ELTE IK - Programtervező Informatikus BSc

Záróvizsga tételek

13. Számításelmélet

Számításelmélet

A Turing gép és a Church-Turing tézis. Turing gépek variánsok: többszalagos, nemdeterminisztikus, számoló, offline. Rekurzív és rekurzívan felsorolható nyelvek. Eldönthetetlen problémák. Idő- és tárbonyolultsági osztályok: P, NP, PSPACE. NP-teljes problémák.

1 Kiszámíthatóság

1.0.1 Algoritmusmodellek

- **Gödel**: rekurzív függvények (primitív rekurzív függvények 1931-ben, majd általánosabb 1934-ben)
- **Church**: λ-kalkulus, λ-definiálható függvények: ekvivalensek a rekurzív függvényekkel (bizonyított)
- **Turing**: Turing-gép (1936), a λ-definiálható és a Turing-géppel kiszámítható függvények megegyeznek (bizonyított)

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

1.0.2 Fogalmak

Kiszámítási problémának nevezünk egy olyan, a matematika nyelvén megfogalmazott kérdést, amire egy algoritmussal szeretnénk megadni a választ. A gyakorlati élet szinte minden problémájához rendelhető, megfelelő absztrakciót használva, egy kiszámítási probléma.

Egy problémát a hozzá tartozó konkrét bementettel együtt a probléma egy példányának nevezzük.

Speciális kiszámítási probléma az eldöntési probléma. Ilyenkor a problémával kapcsolatos kérdés egy eldöntendő kérdés, tehát a probléma egy példányára a válasz "igen" vagy "nem" lesz.

Egy kiszámítási probléma reprezentálható egy $f : A \to B$ függvénnyel. Az A halmaz tartalmazza a probléma egyes bemeneteit, jellemzően egy megfelelő ábécé feletti szóban elkódolva, míg a B halmaz tartalmazza a bemenetekre adott válaszokat, szintén valamely alkalmas ábécé feletti szóban elkódolva. Értelemszerűen, ha eldöntési problémáról van szó, akkor az f értékkészlete, vagyis a B egy két elemű halmaz: $\{igen, nem\}$, $\{1, 0\}$, stb.

Kiszámítható függvény: Egy $f : A \to B$ függvényt *kiszámíthatónak* nevezünk, ha minden $x \in A$ elemre az $f(x) \in B$ függvényérték kiszámítható valamilyen algoritmikus modellel.

Megoldható, eldönthető probléma: Egy kiszámítási probléma *megoldható* (eldöntési probléma esetén azt mondjuk, hogy *eldönthető*), ha az általa meghatározott függvény kiszámítható.

Algoritmusok időigénye: Legyenek $f, g : \mathbb{N} \to \mathbb{N}$ függvények, ahol $\mathbb{N}$ a természetes számok halmaza. Azt mondjuk, hogy f legfeljebb olyan gyorsan nő, mint g (jelölése: $f(n) = \mathcal{O}(g(n))$), ha $\exists c > 0$ és $n_0 \in \mathbb{N}$, hogy $f(n) \leq c * g(n)$ $\forall n \geq n_0$. Az $f(n) = \Omega(g(n))$ jelöli azt, hogy $g(n) = \mathcal{O}(f(n))$ teljesül és $f(n) = \Theta(g(n))$ jelöli azt, hogy $f(n) = \mathcal{O}(g(n))$ és $f(n) = \Omega(g(n))$ is teljesül.

Példa: $3n^3 + 5n^2 + 6 = \mathcal{O}(n^3)$, $n^k = \mathcal{O}(2^n)$ $\forall k \geq 0$, stb.

Tétel: Minden polinomiális függvény lassabban nő, mint bármely exponenciális függvény, azaz minden $p(n)$ polinomhoz és $c > 0$-hoz $\exists n_0$ egész szám, hogy $\forall n \geq n_0$ esetén $p(n) \leq 2^{cn}$

Kiszámítási probléma megfeleltetése eldöntési problémának: Tekintsünk egy P kiszámítási problémát és legyen $f : A \to B$ a P által meghatározott függvény. Ekkor megadható P-hez egy P' eldöntési probléma úgy, hogy P' pontosan akkor eldönthető, ha P kiszámítható. Állítsuk párba ugyanis minden $a \in A$ elemre az a és $f(a)$ elemeket, és kódoljuk el az így kapott párokat egy-egy szóban. Ezek után legyen P' az így kapott szavakból képzett formális nyelv. Nyilvánvaló, hogy ha minden $a \in A$ és $b \in B$ elemre az $(a, b) \in P'$ tartalmazás eldönthető (azaz P' eldönthető), akkor P kiszámítható és fordítva. E megfeleltetés miatt a továbbiakban jellemzően eldöntési problémákkal foglalkozunk.

