ALGORITMUSOK ÉS BONYOLULTSÁGELMÉLET

Matematika MSc hallgatók számára

11. Előadás

Előadó: Hajnal Péter
Jegyzetelő: Vörös Anett

2012. Április 25.

Először elevenítsünk fel néhány definíciót. Kezdjük a $\Pi_i\mathcal{P}, \Sigma_i\mathcal{P}$ nyelvosztályokkal ($i \in \mathbb{N}$).

Definíció. $L \in \Sigma_i\mathcal{P} \overset{\text{def}}{\iff} \exists T$ $|\omega|$-ban polinomiális tanúszalagos Turing-gép úgy, hogy a plusz tanúszalag tartalma $\exists\tau_1\forall\tau_2 \dots Q\tau_i$, ahol Q egy kvantor (a kvantorok szerepe csak elválasztó jel, de a későbbiekből látszik jelentésük). Azaz i alternáló kvantorjel szerepel $\exists$-kel kezdődően. Továbbá

$$\omega \in L \iff \exists\tau_1\forall\tau_2 \dots Q\tau_i \; T(\omega, \tau_1, \tau_2, \dots, \tau_i) = \text{ELFOGAD}.$$

Definíció. $L \in \Sigma_i\mathcal{P} \overset{\text{def}}{\iff} \exists T$ $|\omega|$-ban polinomiális tamúszalagos Turing-gép úgy, hogy a plusz tanúszalag tartalma: $\forall\tau_1\exists\tau_2 \dots Q\tau_i$, és

$$\omega \in L \iff \forall\tau_1\exists\tau_2 \dots Q\tau_i \; T(w, \tau_1, \tau_2, \dots, \tau_i) = \text{ELFOGAD}.$$

Megjegyzés. A fenti elsőre szokatlan jelölés memorizálásához segítséget adunk. Fontos, hogy hányszor alternálnak a kvantorok. Ezt az osztály jelölésében szereplő index adja. Azaz $\Sigma_2\mathcal{P}$-ben egy váltás van két kvantor között, $\Pi_{10}\mathcal{P}$-ben 10 kvantor szerepel váltakozva. A kezdő kvantor is fontos. A $\exists$ és $\forall$ kvantorok szimmetriája különböző: $\exists$ egy vízszintes tengelyre, míg $\forall$ egy függőleges tengelyre szimmetrikus. Hasonló a helyzet a Σ és Π görög betűkkel. A szimmetriatengely irányának azonosságából kiolvasható a jelölés. Azaz $\Pi_{2012}\mathcal{P}$ definiációjában a kvantorokkal való leírás egy $\forall$ kvantorral kezdődik.

Megjegyezzük, hogy a kvantor a tanúszalag egy intervallumára vonatkozik. Ennek hossza előre nem látható. Ha a kvantorokat a tanúszalag Γ ábécéjére vonatkoztatjuk, akkor $\exists\tau$ (ahol $\tau \in \Gamma^\ell$) egy ℓ hosszú kvantorsorozat, ahol minden kvantor egy karakterre vonatkozó $\exists$ kvantor. Két példával világítjuk meg milyen kvantorsorozat szerepel egy fent leírt nyelvosztály definíciójában

$$\Sigma_2\mathcal{P} : \exists_\Gamma\exists_\Gamma \dots \exists_\Gamma\forall_\Gamma\forall_\Gamma \dots \forall_\Gamma,$$

$$\Pi_2\mathcal{P} : \forall_\Gamma\forall_\Gamma \dots \forall_\Gamma\exists_\Gamma\exists_\Gamma \dots \exists_\Gamma.$$

Definíció. $L \in \mathcal{P}^{\text{nem}-\text{uniform}} \overset{\text{def}}{\iff} \exists\{C_n\}$ hálózatsorozat, amelyre a következők teljesülnek:

(i) C_n ($n \in \mathbb{N}$) polinomiális méretű.

