Függvények aszimptotikus viselkedése*

1 Fogalmak

Definíció 1.1. Aszimptotikusan pozitív függvény: Egy f függvény aszimptotikusan pozitív, ha $\exists N \in \mathbb{N}$, hogy $\forall n \geq N$ esetén $f(n) > 0$

Definíció 1.2. O, Ω, Θ Adott $f, g : \mathbb{N} \to \mathbb{R}_0^+$ függvények, definiáljuk a következő függvényhalmazokat:

- $O(g) = \{f | \exists c > 0 \wedge N \in \mathbb{N}$, hogy $\forall n \geq N$ esetén $f(n) \leq c * g(n)\}$

  f-nek g aszimptotikus felső korlátja (f legfeljebb olyan gyorsan nő, mint g), ha $f \in O(g)$ vagy $(f = O(g))$

- $\Omega(g) = \{f | \exists c > 0 \wedge N \in \mathbb{N}$, hogy $\forall n \geq N$ esetén $c * g(n) \leq f(n)\}$

  f-nek g aszimptotikus alsó korlátja, ha $f \in \Omega(g)$ vagy $(f = \Omega(g))$

- $\Theta(g) = \{f | \exists c_1, c_2 > 0 \wedge N \in \mathbb{N}$, hogy $\forall n \geq N$ esetén $c_1 * g(n) \leq f(n) \leq c_2 * g(n)\}$

  f-nek g aszimptotikus éles korlátja, ha $f \in \Theta(g)$ vagy $(f = \Theta(g))$

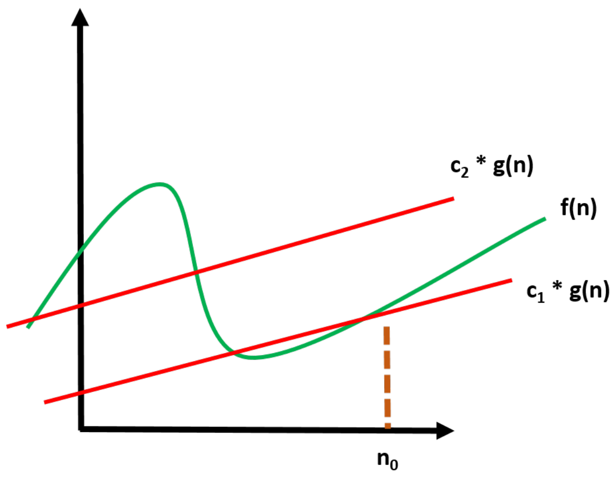
Definíció 1.3. Aszimptotikusan kisebb függvény: Adott $f, g : \mathbb{N} \to \mathbb{R}$ függvények, ekkor f aszimptotikusan kisebb, mint g azaz $f \prec g \iff \lim_{n\to\infty} \frac{f(n)}{g(n)} = 0$.

Ha f, g aszimptotikusan pozitív akkor $f \prec g \iff f \in o(g)$, azaz $o(g) = \{f | f \prec g\}$

Definíció 1.4. Aszimptotikusan nagyobb függvény: Adott $f, g : \mathbb{N} \to \mathbb{R}$ függvények, ekkor f aszimptotikusan nagyobb, mint g azaz $f \succ g \iff g \prec f$.

Ha f, g aszimptotikusan pozitív akkor $f \succ g \iff f \in \omega(g)$, azaz $\omega(g) = \{f | f \succ g\}$

*A jegyzet Dr Tichler Krisztián és Dr Ásványi Tibor anyagai alapján készült



Ábra 1.: $f \in \Theta(g)$

1.1 Tulajdonságok

Az alábbiakban legyenek f, g, h aszimptotikusan pozitív függvények:

- $\Theta(g) = O(g) \cap \Omega(g)$
- $o(g) \subset O(g) \setminus \Omega(g)$
- $\omega(g) \subset \Omega(g) \setminus O(g)$
- ha $f \in O(g)$ és $g \in O(h)$, akkor $f \in O(h)$ (tranzitivitás, Ω és Θ esetén is teljesül)
- $f \in \Theta(g) \iff g \in \Theta(f)$ (szimmetria)
- $f \in O(g) \iff g \in \Omega(f)$ (felcserélt szimmetria)
- $f \in O(f)$ és $f \in \Omega(f)$ és $f \in \Theta(f)$ (reflexivitás)
- ha $f, g \in O(h)$, akkor $f + g \in O(h)$ (Ω és Θ esetén is)
- $f + g \in \Theta(\max\{f, g\})$
- Ha $f \in O(h_1)$ és $g \in O(h_2)$, akkor $f * g \in O(h_1 * h_2)$

Tétel 1.1.

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = 0 \Rightarrow f \prec g \text{ azaz } f \in O(g) \wedge f \notin \Omega(g)$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = c \in \mathbb{R}_0^+ \Rightarrow f \in \Theta(g)$$

$$\lim_{n\to\infty} \frac{f(n)}{g(n)} = \infty \Rightarrow f \succ g \text{ azaz } f \in \Omega(g) \wedge f \notin O(g)$$

1.2 Konkrét függvények

- Ha $P(n) = a_k n^k + \cdots + a_0$ $(a_k > 0)$, akkor $P(n) \in O(n^k)$
- Minden $P(n)$ polinomra és $c > 1$ konstansra $P(n) \in O(c^n)$, de $P(n) \notin \Omega(c^n)$
- Minden $c > d > 1$ konstansokra $d^n \in O(c^n)$, de $d^n \notin \Omega(c^n)$
- Minden $a, b > 1$ esetén $\log_a n \in \Theta(\log_b n)$ (1-nél nagyobb alapú logaritmus függvények aszimptotikusan egyenértékűek)
- $\log(n!) \in \Theta(n \log n)$
- Ha $c, d \in \mathbb{R}$ és $c < d$, akkor $n^c \prec n^d$
- Ha $c \in \mathbb{R}_0^+$, akkor $c^n \prec n! \prec n^n$
- Ha $c \in \mathbb{R}_0^+$, akkor $\log n \prec n^c$
- Ha $c \in \mathbb{R}$ és $d \in \mathbb{R}_0^+$, akkor $n^c \log n \prec n^{c+d}$

Példa aszimptotikusan pozitív függvények nagyságrendjére
$\log n \prec \sqrt{n} \prec n \prec n \log n \prec n^2 \prec n^2 \log n \prec n^3 \prec 2^n \prec n!$

2 Feladatok

1. feladat: Adjunk meg olyan f és g aszimptotikusan pozitív függvényt, melyekre $f \in O(g) \wedge f \notin \Omega(g)$

Legyen $f(n) = 3n + 2$, $g(n) = 2n^2 + 3$

Vizsgáljuk meg, hogy teljesülnek O definíciójának feltételei

$3n + 2 \leq 4n^2 + 6 = 2 * g(n)$, tehát ekkor $c = 2, n \geq 1$, azaz $f \in O(g)$

Ezután ellenőrizzük, hogy $f \notin \Omega(g)$:

Legyen $n \geq 1$, ekkor ha $f \in \Omega(g)$, akkor létezik $c > 0$ konstans, hogy

$$2n^2 + 3 \leq c * (3n + 2) \text{ azaz } \frac{2n^2 + 3}{3n + 2} \leq c$$

Számítsuk ki az egyenlőtlenség baloldalán lévő tört határértékét:

$$\lim_{n\to\infty} \frac{2n^2 + 3}{3n + 2} = \lim_{n\to\infty} \frac{2\frac{n^2}{n} + \frac{3}{n}}{3\frac{n}{n} + \frac{2}{n}} = \lim_{n\to\infty} \frac{2n + \frac{3}{n} \to \infty + 0}{3 + \frac{2}{n} \to 3 + 0} = \infty$$

Azt látjuk, hogy a tört nem korlátos (∞ a határértéke), tehát nem lehet kisebbegyenlő c-nél, ami azt jelenti, hogy $f \notin \Omega(g)$

Megjegyzés: a határérték kiszámolásakor figyelembe vettük, hogy $\frac{polinom}{polinom}$ alakú törtünk van, ebben az esetben a határérték kiszámításához célszerű a nevezőben található polinom domináns tagjával osztani a teljes kifejezést.

2. feladat: Igazoljuk O tranzitivitását!

Legyenek $f, g, h : \mathbb{N} \to R_0^+$ függvények:

$$f \in O(g) : \exists c_1 > 0, N_1 \in \mathbb{N} \text{ hogy } \forall n \geq N_1 \text{ esetén } f(n) \leq c_1 * g(n)$$

$$g \in O(h) : \exists c_2 > 0, N_2 \in \mathbb{N} \text{ hogy } \forall n \geq N_2 \text{ esetén } g(n) \leq c_2 * h(n)$$

A fentiek miatt:

$$\forall n \geq \max\{N_1, N_2\} \text{ esetén } f(n) \leq c_1 * g(n) \leq c_1 * c_2 * h(n)$$

Mivel $c1, c_2 > 0 \Rightarrow c_1 * c_2 > 0$, azaz $f \in O(h)$

3. feladat: Hasonlítsuk össze a következő függvényeket! Hogyan viszonyulnak egymáshoz aszimptotikusan?

$f(n) = 5 * 2^n + n^3$

$g(n) = 3^n + 2n$

Azt sejtjük, hogy $f \prec g$, ennek igazolásához számítsuk ki a következő határértéket:

$$\lim_{n\to\infty} \frac{5 * 2^n + n^3}{3^n + 2n} = \lim_{n\to\infty} \frac{5 * (\frac{2}{3})^n + \frac{n^3}{3^n} \to 0 + 0}{1 + 2\frac{n}{3^n} \to 1 + 0} = 0$$

A határéték 0, tehát az **1.1 tétel** értelmében $f \prec g$, azaz $f \in o(g) = O(g) \setminus \Omega(g)$

Megjegyzés: a határérték kiszámításakor a korábbiakhoz hasonlóan a nevezőben található legnagyobb hatványalapú taggal osztottuk le.

4. feladat: Igazoljuk, a következő állításokat!

1. $P(n) = a_k n^k + \cdots + a_1 n + a_0$ $(a_k > 0)$ esetén $P(n) \in \Theta(n^k)$
2. Ha $c > d > 1$, akkor $d^n \in O(c^n)$
3. Minden $a, b > 1$ esetén $\log_a n \in \Theta(\log_b n)$

1.
$$\frac{P(n)}{n^k} = \frac{a_k n^k + \cdots + a_1 n + a_0}{n^k} = a_k + a_{k-1}\frac{1}{n} + \cdots + a_0\frac{1}{n^k} \to a_k$$
Mivel $(a_k > 0)$ az **1.1 tétel** szerint $P(n) \in \Theta(n^k)$

2. ha $c > d > 1$, akkor:
$$\lim_{n\to\infty} \frac{c^n}{d^n} = \lim_{n\to\infty} (\frac{c}{d})^n = \infty$$
Így a **1.1 tétel** alapján $d^n \in O(c^n) \setminus \Omega(c^n)$

3. Áttérés a alapú logaritmusra:
$$\log_b n = \frac{\log_a n}{\log_a b}$$
$$\log_a n = \log_a b * \log_b n$$
Az utóbbi kifejezésben $\log_a b$ konstans, tehát $\log_a n \in \Theta(\log_b n)$

5. feladat: Igazoljuk Θ szimmetriáját!

Ha $f \in \Theta(g)$, akkor $\exists c_1, c_2, N \in \mathbb{N}$, hogy $\forall n \geq N : c_1 * g(n) \leq f(n) \leq c_2 * g(n)$

Ekkor azonban:

$$\forall n \geq N : \frac{1}{c_2} * f(n) \leq g(n) \leq \frac{1}{c_1} * f(n)$$

6. feladat: Adottak a követkeő függvények. Rendezzük őket aszimptotikusan növekvő sorrendbe, határozzuk meg az egyes függvének nagyságrendjét is!

$log_3(n!)$, $n^{1.01} + 3\sqrt{n}$, $n^{0.03} + 2\ln n$, $(\frac{2}{3})^n$, $100n^{100} + 3^n$, $3^n + 2^n$, $4\log_{17}(n+5)$, $n!$, $n^{3/2}$

Függvény	Nagyságrend	Sorszám
$log_3(n!)$	$\Theta(n \log n)$	4
$n^{1.01} + 3\sqrt{n}$	$\Theta(n^{1.01})$	5
$n^{0.03} + 2 \ln n$	$\Theta(n^{0.03})$	3
$(\frac{2}{3})^n$	-	1
$100n^{100} + 3^n$	$\Theta(3^n)$	7-8
$3^n + 2^n$	$\Theta(3^n)$	7-8
$4 \log_{17}(n + 5)$	$\Theta(\log(n))$	2
$n!$	$\Theta(n!)$	9
$n^{3/2}$	$\Theta(n^{3/2})$	6

7. feladat: Adottak a követkeő függvények. Rendezzük őket aszimptotikusan növekvő sorrendbe, határozzuk meg az egyes függvénék nagyságrendjét is!

$n^4 - 1$, $n \ln(n + 1)$, $\log_{10} 2^n$, $(n + 1) \log_2 n$, $100n^2 + 2n$, $5\sqrt{n} - 100$, $5n^{1/3} + \ln n$

Függvény	Nagyságrend	Sorszám
$n^4 - 1$	$\Theta(n^4)$	7
$n \ln(n + 1)$	$\Theta(n \log n)$	4-5
$\log_{10} 2^n = n \log_{10} 2$	$\Theta(n)$	3
$(n + 1) \log_2 n$	$\Theta(n \log n)$	4-5
$100n^2 + 2n$	$\Theta(n^2)$	6
$5\sqrt{n} - 100$	$\Theta(\sqrt{n})$	2
$5n^{1/3} + \ln n$	$\Theta(n^{1/3})$	1

8. feladat: Adottak a következő algoritmusok és ezek futási ideje. Számítsuk ki, hogy különféle inputméretek esetén hány lépést végeznek az egyes algoritmusok és ez fizikailag mennyi ideig tartana, ha a számítógépünk processzorának órajele $4GHz$.

- Bináris keresés: $\Theta(\log n)$
- Prímszámteszt: $\Theta(\sqrt{n})$
- Maximumkiválasztás (tömb esetén): $\Theta(n)$
- Összefésülő rendezés: $\Theta(n \log n)$

- Buborékrendezés: $\Theta(n^2)$
- Floyd-Warshall algoritmus, CK algoritmus: $\Theta(n^3)$
- Hanoi tornyai: $\Theta(2^n)$
- Utazóügynök probléma: $\Theta(n!)$

Ha a processzor órajele $4GHz$ az azt jelenti, hogy egy másodpercalatt $4 * 10^9$ művelet elvégzésére képes.

input	$\log n$ (ns)	$\sqrt{n}$ (ns)	n (μs)	$n \log n$ (μs)	n^2 (ms)	n^3 (s)	2^n (év)	$n!$ (év)
10	0.83	0.79	0.0025	0.0083	0.000025	0.25 (μs)	0.26 (μs)	0.91 (ms)
100	1.66	2.5	0.025	0.17	0.0025	0.00025	$1.01 * 10^{13}$	$7.84 * 10^{140}$
1000	2.49	7.9	0.25	2.49	0.25	0.25	$8.52 * 10^{283}$	$3.2 * 10^{2549}$

Turing gépek*

1 Történelmi háttér

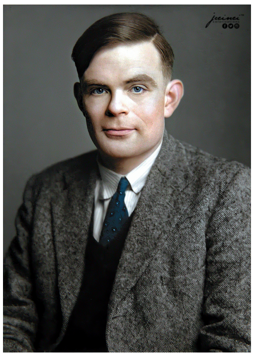
Az algoritmus általánosan egy véges sok lépésben befejeződő utasítássorozatot jelent. Az elmúlt évszázadban számos törekvés volt az algoritmus pontos matematikai definíciójának meghatározására. 1900-ban **David Hilbert**, a kor egyik legelismertebb matematikusa, 23 megoldásra váró matematikai problémát ismertetett, melyek jelentős befolyást gyakoroltak a matematika és az ekkoriban megszülető számításelmélet fejlődésére. Felmerült a kérdés, hogy mit jelent az algoritmikus kiszámíthatóság fogalma, hogyan lehet eldönteni, hogy valami kiszámítható-e. Több különböző modell is készült **Kurt Gödel** osztrák matematikus például, rekurzív függvényekkel modellezte a kiszámíthatóságot. **Alonzo Church** nevéhez kapcsolódik a λ-kalkulus megalkotása, ami a világ első funkcionális nyelvének tekinthető, segítségével a függvények viselkedése és kiszámíthatósága is modellezhető.