2 Turing-gépek

Hasonlóan a véges automatához vagy a veremautomatához, a Turing-gép is egy véges sok állapottal rendelkező eszköz. A Turing-gép egy két irányban végtelen szalagon dolgozik. A szalag cellákra van osztva, tulajdonképpen ez a gép (korlátlan) memóriája. Kezdetben a szalagon csak a bemenő szó van, minden cellán egy betű. A szalag többi cellája egy úgynevezett blank vagy szóköz ($\sqcup$) szimbólumokkal van feltöltve. Kezdetben a gép úgynevezett író-olvasó feje a bemenő szó első betűjén áll és a gép a kezdőállapotában van. A gép az író-olvasó fejet tetszőlegesen képes mozgatni a szalagon. Képes továbbá a fej pozíciójában a szalag tartalmát kiolvasni és átírni. A gépnek van két kitüntetett állapota, a q_i és a q_n állapotok. Ha ezekbe az állapotokba kerül, akkor rendre elfogadja illetve elutasítja a bemenő szót. Formálisan a Turing-gépet a következő módon definiáljuk.

A Turing-gép formális definíciója: A Turing-gép egy olyan $M = (Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n)$ rendszer, ahol:

- Q az állapotok véges, nem üres halmaza,
- $q_0, q_i, q_n \in Q$, q_0 a kezdőállapot, q_i az elfogadó állapot, q_n pedig az elutasító állapot,
- Σ és Γ ábécék, a bemenő jelek és a szalagszimbólumok ábécéje úgy, hogy $\Sigma \subseteq \Gamma$ és $\Gamma - \Sigma$ tartalmaz egy speciális $\sqcup$ szimbólumot,
- $\delta : (Q - \{q_i, q_n\}) \times \Gamma \to Q \times \Gamma \times \{L, R, S\}$ az átmenetfüggvény.

Úgy mint a veremautomaták esetében, egy M Turing-gép működésének fázisait is konfigurációkkal írhatjuk le.

Turing-gép konfigurációja: Az M Turing-gép konfigurációja egy olyan uqv szó, ahol $q \in Q$ és $u, v \in \Gamma^*$, $v \neq \varepsilon$. Ez a konfiguráció az M azon állapotát tükrözi amikor a szalag tartalma uv (uv előtt és után a szalagon már csak $\sqcup$ van), a gép a q állapotban van, és az író-olvasó fej a v első betűjére mutat. M összes konfigurációjának halmazát $\mathcal{C}_M$-el jelöljük.

Turing-gép kezdőkonfigurációja: M kezdőkonfigurációja egy olyan $q_0u\sqcup$ szó, ahol u csak Σ-beli betűket tartalmaz.

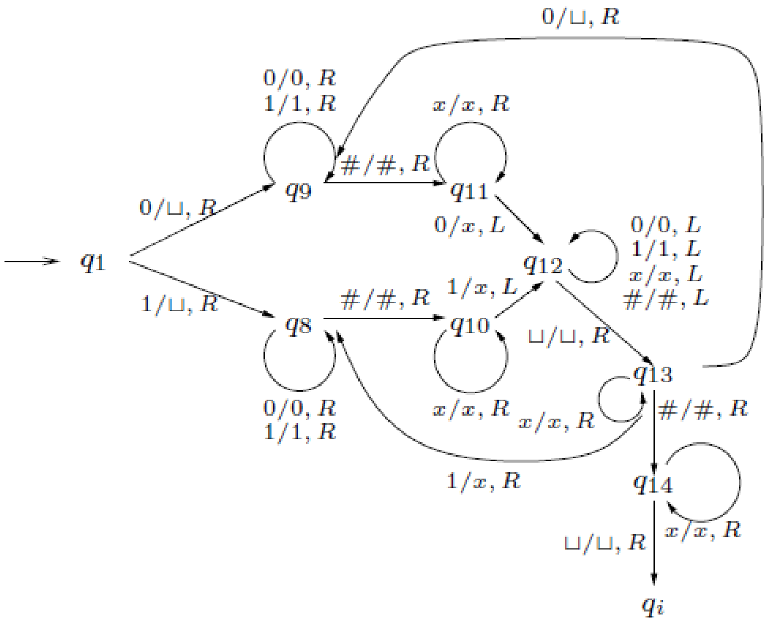
Turing-gép konfigurációátmenete: M konfigurációátmenete egy olyan $\vdash\subseteq \mathcal{C}_M \times \mathcal{C}_M$ reláció, amit a következőképpen definiálunk. Legyen $uqav$ egy konfiguráció, ahol $a \in \Gamma$ és $u, v \in \Gamma^*$. A következő három esetet különböztetjük meg:

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

Azt mondjuk, hogy M véges sok lépésben eljut a C konfigurációból a C' konfigurációba (jele $C \vdash^* C'$), ha létezik olyan $n \geq 0$ és $C_1, ...C_n$ konfigurációsorozat, hogy $C_1 = C$, $C_n = C'$ és minden $1 \leq i < n$-re $C_i \vdash C_{i+1}$.

Ha $q \in \{q_i, q_n\}$, akkor azt mondjuk, hogy az uqv konfiguráció egy megállási konfiguráció. Továbbá, $q = q_i$ esetében elfogadó, míg $q = q_n$ esetében elutasító konfigurációról beszélünk.

Turing-gép által felismert nyelv: Az M Turing-gép által felismert nyelv (jelölése $L(M)$) azoknak az $u \in \Sigma^*$ szavaknak a halmaza, melyekre igaz, hogy $q_0u\sqcup \vdash^* xq_iy$ valamely $x, y \in \Gamma^*$, $y \neq \varepsilon$ szavakra.



ábra 1: Egy, az $L = \left\{u\#u \mid u \in \{0,1\}^+\right\}$ felismerő Turing-gép.

Turing-gépek ekvivalenciája: Két Turing-gépet ekvivalensnek nevezünk, ha ugyanazt a nyelvet ismerik fel.

Turing-felismerhető nyelv, rekurzívan felismerhető nyelvek osztálya: Egy $L \subseteq \Sigma^*$ nyelv Turing-felismerhető, ha $L = L(M)$ valamely M Turing-gépre. A Turing-felismerhető nyelveket szokás *rekurzívan felsorolhatónak* is nevezni. A rekurzívan felsorolható nyelvek osztályát RE-vel jelöljük.

Turing-eldönthető nyelv, rekurzív nyelvek osztálya: Egy $L \subseteq \Sigma^*$ nyelv Turing-eldönthető, ha létezik olyan Turing-gép, amely minden bemeneten megállási konfigurációba jut és felismeri L-et. A Turing-felismerhető nyelveket szokás *rekurzívnak* is nevezni. A rekurzív nyelvek osztályát R-rel jelöljük.

Turing-gép futási ideje, időigénye: Tekintsünk egy $M = (Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n)$ Turing-gépet és annak egy $u \in \Sigma^*$ bemenő szavát. Azt mondjuk, hogy M futási ideje (időigénye) az u szón n $(n \geq 0)$, ha M a $q_0u\sqcup$

kezdőkonfigurációból n lépésben el tud jutni egy megállási konfigurációba. Ha nincs ilyen szám, akkor M futási ideje az u szón végtelen.

Legyen $f : \mathbb{N} \to \mathbb{N}$ egy függvény. Azt mondjuk, hogy M időigénye $f(n)$ (vagy azt, hogy M egy $f(n)$ időkorlátos gép), ha minden $u \in \Sigma^*$ input szóra M időigénye az u szón legfeljebb $f(l(u))$.

2.0.1 Többszalagos Turing-gépek

A többszalagos Turing-gépek, értelemszerűen, egynél több szalaggal rendelkeznek. Mindegyik szalaghoz tartozik egy-egy író-olvasó fej, melyek egymástól függetlenül képesek mozogni a szalagon.

Többszalagos Turing-gép definíciója: Legyen $k > 1$. Egy k-szalagos Turing-gép egy olyan $M = (Q, \Sigma, \Gamma, \delta, q_0, q_i, q_n)$ rendszer, ahol a komponensek a δ kivételével megegyeznek az egyszalagos Turing-gép komponenseivel, δ pedig a következőképpen adódik. $\delta : (Q - \{q_i, q_n\}) \times \Gamma^k \to Q \times \Gamma^k \times \{L, R, S\}^k$. Legyenek $q, p \in Q$, $a_1, a_2, ..., a_k, b_1, b_2, ..., b_k \in \Gamma$ és $D_1, D_2, ..., D_k \in \{L, R, S\}$. Ha $\delta(q, a_1, a_2, ..., a_k) = (p, b_1, b_2, ..., b_k, D_1, D_2, ..., D_k)$, akkor a gép akkor a gép a q állapotból, ha a szalagjain rendre az $a_1, a_2, ..., a_k$ betűket olvassa, át tud menni a p állapotba, miközben az $a_1, a_2, ..., a_k$ betűket átírja a $b_1, b_2, ..., b_k$ betűkre és a szalagokon a fejeket $D_1, D_2, ..., D_k$ irányokba mozgatja.