(ii) $C_n(\lceil\omega\rceil) = \begin{cases} 1, & \text{ha } \omega \in L, \\ 0, & \text{ha } \omega \notin L. \end{cases}$

Definíció. $SAT \overset{\text{def}}{=} \{\lceil\varphi\rceil : \varphi \text{ kielégíthető CNF}\}$.

1. SAT és hálózatok (folytatás): Karp—Lipton—Sipser-tétel

1. Tétel (Karp—Lipton—Sipser-tétel). *Ha* $SAT \in \mathcal{P}^{\text{nem}-\text{uniform}}$, *akkor* $\Pi_2\mathcal{P} \subseteq \Sigma_2\mathcal{P}$.

Bizonyítás. A tétel bizonyításához legyen $L \in \Pi_2\mathcal{P}$ tetszőleges nyelv. Meg fogjuk mutatni, hogy amennyiben $SAT \in \mathcal{P}^{\text{nem}-\text{uniform}}$ teljesül, akkor $L \in \Sigma_2\mathcal{P}$ is. $\Pi_2\mathcal{P}$ definíciója szerint $L \in \Pi_2\mathcal{P}$ azt jelenti, hogy létezik olyan polinomiális tanúszalagos Turing-gép, hogy $\omega \in L$ akkor és csak akkor $\forall x \exists y : T(\omega, x, y)$ futása ELFOGAD állapotba vezet.

Az első lépésben csak a belső részt vizsgáljuk (azaz hagyjuk el úgy a $\forall$ kvantort, mintha ott se lenne):

$$\exists y T(\underbrace{\omega, x}_{\omega^+}, y).$$

Így leírtunk ω^+-ok egy halmazát, egy $\widetilde{L}$ nyelvet. A leírásból nyilvánvaló, hogy $\widetilde{L} \in \mathcal{NP}(= \Sigma_1\mathcal{P})$. A Cook—Levin-tétel szerint SAT $\mathcal{NP}$-teljes, így L nyelv redukálható SAT-ra: $\widetilde{L} \preceq SAT$. Ezek alapján létezik egy polinomiális R redukciós algoritmus úgy, hogy

$$\omega^+ = (\omega, x) \mapsto R(\omega^+),$$

ahol $R(\omega^+)$ egy CNF kód, amelyre teljesül, hogy

$$\omega^+ \in \widetilde{L} \Longleftrightarrow R(\omega^+) \in SAT.$$

A második lépésben azt, hogy $R(\omega^+)$ a SAT-hoz tartozik-e átírjuk a tétel feltétele alapján. Tudjuk, hogy van egy polinomiális méretű $\{C_n\}$ hálózatsorozat, ami a SAT-ot oldja meg

$$\omega^+ \in \widetilde{L} \Longleftrightarrow C_n(\lceil R(\omega^+) \rceil) = 1,$$

ahol $\lceil R(\omega^+) \rceil$ a redukált ω^+ bitekkel való kódolása, n pedig $\lceil R(\omega^+) \rceil$ hossza. Azaz az ω^+ jelölést kibontva

$$(\omega, x) \in \widetilde{L} \Longleftrightarrow C_n(\lceil R(\omega, x) \rceil) = 1.$$

Visszatérve a teljes eredeti L-et leíró formulához kapjuk, hogy

$$\omega \in L \Longleftrightarrow \forall x \; C_n(\lceil R(\omega, x) \rceil) = 1.$$

Egy kvantorunk maradt, de az utána következő rész nem egy polinomiális számolással kiszámolható formula: a C_n hálózatsorozat nem uniform. Mivel a C_n hálózatsorozat különböző értékekre más-más eredményt ad, vagyis hektikusan viselkedhet, így az lehet az ötletünk, hogy tippeljük meg C_n-et:

$$\exists C_n \forall x : C_n(\lceil R(\omega, x) \rceil) = 1.$$

Sajnos az ötlet nem működik. Ha C_n tippünk valóban SAT-ot számolja ki, akkor minden rendben. Ha tippünk rossz, akkor is csak akkor van baj, ha az elfogadó 1 bitet számolja ki, olyan kódra, ami nem kielégíthető formulát kódol. Azaz a hibák közül csak az a veszélyes, ami nem kielégíthető formuláról mondja azt, hogy kielégíthető. Ezen azonban a következő lemma segít:

2. Lemma. *Létezik olyan polinomiális TG, hogy ami egy hálózatból (igazából hálózat kódjából) egy másik hálózatot számol ki:* $\{C_n\} \mapsto \{\widetilde{C}_n\}$ *úgy, hogy*

(i) ha C_n *SAT-ot számolta ki, akkor* $\widetilde{C}_n$ *is SAT-ot fogja eldönteni,*

(ii) ha C_n *nem SAT-ot számolta ki, akkor is* $\widetilde{C}_n(\lceil\varphi\rceil) = 1$ *esetén biztos, hogy* φ *egy kielégíthető CNF kódja.*

A lemma ismeretében egyszerű a bizonyítás befejezése. A Turing-gép úgy módosítja a megtippelt C_n-et, ahogy a lemma írja le. Az új $\widetilde{C}_n$ hibázhat SAT eldöntésében, de csak az egyik irányban. A lemmából következik, hogy

$$L = \{\omega : \exists C_n \forall x \widetilde{C}_n(\lceil R(\omega, x)\rceil) = 1\},$$

polinomiális, amiből azonnal következik, hogy $L \in \Sigma_2\mathcal{P}$, tehát $\Pi_2\mathcal{P} \subseteq \Sigma_2\mathcal{P}$.

A lemmánk bizonyítása teljessé teszi a tétel igazolását.

A lemma bizonyítása: A tétel bizonyításában előforduló C_n-ekkel az lehet a probléma, hogy 1-et adnak, pedig $R(\omega, x)$ nem kielégíthető. Meg fogunk adni $\widetilde{C}_n$ hálózatokat a C_n sorozaton alapulva. Az $\langle\varphi(x_1, \ldots, x_k)\rangle$ inputra megnézzük, mit ad C_n. Ha 0-t, akkor „nem aggódunk": $\widetilde{C_n}$-et úgy definiáljuk, hogy 0-t számoljon ki. Ha C_n 1-et ad (ami a rossz irányú tévedést is megengedi), akkor C_n kap két új inputot:

$$\varphi(x_1, \ldots, x_{k-1}, 0) \quad \text{és} \quad \varphi(x_1, \ldots, x_{k-1}, 1).$$

Ha mindkettőn 0-t ad, akkor C_n rossz, nem aggódunk. Ha valamelyikre 1-et „dob vissza", akkor tovább építjük $\widetilde{C}_n$-et. Legyen ε_k az a bit, amelyikre rögzítve x_k-t a C_n hálózat 1-et számolt ki. Vesszük a következő inputot:

$$\varphi(x_1, \ldots, x_{k-2}, 0, \varepsilon_k), \quad \varphi(x_1, \ldots, x_{k-2}, 1, \varepsilon_k).$$

Ha C_n mindkét inputra 0-t ad, akkor azt mondjuk, hogy C_n „lebukott", nem aggódunk. Ha valamelyik inputra 1-et kapunk, akkor az előzőhöz hasonlóan definiáljuk ε_{k-1}-et, és haladunk tovább.

Ha eljutunk ε_1 definíciójáig, akkor végül kiszámoljuk $\varphi(\varepsilon_1, \ldots, \varepsilon_k)$-t. Ha ezen az input-formula 1-et számol ki, akkor $\widetilde{C}_n$ is 1-et ad, különben $\widetilde{C}_n$ értéke 0 lesz (annak ellenére, hogy az összes kérdésünkre a C_n tippünk kielégíthetőséget mondott).