1936-ban **Alan Mathison Turing** brit matematikus publikálta a róla elnevezett Turing gép modellt, valamint kimondta, hogy a modell megfelel az algoritmikusan kiszámítható függvényeknek. A későbbiekben számos tétel született melyek kimondták, hogy a fenti modellek számítási ereje megegyezik.

Church-Turing tézis: A kiszámíthatóság különféle matematikai modelljei mind az effektíven kiszámítható függvények osztályát definiálják.

Egy a természetesen számokon értelmezett függvényt intuitíven **effektíven kiszámíthatónak** nevezünk, ha "papríron" egy algoritmus utasításait követve és korlátlan erőforrást feltételezve kiszámítható. A fenti tézis nem számít tételnek, de amennyiben elfogadjuk, úgy a Turing gép (és a többi modell is) tekinthető az algoritmus matematikai modelljének.

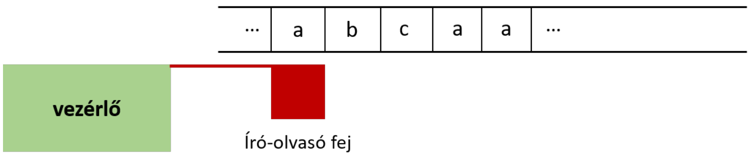
*A jegyzet Dr Tichler Krisztián anyagai alapján készült



Ábra 1.: Alan Mathison Turing

2 Turing gép (TG)

A Turing gép tekinthető az algoritmus lehetséges modelljének, gondolhatunk rá egy egyszerűsített "számítógépként". A TG rendelkezik egy vezérlőegységgel, egy mindkét irányban végtelen, cellákra osztott szalaggal (potenciálisan végtelen tár) és egy mindkét irányba mozogni képes író-olvasó fejjel. Kezdetben az bemenet a szalag celláiban helyezkedik el (a bemenet előti és utáni cellák üresek: $\sqcup$), a TG kezdőállapotban van az író-olvasó fej pedig az input első cellájára "mutat". A TG a lépések során állapotátmeneteket hajt végre, olvassa és írhatja is a szalagot. Amennyiben a TG elfogadóállapotba jut, akkor elfogajda az inputot, amennyiben elutasító állapotba jut, akkor elutasítja, de az is előfordulhat, hogy végtelen ciklusba kerül.



Ábra 2.: Turing gép

Egy $\mathscr{P}$ eldöntési probléma példányait egy megfelelő ábécé felett elkódolva, a probléma "igen-példányainak" halmazából egy $L_{\mathscr{P}}$ formális nyelvet kapunk. $L_{\mathscr{P}}$ eldönthető, ha van olyan mindig termináló gép, amely pontosan $L_{\mathscr{P}}$ szavait fogadja el.

Definíció 2.1. Turing gép: A Turing gép egy $M = \langle Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n \rangle$ rendezett hetes, ahol

- Q az állapotok véges, nemüres halmaza
- $q_0, q_i, q_n \in Q$, ahol q_0 a kezdő, q_i az elfogadó, q_n pedig az elutasító állapot

- Σ az input szimbólumok, Γ a szalagszimbólumok ábécéje és $\Sigma \subseteq \Gamma$ továbbá $\sqcup \in \Gamma \setminus \Sigma$
- $\delta : (Q \setminus \{q_i, q_n\}) \times \Gamma \to Q \times \Gamma \times \{L, S, R\}$ az átmenet függvény

Az átmeneti függvény a TG aktuális állapotának és a szalagról olvasott szimbólumnak megfelelően meghatározza a gép új állapotát, valamint, hogy milyen szimbólumot írjon az aktuális cellára ezután pedig milyen irányba lépjen tovább: L: balra, R: jobbra, S: helyben marad.

3 Kapcsolódó fogalmak

Definíció 3.1. Konfiguráció: A TG konfigurációja egy uqv szó, ahol $q \in Q$ és $u, v \in \Gamma^*$, valamint $v \neq \varepsilon$

A konfiguráció a TG aktuális helyezetét írja le, uqv esetén a gép q állapotban van, az író-olvasó fej v első szimbólumán áll, előtte u szó található.

Definíció 3.2. Kezdőkonfiguráció, elfogadó konfiguráció, elutasító konfiguráció:

- Kezdő konfigurácó: a TG egy $u \in \Sigma^*$ szóhoz tartozó kezdőkonfigurációja a $q_0u\sqcup$ szó
- Elfogadó konfiguráció: a TG azon uqv konfigurációit nevezzük elfogadó konfigurációnak, melyekre $q = q_i$ $(u, v \in \Gamma^*)$
- Elutasító konfiguráció: a TG azon uqv konfigurációit nevezzük elutasító konfigurációnak, melyekre $q = q_n$ $(u, v \in \Gamma^*)$

Az elfogadó és elutasító konfigurációkat közösen **megállási konfiguráció**knak nevezzük.

Definíció 3.3. Konfigurációátmenet: Legyen C_M az M TG-hez tartozó lehetséges konfigurációk halmaza. Az M TG $\vdash C_M \times C_M$ közvetlen (egylépéses) konfigurációátmenet relációját a következőképpen definiáljuk:

Legyen $uqav$ egy konfiguráció, ahol $a \in \Gamma$, $u, v \in \Gamma^*$

- Ha $\delta(q, a) = (r, b, R)$, akkor $uqav \vdash ubrv'$, ahol $v' = v$, ha $v \neq \varepsilon$, különben $v' = \sqcup$
- Ha $\delta(q, a) = (r, b, S)$, akkor $uqav \vdash urbv$
- Ha $\delta(q, a) = (r, b, L)$, akkor $uqav \vdash u'rcbv$, ahol $c \in \Gamma$ és $u'c = u$, ha $u \neq \varepsilon$, egyébként $u' = u$ és $c = \sqcup$

Definíció 3.4. Többlépéses konfigurációátmenet: A $\vdash^* \subset C_M \times C_M$ többlépéses konfigurációátmenet a $\vdash$ reláció reflexív, tranzitív lezártja.

Definíció 3.5. Felismert nyelv: Az M TG által felismert nyelv:
$L(M) = \{u \in \Sigma^* | q_0 u \sqcup \vdash^* x q_i y, x, y \in \Gamma^*, y \neq \varepsilon\}$

Definíció 3.6. Turing-felismerhető nyelv: Egy $L \subseteq \Sigma^*$ nyelv Turing-felismerhető (rekurzívan felsorolható, parciálisan rekurzív), ha $L = L(M)$ valamel M TG-re.

A rekurzívan felsorolható nyelvek osztályát RE-vel jelölik.

Definíció 3.7. Eldönthető nyelv: Egy $L \subseteq \Sigma^*$ nyelv eldönthető (rekurzív), ha létezik olyan M TG, amely minden bemeneten megállási konfigurációba jut és $L = L(M)$.

A rekurzív nyelvek osztályát R jelöli. Nyílván $R \subseteq RE$, de $R \subset RE$ is bizonyítható.

Definíció 3.8. Futási idő: Adott egy M TG és egy $u \in \Sigma^*$ input szó. Az M TG futási ideje az u szón n ($n \geq 0$), ha M a $q_0 u$ kezdőkonfigurációból n lépésben (konfigurációátmenettel) megállási konfigurációba jut. Ha nem létezik ilyen n szám, akkor M futási ideje végtelen u-n.

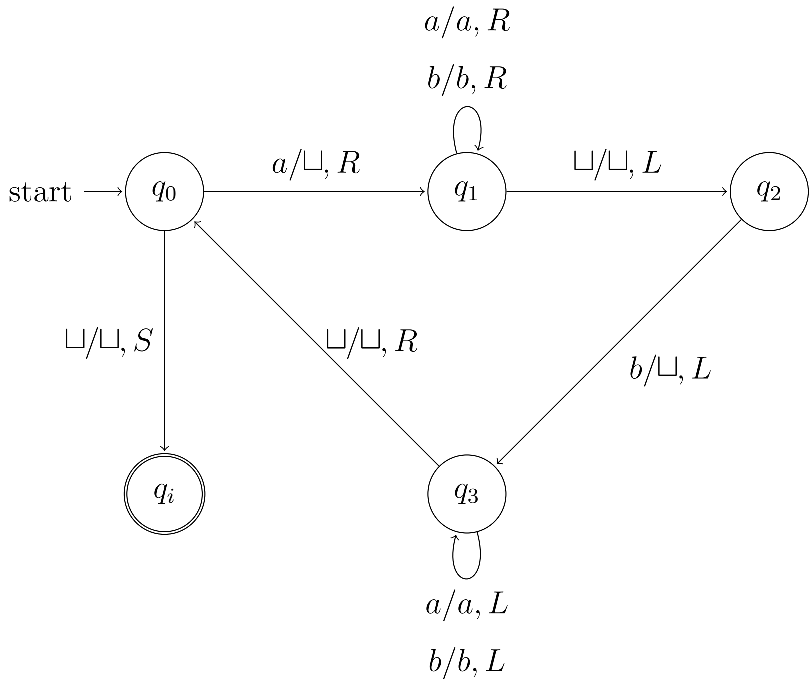
Definíció 3.9. Időkorlát: Adott M TG és $f : \mathbb{N} \to \mathbb{N}$ függvény. Azt mondjuk, hog M egy $f(n)$ időkorlátos gép, ha minden $u \in \Sigma^*$ input szóra M futási ideje az u szón legfeljebb $f(|u|)$.

4 Feladatok

Az alábbi feladatokban az elutasító állapotba irányuló átmeneteket nem jelöljük külön.

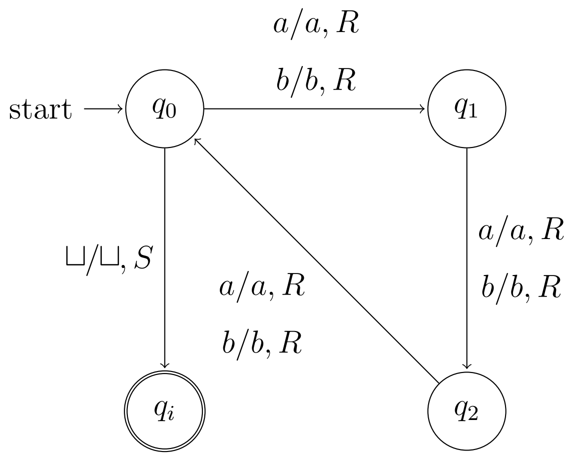
1. feladat: Készítsünk TG-et, amely a következő nyelvet fogadja el:

$L = \{a^n b^n | n \geq 0\}$



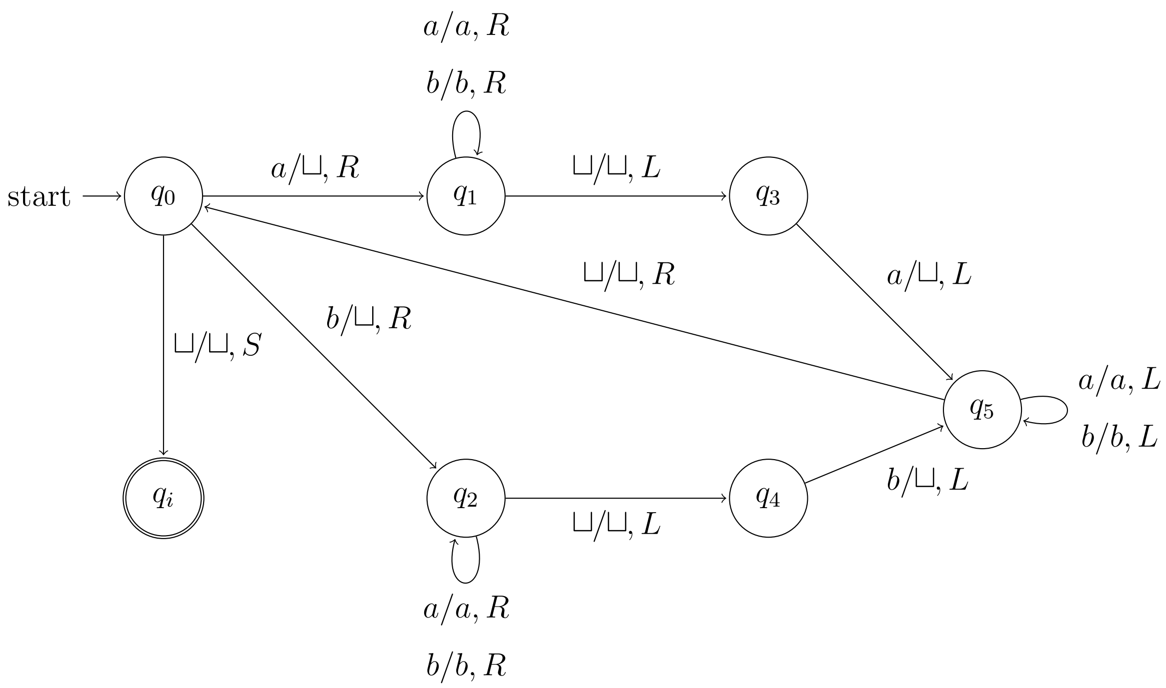
2. feladat: Készítsünk TG-et, amely a következő nyelvet fogadja el:

$L = \{u \in \{a, b\}^* | \ell(u) \mod 3 = 0\}$



3. feladat: Készítsünk TG-et, amely a következő nyelvet fogadja el:

$L = \{uu^{-1} | u \in \{a, b\}^*\}$

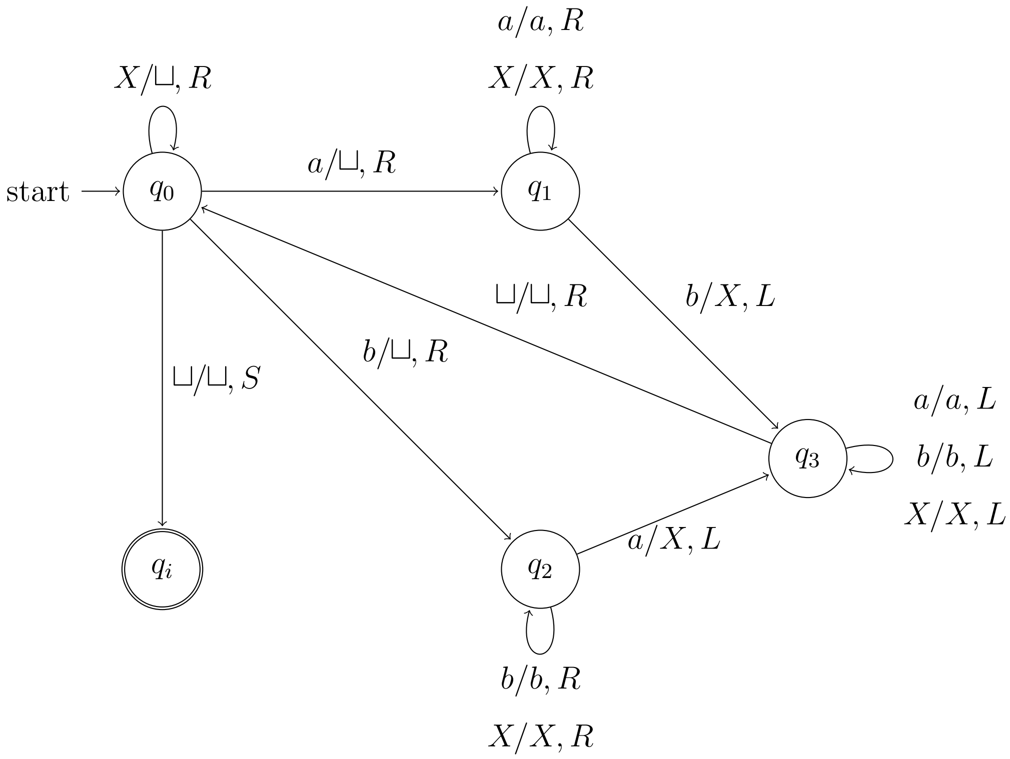


4. feladat: Készítsünk TG-et, amely a következő nyelvet fogadja el:
$L = \{u \in \{a, b\}^* | u = u^{-1}\}$

