

9. Előadás

Előadó: Hajnal Péter

Jegyzetelő: Hajnal Péter

2013. április 11.

1. SAT és hálózatok

Emlékeztető. $L \in_T \mathcal{P}$ akkor, ha létezik $\{C_n\}$ hálózat-sorozat, amely

- (i) n inputbitből számol ki egyet,
- (ii) csúcsainak száma/mérete kisebb, mint $p(n)$, valamely p polinomra,
- (iii) a hálózat input bitjei Σ^ν elemeit kódolják ($n = (\log_2 |\Sigma|) \cdot \nu$, $\omega \in \Sigma^n$ input esetén ω kódja legyen $\lceil \omega \rceil$), továbbá $\omega \in L$ akkor és csak akkor, ha $C_n(\lceil \omega \rceil) = 1$, azaz C_n az ω kódján az elfogadást/elvetést kódoló bitet számolja ki,
- (iv) C_n \mathcal{L} -ben megkonstruálható 1^n -ből.

Megjegyzés. A visszafele irány is igaz. Azaz, ha az L nyelvhez van fenti típusú hálózat, akkor ω -ból felírható az ω -t kódoló bitek, ha ezek száma ν , akkor C_ν a hálózatot is ki tudjuk számolni, végül értékelni. Az eredő eljárás mutatja, hogy $L \in \mathcal{P}$.

Definíció. $L \in \mathcal{P}^{\text{nem-uniform}}$ akkor és csak akkor, ha (i)-(iii) igaz. Azaz \mathcal{P} fenti leírásából csak a hálózat-sorozat „uniformitását” nem követeljük meg.

Megjegyzés. A $\mathcal{P}^{\text{nem-uniform}}$ nyelvosztály „furcsa” az eddigiekhez képest. Nincs benne az eddig ismert legbővebb kiszámíthatósággal kapcsolatos osztályban, \mathcal{S} -ben, azaz a felsorolható nyelvek osztályában.

$$\begin{array}{l} \mathcal{S} \qquad \qquad \supseteq \mathcal{D} \supseteq \text{EXPSPACE} \supseteq \dots \supseteq \mathcal{NP} \supseteq \mathcal{P} \dots \\ \not\subseteq \\ \mathcal{P}^{\text{nem-uniform}} \end{array}$$

Vegyünk egy $\mathcal{B} \subset \mathbb{N}$ bonyolult (nem felsorolható) halmazt. Legyen $L := \{1^\ell : \ell \in \mathcal{B}\}$. L nyilván nincs benne \mathcal{S} -ben, hiszen ha ezt a halmazt fel tudnánk sorolni, akkor \mathcal{B} elemeit is felsorolnánk.

$L \subset \{0, 1\}^*$ benne van $\mathcal{P}^{\text{nem-uniform}}$ -ban: Kétféle inputméret van, \mathcal{B} -beli és nem \mathcal{B} -beli. Egyik inputmérethez semmit sem kell elfogadni, mert olyan hosszban nincs semmi a nyelvben, a másik inputméret pedig olyan, hogy csupa egyes inputot kell elfogadni. Az első esetben hálózatunk, ez az első változót és negáltját ÉS-sel köti össze. A második esetben a hálózat az összes változót ÉS-sel köti össze. Észrevehetjük, hogy a kétféle hálózat önmagában nagyon egyszerű (mérete polinomiális, sőt lineáris), de nagyon bonyolultan, rapszódikusan változik. Mivel nem követeljük meg, hogy megkonstruálható legyen, így beleillik a modellünkbe.

Emlékeztető. A $SAT \in \mathcal{P}$ -ből következik, hogy $\mathcal{P} = \mathcal{NP}$. Sejtés: $SAT \notin \mathcal{P}$. Azaz SAT nem számolható ki \mathcal{L} -ben megkonstruálható polinomiális hálózat-sorozattal.

Kérdés: $SAT \in \mathcal{P}^{nem-uniform}$? **Sejtés:** $SAT \notin \mathcal{P}^{nem-uniform}$. Azaz SAT nem-uniform polinomiális méretű hálózat-sorozattal sem számolható ki.

