géppel  $c^{\log n + f(n)}$  időben.

A konfigurációs gráf elkészíthető „előre” is, de szebb megoldás az, hogy szükség esetén az  $x$  bemenet és  $M$  gép ismeretében eldöntjük minden esetben, hogy két konfiguráció között van-e él.

Következmény

$$\mathbf{L} \subseteq \mathbf{NL} \subseteq \mathbf{P} \subseteq \mathbf{NP} \subseteq \mathbf{PSPACE} \subseteq \mathbf{EXP}.$$

Bizonyítás

$$\mathbf{L} \subseteq \mathbf{NL} \quad (\text{a})$$

$$\mathbf{NL} = \mathbf{NSPACE}(\log n) \subseteq \mathbf{TIME}(k^{\log n}) \subseteq \mathbf{P} \quad (\text{c})$$

$$\mathbf{P} \subseteq \mathbf{NP} \quad (\text{a})$$

$$\mathbf{NP} = \mathbf{NTIME}(n^k) \subseteq \mathbf{SPACE}(n^k) = \mathbf{PSPACE} \quad (\text{b})$$

$$\mathbf{PSPACE} = \mathbf{SPACE}(n^k) \subseteq \mathbf{TIME}(2^{n^k}) = \mathbf{EXP} \quad (\text{c})$$

Mivel  $\mathbf{P} \subsetneq \mathbf{EXP}$ , a  $\mathbf{P} \subseteq \mathbf{NP}$ ,  $\mathbf{NP} \subseteq \mathbf{PSPACE}$ ,  $\mathbf{PSPACE} \subseteq \mathbf{EXP}$  valamelyike biztosan valódi.

Mivel  $\mathbf{L} \subsetneq \mathbf{PSPACE}$ , az  $\mathbf{L} \subseteq \mathbf{NL}$ ,  $\mathbf{NL} \subseteq \mathbf{P}$ ,  $\mathbf{P} \subseteq \mathbf{NP}$ ,  $\mathbf{NP} \subseteq \mathbf{PSPACE}$  valamelyike biztosan valódi.

Sejtés

Mindegyik az.

# Savitch tétele

## Tétel

ELÉRHETŐSÉG  $\in$  SPACE( $\log^2(n)$ ).

## Bizonyítás

(Itt  $n$  akár a csúcsok számát, akár a szomszédsági mátrix tárolásához szükséges bitek számát is jelölheti.)

Legyen  $G$  egy  $n$  csúcsú gráf, azt kérdezzük, létezik-e 1-ből  $n$ -be vagy általánosabban  $x_0$ -ból  $y_0$ -ba.

$\text{ÚT}(x, y, i) \Leftrightarrow$  létezik  $x$ -ből  $y$ -ba vezető  
legfeljebb  $2^i$  hosszúságú út

Világos, hogy akkor és csak akkor érhető el  $y_0$  az  $x_0$ -ból, ha  $\text{ÚT}(x_0, y_0, \lceil \log n \rceil)$ .



# ÚT( $x, y, i$ ) számítása

$i = 0$  :

$$\text{ÚT}(x, y, 0) = \begin{cases} \text{„igen”} & \text{ha } x = y \text{ vagy } (x, y) \text{ él} \\ \text{„nem”} & \text{különben.} \end{cases}$$

$i > 0$  :

Ellenőrizzük minden  $z$ -re, hogy  $\text{ÚT}(x, z, i - 1)$  és

$\text{ÚT}(z, y, i - 1)$  igaz-e. Ha **létezik ilyen  $z$** , akkor

$\text{ÚT}(x, y, i) = \text{„igen”}$ , különben  $\text{ÚT}(x, y, i) = \text{„nem”}$ .

Készíthetünk olyan 3-szavas lyukszalagos Turing-gépet, mely ezt az algoritmust megvalósítja.

- ▶ 1. szalag:  $G, x_0, y_0$  – bemenet
- ▶ 2. szalag: verem, rekurzív hívási lánc, kezdetben  $(x_0, y_0, \lceil \log n \rceil)$
- ▶ 3. szalag: válasz

Hívási lánc hossza:  $\lceil \log n \rceil$

Egy elem hossza:  $\mathcal{O}(\lceil \log n \rceil)$

Összes tárigény:  $\mathcal{O}(\log^2 n)$

## Következmény

Ha  $f(n) \geq \log n$  megengedett bonyolultsági függvény, akkor

$$\text{NSPACE}(f(n)) \subseteq \text{SPACE}( (f(n))^2 ).$$

Speciálisan:

$$\text{PSPACE} = \text{NPSPACE}.$$

## Bizonyítás

Legyen  $N$  nemdeterminisztikus,  $f(n)$  tárkorlátos lyukszalagos Turing-gép. Az  $N$  egy  $x$  bemenetén való szimulálásához **futtassuk le az előző algoritmust az  $N$  konfigurációs gráján.** (Feltehető, hogy  $N$  mindig megáll.) A gráf mérete:  $c^{f(n)}$ , így adódik a  $\log^2(c^{f(n)}) = \mathcal{O}( (f(n))^2 )$  tárkorlát. (A konfigurációs gráfot nem készítjük el előre.)

# Az Immerman-Szelepcsényi tétel

## Tétel

Létezik olyan **nemdeterminisztikus** lyukszalagos Turing-gép, mely **logaritmikus tárral** kiszámítja adott  $G$  gráfra és annak  $x$  csúcsára az  **$x$ -ből elérhető csúcsok számát**.

## Megjegyzés

Egy **nemdeterminisztikus gép akkor számít ki egy  $F(x)$  függvényt**, ha minden  $x$  bemenetre minden számítási sorozat végén vagy  $h$  vagy „nem” állapotban áll meg (tehát nem léteznek végtelen számítási sorozatok), és minden  $x$  bemenetre létezik olyan számítási sorozat, melynek végén  $h$  állapotban áll meg. Ebben az esetben az utolsó szalag tartalma **mindig  $F(x)$** . Ha még a munkaszavak hosszának összege mindig legfeljebb  $f(|x|)$ , akkor a gép  **$f(n)$  tárral** számítja ki  $F(x)$ -et.

## Bizonyítás

Jelölje  $n$  a csúcsok számát és  $k = 0, 1, \dots, n - 1$  esetén legyen  $S(k)$  az  $x$ -ből legfeljebb  $k$  hosszú úton elérhető csúcsok halmaza. Nekünk  $|S(n - 1)|$ -et kell meghatároznunk.

Világos, hogy  $S(0) = \{x\}$ ,  $|S(0)| = 1$ .

Az  $|S(k)|$  meghatározásához  $|S(k - 1)|$ -et használjuk fel, továbbá egy olyan eljárást, mely választ ad az  $u \in S(k)$  kérdésre.  $u = 1, 2, \dots, n$ -re meghívjuk ezt az eljárást:

1.  $m := 0$ ; válasz := nem;
2. Minden  $v = 1, 2, \dots, n$  csúcsra
  - 2.1 ha  $v \in S(k - 1)$ , akkor  $m := m + 1$ ;
  - 2.2 ha továbbá  $(v, u)$  él vagy  $v = u$ , akkor válasz := igen;
3. ha végül  $m < |S(k - 1)|$ , akkor „nem”
4. különben az eredmény válasz értéke.

Itt a  $v \in S(k-1)$  kérdésre egy „pontatlan” nondeterminisztikus algoritmus ad választ, mely megsejt egy legfeljebb  $k$  hosszú csúcssorozatot és ellenőrzi, hogy az egy  $x$ -ből  $v$ -be vezető út-e.

**Tárigény:** arányos az egy csúcs tárolásához szükséges területtel:  $\mathcal{O}(\log n)$ . (Az út egyes pontjait egyenként sejtjük meg.)

Következmény

Ha  $f(n) \geq \log n$  megengedett bonyolultsági függvény, akkor

$$\mathbf{NSPACE}(f(n)) = \mathbf{coNSPACE}(f(n)).$$

## Bizonyítás

Tegyük fel, hogy  $L$ -et eldönti egy  $f(n)$  tárkorlátos nemdeterminisztikus Turing-gép, melyről feltehető, hogy mindig megáll. Adott  $n$  hosszú bemeneten a konfigurációk száma  $\leq c^{f(n)}$ , így nemdeterminisztikus Turing-géppel  $\mathcal{O}(f(n))$  tárral meghatározhatjuk a kezdő konfigurációból elérhető konfigurációk számát, majd ennek alapján – az előző algoritmusban adott módszerhez hasonlóan – eldönthetjük, hogy ezek között van-e („igen”, ...) alakú.

## Megjegyzés

$$\begin{aligned}\text{TIME}(f(n)) &= \text{coTIME}(f(n)) \\ \text{SPACE}(f(n)) &= \text{coSPACE}(f(n))\end{aligned}$$

# Visszavezetések

Legyenek  $A$  és  $B$  problémák.  $B$  legalább olyan nehéz, mint  $A$ , ha létezik olyan hatékony módszer, azaz  $f$  rekurzív függvény, mely az  $A$  tetszőleges  $x$  bemenetéhez hozzárendeli a  $B$  egy  $f(x)$  bemenetét (példányát) úgy, hogy az  $A$ -nak  $x$  akkor és csak akkor „igen” példánya, ha  $B$ -nek  $f(x)$  „igen” példánya. Az  $f$  függvényre további megszorításokat teszünk.

## Definíció

Azt mondjuk, hogy az  $L_1$  nyelv (logaritmikus tárral) visszavezethető az  $L_2$  nyelvre, ha létezik olyan  $\mathcal{O}(\log n)$  tárral kiszámítható  $R$  szófüggvény, hogy minden  $x$  bemenetre

$$x \in L_1 \Leftrightarrow R(x) \in L_2.$$



## Állítás

Ha  $R$  egy  $M$  (lyukszalagos) Turing-géppel kiszámított visszavezetés, akkor  $M$  minden  $x$  bemeneten **polinom időben** megáll.

## Bizonyítás

Adott  $n$  hosszú  $x$  bemeneten a konfigurációk száma  $\mathcal{O}(nc^{\log n}) = \mathcal{O}(n^k)$ . Ezek egyike sem fordulhat elő kétszer egy számítási sorozatban.

Tehát minden logaritmikus tárral kiszámított visszavezetés egyben polinom idejű visszavezetés is. Továbbá  $|R(x)|$  az  $|x|$  polinom függvényével korlátozható.

## Példa

PÁROSÍTÁS visszavezethető MAXIMÁLIS FOLYAM(E)-re.

## HAMILTON-ÚT

- ▶ **Adott:**  $G$  irányított gráf.
- ▶ **Kérdés:** létezik-e olyan út, mely minden csúcsot pontosan egyszer látogat meg?