A 3 feladatban látott TG-et kell kiegészíteni két átmenettel, melyek páratlan hosszú szavak esetén lesznek érvénysek. A TG párosával "törli" a betűket a szóból, amennyiben a szó aktuálisan első és utolsó betűje megegyezik. Ha nézzük például az *aba* szót, akkor a TG letörli az első és az utolsó a-t a középső b marad, ezt elolvassuk, de mivel utána már üres cellák következnek az ő párját nem kell keresnünk.
$\delta(q_3, \sqcup) \rightarrow (q_i, \sqcup, S)$
$\delta(q_4, \sqcup) \rightarrow (q_i, \sqcup, S)$

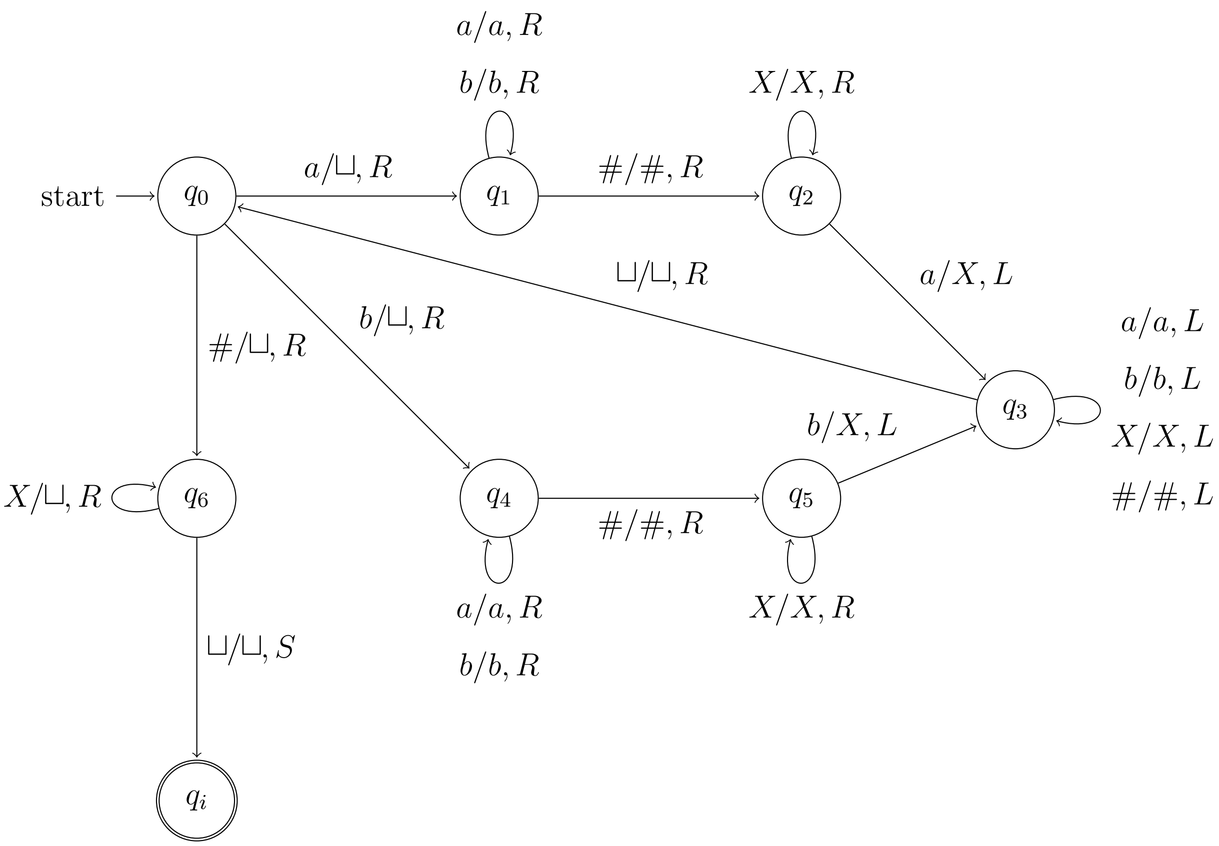
5. feladat: Készítsünk TG-et, amely a következő nyelvet fogadja el:
$L = \{u \in \{a, b\}^* | \ell_a(u) = \ell_b(u)\}$

Ötlet: induljunk el a szó elejéről (q_0 állapot), és keressünk párokat minden betűhöz. Ha először a-t találunk, akkor töröljük és keressünk hozzá egy b-t valahol a szóban (ezzel analóg módon járunk ell, ha b-t találunk először). Ha találunk b-t, akkor annak helyére tegyünk X-et, így jelezve, hogy megvan a "párja" (ne használjunk üres cellákat, mert akkor szakadás lesz a szóban), ezután menjünk vissza a szó elejére, majd váltsunk q_0 állapotba és ugyanezt folytassuk. Ha q_0 állapotban vagyunk és a szó elején X jelek vannak, akkor ezeket töröljük *(közben folyamatosan jobbra lépkedünk a szalagon), ha ezután üres cella jön, az azt jelenti, hogy minden betűnek megvolt a párja, ezért ekkor elfogadhatjuk a szót.*



6. feladat: Készítsünk TG-et, amely a következő nyelvet fogadja el:
$L = \{u\#u | u \in \{a, b\}^*\}$

Ötlet: olvassuk az első betűt, töröljük és megnézzük, hogy a # után is uganolyan betű van-e. Ha igen, akkor X-et írunk a helyére, visszamegyünk a szó elejére és ugyanezt folytatjuk. A következő lépésekben a # olvasása után X-ket is végigolvassuk és az utánuk lévő betűt ellenőrizzük. Legvégül ellenőrizzük, hogy a # után márcsak X-ek vannak-e.


start
q_0
q_1
q_2
q_3
q_4
q_5
q_6
q_i
$a/a, R$
$b/b, R$
$X/X, R$
$a/\sqcup, R$
$\#/\#, R$
$\sqcup/\sqcup, R$
$a/X, L$
$b/\sqcup, R$
$\#/\sqcup, R$
$a/a, L$
$b/b, L$
$X/X, L$
$\#/\#, L$
$b/X, L$
$X/\sqcup, R$
$\#/\#, R$
$a/a, R$
$b/b, R$
$X/X, R$
$\sqcup/\sqcup, S$

Turing gépek*

1 Többszalagos Turing gép

Többszalagos Turing gép esetén az input az első szalagon van és mindegyik szalaghoz tartozik egy-egy író-olvasó fej, melyek egymástól függetlenül tudnak mozogni.

Definíció 1.1. k-szalagos Turing gép A k-szalagos TG csak az átmenetfüggvényében különbözik az egyszalagostól
$\delta : (Q \setminus \{q_i, q_n\}) \times \Gamma^k \to Q \times \Gamma^k \times \{L, S, R\}^k$

Definíció 1.2. k-szalagos TG konfigurációja A k-szalagos TG konfigurációja egy $(q, u_1, v_1, ..., u_k, v_k)$ szó, ahol $q \in Q$ és $u_i, v_i \in \Gamma^*$, valamint $v_i \neq \varepsilon$ $(i \in [1..k])$

Definíció 1.3. Kezdő-, elfogadó-, elutasító konfiguráció Az u szóhoz tartozó kezdőkonfiguráció $u_i = \varepsilon$ $(i \in [1..k])$, $v_1 = u\sqcup$, $v_j = \sqcup$ $(j \in [2..k])$

Az elfogadó/elutasító konfigurácó q_i-t q_n-t tartalmazó konfiguráció.

Az egy- és többlépéses konfigurációátmenetet, az egyszalagos esethez hasonlóan definiáljuk, az egyes szalagokon a fejek eltérő irányba mozoghatnak.

Definíció 1.4. k-szalagos TG által felismert nyelv $L(M) = \{u \in \Sigma^* | (q_0, \varepsilon u, \sqcup, \varepsilon, \sqcup, ...) \vdash^* (q_i, x_1, y_1, ..., x_k, y_k),$ ahol $x_1, y_1 ... x_k, y_k \in \Gamma^*, y_1, ... y_k \neq \varepsilon\}$

Definíció 1.5. Ekvivalencia Két TG ekvivalens, ha ugyanazt a nyelvet ismerik fel.

Tétel 1.1. Minden M k-szalagos TG-hez, megadható vele ekvivalens M' egyszalagos TG. Továbbá, ha M legalább lineáris $f(n)$ időkorlátú TG (azaz $f(n) \in \Omega(n)$), akkor M' $O(f(n)^2)$ időkorlátos.

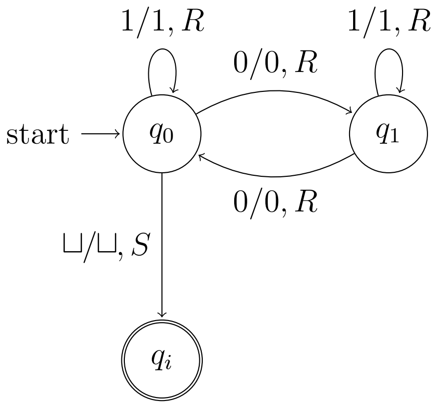
*A jegyzet Dr Tichler Krisztián anyagai alapján készült

2 Feladatok

1. feladat: Készítsünk TG-et, amely a következő nyelvet fogadja el:

$L = \{u \in \{0,1\}^* | \ell_0(u) \mod 2 = 0\}$

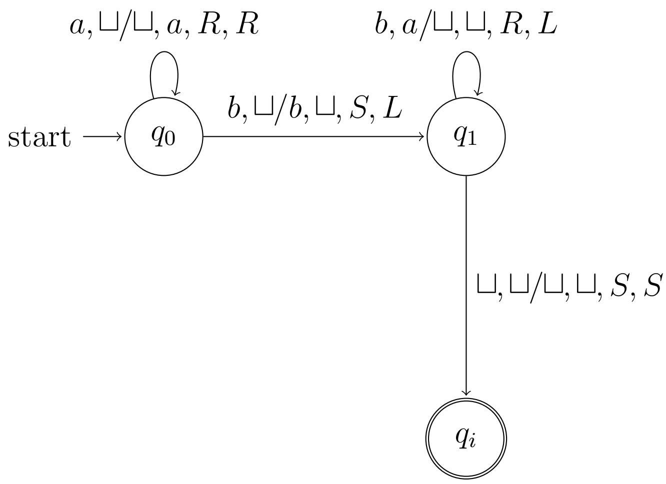
Adjuk meg a második lépés után a TG konfigurációját a 010010 input szó esetén!



$q_0 010010 \vdash 0 q_1 10010 \vdash 01 q_1 0010$

2. feladat: Készítsünk 2-szalagos TG-et, amely a következő nyelvet fogadja el:

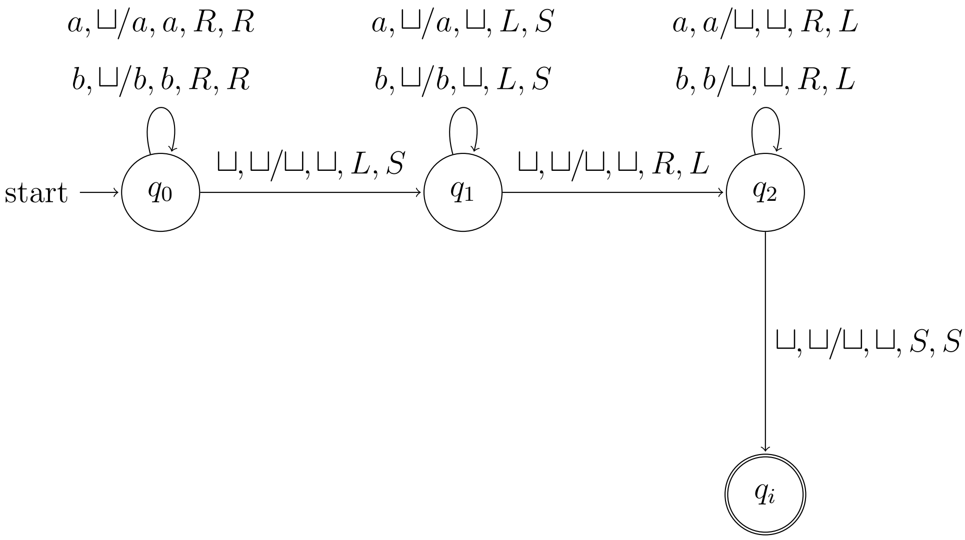
$L = \{a^n b^n | n \geq 1\}$



3. feladat: Készítsünk 2-szalgos TG-et, amely a következő nyelvet fogadja el:

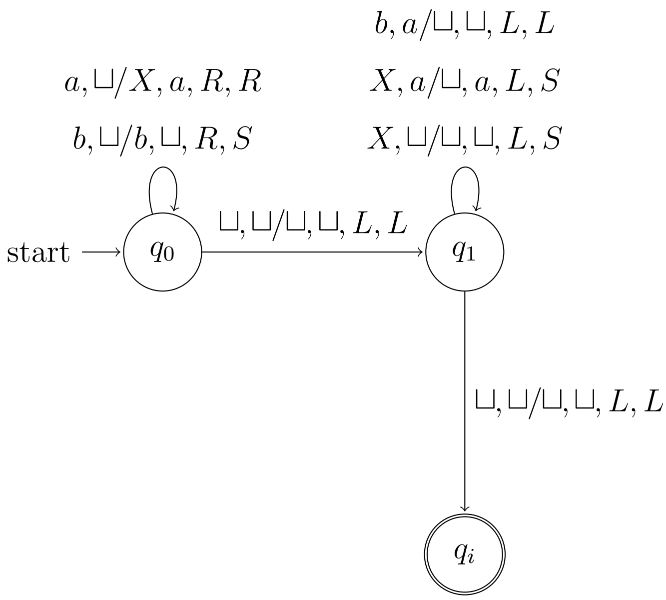
$L = \{u \in \{a,b\}^+ | u = u^{-1}\}$

Ötlet: Másoljuk le a teljes szót a második szalagra. Ezután az első szalagon induljunk el elölről, a másodikon hátulról és ellenőrizzük, hogy azonos betűket olvasunk-e. A második szalagot tulajdonképpen "veremként" használjuk.