A többszalagos Turing-gép konfigurációja, a konfigurációátmenet valamint a felismert illetve eldöntött nyelv definíciója az egyszalagos eset értelemszerű általánosítása. A többszalagos Turing-gép időigényét is az egyszalagoshoz hasonlóan definiáljuk.

Többszalagos és egyszalagos gépek ekvivalenciája: Minden k-szalagos, $f(n)$ időkorlátos Turing-géphez van vele ekvivalens $\mathcal{O}(n * f(n))$ időkorlátos egyszalagos Turing-gép.

2.0.2 Nemdeterminisztikus Turing-gépek

Egy M nemdeterminisztikus Turing-gép állapotfüggvénye $\delta : (Q - \{q_i, q_n\}) \times \mathcal{P}(\Gamma \to Q \times \Gamma \times \{L, R\})$ alakú. Tehát M minden konfigurációjából néhány (esetleg nulla) különböző konfigurációba mehet át. Ily módon M számítási sorozatai egy u szón egy fával reprezentálhatók. A fa csúcsa M kezdőkonfigurációja, a szögpontjai pedig M konfigurációi. A fa minden levele megfelel M egy számítási sorozatának az u-n. M akkor fogadja el u-t, ha a fa valamelyik levele elfogadó konfiguráció. Nevezzük ezt a most leírt fát az M nemdeterminisztikus számítási fájának az u-n. Az M által felismert nyelv a determinisztikus esethez hasonlóan definiálható, a gép által eldöntött nyelv pedig a következőképpen.

Nemdeterminisztikus Turing-gép által eldöntött nyelv: Azt mondjuk, hogy egy nemdeterminisztikus M Turing-gép eldönt egy $L \subseteq \Gamma^*$ nyelvet, ha felismeri, és minden $u \in \Sigma*$ szóra M számítási sorozatai végesek és elfogadási vagy elutasítási konfigurációba vezetnek.

Nemdeterminisztikus Turing-gép időigénye: Legyen $f : \mathbb{N} \to \mathbb{N}$ függvény, M egy nemdeterminisztikus Turing-gép. Az M időigénye $f(n)$, ha egy n hosszú u bemeneten nincsenek M-nek $f(n)$-nél hosszabb számítási sorozatai, azaz az M számítási fája az u-n legfeljebb $f(n)$ magas.

Determinisztikus és nemdeterminisztikus Turing-gépek ekvivalenciája: Minden M nemdeterminisztikus Turing-géphez megadható egy ekvivalens M' determinisztikus Turing-gép. Továbbá, ha M $f(n)$ időigényű valamely $f : \mathbb{N} \to \mathbb{N}$ függvényre, akkor M' $2^{\mathcal{O}(f(n))}$ időigényű.

3 Eldönthetetlen problémák

Ebben a fejezetben megmutatjuk, hogy bár a Turing-gép a lehető legáltalánosabb algoritmus modell, mégis vannak olyan problémák, melyek nem számíthatók ki Turing-géppel.

Emlékeztető: A rekurzívan felsorolható (Turing-felismerhető) nyelvek osztályát RE-vel, a rekurzív (Turing-eldönthető) nyelvek osztályát R-rel jelöljük.

Világos, hogy $R \subseteq RE$. A célunk az, hogy megmutassuk: az R valódi részhalmaza az RE-nek, azaz van olyan nyelv (probléma) ami Turing-felismerhető, de nem eldönthető.

Csak olyan Turing-gépeket fogunk vizsgálni, melyek bemenő ábécéje a $\{0,1\}$ halmaz. Ez nem jelenti az általánosság megszorítását, hiszen ha találunk egy olyan $\{0,1\}$ feletti nyelvet, melyet nem lehet eldönteni ilyen Turing-géppel, akkor ezt a nyelvet egyáltalán nem lehet eldönteni.