## SAT

- ▶ **Adott:** konjunktív normálalakú Boole-formula.
- ▶ **Kérdés:** kielégíthető-e?

## Állítás

HAMILTON-ÚT visszavezethető SAT-ra.

## Bizonyítás

$G$  csúcsai:  $1, 2, \dots, n$ . A formula felírásához az  $x_{i,j}$ ,  $1 \leq i, j \leq n$  változókat használjuk. (Az  $x_{i,j}$  igaz volta annak felel meg, hogy a  $j$  csúcs az  $i$ -edik egy Hamilton-úton.)

A formulát öt részformula konjunkciójaként állítjuk elő:

$$\forall j \exists i x_{i,j} - \bigwedge_j (x_{1,j} \vee x_{2,j} \vee \dots \vee x_{n,j})$$
$$\forall j \forall i_1 < i_2 (\neg x_{i_1,j} \vee \neg x_{i_2,j}) - \bigwedge_j \bigwedge_{i_1 < i_2} (\neg x_{i_1,j} \vee \neg x_{i_2,j})$$

A kettő együtt:  $\forall j \exists! i x_{i,j}$

$$\forall i \exists j x_{i,j} - \bigwedge_i (x_{i,1} \vee x_{i,2} \vee \dots \vee x_{i,n})$$
$$\forall i \forall j_1 < j_2 (\neg x_{i,j_1} \vee \neg x_{i,j_2}) - \bigwedge_i \bigwedge_{j_1 < j_2} (\neg x_{i,j_1} \vee \neg x_{i,j_2})$$

Együtt:  $\forall i \exists! j x_{i,j}$ .

Eddig csak  $n$ -től függ.

$\forall (i,j) \notin E(G)$  az  $i$  csúcs után nem következhet  $j$ :

$$\bigwedge_{(i,j) \notin E(G)} \bigwedge_{k=1}^{n-1} (\neg x_{k,i} \vee \neg x_{k+1,j})$$

Adott  $G$  gráfra a formula logaritmikusan elkészíthető, és  $G$ -ben akkor és csak akkor van Hamilton-út, ha a formula kielégíthető.

# Logikai hálózatok

## Definíció

**Hálózat:** körmentes irányított gráf,  
a csúcsok címkézettek:  $\wedge$ ,  $\vee$ ,  $\neg$ , igaz, hamis,  $x_i$   
 $\wedge$ ,  $\vee$  címke esetén a csúcs befoka 2,  
 $\neg$  címke esetén a csúcs befoka 1,  
igaz, hamis,  $x_i$  címke esetén a csúcs befoka 0.

Általában megköveteljük még, hogy pontosan egy csúcs kifoka legyen 0.

Amennyiben a változók közül az  $x_1, \dots, x_n$  fordulnak elő a hálózatban (legfeljebb), akkor a hálózat kiszámít egy  $\{\text{igaz, hamis}\}^n \rightarrow \{\text{igaz, hamis}\}$  Boole-függvényt.

## HÁLÓZAT-KIÉRTÉKELÉS

- ▶ **Adott:** változómentes hálózat.
- ▶ **Kérdés:** igaz-e az értéke?

## HÁLÓZAT-KIELÉGÍTHETŐSÉG

- ▶ **Adott:** hálózat.
- ▶ **Kérdés:** a változóknak lehet-e úgy értéket adni, hogy a hálózat értéke igaz legyen?

## Állítás

Az ELÉRHETŐSÉG visszavezethető a HÁLÓZAT-KIÉRTÉKELÉS-re.

## Állítás

A HÁLÓZAT-KIELÉGÍTHETŐSÉG visszavezethető a SAT-ra.

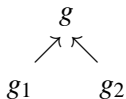
Minden  $g$  csúcsnak feleljen meg a  $g$  változó.

$g$  címkéje  $x$   $\mapsto x \leftrightarrow g$ , azaz  $(\neg x \vee g) \wedge (x \vee \neg g)$

$g$  címkéje igaz  $\mapsto g$

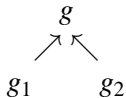
$g$  címkéje hamis  $\mapsto \neg g$

$g$  címkéje  $\vee$ :



$\mapsto g \leftrightarrow (g_1 \vee g_2)$ , azaz  
 $(\neg g \vee g_1 \vee g_2) \wedge (\neg g_1 \vee g) \wedge (\neg g_2 \vee g)$

$g$  címkéje  $\wedge$ :



$\mapsto g \leftrightarrow (g_1 \wedge g_2)$ , azaz  
 $(\neg g \vee g_1) \wedge (\neg g \vee g_2) \wedge (\neg g_1 \vee \neg g_2 \vee g)$

$g$  címkéje  $\neg$ :



$\mapsto g \leftrightarrow (\neg g_1)$ , azaz  
 $(\neg g \vee \neg g_1) \wedge (g_1 \vee g)$

$g$  kimenő kapu  $\mapsto g$

A keresett formula: a fenti formulák konjunkciója.

Adott hálózathoz a formula elkészíthető logaritmikus tárral. Továbbá a hálózat akkor és csak akkor kielégíthető, ha a formula az.

Ha ugyanis a formula kielégíthető, akkor tekintsünk egy  $h$  kielégítő értékelését. Ez minden egyes  $g$  kapuhoz és  $x_i$  változóhoz meghatároz egy logikai értéket. Rendeljük minden kapuhoz ezt a logikai értéket. Ekkor a kimeneti kapu értéke a hálózat értéke, amikor minden  $x_i$  változó értékét  $h(x_i)$ -nek választjuk.

Ha a hálózat kielégíthető, akkor válasszuk meg minden  $x_i$  értékét úgy, hogy a hálózat értéke igaz legyen. Számítsuk ki a hálózat minden kapujának értékét. Az így kapott értékelés kielégíti a formulát.



Az egyik irány: ha  $H_g$ -t kielégíti a  $H_g$  változóinak egy  $h$  kiértékelése, akkor  $h$ -nak az a folytatása, mely  $H_g$  minden  $g'$  csúcsához (mint változóhoz) a  $H_{g'}$  értékét rendeli a  $h$  értékelés mellett, kielégíti  $F_g$ -t.

A másik irány: ha  $F_g$ -t kielégíti egy  $h$  értékelés (mely értéket ad  $H_g$  változóinak és minden csúcsának), akkor  $h$  megszorítása  $H_g$  változóira kielégíti a  $H_g$  hálózatot.

# A visszavezetés tranzitivitása

Világos, hogy a visszavezetés reflexív.

## Tétel

A visszavezetés tranzitív: Ha  $R$  az  $L_1$ -nek  $L_2$ -re,  $R'$  pedig az  $L_2$ -nek  $L_3$ -ra való visszavezetése, akkor az  $R \circ R'$  összetett függvény  $L_1$ -nek  $L_3$ -ra való visszavezetése.

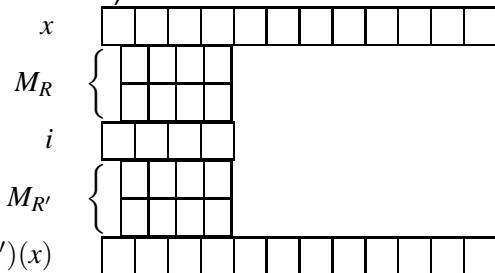
## Bizonyítás

Legyen  $M_R$  és  $M_{R'}$  az  $R$ -et ill.  $R'$ -t logaritmikus tárral kiszámító lyukszalagos Turing-gépek. Elkészítünk egy olyan lyukszalagos Turing-gépet, mely logaritmikus tárral kiszámítja  $R \circ R'$ -t. (Ez nem lehet a hagyományos szuperpozíció, mert  $R(x)$  sokkal hosszabb lehet, mint  $\log |x|$ .)

# A visszavezetés tranzitivitása

## A trükk

Nem tároljuk el  $M_R$  kimenő szalagját, csak a **mutató pozícióját** (kezdetben ez 1).



**$M_{R'}$ -nek szüksége van a következő karakterre:** folytatjuk  $M_R$  szimulálását.

**$M_{R'}$ -nek szüksége van az előző karakterre:** előlről kezdjük  $M_R$  szimulálását.

## Definíció

Azt mondjuk, hogy egy  $\mathcal{C}$  bonyolultsági osztály **zárt a visszavezetésre**, ha valahányszor  $L$  visszavezethető  $L'$ -re és  $L' \in \mathcal{C}$ , mindig teljesül, hogy  $L \in \mathcal{C}$ .

## Állítás

Az **L, NL, P, NP, PSPACE, EXP** osztályok zártak a visszavezetésre.

## Bizonyítás

**L** és **NL** esetén az előző tételhez (tranzitivitás) hasonlóan. A többi osztály esetében a hagyományos szuperpozíció is megfelel.

## Megjegyzés

**L** bármely két nemtriviális  $L, L'$  elemére igaz, hogy  $L$  visszavezethető  $L'$ -re.

## Definíció

Legyen  $\mathcal{C}$  egy bonyolultsági osztály. Egy  $L$  nyelvet  **$\mathcal{C}$ -nehéznek** nevezünk, ha minden  $L' \in \mathcal{C}$  nyelv visszavezethető  $L$ -re. Ha még  $L \in \mathcal{C}$  is teljesül, akkor  $L$   **$\mathcal{C}$ -teljes**.

**Speciálisan:** P-teljes, NP-teljes stb. problémák.

## Állítás

Tegyük fel, hogy  $\mathcal{C}$  zárt a visszavezetésre és az  $L$  nyelv  $\mathcal{C}$ -teljes. Akkor  $\mathcal{C}$  pontosan azon  $L'$  nyelvekből áll, melyek visszavezethetők  $L$ -re. Tehát  $L$  **reprezentálja az egész osztályt!**

## Állítás

Tegyük fel, hogy  $L$  egy  $\mathcal{C}$ -teljes nyelv,  $L' \in \mathcal{C}$  és  $L$  visszavezethető  $L'$ -re. Akkor  $L'$  is  $\mathcal{C}$ -teljes.

## Állítás

$L$  nyelv  $\mathcal{C}$ -teljes  $\Leftrightarrow \bar{L}$  co $\mathcal{C}$ -teljes.

# Egy $\mathbf{P}$ -teljes probléma

## Tétel

A HÁLÓZAT-KIÉRTÉKELÉS probléma  $\mathbf{P}$ -teljes.

## Bizonyítás

Csak azt mutatjuk meg, hogy a probléma  $\mathbf{P}$ -nehéz. E célból legyen  $L \in \mathbf{P}$ , be kell látnunk, hogy  $L$  visszavezethető a HÁLÓZAT-KIÉRTÉKELÉS problémára.

Mivel  $L \in \mathbf{P}$ , létezik olyan polinom időkorlátos egyszavas  $M$  Turing-gép, mely eldönti  $L$ -et. Tegyük fel, hogy  $p(n) - 2$  az  $M$  polinom időkorlátja, ahol  $p(n) > 2$ .