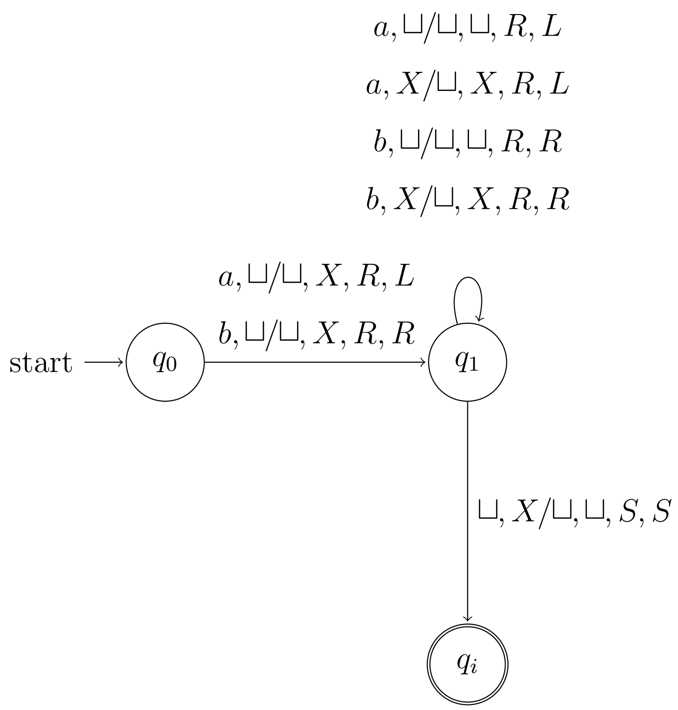


4. feladat: Készítsünk 2-szalagos, TG-et, amely a következő nyelvet fogadja el:
$L = \{u \in \{a, b\}^* | \ell_a(u) = \ell_b(u)\}$

1. változat, *ötlet: a betűket másoljuk le a második szalagra, az első szalagon pedig írjunk X-et a helyükre. Ezután induljunk el visszafelé az első szalagon, ha az első szalagon b-t a második szalagon a-t olvasunk, akkor a második szalagon is balra léphetünk egyet.*

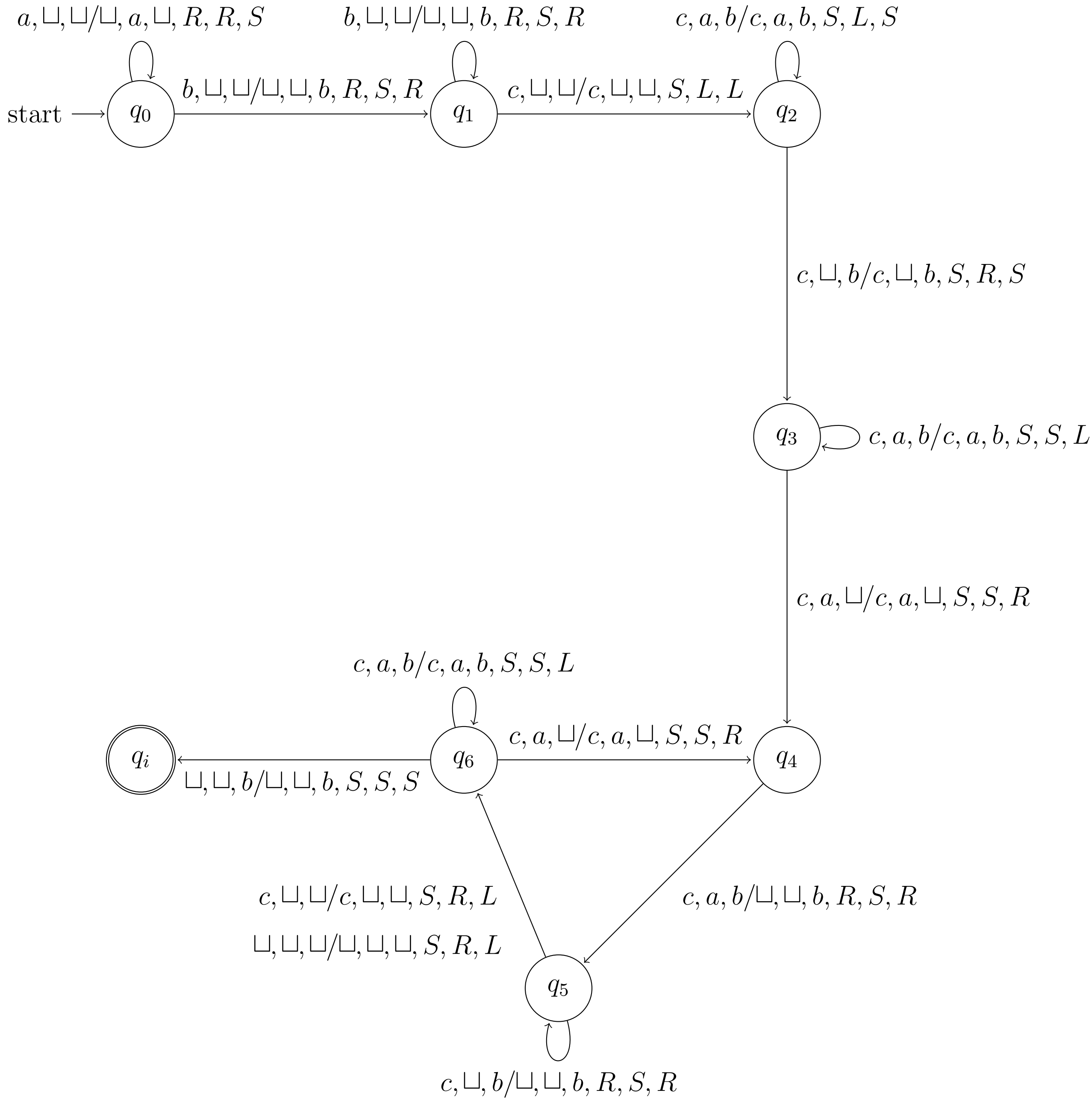


2. változat, *ötlet: a második szalagot használjuk "számlálóként". Induláskor tegyünk le egy X-et ez jelképezi a 0 pontot. Ha a betűt olvasunk az első szalagon, akkor mindig egyet balra, ha b betűt olvasunk, akkor mindig egyet jobbra lépünk a második szalagon. Ha végül a második szalagon épp az X-et tartalmazó cellán állunk, akkor elfogadjuk a szót, egyébként elutasítjuk.*


$a, \sqcup / \sqcup, \sqcup, R, L$
$a, X / \sqcup, X, R, L$
$b, \sqcup / \sqcup, \sqcup, R, R$
$b, X / \sqcup, X, R, R$
$a, \sqcup / \sqcup, X, R, L$
$b, \sqcup / \sqcup, X, R, R$
start
q_0
q_1
$\sqcup, X / \sqcup, \sqcup, S, S$
q_i

5. feladat: Készítsünk 3-szalagos TG-et, amely a következő nyelvet fogadja el:
$L = \{a^i b^j c^k | i * j = k \ (i, j \geq 1)\}$

Ötlet: a betűket másoljuk a második, b betűket a harmadik szalagra. Ezután folyamatosan olvassuk a c-ket és közben ciklikusan lépkedünk a második, illetve harmadik szalagon.

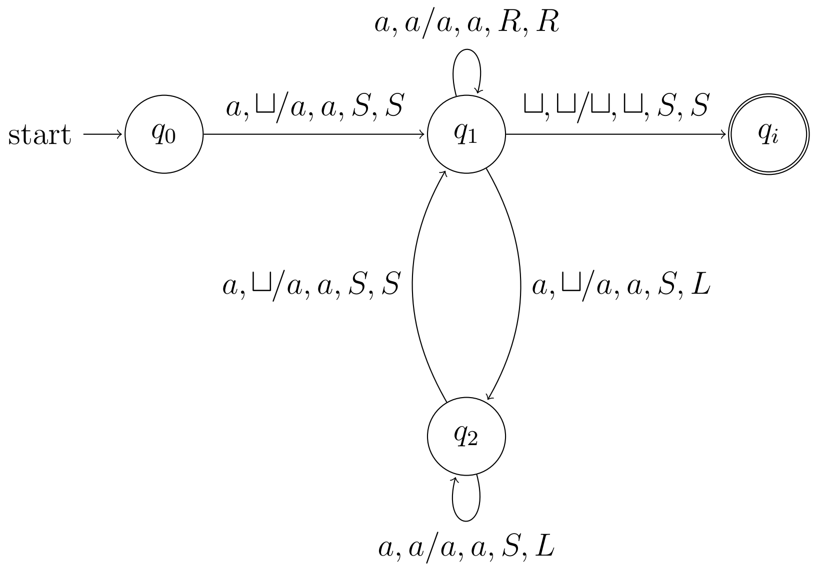


6. feladat: Készítsünk 2-szalgos TG-et, amely a következő nyelvet fogadja el:
$L = \{a^{n^2} | n \geq 0\}$

Használjuk fel a következő összefüggést: $1 + 3 + 5 + ... + (2n - 1) = n^2$

$$\sum_{i=1}^{n}(2i-1) = \frac{(1+(2n-1))n}{2} = n^2$$

Mindig annyi a-t olvasunk az első szalagon, amennyi a másodikon van. A fenti összefüggést felhasználva, a második szalagra kezdetben leteszünk egy a-t, olvasunk egyet az első szalagról is. Ha ezután marad még a az első szalagon, akkor a másodikra leteszünk még kettőt (összesen már 3 lesz), a második szalagot az elejére tekerjük és olvasunk az elsőről hármat, és így tovább. A második szalagra mindig kettővel több a-t teszünk, így a fenti sorozatnak megfelelően haladunk.



Számoló Turing gépek, nemdeterminisztikus Turing gépek*

Eddig olyan Turing gépekkel foglalkoztunk, mely igen/nem kimenetű problémákat oldottak meg. A következőkben tovább általánosítjuk a Turing gépeket számítási problémák megoldásához.

1 Kapcsolódó fogalmak

Definíció 1.1. Adott az $f : \Sigma^* \to \Delta^*$ szófüggvény, azt mondjuk, hogy az $M = \langle Q, \Sigma, \Delta, \delta, q_0, q_i, (q_n) \rangle$ TG kiszámítja f függvényt, ha minden $u \in \Sigma^*$-beli szóra megáll és ekkor $f(u) \in \Delta^*$ olvasható az utolsó szalagján.

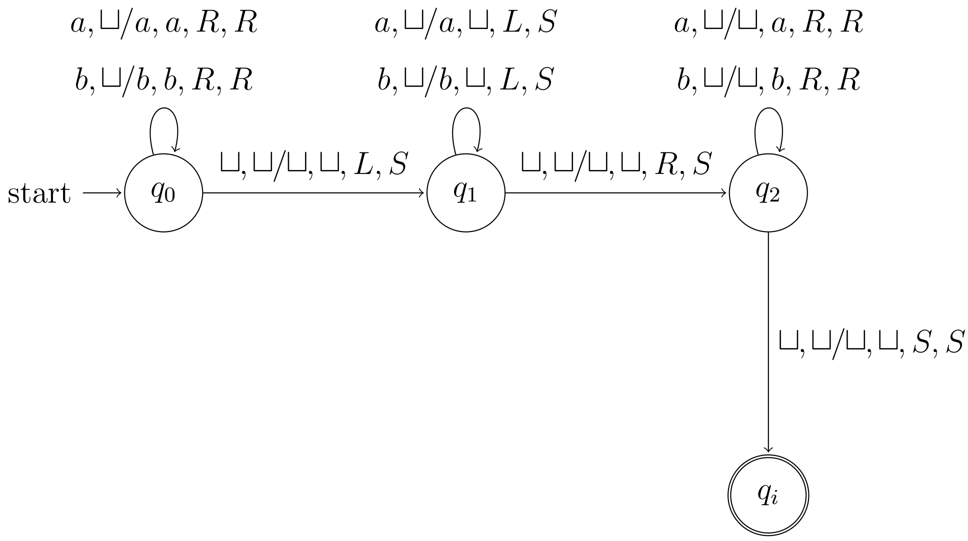
Észrevehető, hogy számítási feladat megoldásánál nincs szükség az elfogadó és elutasító állapotok megkülönböztetésére.

*A jegyzet Dr Tichler Krisztián anyagai alapján készült

2 Feladatok

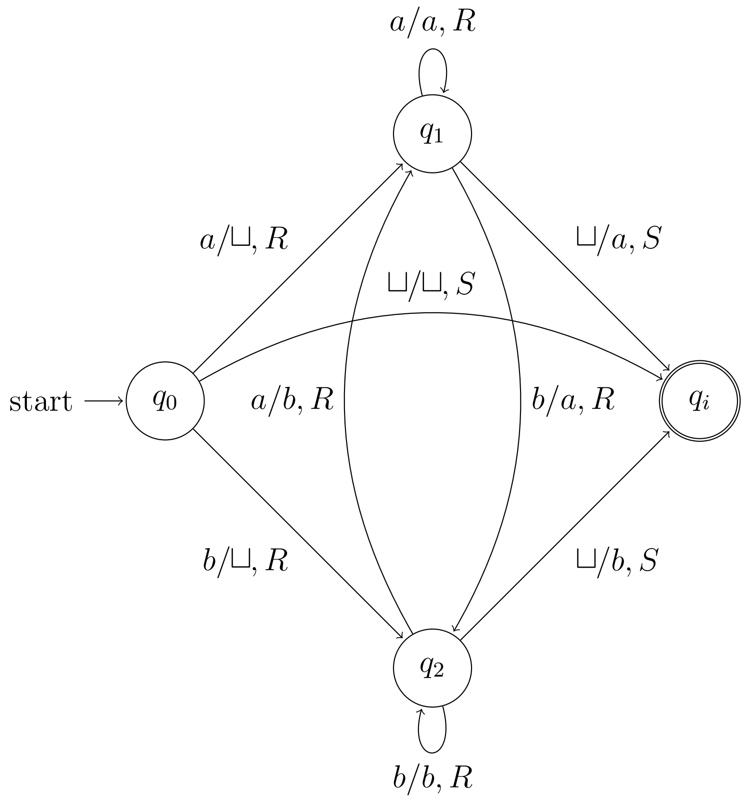
1. feladat: Készítsünk 2-szalagos TG-et, amely a következő függvényt számolja ki: $f : w \rightarrow ww$, ahol $w \in \{a, b\}^*$

Ötlet: Az eredménynek a második szalagon kell lenni, ezért először másoljuk le az első szalag tartalmát a másodikra. Ezután menjünk vissza az első szalag elejére (közben a másodikon helyben maradunk), majd másoljuk le ismét az első szalag tartalmát, közben törölhetjük is.



2. feladat: Készítsünk 1-szalagos TG-et, amely a szalagon lévő $u \in \{a, b\}^*$ szót egy cellával jobbra tolja (shifteli)!

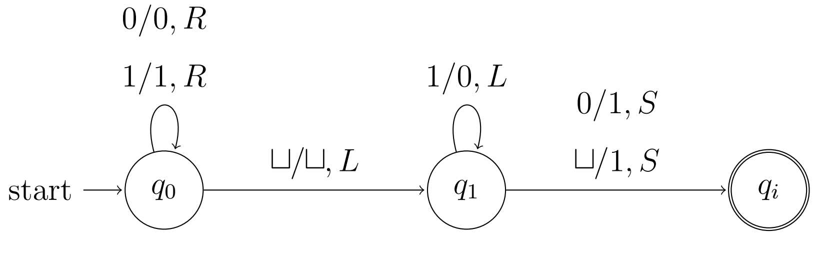
Ötlet: Induljunk el a szó elejéről, töröljük az első karaktert és jegyezzük meg, hogy mi volt az. Ezután olvassuk tovább a szót, ha az előző lépés során a-t olvastunk, akkor az a-kat átugorjuk, ha b-t találunk, akkor átírjuk a-ra és megjegyezzük, hogy felülírtunk egy b-t. Most b-ket ugrunk át egészen amíg, újabb a-t nem találunk, ekkor azt felülírjuk b-re. Ezeket a lépéseket folytatjuk, amíg végig nem érünk a szón. Legutolsó lépésben attól függően, hogy utóljára milyen betűt írtunk felül a szó utáni első üres cellába ezt a betűt írjuk le.



3. feladat: Készítsünk 1-szalagos TG-et, amely a szalagon lévő $x \in \{0,1\}^*$ bináris számhoz hozzáad 1-et!