Mielőtt a kérdést körüljárnánk nézzük $\mathcal{P}^{nem-uniform}$ ekvivalens leírásait.

1. Lemma. *A következők ekvivalensek:*

(i) $L \in \mathcal{P}^{nem-uniform}$.

(ii) Létezik az ω input hosszában polinomiális T nemdeterminisztikus (tanúszalagos) Turing-gép, és létezik $\{t_n\}_{n=1}^{\infty}$ tanúsorozat, hogy $\omega \in L$ akkor és csak akkor, ha $T(\omega, t_{|\omega|}) = 1$ (azaz ω -n a számítás ELFOGAD állapotba vezet).

(iii) $L \preceq_{\mathcal{P}}^{Turing} S$, valamely $S \subset \Sigma^*$ alkalmas ritka nyelvre (definíciók alább).

Definíció. Egy S nyelv ritka, ha benne az n hosszú szavak száma n -ben polinomiális (azaz nagy n -re jóval kevesebb, mint a teljes lehetőségek $|\Sigma|^n$ exponenciális száma). Formálisan van olyan $q(n)$ polinom, hogy $|S \cap \Sigma^n| \leq q(n)$.

Definíció (Turing-redukció). $L \preceq_{\mathcal{P}}^{Turing} S$ akkor és csak akkor van olyan polinomiális idejű Turing-gép, amely ω L -hez tartozását dönti el. Számolása alatt „?” állapotba mehet a gép, aminek van egy kérdező szalagja is. A speciális állapotba kerülés egy kérdés: a kérdező szalagra az előző kérdés óta felírt karaktersorozatról derül ki (egy lépésben) az S -hez tartozás. Az „egy lépésben kiderül” alatt azt értjük, hogy a rákövetkező konfigurációban BENNE-VAN vagy pedig a NINCS-BENNE állapotkomponense lesz, aszerint, hogy a kérdező szalag tartalma S -beli vagy sem.

Az S -et szokták orákulumnak is nevezni. S tulajdonképpen egy „meg nem írt szubrutin”, ami ha valamilyen módon megírható, mondjuk megírása könnyű, akkor L sem lehet nehéz.

Bizonyítás. (i) \Rightarrow (ii): Minden n -hez tartozik egy hossz, amelyben bitekkel kódoljuk az Σ^n elemeit. Legyen ez ν . Ha L nem-uniform polinomiális időben van, akkor tartozik hozzá egy $\{C_n\}$ hálózat-sorozat. Legyen t_n a C_n hálózat kódja. A T Turing-gép a tanúszalag tartalmából kiolvassa C_n -t, az input-szalag tartalmát 0–1 bitekkel kódolja, majd a C_n -t ennek tartalmán kiértékeli. Ha 1-et kap ELFOGAD, ha 0-t ELVET állapotba kerül.

(ii) \Rightarrow (iii): Tegyük fel, hogy létezik egy nem determinisztikus, polinomiális gép az input hosszától függő tanúval. Ezt vissza tudjuk vezetni egy S ritka nyelvre. Ha ismerjük a tanút, \mathcal{P} időben egyszerű a feladat, de ez ritka. Legyen $S := \{(1^k, t_k^{\leq \ell}) : k \in \mathbb{N}, \ell \leq t_k \text{ hossza}\}$. S a következő alakú: $\underbrace{1 \ 1 \ 1 \ \dots \ 1}_{k \text{ db}}$; „ t_k első ℓ' karaktere”

($\ell' \leq \ell$).

2. Állítás. *Az előbb definiált S nyelv ritka. Az 1-es blokk elemszáma meghatározza, hogy melyik tanúnak a része következik.*

3. Állítás. *Adott $\omega \in L$ esetén \mathcal{P} időben az S -hez tartozásra kérdéssel $t_{|\omega|}$ kiszámítható, aminek ismerete megoldja az L -hez tartozás kérdését.*

Beírjuk az $1^{|\omega|}$ karaktert, majd megkérdezzük, hogy $\Sigma := \{\sigma_1, \dots, \sigma_k\}$ valamely eleme S -beli-e.

$$\begin{array}{l} 1^{|\omega|}; \quad \sigma_1 \stackrel{?}{\in} S \\ \quad \quad \quad \sigma_2 \stackrel{?}{\in} S \\ \quad \quad \quad \vdots \\ \quad \quad \quad \sigma_k \stackrel{?}{\in} S \end{array}$$

Ha igent kapunk, akkor megvan a tanúnak az első bitje.