3.0.1 Turing-gépek kódolása

A $\{0,1\}$ feletti szavak felsorolhatóak (vagyis megszámlálhatóak). Valóban, tekintsük azt a felsorolást, amelyben a szavak a hosszuk szerint követik egymást, és két egyforma hosszú szó közül pedig az van előbb, amelyik az alfabetikus rendezés szerint megelőzi a másikat. Ily módon a $\{0,1\}^*$ halmaz elemeinek egy felsorolása a következőképpen alakul: $w_1 = \varepsilon$, $w_2 = 0$, $w_3 = 1$, $w_4 = 00$, $w_5 = 01$ és így tovább. Ebben a fejezetben tehát a w_i szóval a $\{0,1\}^*$ i. elemét jelöljük.

Legyen továbbá M egy $\{0,1\}$ inputábécé feletti Turing-gép. Van olyan $k > 0$ szám, hogy Q-t felírhatjuk $Q = \{p_1, ...p_k\}$ alakban, ahol $p_1 = q_0$, $p_{k-1} = q_i$, $p_k = q_n$. Továbbá, van olyan $m > 0$ szám, hogy Γ-t felírhatjuk $\Gamma = \{X_1, ...X_m\}$ alakban, ahol $X_1 = 0$, $X_2 = 1$, $X_3 = \sqcup$, és $X_4, ...X_m$ az M további szalagszimbólumai. Nevezzük végül az L, R, S szimbólumokat (amelyek irányokat jelölnek) rendre D_1, D_2 és D_3-nak. Ezek után M egy $\delta(p_i, X_j) = (p_r, X_s, D_t)$ $(0 \leq i, r \leq k$, $1 \leq j, s \leq m$ és $1 \leq t \leq 3)$ átmenete elkódolható a $0^i10^j10^r10^s10^t$ szóval. Mivel minden 0-s blokk hossza legalább 1, az átmenetet kódoló szóban nem szerepel az 11 részszó. Tehát az M összes átmenetét kódoló szavakat összefűzhetjük egy olyan szóvá, melyben az átmeneteket az 11 részszó választja el egymástól. Az így kapott szó pedig magát M-et kódolja.

A továbbiakban M_i-vel jelöljük azt a Turing-gépet, amelyet a w_i szó kódol $(i \geq 1)$. Amennyiben w_i nem a fent leírt kódolása egy Turing-gépnek, akkor tekintsük M_i-t olyannak, ami minden input esetén azonnal a q_n állapotba megy, azaz $L(M_i) = \emptyset$.

A későbbiekben szükségünk lesz arra, hogy elkódoljunk egy (M, w) Turing-gép és bemenet párost egy $\{0,1\}$ feletti szóban. Mivel a Turing-gépek kódolása nem tartalmazhat 111-et, ezért (M, w) kódja a következő: M kódja után írunk 111-et, majd utána w-t.

3.0.2 Egy nem rekurzívan felsorolható nyelv

Az $L_{átló}$ nyelv: Az $L_{átló}$ nyelv azon $\{0,1\}$ feletti Turing-gépek bináris kódjait tartalmazza, melyek nem fogadják el önmaguk kódját, mint bemenő szót, azaz $L_{átló} = \{w_i \mid i \geq 1, w_i \notin L(M_i)\}$

Tétel: $L_{átló} \notin RE$.

3.0.3 Egy rekurzívan felsorolható, de nem eldönthető nyelv

Az L_u nyelv: Tekintsük azon (M, w) párok halmazát (egy megfelelő bináris szóban elkódolva), ahol M egy $\{0,1\}$ bemenő ábécé feletti Turing-gép, w pedig egy $\{0,1\}$ feletti szó úgy, hogy $w \in L(M)$, azaz M elfogadja w-t. Ezt a nyelvet jelöljük L_u-val. $L_u = \{\langle w_i, w_j \rangle \mid i, j \geq 1, w_j \in L(M_i)\}$

Tétel: $L_u \in RE$.

Tétel: $L_u \notin R$.

3.0.4 További tételek

1. Legyen L egy nyelv. Ha $L, \bar{L} \in RE$, akkor $L \in R$. Következmény: a rekurzívan felsorolható nyelvek nem zártak a komplementerképzésre.

2. Ha $L \in R$, akkor $\bar{L} \in R$, azaz a rekurzív nyelvek zártak a komplementerképzésre.

3.0.5 További eldönthetetlen problémák

Kiszámítható függvény: Legyen Σ és Δ két ábécé és f Σ^*ból Δ^*-ba képző függvény. Azt mondjuk, hogy f kiszámítható, ha van olyan M Turing-gép, hogy M-et egy $w \in \Sigma^*$ szóval a bemenetén elindítva, M úgy áll meg, hogy a szalagján a $f(w) \in \Delta^*$ szó van.