Ha megengedünk üres átmeneteket is, úgy feltehető, hogy  $M$  minden  $n$  hosszú  $x$  bemeneten pontosan  $p(n) - 2$  lépésben áll meg „igen” vagy „nem” állapotban.

# Egy P-teljes probléma

Azt is feltehetjük, hogy  $M$  a megálláskor a mutató a  $\triangleright$  szimbólumot követő karakteren áll, és ez a karakter az „igen” vagy a „nem” állapot.

$M$  működését az  $n$  hosszú  $x$  bemeneten leírhatjuk egy  $(p(n) - 1) \times p(n)$  méretű  $T = (T_{i,j})$  táblázattal,

$$T_{i,j} \in \Gamma = \Sigma \cup \{\sigma_q : \sigma \in \Sigma, q \in K\}.$$

Itt  $\sigma_q$  jelentése: a gép  $q$  állapotban van, a mutató az adott karaktert jelöli ki, mely  $\sigma$ .

Az egyszerűség kedvéért most feltesszük, hogy induláskor a gép a  $\triangleright$  szimbólumot követő karakterre mutat, és a  $\triangleright$  szimbólumra sosem kerül a mutató.

# Egy P-teljes probléma

|   |     |   |                          |
|---|-----|---|--------------------------|
| ▷ | $x$ | <input type="checkbox"/> <input type="checkbox"/> ... <input type="checkbox"/>                      | <input type="checkbox"/> |
| ▷ |     |   | <input type="checkbox"/> |
| ▷ |     |   | <input type="checkbox"/> |
| ▷ |     |   | <input type="checkbox"/> |
| ▷ |     | <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> <input type="checkbox"/> | <input type="checkbox"/> |
| ▷ |     | <input type="checkbox"/> <input type="checkbox"/>   | <input type="checkbox"/> |
| ▷ |     |   | <input type="checkbox"/> |
| ▷ |     |   | <input type="checkbox"/> |
| ▷ |     |   | <input type="checkbox"/> |

↑  
„igen” v. „nem”



# Egy P-teljes probléma

A szélső elemek adottak. Minden belső elem a fölötte levő három függvénye valamely  $M$ -től függő

$$f : \Gamma^3 \rightarrow \Gamma$$

függvény szerint.

Amennyiben  $\Gamma$  elemeit bitsorozatokkal kódoljuk,  $f$  helyettesíthető

$$f_\ell : \{\text{igaz, hamis}\}^{3m} \rightarrow \{\text{igaz, hamis}\}$$

( $\ell = 1, 2, \dots, m$ ,  $m = \lceil \log |\Gamma| \rceil$ ) függvényekkel.

Az  $x$  bemenethez tartozó  $R(x)$  hálózatot úgy kapjuk, hogy a táblázat minden belső pozícióját helyettesítjük egy egyszerű hálózattal (melyek véges sok,  $x$ -től függetlenül megadható hálózattípusból kerülnek ki).

# Egy P-teljes probléma

**szélső elem:** a megfelelő elem bináris kódjához tartozó, csak kezdőcsúcsokat tartalmazó hálózat.

**belső elem:**  $3m$  bemenettel és  $m$  kimenettel rendelkező, az  $f_1, \dots, f_m$  függvényeket kiszámító hálózat.

A belső elemek helyére kerülő hálózatok bemeneteit rendre azonosítjuk a fölötte levő három hálózat kimeneteivel.

**Az egész hálózat kimenete:** az utolsó sor második hálózatának első kimenete.

Világos, hogy

- ▶  $R(x)$  elkészíthető logaritmikus tárigényű Turing-géppel;
- ▶  $x \in L \Leftrightarrow R(x)$  értéke igaz.

# Cook tétele

## Tétel

A SAT probléma **NP**-teljes.

Az nyilvánvaló, hogy  $SAT \in \mathbf{NP}$ . Mivel a **HÁLÓZAT-KIELÉGÍTHETŐSÉG** visszavezethető SAT-ra, így elegendő belátni, hogy a **HÁLÓZAT-KIELÉGÍTHETŐSÉG** **NP**-nehéz. (Persze akkor **NP**-teljes is.)

# Cook tétele

Tétel'

A HÁLÓZAT-KIELÉGÍTHETŐSÉG **NP**-teljes.

Bizonyítás

Legyen  $L$  egy  $M$  nemdeterminisztikus egyszavas Turing-géppel polinom időben eldöntött nyelv. A HÁLÓZAT-KIÉRTÉKELÉS **P**-teljességének bizonyítását követve, adott  $x$  bemenethez elkészítünk egy  $T$  táblázatot, mely a nemdeterminisztikus választásoktól is függ. Feltehető, hogy minden konfigurációban 2 választás lehetséges, így  $c$  egy  $t = p(|x|)$  hosszú bitsorozat, ahol  $p$  az  $M$  időkorlátját megadó polinom függvény.

# Cook tétele

A belső  $T_{i,j}$  elemek értéke most a  $T_{i-1,j-1}$ ,  $T_{i-1,j}$ ,  $T_{i-1,j+1}$  értékeken kívül a  $c_i$ -től is függ, ami 0 vagy 1.

Így az  $f_\ell$  függvények:

$$\{\text{igaz, hamis}\}^{3m+1} \rightarrow \{\text{igaz, hamis}\}.$$

Az  $R(x)$  hálózatot a korábbiakhoz hasonlóan készítjük el azzal a különbséggel, hogy a hálózat rendelkezik még  $c_1, \dots, c_t$  bemeneti kapukkal, amelyek  $x_1, \dots, x_t$  változókkal címkézettek.

# Az NP osztály egy jellemzése

## Definíció

Legyen  $R \subseteq \Sigma^* \times \Sigma^*$ . Azt mondjuk, hogy  $R$  **polinom időben eldönthető**, ha az

$$L_R = \{x; y : R(x, y)\}$$

nyelv az.

Azt mondjuk, hogy  $R$  **polinomiálisan kiegyensúlyozott**, ha van olyan  $p(n)$  polinom, hogy

$$R(x, y) \Rightarrow |y| \leq p(|x|).$$

## Állítás

$L \in \mathbf{NP} \Leftrightarrow \exists R$  polinom időben eldönthető, polinomiálisan kiegyensúlyozott reláció, hogy

$$L = \{x : \exists y R(x, y)\}.$$

## Bizonyítás

⇒ Tegyük fel, hogy  $L \in \mathbf{NP}$ . Létezik  $L$ -et eldöntő,  $p(n)$  polinom időkorlátos nemdeterminisztikus Turing-gép.

Legyen

$$R = \{(x, y) : y \text{ az } x\text{-hez tartozó elfogadó számítási sorozatot kódol}\}.$$

⇐ Legyen  $L = \{x : \exists y R(x, y)\}$ , ahol  $R$  polinom időben eldönthető és polinomiálisan kiegyensúlyozott:

$$R(x, y) \Rightarrow |y| \leq p(|x|).$$

Az  $N$  nemdeterminisztikus Turing-gép adott  $x$  esetén nemdeterminisztikusan generáljon egy tetszőleges  $y$  szót, amire  $|y| \leq p(|x|)$ , majd az  $L_R$ -et polinom időben eldöntő Turing-gépet szimulálva ellenőrizze, hogy  $R(x, y)$  teljesül-e.

# A kielégíthetőség változatai

## Definíció

Legyen  $k \geq 1$ . A  $k$ SAT a SAT azon speciális esete, amelyben minden tag pontosan  $k$  (nem feltétlenül különböző) literálból áll.



# 3SAT

## Állítás

3SAT NP-teljes, és így  $k \geq 3$  esetén  $k$ SAT NP-teljes.

## Bizonyítás

$$\begin{aligned}l &\mapsto l \vee l \vee l \\l_1 \vee l_2 &\mapsto l_1 \vee l_2 \vee l_2 \\l_1 \vee l_2 \vee l_3 &\mapsto l_1 \vee l_2 \vee l_3 \\l_1 \vee l_2 \vee l_3 \vee l_4 &\mapsto (l_1 \vee l_2 \vee x) \wedge (\neg x \vee l_3 \vee l_4) \\&\vdots \\&\vdots \\l_1 \vee l_2 \vee \dots \vee l_n &\mapsto (l_1 \vee l_2 \vee x) \wedge (\neg x \vee l_3 \vee y) \wedge \\&\quad (\neg y \vee l_4 \vee z) \wedge \dots \wedge (\neg u \vee l_{n-1} \vee l_n)\end{aligned}$$

# 3SAT

## Állítás

Legyen  $\varphi = c_1 \wedge c_2 \wedge \dots \wedge c_k$  konjunktív normálformájú formula. Minden  $c_i$  taghoz legyen  $c'_i$  az előzőekben adott kifejezés, ahol minden kifejezésben új változókat használunk.

Ekkor  $\varphi$  akkor és csak akkor kielégíthető, ha

$\varphi' = c'_1 \wedge c'_2 \wedge \dots \wedge c'_k$  az.

Mivel SAT visszavezethető 3SAT-ra és  $3SAT \in \mathbf{NP}$ , ezért SAT  $\mathbf{NP}$ -teljességéből következik, hogy 3SAT is  $\mathbf{NP}$ -teljes.

# 2SAT

Legyen  $\varphi$  olyan k.n.f., hogy  $\varphi$  minden tagja 2 literálból áll.

$G(\varphi)$

- ▶ csúcsok: a  $\varphi$ -ben előforduló változók és negáltjaik.
- ▶ élek:  $(\alpha, \beta)$  él  $\Leftrightarrow \bar{\alpha} \vee \beta$  vagy  $\beta \vee \bar{\alpha}$  a  $\varphi$  tagja (azaz ha  $\alpha \rightarrow \beta$  vagy  $\bar{\beta} \rightarrow \bar{\alpha}$  a  $\varphi$  tagja.)

Állítás

Akkor és csak akkor vezet  $\alpha$ -ból  $\beta$ -ba irányított út, ha létezik  $\bar{\beta}$ -ból  $\bar{\alpha}$ -ba vezető irányított út.

Tétel

A  $\varphi$  formula akkor és csak akkor kielégíthető, ha nincs olyan  $x$  változó, amelyre létezik  $G(\varphi)$ -ben irányított út  $x$ -ből  $\neg x$ -be és vissza.

## Bizonyítás

Tegyük fel, hogy létezik olyan  $x$  változó, hogy  $x$ -ből elérhető  $\neg x$ , és  $\neg x$ -ből elérhető  $x$ .

Belátjuk, hogy a változók tetszőleges  $T$  kiértékelésére  $\varphi$  hamis.