Ha C_n kiszámolja SAT-ot, akkor az algoritmus azonosít egy kielégítő értékadást és jól fog lefutni. Ha C_n rossz tipp volt, de $\widetilde{C}_n$ az 1 eredményt adja, akkor is kiszámolt egy kielégítő értékadást. Biztosak lehetünk, hogy kielégíthető formulánk van. ■

Megjegyzés. A tétel konkluziója úgy is megfogalmazható, hogy felcseréltünk két kvantort. Két kvantor felcserélhetősége azonban a polinomiális hierarchia összeomlását jelentené. Ezt egy példával világítjuk meg

$$\ldots \exists\tau \forall\tau' \exists\tau'' \varphi \equiv \ldots \exists\tau(\forall\tau' \exists\tau'' \varphi) \equiv \ldots \exists\tau(\exists\lambda \forall\lambda' \psi) \equiv \ldots (\exists\tau\exists\lambda)\forall\lambda'\psi.$$

Azaz

$$\Sigma_i\mathcal{P} = \Pi_i\mathcal{P} = \Sigma_{i-1}\mathcal{P} = \Pi_{i-1}\mathcal{P} = \ldots = \Sigma_3\mathcal{P} = \Pi_3\mathcal{P} \subset \Sigma_2\mathcal{P},$$

így

$$\mathcal{PH} = \Sigma_2\mathcal{P},$$

amit úgy is szoktak leírni, hogy „a hierarchia a második szintre esik össze".

A továbbiakban felidézzük a múlt óra anyagából a Lipton-Sipser-tétel egy alternatív kimondását.

3. Tétel (Lipton-Sipser). *$SAT \preceq_{\mathcal{P}}^{\text{Turing}} S$, ahol S ritka nyelv (azaz alkalmas polinomra $|S \cap \Sigma^n| \leq p(n)$ minden n természetes számra). Ekkor $\Pi_2\mathcal{P} \subseteq \Sigma_2\mathcal{P}$.*

2. SAT és hálózatok: Mahaney-tétel

A Karp—Liptin—Sipser-tétel alternatív megfogalmazása vezet el Mahaney tételéhez, ami az előbbihez képest gyöngébb redukciót tételez fel SAT-ra (erősebb a feltétel). De a következmény is erősebb.

4. Tétel (Mahaney-tétel). *$SAT \preceq_{\mathcal{P}}^{\text{Karp}} S$, ahol S ritka nyelv (azaz alkalmas polinomra $|S \cap \Sigma^n| \leq p(n)$ minden n természetes számra). Ekkor $\mathcal{P} = \mathcal{NP}$.*

A bizonyítás előtt átfogalmazzuk a tételt.

Definíció. $\Sigma^{\leq n} \stackrel{\text{def}}{=} \Sigma^0 \cup \ldots \cup \Sigma^n = \{\omega = a_1 \ldots a_e | a_i \in \Sigma, \ell \leq n\}$, (ahol Σ^0 csak az üres szót tartalmazó egyelemű halmaz).

Feltesszük, hogy Σ rendezett (karaktereink között van egy rendezés, ábécé sorrend): Ekkor Σ^n-nek is van egy természetes rendezése: a lexikografikus sorrend. Ezt egy kissé szokatlan módon terjesztjük ki a $\Sigma^{\leq n}$ halmazre:

Legyen $\omega_1 = a_1 \ldots a_k$, $\omega_2 = b_1 \ldots b_\ell$ két tetszőleges eleme $\Sigma^{\leq n}$-nek. Legyen $m = \min\{k, \ell\}$. Ekkor $\omega_1 \prec \omega_2$ pontosan akkor teljesül, ha a következő két eset valamelyike fennáll:

(1) $a_1 \ldots a_m$ szigorúan előbb van a lexikografikus sorozatban, mint $b_1 \ldots b_m$,

(2) ω_1 az ω_2 szó kiterjesztése (ekkor $a_1 \ldots a_m = b_1 \ldots b_m$, (1) nem döntött).