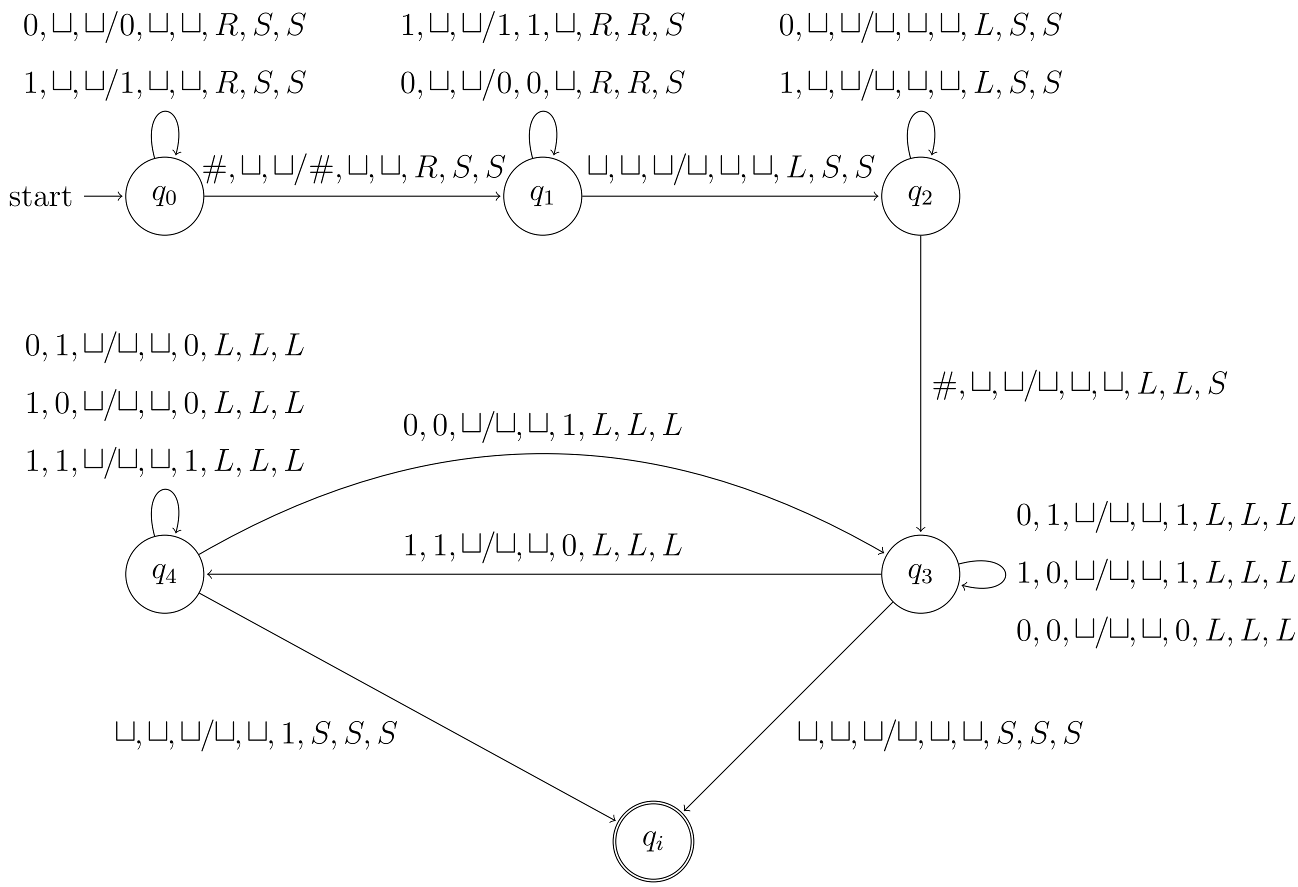
Bináris összeadás, emlékeztető: két bináris számot a helyiértékek szerint, jobbról-balra (a legkisebb helyiérték felől haladva) szoktunk összeadni. Az adott helyiértéken az eredmény a következők szerint alakul: két 0-ás bit esetén 0, egy 1-es és egy 0-ás bit esetén 1, két 1-es bit esetén 0 és egy 1-es bitet átviszünk a következő helyiértékre. Pl.: $1010 + 0111 = 10001$

Ötlet: Menjünk végig a szalagon és hátulról visszafelé haladva a bináris összeadás szabályainak megfelelően írjuk át a biteket, ahol szükséges, ha bárhol 0-ás bitet találunk, akkor megállhatunk. Ha végigértünk a számon, akkor figyelni kell, hogy a szám előtti első üres cellába írjunk egy 1-es bitet, hiszen csak akkor érhetünk végig a szalagon, ha nem találtunk 0-ás bitet, azaz átvitel van.



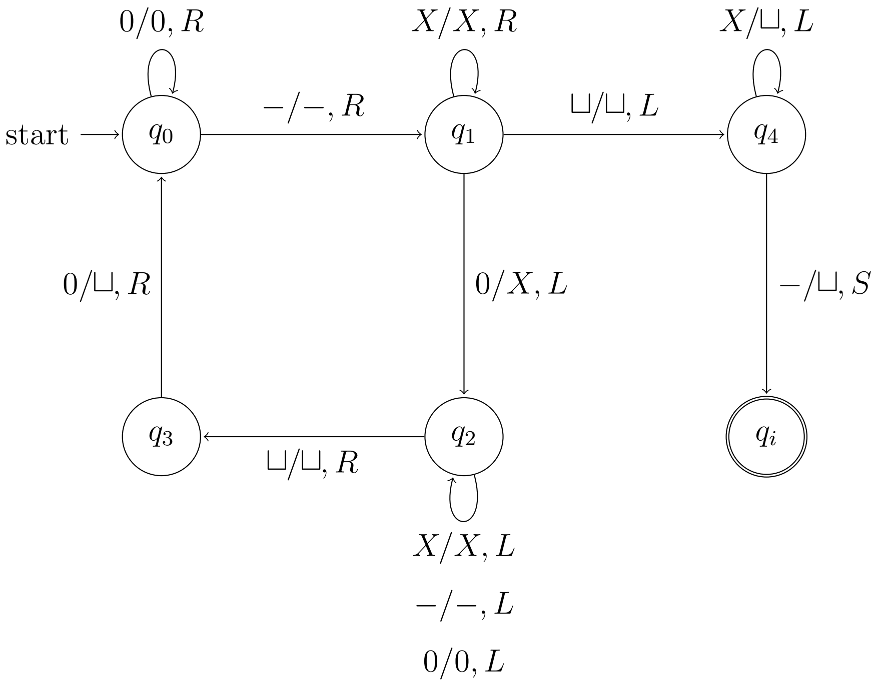
4. feladat: Készítsünk 3-szalagos TG-et, amely bináris összeadást valósít meg: $f(x\#y) \to x + y$, ahol $x, y \in \{0,1\}^+$ és $\ell(x) = \ell(y)$

Ötlet: először lemásoljuk a # utáni részt a második szalagra, majd bitenként elvégezzük az összeadást az eredmény a harmadik szalagra kerül.



5. feladat: Készítsünk 1-szalagos TG-et, amely unáris kivonást valósít meg:
$f(x-y) \to x-y$, ahol $x, y \in \{0\}^+$ és $\ell(x) \geq \ell(y)$

Ötlet: Keressük meg a - jel utáni első 0-t és írjuk felül egy X-el, majd menjünk vissza a szalag elejére és töröljük le az első 0-át. Ezeket a lépéseket ismételjük, amíg még van 0 a - jel után, azaz, annyiszor törlünk le egy 0-át a szó elejéről, ahány 0 szerepel a - jel után.



3 Nemdeterminisztikus Turing gépek

Definíció 3.1. Nemdeterminisztikus Turing gép (NTG): Az (egyszalagos) nemdeterminisztikus Turing gép csak átmenetfüggvényében különbözik a determinisztikustól:
$\delta : (Q \setminus \{q_i, q_n\}) \times \Gamma \to \mathscr{P}(Q \times \Gamma \times \{L, R, S\})$, ahol $\mathscr{P}(X) = \{Y | Y \subseteq X\}$, az X halmaz hatványhalmaza.

Definíció 3.2. Konfigurációátmenet: Legyen C_M az M TG-hez tartozó lehetséges konfigurációk halmaza. Az M TG $\vdash C_M \times C_M$ közvetlen (egylépéses) konfigurációátmenet relációját a következőképpen definiáljuk:

Legyen $uqav$ egy konfiguráció, ahol $a \in \Gamma$, $u, v \in \Gamma^*$

- Ha $(r, b, R) \in \delta(q, a)$, akkor $uqav \vdash ubrv'$, ahol $v' = v$, ha $v \neq \varepsilon$, különben $v' = \sqcup$
- Ha $(r, b, S) \in \delta(q, a)$, akkor $uqav \vdash urbv$
- Ha $(r, b, L) \in \delta(q, a)$, akkor $uqav \vdash u'rcbv$, ahol $c \in \Gamma$ és $u'c = u$, ha $u \neq \varepsilon$, egyébként $u' = u$ és $c = \sqcup$

Definíció 3.3. Többlépéses konfigurációátmenet: A $\vdash^* \subset C_M \times C_M$ többlépéses konfigurációátmenet a $\vdash$ reláció reflexív, tranzitív lezártja.

Nemdeterminisztikus Turing gép esetén töb számítás is létezhet ugyanarra a szóra, ezek közül lehetnek olyanok melyek elutasító lehetnek olyanok melyek elfogadó állapotba érnek. Ha egy NTG legalább egy számítása elfogadó állapotba ér egy adott input szón, akkor azt mondjuk, hogy elfogadja a szót.

Definíció 3.4. Felismert nyelv: Az M NTG felismeri az $L \subseteq \Sigma^*$ nyelvet, ha $L(M) = L$

Az M NTG által felismert nyelv:
$L(M) = \{u \in \Sigma^* | q_0 u \sqcup \vdash^* x q_i y, x, y \in \Gamma^*, y \neq \varepsilon\}$

Definíció 3.5. Nemdeterminisztikus számítási fa: Egy M TG egy $u \in \Sigma^*$ inputjához tartozó nemdeterminisztikus számítási fája egy gyökeres fa, melynek csúcsai M konfigurációival címkézettek. $q0u\sqcup$ a gyökér címkéje. Ha C egy csúcs címkéje, akkor $|\{C' | C \vdash C'\}|$ gyereke van és ezek címkéi éppen $\{C' | C \vdash C'\}$ elemei.

Definíció 3.6. Egy M NTG eldönti az $L \subseteq \Sigma^*$ nyelvet, ha felismeri, továbbá minden $u \in \Sigma^*$ szóhoz tartozó nemdeterminisztikus számítási fa véges és a fa minden levele elfogadó vagy elutasító konfiguráció.

Definíció 3.7. Időkorlát Az M NTG $f(n)$ időigényű, ha minden $u \in \Sigma^*$ n hosszú szóra u számítási fája legfeljebb $f(n)$ magas.

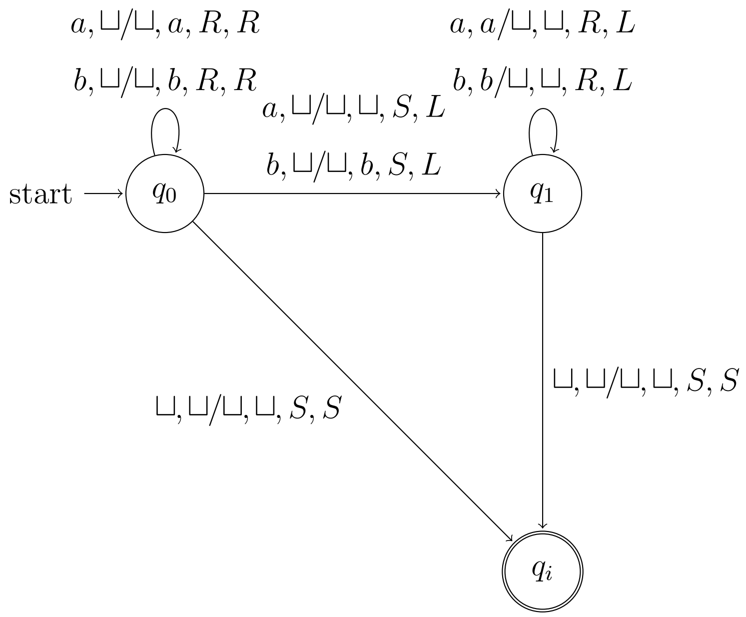
Tétel 3.1. Minden M nemdeterminisztikus Turing géphez megadható vele ekvivalens $O(2^{(f(n))})$ időigényű determinisztikus TG.

3.1 Feladatok

6. feladat: Készítsünk 2-szalagos nemdeterminisztikus TG-et, amely a következő nyelvet ismeri fel:

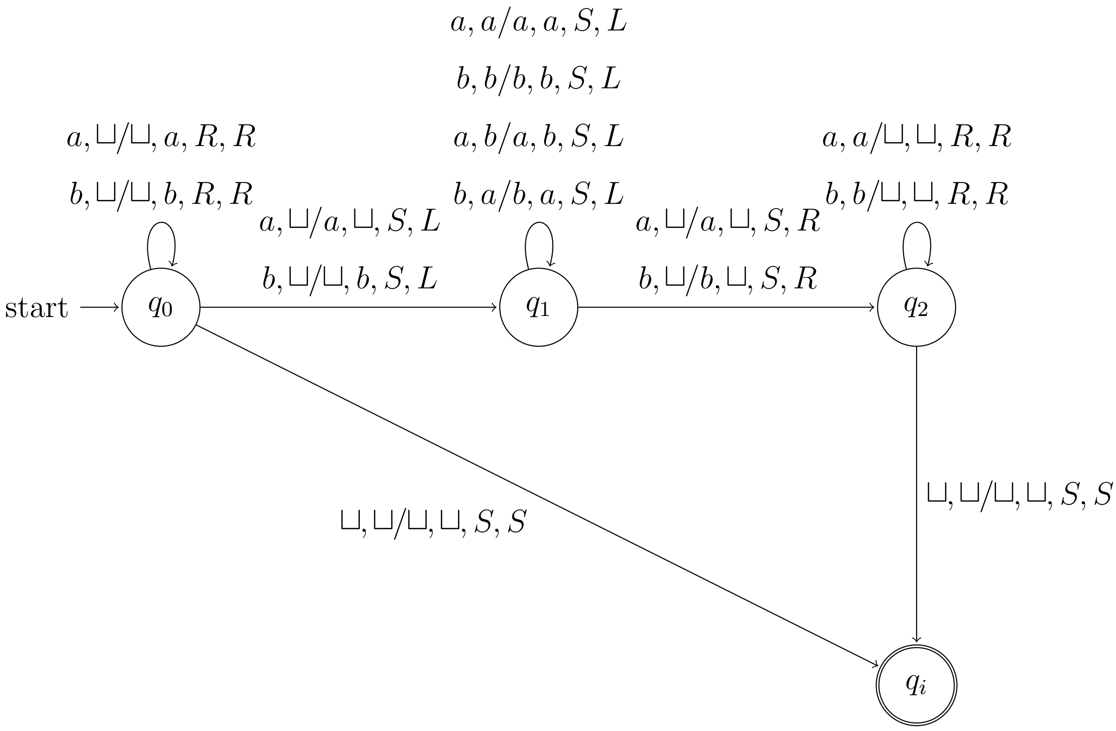
$L = \{uu^{-1} | u \in \{a, b\}^*\}$

Ötlet: nemdeterminisztikusan másoljunk át valahány karaktert a második szalagra. Ezután induljunk visszafelé a második szalagon, az elsőn továbbra is előre és ha egyező karaktereket találunk, akkor töröljük őket.