**Tegyük fel, hogy**  $T(x)$  igaz (a  $T(x)$  hamis eset hasonlóan kezelhető). Mivel  $T(\neg x)$  hamis, az  $x$ -ből  $\neg x$ -be vezető úton létezik olyan  $(\alpha, \beta)$  él, hogy  $T(\alpha)$  igaz és  $T(\beta)$  hamis. Ekkor  $G(\varphi)$  konstrukciója szerint a megfelelő tag  $(\bar{\alpha} \vee \beta$  vagy  $\beta \vee \bar{\alpha})$  hamis, így  $\varphi$  hamis.

**Tegyük fel, hogy** egyetlen  $x$  változóra sem létezik  $x$ -ből  $\neg x$ -be és  $\neg x$ -ből  $x$ -be vezető út. Elkészítjük a változók egy olyan  $T$  kiértékelését, melyre  $\varphi$  igaz.

## 2SAT

Válasszunk egy olyan  $x$  változót, amelynek igazságértékét még nem rögzítettük. Ha  $x$ -ből nem érhető el  $\neg x$ , akkor legyen  $\alpha := x$ , különben (akkor  $\neg x$ -ből nem érhető el  $x$ )  $\alpha := \neg x$ .  $\alpha$ -hoz és minden  $\alpha$ -ból elérhető csúcshoz rendeljük az igaz értéket és tagadásaikhoz rendeljük a hamis értéket.

(Ezek pontosan azok, melyekből elérhető  $\bar{\alpha}$ .)

$T$  jól definiált, **nem lehet**:  $\alpha \rightsquigarrow \beta$  és  $\alpha \rightsquigarrow \bar{\beta}$ , mert ekkor  $\bar{\beta} \rightsquigarrow \bar{\alpha}$  és  $\beta \rightsquigarrow \bar{\alpha}$  miatt  $\alpha \rightsquigarrow \bar{\alpha}$  lenne.

$\alpha \rightsquigarrow \beta$  esetén **nem lehet**, hogy  $\beta$  korábban hamis értéket kapjon, mert akkor  $\bar{\beta} \rightsquigarrow \bar{\alpha}$  alapján  $\alpha$  már korábban hamis értéket kapott volna.

Valahányszor  $(\alpha, \beta)$  él és  $T(\alpha)$  igaz,  $T(\beta)$  is igaz. Ezért  $\varphi$  **a  $T$  kiértékelés mellett igaz**.

# 2SAT

## Következmény

$2SAT \in \mathbf{NL}$ , így  $2SAT \in \mathbf{P}$ .

## Bizonyítás

Mivel  $\mathbf{NL} = \mathbf{coNL}$ , elegendő azt belátni, hogy

$2SAT - \text{KOMPLEMENTER} \in \mathbf{NL}$ .

Ehhez azt kell eldönteni adott  $\varphi$  esetén, hogy létezik-e  $G(\varphi)$ -ben valamely  $x$  változóra  $x$ -ből  $\neg x$ -be és vissza vezető út. Ez ugyanúgy eldönthető nemdeterminisztikusan logaritmikus tárral, mint az ELÉRHETŐSÉG.

# NP-teljes gráfelméleti problémák

## FÜGGETLEN CSÚCSHALMAZ

**Adott:**  $G = (V, E)$  irányítatlan gráf,  $K$  szám.

**Kérdés:** Létezik-e  $K$  elemű független csúcshalmaz?

## Tétel

A FÜGGETLEN CSÚCSHALMAZ probléma NP-teljes.

## Bizonyítás

Világos, hogy a probléma NP-ben van. Megmutatjuk, hogy 3SAT visszavezethető a FÜGGETLEN CSÚCSHALMAZ-ra. Legyen  $\varphi$  a 3SAT egy példánya,  $\varphi = c_1 \wedge \dots \wedge c_m$ . A  $G$  gráf álljon  $m$  darab **háromszögből**, melyek a  $c_i$  tagoknak felelnek meg, s melyek csúcsai a  $c_i$ -kben lévő literáloknak felelnek meg. Ezen kívül még kössünk össze éllel **minden ellentétes literált**. Végül legyen  $K = m$ .

$\varphi$  kielégíthető  $\Leftrightarrow G$ -ben létezik  $K$  elemű független csúcshalmaz.

# NP-teljes gráfelméleti problémák

## KLIKK

**Adott:**  $G = (V, E)$  irányítatlan gráf,  $K$  szám.

**Kérdés:** Létezik-e  $K$  elemű klikk (teljes részgráf)?

## Tétel

KLIKK NP-teljes.

## Bizonyítás

Világos, hogy  $\text{KLIKK} \in \text{NP}$ .

Legyen  $(G = (V, E), K)$  a FÜGGETLEN CSÚCSHALMAZ probléma egy példánya. Feltehető, hogy  $K \leq |V|$ .

Legyen  $G^c = (V, E^c)$  a  $G$  **komplementer gráfja**. Világos, hogy  $G$ -ben akkor és csak akkor létezik  $K$  elemű független csúcshalmaz, ha  $G^c$ -ben létezik  $K$  elemű klikk.



# NP-teljes gráfelméleti problémák

## CSÚCSLEFEDÉS

**Adott:**  $G = (V, E)$  irányítatlan gráf,  $K$  szám.

**Kérdés:** Létezik-e olyan  $K$  elemű csúcshalmaz, hogy minden él illeszkedik a halmaz egy csúcsához?

## Tétel

A CSÚCSLEFEDÉS probléma NP-teljes.

## Bizonyítás

Legyen  $(G = (V, E), K)$  a FÜGGETLEN CSÚCSHALMAZ probléma egy példánya, ahol  $K \leq |V|$ .

$G$ -nek akkor és csak akkor létezik  $K$  elemű független csúcshalmaza, ha élei lefedhetőek  $|V| - K$  elemű csúcshalmazzal.

## HAMILTON-ÚT

**Adott:**  $G = (V, E)$  irányítatlan gráf.

**Kérdés:** Létezik-e olyan út, melyben minden csúcs pontosan egyszer fordul elő?

### Tétel

A HAMILTON-ÚT probléma **NP**-teljes.

### Tétel

A TSP(E) probléma **NP**-teljes.

### Bizonyítás

A HAMILTON-ÚT problémát visszavezetjük a TSP(E) problémára.

Legyen  $G$  egy  $n$  csúcsú gráf az  $\{1, \dots, n\}$  halmazon. Ha  $i \neq j$  és  $(i, j)$  él, akkor  $d_{i,j}$  legyen 1, különben legyen  $d_{i,j} = 2$ .  $G$ -ben akkor és csak akkor létezik Hamilton-út, ha létezik  $\leq n + 1$  összköltségű körút.

# NP-teljes gráfelméleti problémák

## 3-SZÍNEZÉS

**Adott:**  $G = (V, E)$  irányítatlan gráf.

**Kérdés:** Kiszínezhetőek-e a gráf csúcsai 3 színnel úgy, hogy a szomszédos csúcsok különböző színűek legyenek?

## Tétel

A 3-SZÍNEZÉS probléma **NP**-teljes.

## Megjegyzés

A 2-SZÍNEZÉS probléma **P**-ben van.

A 4-SZÍNEZÉS probléma **triviális** síkbarajzolható gráfokra.

# NP-teljes problémák halmazokra és számokra

## HÁRMASÍTÁS

**Adott:** Három azonos méretű halmaz,  $F$  (fiúk),  $L$  (lányok),  $H$  (házak), és egy  $R \subseteq F \times L \times H$  reláció.

**Kérdés:** Megadható-e az  $R$ -beli hármások egy olyan részhalmaza, melyben minden fiú, lány és ház pontosan egyszer szerepel?

Világos, hogy HÁRMASÍTÁS  $\in$  NP.

Tétel

A HÁRMASÍTÁS probléma NP-teljes.

Megjegyzés

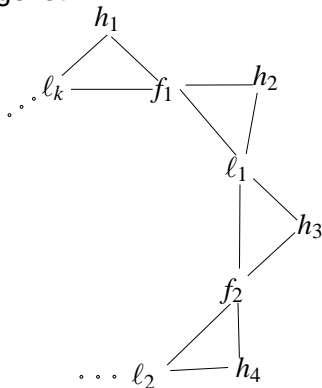
PÁROSÍTÁS  $\in$  P.

## Bizonyítás

A 3SAT-ot vezetjük vissza HÁRMASÍTÁS-ra.

$$\varphi = c_1 \wedge \dots \wedge c_m$$

Adott  $x$  változóra jelölje  $k$  az  $x$  és  $\neg x$  előfordulásainak maximumát. Az  $x$  változóhoz felvesszünk  $2k$  hármast, az ábrán látható háromszögeket:



# HÁRMASÍTÁS

## Bizonyítás folytatása

Az  $x$  minden egyes előfordulásához hozzárendelünk egy páratlan indexű  $h_i$ -t, a  $\neg x$  minden előfordulásához pedig egy páros indexű  $h_i$ -t.

Mivel az ábrán szereplő fiúk és lányok más hármásokban nem szerepelnek, minden hármásításban minden második háromszög szerepel:

|   |           |
|---|-----------|
| $(f_1, \ell_1, h_2), (f_2, \ell_2, h_4), \dots$ | $x$ igaz  |
| $(f_1, \ell_k, h_1), (f_2, \ell_1, h_3), \dots$ | $x$ hamis |

Minden  $c_j$  taghoz felvesszünk még 3 hármast, melyek  $(f, \ell, h)$  alakúak, ahol  $f, \ell$  csak a  $c_j$ -től függnek,  $h$  pedig azon három ház valamelyike, melyek a tagban szereplő literáloknak felelnek meg.

Az eddig elkészített  $R$  relációra érvényes:

**Akkor és csak akkor adható meg az  $R$  egy olyan részhalmaza, melyben minden fiú és lány pontosan egyszer, és minden ház legfeljebb egyszer szerepel, ha  $\varphi$  kielégíthető.**

# HÁRMASÍTÁS

## A konstrukció befejezése

Eddig  $H \geq 3m$  házat és  $\frac{H}{2} + m \leq \frac{H}{2} + \frac{H}{3}$  fiút és ugyanannyi lányt használtunk.

Jelölje  $K$  a  $H - (\frac{H}{2} + m)$  különbséget. Felveszünk még  $K$  lányt és fiút, melyek mindegyik házba belerakhatók.