Példa. Legyen $\Sigma = \{0, 1\}$, $n = 3$

Ekkor $\Sigma^{\leq 3} = \{\varepsilon, 0, 1, 00, 01, 10, 11, 000, 001, 010, 011, 100, 101, 111, 110\}$.

$\Sigma^{\leq 3}$ rendezésében az első szó 000, az utolsó ε. A teljes rendezés (feltüntettük, hogy a soorendet melyik szabály határozza meg):

$$000 \underset{(1)}{\prec} 001 \underset{(2)}{\prec} 00 \underset{(1)}{\prec} 010 \underset{(1)}{\prec} 01 \underset{(2)}{\prec} 0 \underset{(1)}{\prec} 100 \underset{(1)}{\prec}$$

$$\underset{(1)}{\prec} 101 < \underset{(2)}{\prec} 10 \underset{(1)}{\prec} 110 \underset{(1)}{\prec} 111 \underset{(2)}{\prec} 11 \underset{(2)}{\prec} 1 \underset{(2)}{\prec} \varepsilon.$$

Bevezetjük SAT egy nehezítését:

Definíció.

$SAT^* = \{\langle \varphi(x_1, \ldots, x_n), t \rangle : t \in \{0,1\}^{\leq n}, \varphi$ CNF, φ-nek létezik

$k \in \{0,1\}^n$ kielégítő kiértékelése, amelyre $k \preceq t\}$.

Ez valóban nehezítés, hiszen $\varphi \in SAT$ akkor és csak akkor, ha $(\varphi, \varepsilon) \in SAT^*$. Vegyük észre, hogy $SAT^* \in \mathcal{NP}$, így SAT^* $\mathcal{NP}$-teljes. Ezek után újrafogalmazhatjuk Mahaney tételét:

5. Tétel (Mahaney). *$SAT^* \preceq_{\mathcal{P}}^{\text{Karp}} S$, ahol S ritka. Ekkor $SAT \in \mathcal{P}$, azaz $\Rightarrow \mathcal{P} = \mathcal{NP}$.*

Bizonyítás. A tétel feltétele: $\text{SAT}^* \preceq S$, azaz egy R redukciós algoritmus. A tétel állítása egy SAT-ot eldöntő polinomiális algoritmus, ezt meg fogjuk adni.

A SAT-ot megoldó algoritmus (adott $\varphi(x_1, \ldots, x_n)$ CNF esetén) egy $L_0, L_1, \ldots, L_n$ listasorozatot számol ki polinom időben, amelyre a következők teljesüljenek:

(i) Az L_i lista elemei i hosszú 0-1 sorozatok. Azaz $L_i \subseteq \{0,1\}^i \quad i \in \{0, 1, \ldots, n\}$

(ii) L_i hossza polinomiális. Azaz van olyan q polinom, hogy $|L_i| < q(n)$.

(iii) $\forall i \quad \exists \ell_i \in L_i$: ha φ kielégíthető, akkor legyen olyan kielégítése is, amelyre $\preceq \ell_i$.

Az algoritmus során meg fogjuk adni hogyan generáljuk a $L_0, L_1, L_2, \ldots, L_n$ sorozatot.

Kérdések:

1. Mi lesz L_0?
2. L_i ismeretében hogyan számolható ki L_{i+1}?

Válaszok:

1. $L_0 \stackrel{\text{def}}{=} \{\varepsilon\}$.
2. L_{i+1} leírását több lépésben adjuk meg

$$L_i \to L_i^+ \stackrel{\text{def}}{=} \{L_i \text{ elemeinek kiterjesztései egy bittel}\},$$

vagyis L_i egy eleméhez hozzáírjuk a 0-t vagy az 1-et az összes lehetséges módon. Így $|L_i^+| = 2|L_i|$. Ha csak annyiben maradnánk $|L_i|$ exponenciálisan nőne. valamikor L_i^+-t csökkentenünk kell.