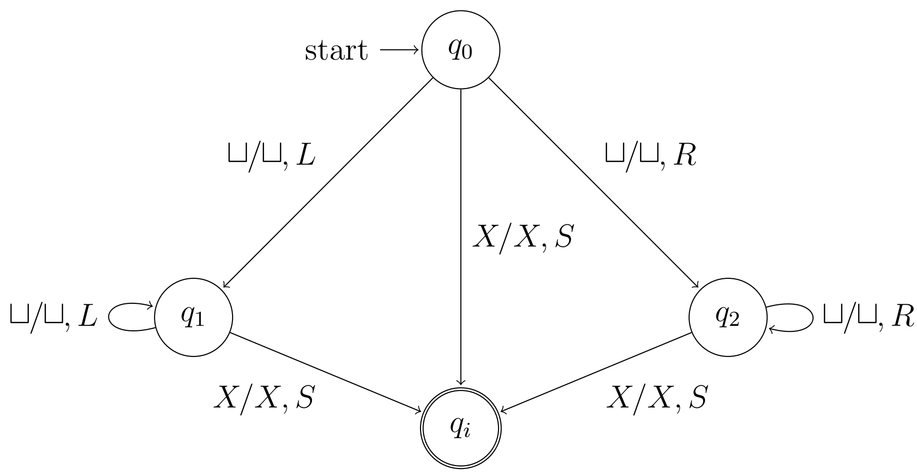
7. feladat: Készítsünk 2-szalagos nemdeterminisztikus TG-et, amely a következő nyelvet ismeri fel:

$L = \{uu | u \in \{a, b\}^*\}$

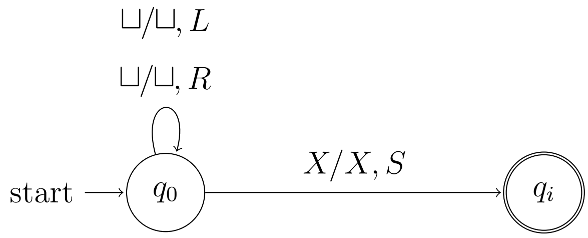


8. feladat: Kincskeresés: adott egy 1-szalagos TG, melynek szalgján elrejtettünk valahol egy "kincset". A kincset X jelöli a szalagon. Az olvasófej kezdetben tetszőleges helyen áll. Adjunk nemdeterminisztikus és determinisztikus módszereket a kincs megtalálására.

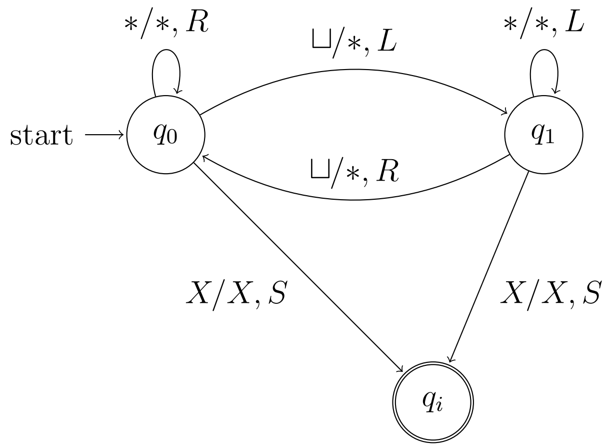
Nemdeterminisztikus 1. *ötlet: nemdeterminisztikusan döntsük el, hogy a fejtől balra vagy jobbra keressük a kincset és ennek megfelelően lépkedjünk balra vagy jobbra, amíg meg nem találjuk. Előfordulhat, hogy nem találjuk meg a kincset.*



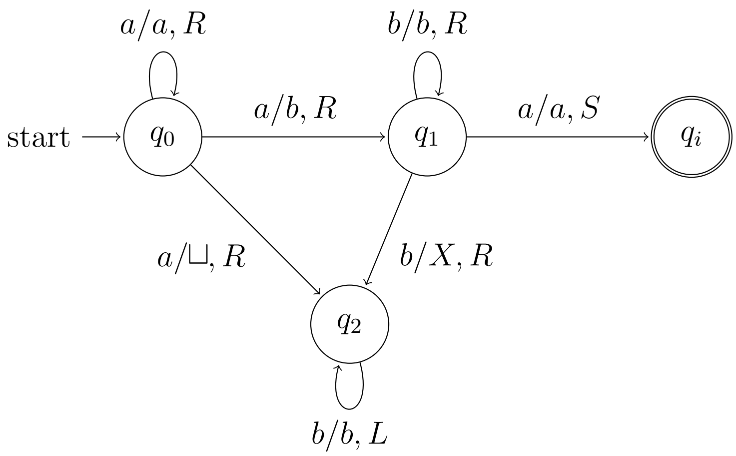
Nemdeterminisztikus 2. *ötlet: nemdeterminisztikusan lépjünk balra vagy jobbra, amíg meg nem találjuk a kincset.*

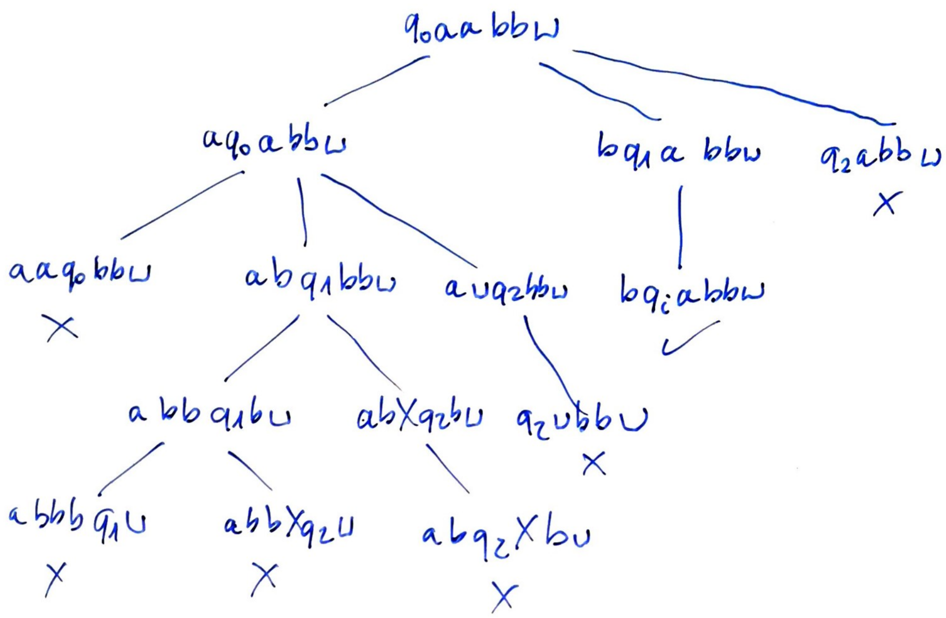


Determinisztikus *ötlet: *-okat helyezünk el az üres cellákba, amíg nem találjuk meg a kincset. Lehelyezünk egy *-ot, majd elindulunk balra, amíg üres cellát nem találunk, ha megvan, lehelyezünk egy *-ot és elindulunk jobbra hasonlóan egy üres celláig.*



9. feladat: Adott az alábbi NTG és az $u = aabb$ szó. Rajzoljuk fel az u-hoz tartozó nemdeterminisztikus számítási fát!





Ábra 1.: Megoldás

Algoritmikus eldönthetetlenség*

A korábban már megfogalmazott **Church-Turing** tézis alapján azok a problémák oldhatók meg algoritmikusan, melyekhez Turing gépet tudunk készíteni. A következőkben megvizsgáljuk, hogy vannak-e olyan problémák amik nem dönthetők el algoritmikusan és ha igen, akkor, hogyan tudjuk ezt belátni. Először a halmazok számosságával foglalkozunk, az ide tartozó ismeretek szükségesek lesznek a későbbiekben.

1 Turing gépek elkódolása

Definíció 1.1. Turing gépek egy elkódolása Tegyük fel, hogy $\Sigma = \{0, 1\}$, egy M TG kódja (jelölés: $\langle M \rangle$):

Legyen $M = \langle Q, \{0, 1\}, \Gamma, \delta, q_0, q_i, q_n \rangle$, ahol

- $Q = \{p_1, ..., p_k\}$, $\Gamma = \{X_1, ..., X_m\}$
- $D_1 = R$, $D_2 = S$, $D_3 = L$
- $p_1 = q_0$, $p_{k-1} = q_i$, $p_k = q_n$ (ahol $k \geq 3$)
- $X_1 = 0$, $X_2 = 1$, $X_3 = \sqcup$ ($m \geq 3$)

Ekkor egy $\delta(p_i, X_j) = (p_r, X_s, D_t)$ átmenet kódja a következő: $0^i 1 0^j 1 0^r 1 0^s 1 0^t$

M kódja ($\langle M \rangle$) az átmenetek felsorolása 11 töredékekkel elválasztva.

Észrevehető, hogy egy TG kódja 0-val kezdődik és végződik.

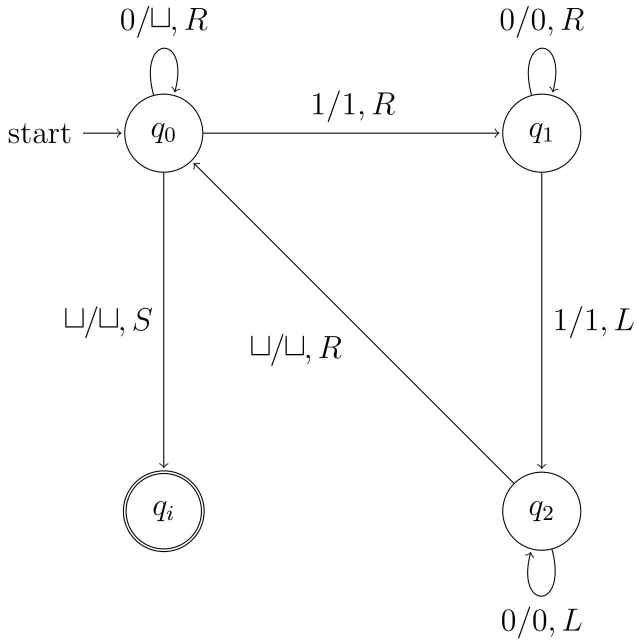
TG és szó kódja: $\langle M, w \rangle := \langle M \rangle 111w$

*A jegyzet Dr Tichler Krisztián anyagai alapján készült

1.1 Feladatok

1. feladat: Adott az alábbi TG, $\Sigma = \{0, 1\}$, $\Gamma = \{0, 1, \sqcup\}$

- q_0: 1
- q_1: 2
- q_2: 3
- q_i: 4



01010100010**11**010010010010**11**0100010000100010**0****11**00101001010**11**00100100010010000**11**

0001010001010000**11**000100010100010

Nézzük meg az első átmenet kódját: 01010100010, az első 0 q_0-t azonosítja, a második a 0-ás karaktert, a harmadik szintén q_0-t. A következő három 0 az üres cellát jelöli, az utolsó 0 pedig a jobbra mozgást.

2 Számosság

Definíció 2.1. Hossz-lexikografikus rendezés: Legyen $X = \{x_1 \prec x_2 \prec \cdots \prec x_s\}$ egy rendezett ábécé. Ekkor X^* szavainak **hossz-lexikografikus** (shortlex) rendezése alatt azt a $\prec_{\text{shortlex}}$ rendezést értjük, melyre a következők teljesülnek. Minden $u_1 \cdots u_n, v_1 \cdots v_m \in X^*$-ra $u_1 \cdots u_n \prec_{\text{shortlex}} v_1 \cdots v_m \Leftrightarrow (n < m) \vee$

$((n = m) \wedge (u_k < v_k)$, ahol k a legkisebb olyan i, melyre $u_i \neq v_i)$

Például: $X = \{0, 1\}^*$ és $0 \prec 1$, ekkor X^* szavainak hossz-lexikografikus sorrendje: $0, 1, 00, 01, 10, 11, 000, ...$

Definíció 2.2. A és B halmazoknak **megegyezik a számosságuk** ($|A| = |B|$), ha létezik köztük bijekció (kölcsönösen egyértelmű leképzés).

A-nak **legalább annyi a számossága**, mint B-nek ($|A| \geq |B|$), ha létezik injekció B-ből A-ba.

A-nak **nagyobb a számossága**, mint B-nek ($|A| > |B|$), ha létezik injekció B-ből A-ba, de nem létezik bijekció.

Képzeljük el azt az esetet, hogy egy repülőgépen mászkáló utasok számát szeretnénk meghatározni. Alapvetően nem egyszerű feladat, de minden utashoz hozzárendelhetjük az ülőhelyét, ami egy kölcsönösen egyértelmű leképezés. [1]

Tétel 2.1. Cantor-Bernstein-Schröder: Ha létezik injekció A-ból B-be és B-ből A-ba is, akkor létezik bijekció A és B között, azaz ha $|A| \leq |B|$ és $|A| \geq |B|$, akkor $|A| = |B|$.

2. feladat: Lássuk be, hogy az alábbi halmazok számossága megegyezik!

- Pozitív egész számok ($\mathbb{Z}^+$) és negatív egész számok ($\mathbb{Z}^-$):

  A feladat, hogy találjunk egy bijektív (kölcsönösen egyértelmű) lekérdezést a két halmaz között:

  $f(x) = -x$ azaz minden pozitív egész számhoz rendeljük hozzá az ellentetjét.

- Pozitív egész számok ($\mathbb{Z}^+$) és pozitív páros számok:

  Nézzük meg a következő történetet [1]: van valahol egy távoli szálloda, aminek az a fő érdekessége, hogy végtelen sok szobája van. Egy napon végtelen sok vendég érkezik a szállodába és de a portás közli, hogy sajnos az összes szoba foglalt. Kis gondolkodás *után azonban eszében jut a portásnak, hogy mégis el tudja helyezni a vendégeket a következő "csel" segítségével: a szálló összes vendégét megkéri, hogy költözzön át egy kétszer nagyobb sorszámú szobába, azaz az 1-esben lévő vendég a 2-esbe, a 2-esben lévő a 4-esbe, stb. Így felszabadulnak a páratlan sorszámú szobák és mivel végtelen sok páratlan szám van, az újonnan érkező végtelen számú vendég be tud költözni.*

$f(x) = 2x$, minden pozitív egészhez hozzárendeljük a kétszeresét.

- Természetes számok ($\mathbb{N}$) és pozitív egész számok ($\mathbb{Z}^+$):

  $f(x) = x + 1$

- $\mathbb{N}$ és $\mathbb{Z}$

$$f(x) = \begin{cases} \frac{x}{2}, \text{ ha } x \text{ páros} \\ -\lceil \frac{x}{2} \rceil, \text{ ha } x \text{ páratlan} \end{cases}$$

Definíció 2.3. Megszámlálhatóan végtelen számosság: Egy X halmaz megszámlálhatóan végtelen számosságú, ha létezik bijekció X és $\mathbb{N}$ között.

Ez alapvetően azt jelenti, hogy X elemei természetes számokkal sorszámozhatók.

Definíció 2.4. Megszámlálható halmaz: Egy halmaz megszámlálható, ha a számossága véges vagy megszámlálhatóan végtelen.

Definíció 2.5. Continuum számosság: Egy X halmaz continuum számosságú, ha létezik bijekció X és $\mathbb{R}$ között.

2.1 Összefüggések

- $|\mathbb{R}| > |\mathbb{N}|$
- $|\mathbb{R}| = |(0, 1)|$
- $|\{0, 1\}^*| = |\mathbb{N}|$

  Ebben az esetben a hossz-lexikografikus (shortlex) rendezés ad egy bijekciót, minden i természetes számhoz hozzárendeli a shortlex rendezés szerinti i-edik szót.