## HALMAZLEFEDÉS

**Adott:** Egy  $U$  halmaz részhalmazainak  $C = (S_1, \dots, S_n)$  rendszere és egy  $K$  szám.

**Kérdés:** Kiválasztható-e  $K$  darab az  $S_i$ -k közül úgy, hogy ezek egyesítése  $U$ ?

## HALMAZPAKOLÁS

**Adott:** Egy  $U$  halmaz részhalmazainak  $C = (S_1, \dots, S_n)$  rendszere és egy  $K$  szám.

**Kérdés:** Kiválasztható-e az  $S_i$ -k közül  $K$  páronként diszjunkt halmaz?

## PONTOS LEFEDÉS HÁRMASOKKAL

**Adott:** Egy  $3m$  elemű  $U$  halmaz és 3-elemű részhalmazainak  $(S_1, \dots, S_n)$  rendszere.

**Kérdés:** Kiválasztható-e  $m$  darab  $S_i$  úgy, hogy ezek egyesítése  $U$ ? (A kiválasztott  $S_i$ -k nyilván diszjunktak.)



# NP-teljes problémák halmazokra

## Tétel

A HALMAZLEFEDÉS, HALMAZPAKOLÁS, PONTOS LEFEDÉS HÁRMASOKKAL problémák NP-teljesek.

## Bizonyítás

A HÁRMASÍTÁS a PONTOS LEFEDÉS HÁRMASOKKAL **speciális esete**.

A PONTOS LEFEDÉS HÁRMASOKKAL a HALMAZLEFEDÉS, valamint a HALMAZPAKOLÁS speciális esete is.

## EGÉSZ ÉRTÉKŰ PROGRAMOZÁS

**Adott:** Egy  $n$ -változós, egész együtthatós lineáris egyenlőtlenség-rendszer.

**Kérdés:** Létezik-e **egész értékű** megoldás?

A HALMAZLEFEDÉS felfogható az EGÉSZ ÉRTÉKŰ PROGRAMOZÁS speciális eseteként:

$$Ax \geq 1, \quad \sum_{i=1}^n x_i \leq B, \quad 0 \leq x_i \leq 1,$$

ahol  $A$  oszlopai a halmazrendszer elemeinek felelnek meg. Azt is meg lehet mutatni, hogy az EGÉSZ ÉRTÉKŰ PROGRAMOZÁS NP-ben van.

**Tétel**

Az EGÉSZ ÉRTÉKŰ PROGRAMOZÁS probléma NP-teljes.

# NP-teljes problémák halmazokra és számokra

Az EGÉSZ ÉRTÉKŰ PROGRAMOZÁS egy másik speciális esete:

## HÁTIZSÁK

**Adott:**  $n$  elem mindegyikének  $w_i$  súlya és  $c_i$  értéke, valamint  $W$  és  $K$ .

**Kérdés:** Kiválasztható-e ismétlés nélkül néhány elem úgy, hogy összértékük  $\geq K$  és összsúlyuk  $\leq W$ ?

## Tétel

A HÁTIZSÁK probléma NP-teljes.

# A coNP osztály

Definíció

$$\text{coNP} = \{L : \bar{L} \in \text{NP}\}$$

Példák

**ÉRVÉNYESSÉG**

**Adott:**  $\varphi$  Boole-formula

**Kérdés:** Érvényes-e, azaz azonosan igaz-e  $\varphi$ ?

**HAMILTON-ÚT KOMPLEMENTER**

**Adott:**  $G$  gráf

**Kérdés:** Igaz-e, hogy  $G$  nem tartalmaz Hamilton-utat?

Állítás

Az ÉRVÉNYESSÉG és HAMILTON-ÚT KOMPLEMENTER problémák coNP-ben vannak.

# NP és coNP

Egy probléma akkor van

- ▶ **NP**-ben, ha minden igen példányához létezik tömör, polinom időben ellenőrizhető bizonyíték, és csak az igen példányokhoz létezik ilyen bizonyíték.
- ▶ **coNP**-ben, ha minden nem példányhoz létezik tömör, polinom időben ellenőrizhető cáfolat, és csak a nem példányokhoz létezik ilyen cáfolat.

Állítás

Az ÉRVÉNYESSÉG és HAMILTON-ÚT KOMPLEMENTER problémák **coNP**-teljesek.

# NP és coNP

Állítás

$$\mathbf{P} \subseteq \mathbf{NP} \cap \mathbf{coNP}.$$

Állítás

Tegyük fel, hogy  $\mathcal{C}$  zárt a visszavezetésre és  $L$   $\mathcal{C}$ -teljes.

Ekkor  $\mathcal{C} = \mathbf{co}\mathcal{C} \Leftrightarrow L \in \mathbf{co}\mathcal{C}$ .

Következmény

$$\mathbf{NP} = \mathbf{coNP} \Leftrightarrow \mathbf{SAT} \in \mathbf{coNP} \Leftrightarrow \mathbf{ÉRVÉNYESSÉG} \in \mathbf{NP}.$$

## PRÍMEK

**Adott:**  $p$  szám (binárisan).

**Kérdés:** Prímszám-e  $p$ ?

### Állítás

PRÍMEK  $\in$  coNP.

### Tétel (Lucas)

Egy  $p > 1$  szám akkor és csak akkor prímszám, ha létezik olyan  $1 \leq r < p$ , amelyre:

(a)  $r^{p-1} \equiv 1 \pmod{p}$

(b) A  $p - 1$  minden  $q$  prímszám osztójára  $r^{(p-1)/q} \not\equiv 1 \pmod{p}$ .

### Példa

$$p = 5 \rightarrow r = 3: 3^4 \equiv 1 \pmod{5}, 3^2 \equiv 4 \pmod{5}.$$

# Pratt tétele

Tétel (Pratt, 1975)

PRÍMEK  $\in \mathbf{NP} \cap \mathbf{coNP}$ .

Bizonyítás

Legyen adott  $p$ . Feltehető, hogy  $p > 2$ .

A  $p$  prímszám voltának bizonyítéka:

$$(r, q_1, \dots, q_k),$$

ahol  $1 \leq r < p$ ,  $r^{p-1} \equiv 1 \pmod{p}$ , továbbá  $q_1, \dots, q_k$  a  $p-1$  összes prímosztói, és így  $k \leq \lceil \log p \rceil$  és  $r^{(p-1)/q_i} \not\equiv 1 \pmod{p}$ ,  $i = 1, \dots, k$ .



# Pratt tétele

Az input mérete

$$n = \lceil \log p \rceil.$$

Állítás

$r^{p-1} \bmod p$  meghatározható valamely  $m$ -re  $\mathcal{O}(n^m)$  időben.

Bizonyítás

Képezzük az  $r \bmod p$ ,  $r^2 \bmod p$ ,  $\dots$ ,  $r^{2^n} \bmod p$  számokat, majd ezek közül összeszorozzuk azon  $r^{2^i}$ -ket, melyekre  $p - 1$  bináris reprezentációjában az  $i$ -edik helyiértékű bit 1.

# Pratt tétele

## Állítás

$n$ -ben polinom időben ellenőrizhető, hogy eljuthatunk-e  $p - 1$ -ből az 1-be  $q_i$ -kkel való osztások segítségével, ahol minden egyes  $q_i$ -t legalább egyszer felhasználjuk.

Továbbá  $n$ -ben polinom időben ellenőrizhető az is, hogy minden  $i$ -re érvényes-e az  $r^{(p-1)/q_i} \not\equiv 1 \pmod p$  egyenlőtlenség.

De  $(r, q_1, \dots, q_k)$  **nem teljes bizonyíték**, mert valahányszor  $q_i \neq 2$ , annak is bizonyítékát kell adni, hogy  $q_i$  prím.

A teljes bizonyíték:

$$C(p) = (r, q_1, C(q_1), \dots, q_k, C(q_k)),$$

ahol  $C(q_i)$  a  $q_i$  prímszám voltának bizonyítéka (ez üres, ha  $q_i = 2$ ).

# Pratt tétele

## Állítás

A teljes bizonyíték hossza  $\mathcal{O}(n^2)$ .

## Állítás

Adott  $p$ -re a  $C(p)$  bizonyíték helyessége  $n$ -ben polinom időben leellenőrizhető.

## Bizonyítás

A fentiek alapján, ld. könyv.

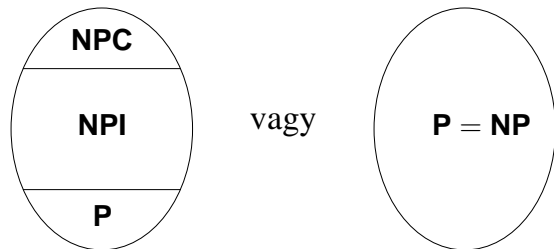
Tétel (M. Agrawal, N. Kayal, N. Saxena, 2002)

**PRÍMEK  $\in$  P.**

# Néhány további eredmény a **P** és **NP** osztályokról

## Tétel

Ha  $\mathbf{P} \neq \mathbf{NP}$ , akkor létezik olyan  $L \in \mathbf{NP} - \mathbf{P}$ , mely nem **NP**-teljes.



**NPC**: **NP**-teljes nyelvek (**NP**-complete)

**NPI**: azon  $(\mathbf{NP} - \mathbf{P})$ -beli nyelvek, amelyek nem **NP**-teljesek (**NP**-intermediate).

# Orákulumos Turing-gép

## Definíció

Az  $M^?$  **orákulumos**, determinisztikus vagy nemdeterminisztikus **Turing-gép** olyan többszavas Turing-gép, mely rendelkezik egy speciális kérdés szóval (szalaggal), valamint  $q?$ ,  $q_{igen}$ ,  $q_{nem}$  állapotokkal.

## Definíció

Legyen  $A \subseteq \Sigma^*$ , ahol  $\Sigma$  az  $M$  ábécéje. Ekkor az  $M^A$  Turing-gép, melyben **az orákulumot az  $A$  nyelvvel realizáljuk**, úgy működik, mint egy közönséges Turing-gép, kivéve, ha a  $q?$  állapotban van. Ekkor a  $q_{igen}$  vagy a  $q_{nem}$  állapotba megy át annak megfelelően, hogy a kérdés szó az  $A$  nyelvben van-e.

## Időbonyolultság

Mint a közönséges esetben, **minden kérdés egyetlen lépésnek számít.**

# Orákulumos Turing-gép

## Definíció

Legyen  $\mathcal{C}$  bonyolultsági osztály,  $A$  tetszőleges nyelv.

$\mathcal{C}^A$ : Azon nyelvek, amelyek eldönthetők ugyanolyan típusú orákulumos Turing-géppel az orákulum  $A$ -val való realizálása mellett, mint a  $\mathcal{C}$  osztályba eső nyelveket eldöntő Turing-gépek. Legyen  $\mathcal{C}'$  is bonyolultsági osztály. Ekkor

$$\mathcal{C}^{\mathcal{C}'} = \bigcup_{A \in \mathcal{C}'} \mathcal{C}^A.$$

## Példa

- ▶  $\mathbf{P}^{\mathbf{P}} = \mathbf{P}$
- ▶  $\mathbf{NP} \cup \mathbf{coNP} \subseteq \mathbf{P}^{\mathbf{NP}} = \mathbf{P}^{\mathbf{SAT}}$

# Orákulumos Turing-gép

## Tétel

Van olyan  $A$  orákulum, melyre  $\mathbf{P}^A = \mathbf{NP}^A$ .