Először L_i^+-t úgy csökkentjük, hogy mindegyikre alkalmazzuk az R redukciót. Majd azon elemekből, amiket R ugyanarra a képre képez, csak egyet tartunk meg, mégpedig a lexikongrafikusan elsőt. Legyen L_i^+ az így kapott lista.

A 3. ígérethez elég lenne csak azokat az elemeket megtartani, amik S-beli elemre redukálódnak. Ezek számát könnyen becsülhetjük: $R(\varphi, t)$ hosszát polinommal becsülhetjük (R polinomiális algoritmus) és az ilyen hosszú S-beli lehetőségek számát is becsülhetjük S ritkasága miatt. Legyen $\overline{p}$ az a polinom becslés, amit adhatunk azokra a listabeli elemekre, amik S-be képződnek. Sajnos S „bonyolult", a redukciókat elvégezve nem tudjuk azonsítani

Tekintsük L_i^{+-} rendezett elemeit:

$$\begin{array}{ccccccccc} m_1 & \preceq & m_2 & \preceq & m_3 & \preceq & \ldots & \preceq & m_s \\ \downarrow R & & \downarrow R & & \downarrow R & & \ldots & & \downarrow R \\ \rho_1 & & \rho_2 & & \rho_3 & & \ldots & & \rho_3 \end{array}$$

A SAT^* nyelvet úgy definiáltuk, hogy a ρ_i-k egy ideig S-en kívül lehetnek, de az első S-be eső ρ_i után az összes S-beli lesz.

A végső ritkítás az lesz, hogy a fenti rendezett sorban L_i^{+-} utolsó $\overline{p}(n)$ elemét megtartjuk. Az így kapott lista legyen L_i^{+--}.

Legyen $L_{i+1} = L_i^{+--}$.

Ezzel látjuk, hogy a listáinkat hogyan generáljuk. A megígért (i)-(iii) tulajdonságok nyilvánvalók.

Az algoritmus befejezése: A listák generálása után L_n elemeiről ellenőrizzük, hogy kielégítik-e φ-t. Ha valamelyik igen, akkor elfogadjuk az inputot (és ebben az esetben válaszunk nyilvánvalóan korrekt). Ha egyik sem elégíti ki, akkor az inputot elvetjük.

A bizonyítás vége annak igazolása, hogy algoritmusunk a SAT-ot oldja meg. Azaz, ha a φ input olyan, hogy az L_n listán nincs kielégítő kiértékelés, akkor φ nem is elégíthető ki.

Indirekten bizonyítunk. Tegyük fel, hogy L_n-ben nincs kielégítő értékelés, de φ mégis kielégíthető. Legyen k a lexikografikus sorrend szerinti első kielégítő értékadás. Legyen k_i a k bitsorozat első i bitje. Belátjuk, hogy $k_i \in L_i$. Speciálisan $k = k_n \in L_n$, ami ellentmond annak, hogy L_n elemeit végignézve nem találtunk kielégítő értékadást.

A $k_i \in L_i$ állítást indukcióval igazoljuk. $k_0 = \varepsilon \in L_0$. Tegyük fel, akkor $k_i \in L_i$. Nyilván $k_{i+1} \in L_i^+$. Azt kell igazolnunk, hogy k_{i+1} az összes ritkítást „túléli".

Azt kell észrevenni, hogy k_i a Σ^i lexikografikusan első olyan eleme, amely SAT^* eleme, azaz az R redukciós algoritmus az S halmazba képzi. Az első rikításnál az ugyanazon elemre képződő szavak közül a lexikografikusan elsőt hagytuk meg, k_{i+1} túlélő. A második ritkításnál az S halmazba képződő elemek garantáltan megmaradtak, k_{i+1} ismét túlélő. ■