- $|\{L|L \subseteq \{0,1\}^*\}| = |\{0,1\}^{\mathbb{N}}| = |\mathbb{R}|$

  Itt $\{0,1\}^{\mathbb{N}}$ a megszámlálhatóan végtelen hosszúságú $\{0,1\}$ sorozatokat jelöli. Egy L nyelvhez hozzárendelhetünk egy megszámlálhatóan végtelen hoszúságú $b_L = (b_1, ..., b_i, ...)$ bitsorozatot, amelyre $b_i = 1 \iff w_i \in L$, ahol w_i a $\{0,1\}^*$ shortlex rendezésének i. eleme. b_L-t az L nyelv **karakterisztikus sorozatának** nevezzük, alapvetően azt írja le, hogy a nyelv mely szavakat tartalmazza.

- A $\{0,1\}$ feletti nyelvek halmazának számossága nagyobb, mint a $\{0,1\}$ feletti szavak számossága

- Minden H halmazra $|\mathscr{P}(H)| > |H|$

  Ebből az következik, hogy minden számosságnál van nagyobb számosság, azaz végtelen sok számosság van.

  Jelölések: $\aleph_0 = |\mathbb{N}|$ (alef-null), $\aleph_1 = |\mathscr{P}(\mathbb{N})| = |\mathbb{R}|$

  Ha valemely X halmazra $|X| = \aleph_i$, akkor $\aleph_{i+1} = |\mathscr{P}(X)|$

3 Turing-felismerhetőség

Tétel 3.1. Létezik nem Turing-felismerhető nyelv.

Bizonyítás. Korábban megnéztük, hogy a Turing gépek hatékonyan elkódolhatók a $\{0,1\}$ ábécé felett, azaz minden Turing géphez egyértelműen hozzárendelhető egy $w \in \{0,1\}^*$ szó. Ez azt jelenti, hogy a Turing gépek számossága megszámlálható.

Azt is tudjuk, hogy a $\{0,1\}$ feletti nyelvek számossága continuum, azaz van olyan nyelv, amihez nem létezik őt felismerő TG.

□

3.1 A diagonális nyelv

Jelölések: minden $i \geq 1$ esetén

- w_i a $\{0,1\}^*$ halmaz i. eleme ashortlex rendezés szerint

- M_i a w_i által kódolt TG (ha w_i nem kódol TG-et, akkor M_i egy tetszőleges olyan TG, ami nem fogad el semmit.)

Definíció 3.1. Diagonális nyelv: $L_{\text{átló}} := \{w_i | w_i \notin L(M_i)\}$

Nézzük meg, hogy miről kapta a nevét a diagonális nyelv. Vizsgáljuk meg a következő T táblázatot:

	w_1	w_2	w_3	...
M_1	0	1	1	...
M_2	1	1	0	...
M_3	0	0	1	...
...	...	...	...	...

A táblázat oszlopiban $w_1, w_2, ...$ szavak találhatók, soraiban pedig az $M_1, M_2, ...$ Turing gépek. A táblázat mindkét dimenziójában megszámlálhatóan végtelen.
$T(i, j) = 1 \iff w_j \in L(M_i)$, azaz a táblázat i. sorának j. cellája megmutatja, hogy M_i TG elfogadja-e w_j szót vagy sem, ennél fogva a táblázat i. sora az $L(M_i)$ nyelv karakterisztikus sorozatának tekinthető.

A táblázat átlójában látható bitsorozat komplementere pontosan az $L_{\text{átló}}$ nyelv karakterisztikus sorozata található, ami magyarázatot ad a nyelv elnevezésére.

Tétel 3.2. $L_{\text{átló}} \notin RE$, azaz az $L_{\text{átló}}$ nyelv nem Turing-felismerhető.

Bizonyítás. Tekintsük a fenti T táblázatot. A táblázat átlójában szereplő bitsorozatot jelöle x. Tudjuk, hogy x komplementere ($\overline{x}$) az $L_{\text{átló}}$ nyelv karakterisztikus sorozata. Minden Turing géppel felismerhető (RE-beli) nyelv karakterisztikus sorozata megegyzik a táblázat valamelyik sorával. Azonban minden $i \geq 1$ esetén $\overline{x}$ különbözik T i. sorától, hiszen az átló komplementeréről van szó, azaz az i. sor esetén legalább az i. cella értéke nem egyezik meg $\overline{x}$ i. bitjével. Ez azonban azt jelenti, hogy a diagonális nyelv karakterisztikus sorozata különbözik a táblázat összes sorától, azaz az összes RE-beli nyelv karakterisztikus sorozatától is. Tehát nincs olyan TG, ami a diagonális nyelvet ismeri fel. $\square$

3.2 Az univerzális nyelv

Vizsgáljuk meg, hogy algoritmikusan el tudjuk-e dönteni, hogy egy tetszőleges TG elfogad-e egy tetszőleges szót.

Definíció 3.2. Univerzális nyelv: $L_u = \{\langle M, w\rangle | w \in L(M)\}$

Az univerzális nyelv elkódolt Turing gép és szó párosokat tartalmaz, melyekre az igaz, hogy a párosban lévő TG elfogajda a párosban lévő szót.

Tétel 3.3. $L_u \in RE$

Bizonyítás. [2] Egy úgynevezett **univerzális** Turing gépet készítünk, ami L_u-t ismeri fel. Az U univerzális TG négy szalaggal rendelkezik és szimulálja a bemenetén kapott M TG működését a w szón. Feltesszük, hogy M egyszalagos.

Először is nézzük meg, hogy mire használjuk az egyes szalagokat:

1. szalag: itt található a bemenet $\langle M, w \rangle$, ezt a szalagot U csak olvassa.
2. szalag: ezen a szalgon M szalagjának tartalmát és a fej pozícióját tároljuk. Az egyes szimbólumokat elkodolva tartalmazza a szalag 1-esek választják el őket.
3. szalag: M aktuális állapota elkódolva.
4. szalag: segédszalag.

Most nézzük meg U működését egy $x = \langle M, w \rangle$ bemeneten:

1. Először is U megvizsgálja, hogy az első szalagján lévő bement megfelelő formájú-e, azaz $\langle M \rangle 111w$ alakú-e. Ha a bemenet nem megfelelő formájú, akkor U elutasítja azt. Ebben az esetben $x \notin L_u$.
2. U átmásolja w-t a második szalagjára, majd 0-t ír a harmadik szalagjára, ez jelöli, hogy M kezdőállapotba van.
3. U szimulálja M egy lépését. Először is keres egy $0^i10^j10^r10^s10^t$ alakú részt az első szalagon természetesen úgy, hogy 0^i-en megegyezzen a harmadik szalag tartalmával. Azaz, ha M i állapotban van, akkor olyan részszavakat keresünk, amik 0^i-vel kezdődnek. 0^j-nek pedig a második szalagon M fejének a pozíciójában lévő sorozattal kell megegyeznie.
   (a) U a harmadik szalagjáról törli 0^i-t és átírja 0^r-re. Ez szimulálja M állapotváltását.
   (b) U a második szalagon 0^j-t átírja 0^s-re. Ehhez a lépéshez U felhasználhatja a negyedik szalagot is, amennyiben $j \neq s$. Ebben az esetben U lemásolja a második szalagról a 0^j utáni részt a negyedik szalagra, majd a második szalagon elhelyezi 0^s-t ezután pedig visszamásolja a negyedik szalag tartalmát.

(c) t értékétől függően U beállítja a fejet a második szalagon.

4. Ha a szimuláció során U azt látja, hogy M elfogadó vagy elutasító konfigurációba lép, akkor U is elfogad illetve elutasít.

Ha M nem áll meg w-n, akkor U sem, ezért U nem dönti el L_u-t csak felismeri. □

Tétel 3.4. $L_u \notin R$

Bizonyítás. [2] A tételt indirekt módon bizonyítjuk. Indirekt módon tegyük fel, hogy L_u eldönthető és legyen M egy olyan TG, ami eldönti. M segítségével készítünk egy olyan M' TG-et, ami $L_{\text{átló}}$-t fogadja el.

Egy u bemenet esetén M' működése legyen a következő:

- M' u-ból előállítja $u' = u111u$ szót. M' azt akarja eldönteni, hogy u-t elfogadja-e az a TG, amit u kódol el.
- Az univerzális TG-nél látottaknak megfelelően M' szimulálja M működését u-n.
- Ha M elfogadja u-t, akkor M' elutasítja, amennyiben M elutasítja, akkor M' elfogadja, hiszen u akkor szerepel $L_{\text{átló}}$-ban, ha az általa elkódolt TG elutastja őt.

$u \in L(M') \iff u111u \notin L(M) \iff u \in L_{\text{átló}}$

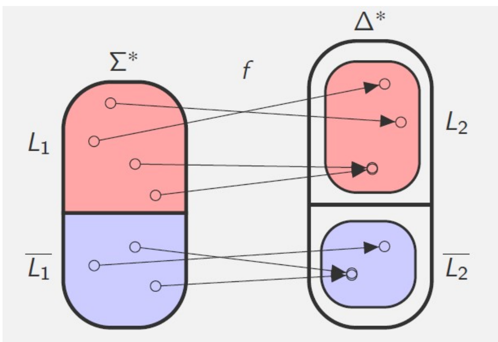
Ebből az következik, hogy $L(M') = L_{\text{átló}}$, ez a diagonális nyelvnél megfogalmazott tétel miatt, azonban lehetetlen, mivel nincs olyan TG, ami felismeri a diagonális nyelvet. Következésképpen ellentmondásra jutottunk, azaz az indirekt feltevésünk, hogy L_u eldönthető nem igaz. □

4 Visszavezetés

Adott két probléma: P_1 és P_2, P_1 eldöntéséhez van algoritmusunk, P_2-höz nincs. Ha azonban meg tudunk adni egy olyan leképezést, amely P_2 probléma minden u bemenetéhez hozzárendeli P_1 egy u' bemenetét, úgy, hogy u pontosan akkor igen példánya P_2-nek, ha u' igen példánya P_1-nek, akkor lényegében P_2 problémát is el tudjuk dönteni. Ehhez a következő algoritmust készíthetjük: u bemenet esetén határozzuk meg u'-t és szimuláljuk P_1-et megoldó algoritmus működését u'-n, ha ez elfogad, akkor a P_2-t megoldó algoritmus is fogadjon el [2]. Angol nelvű szakirodalomban *many-one reduction* néven találkozhatunk a visszavezetéssel.

Definíció 4.1. Kiszámítható függvény: Az $f : \Sigma^* \to \Delta^*$ szófüggvény kiszámítható, ha van olyan TG, ami kiszámítja.

Definíció 4.2. Visszavezetés: Az $L_1 \subseteq \Sigma^*$ nyelv visszavezethető az $L_2 \subseteq \Delta^*$ nyelvre ($L_1 \leq L_2$), ha van olyan $f : \Sigma^* \to \Delta^*$ kiszámítható szófüggvény, hogy
$w \in L_1 \iff f(w) \in L_2$



Ábra 1.: Visszavezetés [3]

A visszavezetés jelölése $L_1 \leq L_2$ intuitíven azt jelenti, hogy L_1 probléma "könnyebben" eldönthető, mint L_2, éppen ezért vezetjük vissza a "nehezebb" L_2-re. Alapvetően "egszerűsítjük" (redukáljuk) L_2-t a visszavezetéssel, azáltal, hogy memutatjuk a kapcsolatot egy már "megoldott" problémával.

Tétel 4.1. Legyen $L_1 \subseteq \Sigma^*$ és $L_2 \subseteq \Delta^*$, ekkor:

- ha $L_1 \leq L_2$ és $L_2 \in RE$, akkor $L_1 \in RE$
- ha $L_1 \leq L_2$ és $L_2 \in R$, akkor $L_1 \in R$

Tétel 4.2. Legyen $L_1 \subseteq \Sigma^*$ és $L_2 \subseteq \Delta^*$, ekkor:

- ha $L_1 \leq L_2$ és $L_1 \notin RE$, akkor $L_2 \notin RE$
- ha $L_1 \leq L_2$ és $L_1 \notin R$, akkor $L_2 \notin R$

Definíció 4.3. Megállási probléma: $L_h = \{\langle M, w\rangle | M \text{ megáll a } w \text{ bemeneten}\}$

Tétel 4.3. $L_h \in RE$

Tétel 4.4. $L_h \notin R$

4.1 Feladatok

3. feladat: [2] Adott a következő nyelv:
$L_{\neg\emptyset} = \{\langle M\rangle | L(M) \neq \emptyset\}$

Visszavezetéssel igazoljuk, hogy $L_{\neg\emptyset} \notin R$

- Vezzessük vissza L_u-t $L_{\neg\emptyset}$-re! Megadunk egy olyan f leképezést, amely az L_u egy $\langle M, w\rangle$ bemenetéhez megkonstruál egy $\langle M'\rangle$ TG kódot.
- M' a következőképp működik: szimulálja M működését w-n. Ha M elfogadja w-t, akkor M' is elfogad, ha M elutasítja w-t, akkor M' is elutasít.

$$\langle M, w\rangle \in L_u \iff w \in L(M) \iff L(M') \neq \emptyset \iff \langle M'\rangle \in L_{\neg\emptyset}$$

- Ez tahát azt jelenti, hogy $L_u \leq L_{\neg\emptyset}$ és mivel $L_u \notin R$ ezért $L_{\neg\emptyset} \notin R$ (korábbi tétel alapján).

5 Rice tétele

Definíció 5.1. RE tulajdonsága: Tetszőleges $P \subseteq RE$ halmazt a rekurzívan felsorolható nyelvek egy tulajdonságának nevezzük. P triviális, ha $P = \emptyset$ vagy $P = RE$.

$L_P = \{\langle M\rangle | L(M) \in P\}$

Tétel 5.1. Rice: Ha $P \subseteq RE$ egy nemtriviális tulajdonság, akkor $L_P \notin R$

A rekurzíven felsorolható nyelvek egy P tulajdonsága alapvetően RE egy osztálya. Ha egy P tulajdonság triviális, akkor vagy RE összes elemére teljesül vagy egyikre sem. Egy $L \in RE$ nyelv rendelkezik egy P tulajdonsággal, ha $L \in P$

A tétel következménye, hogy nem dönthetők el algoritmikusan pl. a következő problémák:

- Egy M TG az üres nyelvet ismeri-e fel: $P = \{\emptyset\}$
- Egy M TG véges nyelvet ismer-e fel: $P = \{L | L \text{ véges }\}$
- Egy M TG környezetfüggetlen nyelvet ismer-e fel:
  $P = \{L | L \text{ környezetfüggetlen nyelv}\}$

6 A Post megfelelkezési probléma (PMP)

A korábbiakban olyan algoritmikusan eldönthetetlen problémákat néztünk meg, melyek közvetlenül a Turing gépekhez kapcsolódnak. Most egy másik esetet nézünk meg.

Definíció 6.1. Dominókészlet: Adott a Σ ábécé és $u_1, ..., u_n, v_1, ..., v_n \in \Sigma^+$ $(n \geq 1)$, a $D = \{\frac{u_1}{v_1}, ..., \frac{u_n}{v_n}\}$ halmazt dominókészletnek nevezzük.

Definíció 6.2. Dominókészlet megoldása: Az $\frac{u_{i_1}}{v_{i_1}} \cdots \frac{u_{i_m}}{v_{i_m}}$ $(m \geq 1$ és $i_1, ...i_m \geq 1)$ dominósorozat a $D = \{\frac{u_1}{v_1}, ..., \frac{u_n}{v_n}\}$ dominókészlet egy megoldása, ha $u_{i_1} \cdots u_{i_m} = v_{i_1} \cdots v_{i_m}$

Egy dominókészletnek több megoldása is lehet, ugyanazt a dominót többször is felhasználjuk és nem kell a készlet összes dominóját felhasználni.