## Lemma

Ha  $A \in \mathbf{PSPACE}$ , akkor  $\mathbf{P}^A \subseteq \mathbf{PSPACE}$  és  $\mathbf{NP}^A \subseteq \mathbf{NPSPACE} = \mathbf{PSPACE}$ .

## Lemma

Ha  $A$   $\mathbf{PSPACE}$ -teljes, akkor  $\mathbf{PSPACE} \subseteq \mathbf{P}^A$ .

## A tétel bizonyítása

Legyen  $A$   $\mathbf{PSPACE}$ -teljes, később látni fogjuk, hogy ilyen  $A$  létezik. Ekkor:

$$\mathbf{PSPACE} \subseteq \mathbf{P}^A \subseteq \mathbf{NP}^A \subseteq \mathbf{PSPACE}.$$

# Orákulumos Turing-gép

## Tétel

Van olyan  $B$ , melyre  $\mathbf{P}^B \neq \mathbf{NP}^B$ .

Tehát a  $\mathbf{P} \neq \mathbf{NP}$  sejtés csak olyan gondolatmenettel bizonyítható, mely **nem relativizálható** tetszőleges orákulumra.



# Logaritmikus tár

## Tétel

Az ELÉRHETŐSÉG probléma **NL**-teljes.

## Bizonyítás

Azt már tudjuk, hogy a probléma **NL**-ben van. Legyen most az  $L$  egy **NL**-beli nyelv. Ekkor  $L$  eldönthető egy  $N, \log n$  tárkorlátos nemdeterminisztikus lyukszalagos Turing-géppel.

Egy  $n$  hosszú  $x$  bemeneten a konfigurációk száma  $\mathcal{O}(n^k)$  valamely  $k$ -ra. Feltehető, hogy csak egy elfogadó konfiguráció van.

Így a konfigurációs gráfot tekintve az ELÉRHETŐSÉG egy példányához jutunk, mely akkor és csak akkor igen példány, ha  $x \in L$ .

Mivel a konfigurációs gráf elkészíthető logaritmikus tárral, készen vagyunk.

# Logaritmikus tár

Következmény

Az ELÉRHETETLENSÉG probléma **NL**-teljes.

Bizonyítás

**NL** = co**NL**

## Tétel

A 2SAT probléma **NL**-teljes.

### Bizonyítás

Azt már tudjuk, hogy  $2SAT \in \mathbf{NL}$ . Tekintsük az ELÉRHETETLENSÉG egy példányát. Az előző tétel bizonyítása miatt feltehető, hogy az adott gráf körmentes.

Minden csúcshoz rendeljünk egy  $x$  változót. A 2SAT azon  $\varphi$  példányát készítjük el, mely az összes  $\neg x \vee y$  tagokból áll, ahol  $(x, y)$  él, továbbá az  $s$  és  $\neg t$  tagokból, ahol  $s$  a kezdőcsúcs és  $t$  a cél.

Így  $\varphi$  kielégíthető  $\Leftrightarrow s$ -ből nem érhető el  $t$ .

Tegyük fel, hogy  $T$  a  $\varphi$  kielégítő kiértékelése. Ekkor  $T(x) = 1$ , valahányszor  $x$  elérhető  $s$ -ből, és  $T(t) = 0$ . Így  $t$  nem érhető el.

Tegyük fel, hogy  $t$  nem érhető el  $s$ -ből. Legyen  $T$  azon kiértékelés, melyre  $T(x) = 1$  akkor és csak akkor, ha  $x$  elérhető  $s$ -ből. Ekkor  $T$  kielégíti  $\varphi$ -t.

# Alternáló Turing-gép

## Definíció

Az **alternáló Turing-gép** olyan  $N = (K, \Sigma, \Delta, s)$  nemdeterminisztikus Turing-gép, melyben  $K$  a  $K_{\text{ÉS}}$  és  $K_{\text{VAGY}}$  diszjunkt halmazok egyesítése, és amelynek minden számítási sorozata véges (feltehető, hogy a gép pontos).

Tekintsük az  $N$  számítási fáját az  $x$  bemeneten.

Azt mondjuk, hogy egy  $C$  konfiguráció **végül elfogadó**, ha

- ▶  $C$ -ben a gép „igen” állapotban van, vagy
- ▶  $K_{\text{ÉS}}$ -állapotban van, és **minden** közvetlen leszármazottja végül elfogadó, vagy
- ▶  $K_{\text{VAGY}}$ -állapotban van, és **valamilyen** közvetlen leszármazottja végül elfogadó.

Az  $N$  gép akkor **fogadja el** az  $x$  bemenetet, ha a kezdő konfiguráció végül elfogadó, különben elutasítja azt. Az  $f(n)$  idő- vagy tárkorlátos Turing-gép fogalmát értelemszerűen definiáljuk.

# Alternáló Turing-gép

## Definíció

Legyen  $f(n)$  megengedett bonyolultsági függvény. (Idő esetén  $f(n) > n$ .)

**ATIME**( $f(n)$ ): az összes  $f(n)$  időkorlátos alternáló Turing-géppel eldönthető nyelvek.

**ASPACE**( $f(n)$ ): az összes  $f(n)$  tárkorlátos alternáló Turing-géppel eldönthető nyelvek.

**AL** = ASPACE( $\log n$ )

**AP** = ATIME( $n^k$ )

Világos, hogy  $\text{NTIME}(f(n)) \subseteq \text{ATIME}(f(n))$  és  $\text{NSPACE}(f(n)) \subseteq \text{ASPACE}(f(n))$ .

$$\mathbf{AL} = \mathbf{P}$$

Tétel

$$\mathbf{AL} = \mathbf{P}.$$

Bizonyítás

Láttuk, hogy a HÁLÓZAT-KIÉRTÉKELÉS probléma  $\mathbf{P}$ -teljes.  
Valójában már a MONOTON-HÁLÓZAT-KIÉRTÉKELÉS is  $\mathbf{P}$ -teljes, mert minden változómentes hálózat monotonná tehető.

Mivel mindkét osztály zárt a visszavezetésre,  $\mathbf{AL} = \mathbf{P}$  teljesül, ha a MONOTON-HÁLÓZAT-KIÉRTÉKELÉS  $\mathbf{AL}$ -teljes.

# $P \subseteq AL$

## Tétel

MONOTON-HÁLÓZAT-KIÉRTÉKELÉS  $\in AL$ .

## Bizonyítás

Adott monoton hálózat kiértékelését kezdje az alternáló Turing-gép a kimeneti kapunál.

Ha ez igaz vagy hamis, a kiértékelésnek vége.

Ha ÉS-kapu, akkor a gép ÉS-állapotba, különben VAGY-állapotba lép, és nemdeterminisztikusan a két ős valamelyikénél folytatja a kiértékelést.

Ha végül igaz kapuhoz ér, „igen” állapotba, különben „nem” állapotba megy át.

Eközben mindig csak az aktuális kapu sorszámát kell tárolni.

# $\mathbf{AL} \subseteq \mathbf{P}$

## Tétel

A MONOTON-HÁLÓZAT-KIÉRTÉKELÉS probléma  $\mathbf{AL}$ -nehéz.

## Bizonyítás

Megmutatjuk, hogy tetszőleges  $L \in \mathbf{AL}$  nyelv visszavezethető a MONOTON-HÁLÓZAT-KIÉRTÉKELÉS-re.

Legyen  $N = (K, \Sigma, \Delta, s)$  olyan pontos, alternáló Turing-gép, mely  $\log n$  tárral eldönti az  $L$  nyelvet. Feltehető, hogy minden nem elfogadó vagy elutasító konfigurációnak 2 leszármazottja van. A konfigurációs gráf adott  $x$  bemenet esetén körmentes. Minden  $C$  csúcsot helyettesítünk egy  $C'$  kapuval. Ez

**ÉS** kapu, ha az állapot  $K_{\text{ÉS}}$ -ben van,

**VAGY** kapu, ha az állapot  $K_{\text{VAGY}}$ -ban van,

**igaz** kapu, ha az állapot „igen”,

**hamis** kapu, ha az állapot „nem”.



# **AL** $\subseteq$ **P**

Amennyiben  $C$  közvetlen leszármazottai  $C_1$  és  $C_2$ , akkor a hálózatban  $C'$  ősei  $C'_1$  és  $C'_2$ . A kimeneti csúcs a kezdőkonfiguráció.

Világos, hogy a hálózat értéke akkor és csakis akkor igaz, ha  $x \in L$ .

Minden konfiguráció mérete  $\mathcal{O}(\log n)$ . Így könnyen látható, hogy a hálózat elkészíthető logaritmikus tárral.

## **Tétel**

Ha  $f(n) \geq \log n$  megengedett, akkor  $\text{ASPACE}(f(n)) = \text{TIME}(k^{f(n)})$ .

# A polinomiális tár

## QSAT

**Adott:**  $\exists x_1 \forall x_2 \dots Q_m x_m \varphi$  kvantifikált Boole-formula prenex normálalakban úgy, hogy a  $\varphi$  konjunktív normálalakú kvantormentes formula, melyben legfeljebb az  $x_1, \dots, x_m$  változók fordulnak elő.

**Kérdés:** Igaz-e az adott formula?

## Megjegyzés

- ▶ A kvantorok szigorú alternálása nem lényeges.
- ▶ Az sem lényeges, hogy az első kvantor  $\exists$ .

## Állítás

A QSAT probléma **PSPACE**-ben van.

## Bizonyítás

Legyen adott a  $\exists x_1 \forall x_2 \dots Q_m x_m \varphi$  formula. Ebből  $\mathcal{O}(m)$  tárral elkészítünk egy olyan hálózatot, mely gráfja egy  $m$  mélységű, kiegyensúlyozott bináris fa.

- ▶ Páratlan szinten: **VAGY** kapu
- ▶ Páros szinten: **ÉS** kapu
- ▶ Levél: **igaz** vagy **hamis** attól függően, hogy a „megfelelő” kiértékelés mellett  $\varphi$  igaz vagy hamis.

Egy rekurzív algoritmussal a mélységgel arányos, tehát  $\mathcal{O}(m)$  tárral kiértékeljük a hálózatot.