Eldöntési problémák visszavezetése: Legyen $L_1 \subseteq \Sigma^*$ és $L_2 \subseteq \Delta^*$ két eldöntési probléma. L_1 visszavezethető L_2-re ($L_1 \leq L_2$), ha van olyan $f : \Sigma^* \to \Delta^*$ kiszámítható függvény, hogy minden $w \in \Sigma^*$ szóra $w \in L_1$ pontosan akkor teljesül, ha $f(w) \in L_2$ is teljesül.

Tétel: Legyen $L_1 \subseteq \Sigma^*$ és $L_2 \subseteq \Delta^*$ két eldöntési probléma és tegyük fel, hogy L_1 visszavezethető L_2-re. Ekkor igazak a következő állítások:

1. Ha L_1 eldönthetetlen, akkor L_2 is.

2. Ha $L_1 \notin RE$, akkor $L_2 \notin RE$.

A megállási probléma: Legyen $L_h = \{\langle M, w\rangle \mid M\ \textit{megáll a}\ w\ \textit{bemeneten}\}$, azaz L_h azon $\langle M, w\rangle$ Turing-gép és bemenet párosokat tartalmazza elkódolva, melyekre M megáll a w bemeneten. L_h eldönthetetlen (L_u visszavezethető L_h-ra), viszont $L_h \in RE$.

Az $L_{üres}$ probléma: Legyen $L_{üres} = \{\langle M\rangle \mid L(M) = \emptyset\}$. $L_{üres}$ eldönthetetlen (L_u visszavezethető $L_{üres}$-re), valamint $L_{üres} \notin RE$.

Rekurzívan felsorolható nyelvek (nem triviális) tulajdonsága: Ha $\mathcal{P}$ a rekurzívan felsorolható nyelvek egy halmaza, akkor $\mathcal{P}$ a rekurzívan felsorolható nyelvek egy tulajdonsága. Ha $\mathcal{P} \neq \emptyset$ és $\mathcal{P} \neq RE$, akkor $\mathcal{P}$ nem triviális tulajdonsága a rekurzívan felsorolható nyelveknek.

Rice tétele: Adott $\mathcal{P}$ tulajdonságra jelöljük $L_\mathcal{P}$-vel azon Turing-gépek kódjainak halmazát, amelyek $\mathcal{P}$-beli nyelvet ismernek fel. Ha $\mathcal{P}$ a rekurzívan felsorolható nyelvek egy nem triviális tulajdonsága, akkor $L_\mathcal{P}$ eldönthetetlen.

Post Megfelelkezési Probléma (röviden PMP): A PMP problémát a következőképpen definiáljuk. Legyen Σ egy legalább két betűt tartalmazó ábécé és legyen $D = \left\{[\frac{u_1}{v_1}], ..., [\frac{u_n}{v_n}]\right\}$ egy dominóhalmaz, melyben $n \geq 1$ és $u_1, ..., u_n, v_1, ..., v_n \in \Sigma^+$. A kérdés az, hogy van-e egy olyan $1 \leq i_1, ..., i_m \leq m$ ($m \geq 1$) indexsorozat, melyre teljesül, hogy a $[\frac{u_{i_1}}{v_{i_1}}], ..., [\frac{u_{i_m}}{v_{i_m}}]$ dominókat egymás mellé írva alul és felül ugyanaz a szó adódik, azaz $u_{i_1}...u_{i_m} = v_{i_1}...v_{i_m}$. Ebben az esetben a fenti dominósorozatot a D egy megoldásának nevezzük.

Formális nyelvként a következőképpen definiálhatjuk a PMP-t: PMP $= \{\langle D\rangle \mid D - nek\ van\ megoldása\}$. PMP eldönthetetlen.

4 Bonyolultságelmélet

A bonyolultságelmélet célja a megoldható (és ezen belül az eldönthető) problémák osztályozása a megoldáshoz szükséges erőforrások (jellemzően az idő és a tár) mennyisége szerint.