Definíció 6.3. PMP: $L_{PMP} = \{\langle D \rangle | D\text{-nek van megoldása}\}$

Tétel 6.1. $L_{PMP} \in RE$

Tétel 6.2. $L_{PMP} \notin R$

6.1 Feladatok

4. feladat: Adjunk meg egy olyan dominókészletet, aminek van megoldása! Adjunk meg egy megoldást is!

Legyen a dominókészlet a következő:

$$\left\{\frac{b}{ca}, \frac{dd}{e}, \frac{a}{ab}, \frac{ca}{a}, \frac{abc}{c}\right\}$$

Egy megoldás:

$$\frac{a}{ab}\frac{b}{ca}\frac{ca}{a}\frac{a}{ab}\frac{abc}{c}$$

Irodalomjegyzék

[1] Mosóczi András *A gondolkodás forradalma* (Typotex kiadó)

[2] Gazdag Zsolt *Bevezetés a számításelméletbe* – egyetemi jegyzet

[3] Tichler Krisztián *A számításelmélet alapjai II.* – előadás

Bonyolultságelmélet*

A bonyolultságelmélet feladata az eldönthető problémák "csoportosítása" általában idő- és tárigény alapján.

1 Időbonyolultsági osztályok

Definíció 1.1. Determinisztikus időbonyolultsági osztályok:

- $TIME(f(n)) = \{L|L$ eldönthető O(f(n)) időigényű determinisztikus Turing géppel$\}$
- $P = \bigcup_{k\geq 1} TIME(n^k)$

Definíció 1.2. Nemdeterminisztikus időbonyolultsági osztályok:

- $NTIME(f(n)) = \{L|L$ eldönthető O(f(n)) időigényű determinisztikus Turing géppel$\}$
- $NP = \bigcup_{k\geq 1} NTIME(n^k)$

P azokat a problémákat tartalmazza, melyek polinom időben eldönthetők determinisztikus Turing géppel, azaz ismert olyan algoritmus, ami polinom időben megoldja. NP olyan problémákat tartalmaz, melyek polinom időben eldönthetők nemdeterminisztikus Turing géppel. Egy P NP-beli probléma esetén ha rendelkezésünkre áll a probléma egy példánya, akkor polinom időben eldönthető, hogy a példány megoldása-e a problémának. Ebben az esetben a nemdeterminisztikus TG nemdeterminisztikusan "megsejti" a probléma egy megoldását majd ellenőrzi, hogy az megfelel-e.

Intuitíven azt mondhatjuk, hogy P osztályban szerepelnek a "hatékonyan" megoldható problémák, NP-ben a hatékonyan ellenőrizhetők.

Észrevehető, hogy a $P \subseteq NP$ összefüggés fennáll, de felmerül a kérdés, hogy $P = NP$ teljesül-e. Erre a kérdésre jelenleg nem ismert a válasz, az évszázad egyik legjelentősebb

*A jegyzet Dr Tichler Krisztián anyagai alapján készült

matematikai problémája. A sejtés az, hogy $P \neq NP$ de ehhez nincs bizonyítás. Amennyiben $P = NP$ igaz lenne, az azt jelentetné, hogy bizonyos bonyolúlt problémákhoz megadható polinomiális, determinisztikus megoldás, ez például azt is jelentené, hogy a jelenleg ismert titkosítások könnyen feltörhetővé válnának. Persze, ha $P = NP$ igaz lenne, csak egy elméleti lehetőséget adna a problémák megoldhatóságára, konkrét megoldást attól még nem lenne.

2 Visszavezetés polinom időben

Definíció 2.1. Polinom időben kiszámítható függvény: Egy $f : \Sigma^* \to \Delta^*$ függvény polinom időben kiszámítható, ha létezik olyan TG, ami polinom időben kiszámítja.

Definíció 2.2. Polinom idejű visszavezetés: Egy $L_1 \subseteq \Sigma^*$ nyelv polinom időben visszavezethető $L_2 \subseteq \Delta^*$ nyelvre, ha létezik $f : \Sigma^* \to \Delta^*$ polinom időben kiszámítható szófüggvény, hogy $w \in L_1 \iff f(w) \in L_2$ ($L_1 \leq_p L_2$)

Tétel 2.1. Ha $L_1 \leq_p L_2$ és $L_2 \in P$, akkor $L_1 \in P$

Tétel 2.2. Ha $L_1 \leq_p L_2$ és $L_2 \in NP$, akkor $L_1 \in NP$

Definíció 2.3. NP-teljesség: Egy L probléma NP-teljes, ha NP-beli és minden NP-beli probléma polinom időben visszavezethető rá.

Tétel 2.3. Ha L NP-teljes, és $L \in P$, akkor $P = NP$

A fenti tétel alapján, ha egy NP-teljes problémáról be tudnánk látni, hogy P-beli (azaz tudunk hozzá polinom idejű determinisztikus megoldást adni), akkor mivel egy NP-teljes problémára minden NP-beli probléma polinom időben visszavezethető $P = NP$ következne.

Tétel 2.4. Ha L_1 NP-teljes és $L_2 \in NP$ valamint $L_1 \leq_p L_2$, akkor L_2 is NP-teljes.

2.1 Elérhetőség probléma

Példaként nézzünk megy egy tipikus P-beli problémát.

$$L_{\text{Elér}} = \{\langle G, s, t\rangle | G \text{ irányított gárf és s-ből elérhető t}\}$$

Az elérhetőség probléma könnyen megoldható a szélességi bejárás segítségével, tudunk készíteni, olyan Turing gépet, ami ezt az algoritmust követi. A szélességi bejárásról pedig ismert, hogy polinomiális műveletigényű: $O(n + m)$

3 NP-teljes nyelvek

3.1 SAT

A SAT probléma a nulladrendű logika témaköréhez tartozik.

Definíció 3.1. Literál: Egy itéletváltozót vagy annak negáltját literálnak nevezzük.

Pl.: X $\neg X$

Definíció 3.2. Elemi diszjunkció: Lierálok diszjunkcióját elemi diszjunkciónak másnéven klóznak nevezzük.

Pl.: $(X \vee Y)$

Definíció 3.3. Konjunktív normálforma (KNF): Elemi diszjunkciók konjunkciója.

Pl.: $(X \vee Y) \wedge (X \vee \neg Z)$

Definíció 3.4. SAT: $SAT = \{\langle\phi\rangle | \phi \text{ kielégíthető KNF}\}$

Tétel 3.1. Cook-Levin: A SAT probléma NP-teljes.

3.2 További NP-teljes nyelvek

- $3SAT = \{\langle\phi\rangle | \phi$ kielégíthető KNF és minden klóz pontosan 3 literált tartalmaz$\}$

  Pl.: $\phi = (X \vee Y \vee \neg Z) \wedge (\neg X \vee \neg Y \vee Z)$

- 3Színezés $= \{\langle G\rangle | G$ 3-színezhető$\}$

  Egy G gráf 3-színezhető, ha a csúcsai 3 színnel színezhetők úgy, hogy a szomszédos csúcsok színei különböznek.

- $KLIKK = \{\langle G, k\rangle | G$-nek van k méretű teljes részgráfja$\}$

- HÚ $= \{\langle G, s, t\rangle | G$-ben van s-ből t-be irányított Hamilton út$\}$

- $TSP = \{\langle G, k\rangle |$a nemnegatív élekkel súlyozott G-ben ven-e legfelejebb k súlyú Hamilton kör$\}$

  Ez az úgynevezett utazóügynök probléma eldöntési verziója. A számítási verzióban a legkisebb összsúlyú Hamilton kört keressük.

4 NP lehetséges szerkezete

Definíció 4.1. NP-köztes nyelv: L NP-köztes nyelv, ha $L \in NP$ és $L \notin P$, valamint L nem NP-teljes.

Tétel 4.1. Ladner: Ha $P \neq NP$, akkor létezik NP-köztes nyelv.

Az alábbi két problémáról nem tudjuk, hogy P-beliek-e illetve, NP-teljesek-e, íg NP-köztes nyelvet jelölnek:

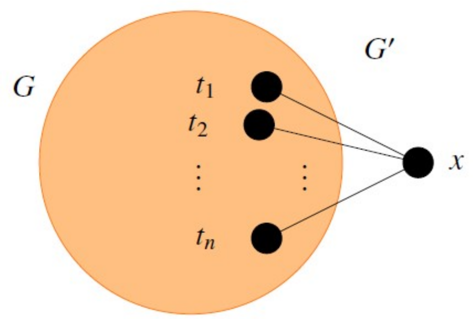
- Gráfizomorfizmus = $\{\langle G_1, G_2\rangle | G_1$ és G_2 irányítatlan izomorf gráfok$\}$
- Prímfaktorizáció: adjuk meg egy egész szám prímtényezős felbontását.

5 Feladatok

1. feladat: Igazoljuk, hogy a 4Színezés NP-teljes.

A megoldás során a 2.4 tételt fogjuk felhasználni. Ennek megfelelően belátjuk, hogy a 4Színezés NP-beli majd azt, hogy 3Színezés $\leq_p$ 4Színezés

- a 4Színezés $\in NP$, mivel megadható olyan NTG, amely egy számítási ágán polinom időben előállítja a csúsok egy színezését. Nem biztos, hogy a színezés helyes, ez viszont szintén polinom időben ellenőrizhető (pl. szélességi kereséssel).
- Vezessük vissza a a 3Színezés-t a 4Színezés-re, ehhez szükségünk van egy $f : \langle G\rangle \rightarrow \langle G'\rangle$ polinom időben kiszámítható függvényre, úgy, hogy G akkor és csak akkor 3-színezhető, ha G' 4-színezhető.
- Vegyünk az eredeti G gráfhoz egy új x csúcsot, úgy, hogy G összes csúcsával összekötjük: 1. ábra (ez polinom időben megtehető)
- Ha G 3-színezhető, akkor G'-ben x csúcsot színezzük ki, a 4. színnel, így G' 4-színezését kapjuk.
- Ha G' 4-színezhető, akkor G egy csúcsának a színe sem egyezhet meg x színével, hiszen x minden más csúccsal szomszédos, ami azt jelenti, hogy G csúcsait a maradék 3 színnel színezhetjük. Ez azonban azt jelenti, hogy ez a 4-színezés a G csúcsain valójában egy 3-színezés, tehát G 3-színezhető.



Ábra 1.: 4-színezés [2]

5.1 Hamilton út problémák

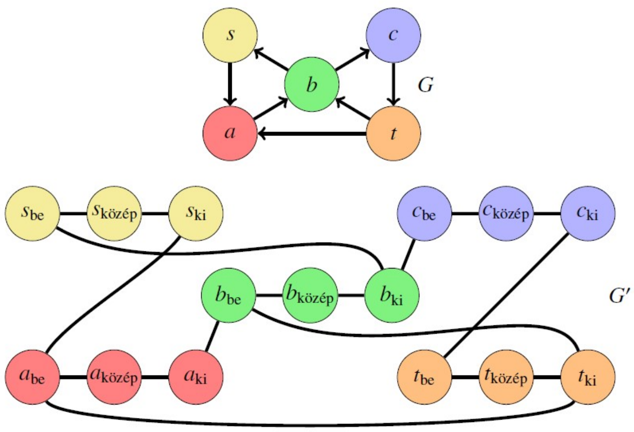
Egy n csúcs gráf esetén kereshetünk Hamilton utakat / köröket, ha a csúcsokat valamilen módon sorbarendezzük és ellenőrizzük, hogy ez megfelelő megoldás-e (élsúlyozást is figyelembe lehet venni). Ez azonban n csúcs esetén $n!$ lehetőséget jelent, ami nem polinomiális. Egy NTG egy számítási ágán előállíthatja a gráf csúcsainak egy sorrendjét ez polinom időbe telik majd szintén polinom időben ellenőrizhető, hogy a kapott sorrend megfelel-e. Ez azt jelenti, hogy a Hamilton út problémák NP-beliek.

2. feladat: Igazoljuk, hogy HÚ $\leq_p$ IHÚ

HÚ $= \{\langle G, s, t\rangle | G$-ben van s-ből t-be irányított Hamilton út$\}$

IHÚ $= \{\langle G, s, t\rangle | G$-ben van s-ből t-be irányítatlan Hamilton út$\}$

- Megadunk egy $f : \langle G, s, t\rangle \to \langle G', s', t'\rangle$ polinom időben kiszámítható függvényt, úgy, hogy G-ben akkor és csak akkor van s csúcsból t csúcsba vezető irányított Hamilton út, ha G'-ben van s'-ből t'-be vezető irányítatlan Hamilton út.
- G minden u csúcsához 3 csúcs tarttozzon G'-ben a következők szerint (2.):
  - a 3 csúcs: $u_{be}, u_{\text{közép}}, u_{ki}$
  - G' élei közé vegyük fel $\{u_{be}, u_{\text{közép}}\}$ és $\{u_{\text{közép}}, u_{ki}\}$ éleket
  - Minden G-beli (u, v) irányított él esetén, G'-höz adjuk hozzá $\{u_{ki}, v_{be}\}$ élt
  - $s' := s_{be}$, $t' := t_{ki}$
- G' konstrukciója polinomiális idejű



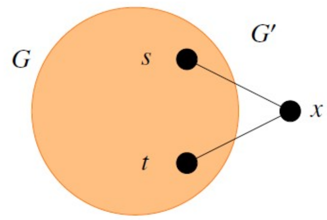
Ábra 2.: Irányítatlan Hamilton út [2]

- Ha G-ben van s és t között irányított p Hamilton út, akkor G'-ben p szerint haladva $v_{be}, v_{\text{közép}}, v_{ki}$ sorrendben szintén Hamiltun utat találunk s' és t' között
- Ha G'-ben van irányítatlan p Hamilton út s'-ből t'-be, akkor p minden csúcs esetén $v_{be}, v_{\text{közép}}, v_{ki}$ sorrendben kell, hogy haladjon, mivel $v_{\text{közép}}$ egy másodfokú csúcs és ezért másképp nem lehetne rajt a Hamilton úton (nem tudunk róla más irányban elindulni). Ha összevonjuk a $v_{be}, v_{\text{közép}}, v_{ki}$ hármasokat, valamint ha az $\{u_{ki}, v_{be}\}$ jellegű éleket irányítjuk, akkor G egy irányított Hamilton útját kapjuk s és t között.

3. feladat: Igazoljuk, hogy IHÚ $\leq_p$ IHK

IHK $= \{\langle G\rangle | G$ irányítatlan gráfban van Hamilton kör$\}$

- Megadunk egy $f : \langle G, s, t\rangle \rightarrow \langle G'\rangle$ szófüggvényt, úgy, hogy G akkor és csak akkor van s csúcsból t-be irányítatlan Hamilton út, ha G'-ben van irányítatlan Hamilton kör.
- G'-ben vegyünk fel egy új x csúcsot, valamint $\{s, x\}$ és $\{t, x\}$ irányítatlan éleket (ez polinomiális időben megtehető) (3. ábra)



Ábra 3.: Irányítatlan Hamilton kör [2]

- Ha G-ben van s és t csúcsok között irányítatlan Hamilton út, akkor ehhez hozzávéve $\{s, x\}$ és $\{t, x\}$ éleket G'-ben egy irányítatlan Hamilton kört kapunk.
- Ha G'-ben van irányítatlan Hamilton kör, akkor az mindenképp tartalmazza $\{s, x\}$ és $\{t, x\}$ éleket, hiszen x csúcs másodfokú és csak ez a két él illeszkedik rá (azaz másképp x nem szerepelhetne a Hamilton körben). Ha elhagyjuk x csúcsot, valamint $\{s, x\}$ és $\{t, x\}$ éleket, akkor egy G-beli s és t között vezető irányítatlan Hamilton utat kapunk.

Irodalomjegyzék

[1] Gazdag Zsolt *Bevezetés a számításelméletbe* – egyetemi jegyzet

[2] Tichler Krisztián *A számításelmélet alapjai II.*