A visszavezetés tranzitivitásának bizonyításában megismert módszer szerint a két algoritmust összetehetjük –  $\mathcal{O}(m)$  **tárkorlátos Turing-gépet kapunk.**

## Megjegyzés

A probléma akkor is **PSPACE**-ben van, ha

- ▶  $\varphi$  tetszőleges kvantormentes Boole-formula.
- ▶ a kvantorok nem feltétlenül váltakoznak.
- ▶  $\varphi$ -ben az  $x_1, \dots, x_m$  változókon kívül egyéb változók is előfordulnak, és azt kérdezzük, kielégíthető-e az egész formula.
- ▶ a formula nem feltétlenül prenex alakú.
- ▶ azt kérdezzük, hogy a formula hamis-e.

## Tétel

A QSAT probléma **PSPACE**-teljes.

### Bizonyítás

Mivel **PSPACE** = **coPSPACE**, a QSAT probléma akkor és csakis akkor **PSPACE**-teljes, ha QSAT-KOMPLEMENTER **PSPACE**-teljes.

Ez utóbbi probléma akkor és csakis akkor **PSPACE**-teljes, ha a QSAT azon változata **PSPACE**-teljes, melyben az adott formula **magja** (a  $\varphi$  a  $\exists x_1 \forall x_2 \dots Q_m x_m \varphi$  formulában) **diszjunktív normálformában** van.

Legyen adott az  $L \in \mathbf{PSPACE}$  nyelv, melyet eldönt az  $\mathcal{O}(n^{k'})$  tárkorlátos  $M$  Turing-gép. Adott  $x$ ,  $n$  hosszú bemeneten a konfigurációk száma  $\mathcal{O}(2^{n^k})$  valamely  $k$ -ra.

Az egyszerűség kedvéért feltesszük, hogy a konfigurációk száma legfeljebb  $2^{n^k}$  és hogy pontosan egy elfogadó konfiguráció van.

Minden konfigurációt kódoljunk egy  $n^k$  hosszú bitsorozattal.  
Legyen

$$\text{ÚT}(a, b, i)$$

akkor és csakis akkor, ha  $a$ -ból elérhető a  $b$  legfeljebb  $2^i$  hosszú úton.

Megmutatjuk, hogyan lehet felírni olyan

$$\psi_i(A, B)$$

formulát az  $A = \{a_1, \dots, a_{n^k}\}$ ,  $B = \{b_1, \dots, b_{n^k}\}$  szabad változókkal, hogy  $\psi_i$  akkor és csakis akkor igaz a változók valamely kiértékelése mellett, ha a változók értékei olyan  $a, b$  konfigurációkat kódolnak, hogy  $\text{ÚT}(a, b, i)$  teljesül.

Így  $x \in L \Leftrightarrow \psi_{n^k}(A, B)$  igaz, amikor  $A$  helyébe a kezdő konfigurációt,  $B$  helyébe az elfogadó konfigurációt kódoló kiértékelést helyettesítjük.

A formulák logaritmikus tárral elkészíthetők lesznek.

## A konstrukció

$\psi_0(A, B)$ : azt fejezi ki, hogy minden  $i$ -re  $a_i = b_i$  vagy a  $B$  konfiguráció egy lépésben következik  $A$ -ból.

Megadható  $\mathcal{O}(n^{2k})$  hosszú kvantifikált formulával, aminek a magja diszjunktív normálalakú.

$$\psi_{i+1}(A, B) : \exists Z(\psi_i(A, Z) \wedge \psi_i(Z, B)).$$

**A formulák exponenciálisan hosszúak lesznek!**

$\psi_{i+1}(A, B)$ :

$$\exists Z \forall X \forall Y (((X = A \wedge Y = Z) \vee (X = Z \wedge Y = B)) \rightarrow \psi_i(X, Y)).$$

$\psi_i$  kvantorai előre hozhatóak. A mag diszjunktív normálformája

$\psi_i$  magjának normálformája, plusz egy  $\mathcal{O}(n^{2k})$  hosszú tag.

Most egy összefüggést igazolunk az alternáló polinom idő és a polinomiális tár között.

Már definiáltuk az  $\mathbf{AP} = \text{ATIME}(n^k)$  osztályt.

Tétel

A QSAT probléma  $\mathbf{AP}$ -teljes.

Bizonyítás

A  $\text{QSAT} \in \mathbf{AP}$  bizonyításához: egy alternáló Turing-gép egymás után megsejti az  $x_1, x_2, \dots$  változók értékét, az egzisztenciálisan kvantifikált változókét  $K_{\text{VAGY}}$  állapotban, az univerzálisan kvantifikált változókét  $K_{\text{ÉS}}$  állapotban. Végül ellenőrzi, hogy a megtalált kiértékelés kielégíti-e a formula magját.

Időigény: polinomiális.



## Teljesség

A Cook-tétel bizonyításának változata. A nemdeterminisztikus választásokhoz tartozó változókat lekötjük  $\exists$  vagy  $\forall$  kvantorral aszerint, hogy az állapot a  $K_{\text{VAGY}}$  vagy  $K_{\text{ÉS}}$  halmazba esik-e. Feltehető, hogy csak alternálnak – minden második azonosan kvantifikált.

## Következmény

**AP = PSPACE.**

## Bizonyítás

Mindkét osztály zárt a visszavezetésre és QSAT mindkettőben teljes.

# Kétszemélyes játékok

A QSAT felfogható **kétszemélyes játékként**:

- ▶ Játékosok:  $\exists, \forall$
- ▶ Lépések felváltva:  $x_1$  értéke,  $x_2$  értéke, ...
- ▶  $\exists$  célja: kielégíteni a formula magját
- ▶  $\forall$  célja: hamissá tenni

## FÖLDRAJZI JÁTÉK

**Adott:**  $G = (V, E)$  irányított gráf, 1 kezdőcsúcs.

**Kérdés:** Az I. játékos nyer-e az alábbi játékban?

- ▶ Az I. játékos kezd az 1 csúcs megnevezésével.
- ▶ Minden lépésben minden játékosnak egy olyan csúcsot kell választani, amely még nem szerepelt, és amelybe vezet él az előzőleg megnevezett csúcsból.
- ▶ Egy játékos veszít, ha végül nem tud ilyen csúcsot megnevezni.

## Állítás

A FÖLDRAJZI JÁTÉK probléma **PSPACE**-ben van.

## Bizonyítás

- ▶ A lépéssorozatok hossza a bemenet méretében polinomiális.
- ▶ Létezik olyan polinom tárigényű algoritmus, mely adott állásra megkonstruálja az állás rákövetkezőit, vagy ha ilyen nincs, eldönti, melyik játékos nyert.

Minden ilyen kétszemélyes játék **PSPACE**-ben van:

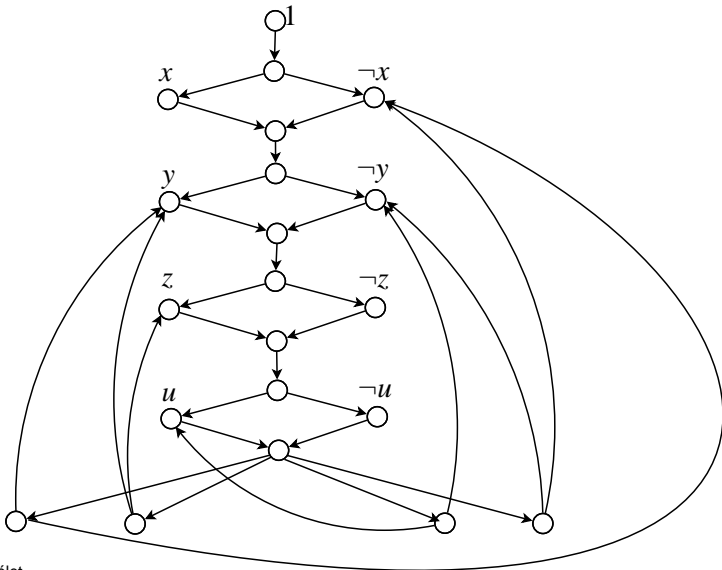
- ▶ Polinom tárral elkészítjük az adott játék fáját.
- ▶ Mélységgel arányos tárral eldöntjük, kinek van nyerő stratégiája.

A két algoritmus összetettheő: polinom tárkorlát.

## Állítás

A FÖLDRAJZI JÁTÉK probléma **PSPACE**-teljes.

$$\exists x \forall y \exists z \forall u (y \vee \neg x) \wedge (y \vee z) \wedge (\neg y \vee u) \wedge (\neg x \vee \neg y)$$



- ▶ Az első játékos dönti el az egzisztenciálisan, a második az univerzálisan kvantifikált változó értékét, mely a címkével ellentétes.
- ▶ A második játékos választ egy tagot.
- ▶ Az első játékos akkor és csak akkor nyer, ha a tag valamely literálja igaz (visszavezető él választható).
- ▶ Mivel ez minden tagra igaz, az első játékos nyer  $\Leftrightarrow$  a formula igaz.

## HELYBEN ELFOGADÁS

**Adott:**  $M$  determinisztikus Turing-gép és  $x$  bemenő szó.

**Kérdés:** Elfogadja-e  $M$  az  $x$ -et úgy, hogy szava valamikor is hosszabb lenne, mint  $|x| + 1$ ?

### Tétel

A HELYBEN ELFOGADÁS probléma **PSPACE**-teljes.

## REGULÁRIS KIFEJEZÉSEK EKVIVALENCIÁJA

**Adott:** Két reguláris kifejezés.

**Kérdés:** Ugyanazt a nyelvet jelölik-e?

### Tétel

A REGULÁRIS KIFEJEZÉSEK EKVIVALENCIÁJA probléma **PSPACE**-teljes.

## VÉGES AUTOMATÁK EKVIVALENCIÁJA

**Adott:**  $M_1$  és  $M_2$  véges **nemdeterminisztikus** automaták.

**Kérdés:**  $M_1$  és  $M_2$  ekvivalensek-e?

### Tétel

A VÉGES AUTOMATÁK EKVIVALENCIÁJA probléma  
**PSPACE**-teljes.



# Túl a **PSPACE** osztályon

Definíció

$$\mathbf{EXP} = \text{TIME}(2^{n^k})$$

$$\mathbf{NEXP} = \text{NTIME}(2^{n^k})$$

Világos, hogy  $\mathbf{EXP} \subseteq \mathbf{NEXP}$  és tudjuk, hogy  $\mathbf{PSPACE} \subseteq \mathbf{EXP}$ .  
Nem ismert, hogy az  $\mathbf{EXP}$  és  $\mathbf{NEXP}$  osztályok megegyeznek-e.

Tétel

Ha  $\mathbf{P} = \mathbf{NP}$ , akkor  $\mathbf{EXP} = \mathbf{NEXP}$ .

## Gráfok tömör reprezentációja

- ▶  $G = (V, E)$ ,  $V = \{0, 1, \dots, 2^m - 1\}$
- ▶ Megadható egy  $f(\vec{x}, \vec{y})$ ,  $2m$ -változós Boole-függvénnyel.
- ▶ Az  $f$  megadható egy  $2m$  bemeneti kapuval rendelkező hálózattal.