4.0.1 Időbonyolultsági fogalmak

TIME: Legyen $f : \mathbb{N} \to \mathbb{N}$ függvény. $\mathbf{TIME}(f(n)) = \{L \mid L\ eldönthetõ\ \mathcal{O}(f(n))\ idõigényũ\ Turing - géppel\}$

$\mathbf{P} = \bigcup_{k \geq 1} \mathbf{TIME}(n^k)$. Tehát **P** azon nyelveket tartalmazza, melyek eldönthetőek polinom időkorlátos determinisztikus Turing-géppel. Ilyen például a jól ismert ELÉRHETŐSÉG probléma, melynek bemenete egy G gráf és annak két kitüntetett csúcsa (s és t). A kérdés az, hogy van-e a G-ben út s-ből t-be. Ha az ELÉRHETŐSÉG problémára nyelvként tekintünk, akkor írhatjuk azt, hogy

$$\text{ELÉRHETŐSÉG} = \{\langle G, s, t\rangle \mid G - ben\ van\ út\ s - bõl\ t - be\}.$$

Könnyen megadható az ELÉRHETŐSÉG problémáját polinom időben eldöntő determinisztikus Turing-gép, tehát ELÉRHETŐSÉG $\in \mathbf{P}$.

NTIME: Legyen $f : \mathbb{N} \to \mathbb{N}$ függvény.
$\mathbf{NTIME}(f(n)) = \{L \mid L\ eldönthetõ\ \mathcal{O}(f(n))\ idõigényũ\ nemdeterminisztikus\ Turing - géppel\}$

$\mathbf{NP} = \bigcup_{k \geq 1} \mathbf{NTIME}(n^k)$. Az **NP**-beli problémák rendelkeznek egy közös tulajdonsággal az alábbi értelemben. Ha tekintjük egy **NP**-beli probléma egy példányát és egy lehetséges "bizonyítékot" arra nézve, hogy ez a példány "igen" példánya az adott problémának, akkor ezen bizonyíték helyességének leellenőrzése polinom időben elvégezhető. Ennek megfelelően egy **NP**-beli problémát eldöntő nemdeterminisztikus Turing-gép általában úgy működik, hogy "megsejti" a probléma bemenetének egy lehetséges megoldását, és polinom időben leellenőrzi, hogy a megoldás helyes-e.

Tekintsük a SAT problémát, amit a következőképpen definiálunk. Adott egy ϕ ítéletlogikai KNF. A kérdés az, hogy kielégíthető-e. Annak a bizonyítéka, hogy a ϕ kielégíthető, egy olyan változó-hozzárendelés, ami mellett kiértékelve a ϕ-t igaz értéket kapunk. Egy tetszőleges változó-hozzárendelés tehát a ϕ kielégíthetőségének egy lehetséges bizonyítéka .Annak leellenőrzése pedig, hogy ez a hozzárendelés tényleg igazzá teszi-e ϕ-t, polinom időben elvégezhető. A SAT **NP**-beli probléma.

Az a definíciókból következik, hogy fennáll a $\mathbf{P} \subseteq \mathbf{NP}$ tartalmazás.

4.0.2 NP-teljes problémák

Polinom időben kiszámítható függvény: Legyen Σ és Δ két ábécé és f Σ^*ból Δ^*-ba képző függvény. Azt mondjuk, hogy f polinom időben kiszámítható, ha kiszámítható egy polinom időigényű Turing-géppel.

Eldöntési problémák polinom idejű visszavezetése: Legyen $L_1 \subseteq \Sigma^*$ és $L_2 \subseteq \Delta^*$ két eldöntési probléma. L_1 polinom időben visszavezethető L_2-re ($L_1 \leq_p L_2$), ha $L_1 \leq L_2$ és a visszavezetésben használt f függvény polinom időben kiszámítható.

Tétel: Legyen L_1 és L_2 két probléma úgy, hogy $L_1 \leq_p L_2$. Ha L_2

1. **P**-beli, akkor L_1 is **P**-beli.
2. **NP**-beli, akkor L_1 is **NP**-beli.

NP-teljes probléma: Legyen L egy probléma. Azt mondjuk, hogy L **NP**-teljes, ha

1. **NP**-beli, és
2. minden további **NP**-beli probléma polinom időben visszavezethető L-re.

Tétel: Legyen L egy **NP**-teljes probléma. Ha $L \in \mathbf{P}$, akkor $\mathbf{P} = \mathbf{NP}$.

Megjegyzés: Jelenleg **NEM** tudunk **P**-beli **NP**-teljes problémáról!!!

Tétel: Legyen L_1 egy **NP**-teljes, L_2 pedig **NP**-beli probléma. Ha $L_1 \leq_p L_2$, akkor L_2 is **NP**-teljes.

Cooke tétele: Sat **NP**-teljes.

Legyen $k \geq 1$. kSat $= \{\langle\phi\rangle \mid \phi \text{ } \textit{minden tagjában k literál van.}\}$

Tétel: 3Sat **NP**-teljes, ugyanis Sat $\leq_p$ 3Sat.

Teljes részgráf $= \{\langle G, k\rangle \mid G \text{ } \textit{véges gráf}, k \geq 1, \ G - \textit{nek} \ \exists \ k \text{ } \textit{csúcsú részgráfja}\}$. Tehát a Teljes részgráf azon G és k párokat tartalmazza, megfelelő ábécé feletti szavakban elkódolva, melyekre igaz, hogy G-ben van k csúcsú teljes részgráf, azaz olyan részgráf, melyben bármely két csúcs között van él.

Teljes részgráf$= \{\langle G, k\rangle \mid G \text{ } \textit{véges gráf}, k \geq 1, \ G - \textit{nek} \ \exists \ k \text{ } \textit{csúcsú részgráfja}\}$. Tehát a Teljes részgráf azon G és k párokat tartalmazza, megfelelő ábécé feletti szavakban elkódolva, melyekre igaz, hogy G-ben van k csúcsú teljes részgráf, azaz olyan részgráf, melyben bármely két csúcs között van él.

Független csúcshalmaz $= \{\langle G, k\rangle \mid G \text{ } \textit{véges gráf}, k \geq 1, \ G - \textit{nek} \ \exists \ k \text{ } \textit{elemũ független csúcshalmaza}\}$. Vagyis a Független csúcshalmaz azon G és k párokat tartalmazza, melyekre igaz, hogy G-ben van k olyan csúcs, melyek közül egyik sincs összekötve a másikkal.

Csúcslefedés =
$$\left\{\langle G, k\rangle \mid \begin{array}{l} G \text{ } \textit{véges gráf}, k \geq 1, \ G - \textit{nek van olyan k elemũ csúcshalmaza,} \\ \textit{mely tartalmazza G minden élének legalább } 1 \text{ } \textit{végpontját.} \end{array}\right\}.$$

Teljes részgráf, Független csúcshalmaz és Csúcslefedés **NP**-teljesek (Teljes részgráf $\leq_p$ Független csúcshalmaz $\leq_p$ Csúcslefedés).

Utazóügynök =
$$\left\{\langle G, k\rangle \mid \begin{array}{l} G \text{ } \textit{véges irányítatlan gráf, az éleken egy} - \textit{egy pozitív egész súllyal és} \\ \textit{van } G - \textit{ben legfeljebb k összsúlyú Hamilton kör} \end{array}\right\}.$$

Tétel: Az Utazóügynök probléma **NP**-teljes.

4.0.3 Tárbonyolultság

A tárbonyolultságot egy speciális, úgynevezett offline Turing-gépen vizsgáljuk.

Off-line Turing-gép: Offline Turing-gépnek nevezünk egy olyan többszalagos Turing-gépet, mely a bemenetet tartalmazó szalagot csak olvashatja, a többi, ún. munkaszalagokra pedig írhat is. Az offline Turing-gép tárigényébe csak a munkaszalagokon felhasznált terület számít be.

A továbbiakban Turing-gép alatt minidig offline Turing-gépet értünk. Most definiáljuk a tárbonyolultsággal kapcsolatos nyelvosztályokat.

SPACE(f(n)) $= \{L \mid L \text{ } \textit{eldönthetõ } \mathcal{O}(f(n)) \text{ } \textit{tárigényũ determinisztikus Turing} - \textit{géppel}\}$

NSPACE(f(n)) $= \{L \mid L \text{ } \textit{eldönthetõ } \mathcal{O}(f(n)) \text{ } \textit{tárigényũ nemdeterminisztikus Turing} - \textit{géppel}\}$

$\mathbf{PSPACE} = \bigcup_{k>0} \mathbf{SPACE}(n^k)$

$\mathbf{NPSPACE} = \bigcup_{k>0} \mathbf{NSPACE}(n^k)$

$\mathbf{L} = \mathbf{SPACE}(\log_2 n)$

$\mathbf{NL} = \mathbf{NSPACE}(\log_2 n)$

Savitch tétele: Ha $f(n) \geq \log n$, akkor $\mathbf{NSPACE}(f(n)) \subseteq \mathbf{SPACE}(f^2(n))$.