Hálózatok hasonló módszerrel megadhatók tömören.

### Tétel

A TÖMÖR HAMILTON-ÚT és  
TÖMÖR HÁLÓZAT-KIELEGÍTHETŐSÉG problémák  
**NEXP**-teljesek.

### Tétel

A TÖMÖR HÁLÓZAT-KIÉRTÉKELÉS probléma **EXP**-teljes.

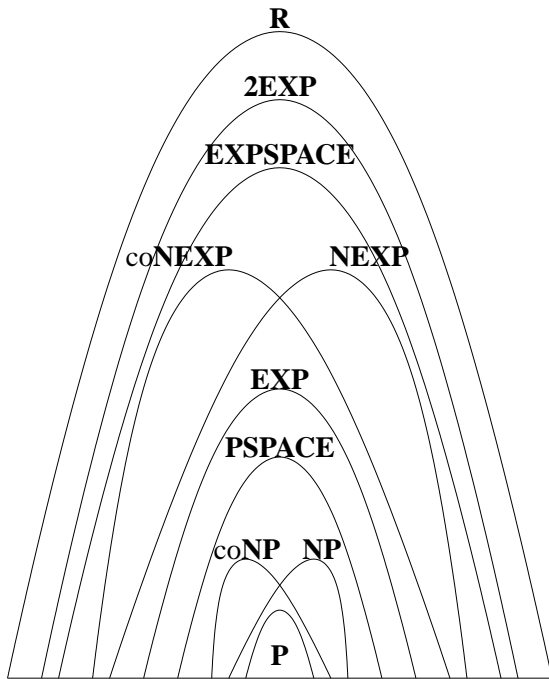
## Definíció

$$\mathbf{EXPSPACE} = \text{SPACE}(2^{n^k})$$

Világos, hogy  $\mathbf{NEXP} \subseteq \mathbf{EXPSPACE}$ .

## Tétel

Az alábbi probléma  $\mathbf{EXPSPACE}$ -teljes: adott két reguláris kifejezés, melyekben négyzetreemelés is szerepelhet, ekvivalensek-e a kifejezések?



# Randomizált számítások

## PÁROSÍTÁS

**Adott:**  $G = (U, V, E)$  páros gráf, ahol

$$U = \{u_1, \dots, u_n\}, \quad V = \{v_1, \dots, v_n\}, \quad E \subseteq U \times V$$

**Kérdés:** Létezik-e teljes párosítás, azaz olyan  $M \subseteq E$  halmaz, hogy

$$\forall u_i \exists! v_j \quad (u_i, v_j) \in M$$

$$\forall v_j \exists! u_i \quad (u_i, v_j) \in M?$$

## Szimbolikus determináns

Legyen  $A^G$  az az  $n \times n$ -es mátrix, melynek  $(i,j)$ -dik eleme az  $x_{i,j}$  változó, ha  $(u_i, v_j) \in E$ ,  $0$  különben.

### Állítás

A  $\det(A^G)$  **szimbolikus determináns** akkor és csakis akkor nem nulla, ha létezik teljes párosítás.

### Bizonyítás

- ▶  $\det(A^G) = \sum_{\pi} \sigma(\pi) \prod_{i=1}^n A_{i,\pi(i)}^G$ .
- ▶  $\forall \pi : \prod_{i=1}^n A_{i,\pi(i)}^G \neq 0 \Leftrightarrow \pi$  teljes párosítás.
- ▶  $\pi_1 \neq \pi_2, \neq 0 \Rightarrow \prod_{i=1}^n A_{i,\pi_1(i)}^G$  tartalmaz olyan változót, mely nem fordul elő  $\prod_{i=1}^n A_{i,\pi_2(i)}^G$ -ben.

## Lemma

Legyen  $\pi(x_1, \dots, x_m)$  adott  $m$ -változós, nem azonosan nulla, minden változójában legfeljebb  $d$ -ed fokú polinom, és legyen  $M > 0$ . Ekkor azon  $(x_1, \dots, x_m) \in \{0, 1, \dots, M-1\}^m$  rendezett  $m$ -esek száma, melyekre  $\pi(x_1, \dots, x_m) = 0$ , legfeljebb  $mdM^{m-1}$ .

## Bizonyítás

$m$  szerinti teljes indukcióval.

$m = 1$ : Legfeljebb  $d$  zérushely.

$m > 1$ : Legyen

$$\begin{aligned}\pi(x_1, \dots, x_m) &= p_0(x_1, \dots, x_{m-1})x_m^r + p_1(x_1, \dots, x_{m-1})x_m^{r-1} + \\ &\quad \dots + p_r(x_1, \dots, x_{m-1}).\end{aligned}$$

Tegyük fel, hogy  $(x_1, \dots, x_m) \in \{0, 1, \dots, M-1\}^m$ , melyre  $\pi(x_1, \dots, x_m) = 0$ .

(a)  $p_0(x_1, \dots, x_{m-1}) = 0 \Rightarrow (m-1)dM^{m-2} \cdot M$

(b)  $p_0(x_1, \dots, x_{m-1}) \neq 0 \Rightarrow dM^{m-1}$

**Összesen:**  $mdM^{m-1}$ .

## Következmény

Az előző feltételek esetén, ha  $M = 2m$  és  $d = 1$ , akkor azon  $(x_1, \dots, x_m) \in \{0, 1, \dots, M - 1\}^m$  rendezett  $m$ -esek száma, melyekre  $\pi(x_1, \dots, x_m) = 0$ , **legfeljebb**  $\frac{M^m}{2}$ , azaz az összes  $m$ -esek **fele**.

## Randomizált algoritmus PÁROSÍTÁS-ra

1. Válasszunk véletlenszerűen  $i_1, \dots, i_m$  egész számokat a  $\{0, 1, \dots, 2m - 1\}$  halmazból, ahol  $m$  az élek száma.
2. Számítsuk ki Gauss eliminációval a  $\det(A^G(i_1, \dots, i_m))$  értéket.
3. Ha ez nem 0, akkor a válasz **igen**: létezik teljes párosítás.
4. Különben a válasz **nem**:  $G$ -ben (valószínűleg) nem létezik teljes párosítás.



## Megjegyzés

- ▶ Igen válasz sohasem téves.
- ▶ Nem válasz esetén a tévedés valószínűsége  $\leq 1/2$ .
- ▶ Így  $k$ -szori megismétlés esetén nem válasz esetén a tévedés valószínűsége  $\leq 1/2^k$ .
- ▶ Polinom időigény.

# A „Fermat próba”

## Tétel

Tegyük fel, hogy  $p$  prímszám és  $0 < a < p$  egész. Ekkor

$$a^{p-1} \equiv 1 \pmod{p}.$$

## Hipotézis

Ha  $n > 1$  nem prímszám, akkor a nemnulla  $a$  maradékok legalább felére igaz, hogy

$$a^{n-1} \not\equiv 1 \pmod{n}.$$

Adódna egy randomizált polinomidejű algoritmus az

$$\overline{\text{ÖSSZETETT-SZÁM}} = \overline{\text{PRÍMEK}}$$

problémára.

# A „Fermat próba”

Adott  $n > 2$  esetén:

1. Válasszunk ki egy  $a \neq 0$  véletlen maradékot.
2. Ha  $a^{n-1} \not\equiv 1 \pmod n$ , akkor  $n$  **összetett szám**.
3. Ellenkező esetben  $n$  **valószínűleg prím**.

Téves (negatív) válasz valószínűsége  $\leq 1/2$ ,  $k$ -szori ismétlés után pedig legfeljebb  $1/2^k$ .

De a HIPOTÉZIS **nem igaz**...

# Legendre-szimbólum

## Definíció

Legyen  $p$  páratlan prímszám, az  $a$  pedig  $p$  nemnulla maradéka.  
Ekkor az

$$(a | p) = a^{(p-1)/2} \bmod p$$

kifejezést az  $a$  és  $p$  **Legendre-szimbólumának** nevezzük.

## Állítás

$$(a | p) \in \{p - 1, 1\}$$

$$((a | p) \in \{-1, 1\})$$

# Legendre-szimbólum

## Definíció

Legyenek  $M, N$  relatív prímek,  $N$  páratlan. Legyen  $N = q_1 \dots q_k$ , ahol minden  $q_i$  prím. Ekkor

$$(M | N) = \prod_{i=1}^k (M | q_i).$$

## Tétel

Ha  $N$  páratlan összetett szám, akkor az  $N$ -nel relatív prím  $M < N$  számok **legalább felére teljesül**, hogy

$$(M | N) \neq M^{(N-1)/2} \pmod N.$$

# Az algoritmus

Adott  $N \geq 1$  páratlan egész.

1. Generáljunk 1 és  $N - 1$  között egy véletlen egész számot, legyen ez  $M$ .
2. Számoljuk ki  $(M, N)$  értékét.
3. Ha  $(M, N) > 1$ , akkor a válasz **igen**:  $N$  összetett szám.
4. Ellenkező esetben számítsuk ki  $(M | N)$  és  $M^{(N-1)/2} \bmod N$  értékét, majd hasonlítsuk össze a két eredményt.
  - ▶ Ha nem egyeznek meg, a válasz ismét **igen**:  $N$  összetett szám.
  - ▶ Ellenkező esetben a válasz **nem**:  $N$  valószínűleg prím.

## Definíció

Legyen  $L$  nyelv  $L$ -hez tartozó polinom idejű, **Monte-Carlo típusú Turing-gép** olyan pontos, nemdeterminisztikus,  $p(n)$  polinom idejű Turing-gép, melynek minden nem végső konfigurációjában pontosan két nemdeterminisztikus választása van, és amelyekre teljesül:

- ▶  $x \in L$  esetén a számítási sorozatok **legalább fele** „igen” válasszal végződik.
- ▶  $x \notin L$  esetén **minden** számítási sorozat „nem” válasszal végződik.

## Definíció

**RP**: polinom idejű, Monte-Carlo típusú Turing-géppel eldönthető nyelvek.

## Állítás

$$\mathbf{P} \subseteq \mathbf{RP} \subseteq \mathbf{NP}.$$

## Következmény

$$\mathbf{P} \subseteq \mathbf{coRP} \subseteq \mathbf{coNP}.$$

## Definíció

$$\mathbf{ZPP} = \mathbf{RP} \cap \mathbf{coRP}.$$

Polinom idejű, Las Vegas-algoritmussal (Turing-géppel) eldönthető nyelvek, ill. problémák.

Ha egy ilyen algoritmust  $k$ -szor függetlenül végrehajtunk, annak valószínűsége, hogy nem kapunk **végleges választ**, legfeljebb  $1/2^k$ .