Ha azt kapjuk, hogy nem végigmenve az ábécén, akkor az eddigi bitek megadják a tanút. Ez polinom időben eldönthető T -vel, ahol T (ii)-ből való.

(iii) \Rightarrow (i): Konstruálunk egy hálózatsorozatot. Korábban láttuk, hogy egy Turing-gépből (adott inputhossz esetén) hogyan lehet egy a számolását szimuláló hálózatot konstruálni. A gondolatmenetet megismételjük a redukciót bizonyító orákulumos Turing-gépre.

Természetesen a konfigurációban ott szerepel e kérdező-szalag tartalma is. Egyetlen lényeges különbség van. Kezelnünk kell a ‘?’ állapotot. Az időkorlát miatt tudjuk a kérdés hosszát becsülni. Az S -hez tartozás így egy polinomiális méretű S -beli részhalmazhoz tartozással ekvivalens. Ha egy ℓ hosszú k kérdést teszünk fel az orákulumnak, akkor az

$$(s_1 = k) \vee (s_2 = k) \vee \dots \vee (s_{q(\ell)} = k),$$

ahol s_i az S ritka halmaz „rövid” elemei. A lista ismeretében egy polinomiális méretű, ezt eldöntő hálózat megtervezhető, bármi is legyen az inputméret.

A bizonyítás befejezése a korábbiak alapján standard módon megy. ■

A fent definiált hálózatról nem állíthatjuk/állítjuk, hogy uniform. Egy konfigurációból a következő kiszámolása esetén számolnunk kell, hogy egy S -hez tartozást döntünk el. Ez S ritkasága miatt megtehető, de uniformitásról szó sincs.

2. Karp—Lipton—Sipser-tétel

Tudjuk, hogy $SAT \in \mathcal{P}$ -nek nem várt következményei lennének. Mi a helyzet $SAT \in \mathcal{P}^{\text{nem-uniform}}$ esetén?

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

Definíció. $L \in \Sigma_i \mathcal{P} \stackrel{\text{def}}{\iff} \exists T \ |\omega|$ -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. Azaz i -szer váltakoznak a kvantorjelek \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. $L \in \Pi_i \mathcal{P} \stackrel{\text{def}}{\iff} \exists T \ |\omega|$ -ban polinomiális tanúszalagos Turing-gép úgy, hogy a plusz tanúszalag tartalma: $\forall \tau_1 \exists \tau_2 \dots Q \tau_i$, és

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

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 azonoságából 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

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

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

Ezekután lássuk a főtételeinket:

4. Tétel (Karp—Lipton, Sipser tétele). *Ha $SAT \in \mathcal{P}^{nem-uniform}$, akkor*

$$\Pi_2\mathcal{P} \subset \Sigma_2\mathcal{P}.$$

Bizonyítás. A tétel bizonyításához legyen $L \in \Pi_2\mathcal{P}$ tetszőleges nyelv. Meg fogjuk mutatni, hogy amennyiben $SAT \in \mathcal{P}^{nem-uniform}$ teljesül, akkor $L \in \Sigma_i\mathcal{P}$ 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):

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

Így leírtunk ω^+ -ok egy halmazát, egy \tilde{L} nyelvet. A leírásból nyilvánvaló, hogy $\tilde{L} \in \mathcal{NP} = \Sigma_1\mathcal{P}$. A Cook—Levin-tétel szerint $SAT \mathcal{NP}$ -teljes, így L nyelv redukálható SAT -ra: $\tilde{L} \preceq SAT$. 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

$$\omega^+ \in \tilde{L} \iff R(\omega^+) \in SAT.$$

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

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

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

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

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

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

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

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

Ssajnos 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:

5. Lemma. *Létezik olyan polinomiális TG, hogy ami egy halózatból (igazából halózat kódjából) egy másik halózatot számol ki: $\{C_n\} \mapsto \{\tilde{C}_n\}$ úgy, hogy*

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

(ii) *ha C_n nem SAT-ot számolta ki, akkor is $\tilde{C}_n(\lceil \varphi \rceil) = 1$ 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 \tilde{C}_n hibázhat SAT eldöntésében, de csak az egyik irányban. A lemmából következik, hogy

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

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

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 \tilde{C}_n -okat rekurzívan, input a $\langle \varphi(x_1, \dots, x_k) \rangle$. Erre az inputra megnézzük, mit ad C_n . Ha 0-t, akkor „nem aggódunk”, ha 1-et (ami a rossz irányú tévedést is megengedi), akkor C_n kap két új inputot:

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

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

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

ahol ε_k az előző „siker” x_k érték: $\varepsilon_k \in \{0, 1\}$. 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 $\varphi(\varepsilon_1, \dots, \varepsilon_k)$ -t. Ha ezen az input-formula 1-et számol ki, akkor \tilde{C}_n is 1-et ad, különben \tilde{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 \tilde{C}_n az 1 eredményt adja, akkor ki is számolt 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, mivel ha kvantoroknak legalább három blokkja van (legalább kétszer alternálunk), akkor a polinomiális hierarchia nem valódi. Ezt egy példával világítjuk meg

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

Azaz

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

így

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

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

3. Mahaney-tétel

A koárbbi karakterizációink alapján Karp—Lipton, Sipser tétele megfogalmazható a következő alakban.

6. Tétel (Karp—Lipton, Sipser tétele). *Ha $SAT \preceq_{\mathcal{P}}^{Turing} S$, ahol S ritka nyelv, akkor $\Pi_2 \mathcal{P} \subseteq \Sigma_2 \mathcal{P}$.*

A Karp-redukció felfogható Turing-redukcióként is, orákulum nélkül számolunk, egyetlen kérdést generálunk és csak ezt az egy kérdést tehetjük fel S -nek, amire kapott válasz az outputja. Mahaney vette észre, ha a Karp—Lipton, Sipser tételének feltételben szereplő Turing-redukciónál erősebb Karp-redukciót tesszük fel, akkor erősebb következményt igazolhatunk.

7. Tétel (Mahaney-tétel). *Ha $SAT \preceq_{\mathcal{P}}^{Karp} S$, akkor $\mathcal{P} = \mathcal{NP}$.*

A bizonyítás előtt átfogalmazzuk a tételt.

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

Feltesszük, hogy Σ rendezett (karaktereink között van egy rendezés, ábécé sorrend): $(\Sigma^{\leq n}, \leq)$. 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 $\omega_1 = a_1 \dots a_k, \omega_2 = b_1 \dots b_\ell \in \Sigma^{\leq n}$ két tetszőleges eleme $\Sigma^{\leq n}$ -nek. Legyen $m = \min\{k, \ell\}$. Ekkor $\omega_1 < \omega_2$ pontosan akkor teljesül, ha a következő két eset valamelyike fennáll:

- (1) $a_1 \dots a_m$ szigorúan előbb van a lexikografikus sorozatban, mint $b_1 \dots b_m$,
- (2) ω_1 az ω_2 szó kiterjesztése (ekkor $a_1 \dots a_m = b_1 \dots b_m$, (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 (feltüntetettük, hogy a sorendet melyik szabály határozza meg):

$$\begin{array}{cccccccccccc} 000 & \underset{(1)}{<} & 001 & \underset{(2)}{<} & 00 & \underset{(1)}{<} & 010 & \underset{(1)}{<} & 01 & \underset{(2)}{<} & 0 & \underset{(1)}{<} & 100 & \underset{(1)}{<} \\ & & & & & & & & & & & & & & \\ \underset{(1)}{<} & & 101 & \underset{(2)}{<} & 10 & \underset{(1)}{<} & 110 & \underset{(1)}{<} & 111 & \underset{(2)}{<} & 11 & \underset{(2)}{<} & 1 & \underset{(2)}{<} & \varepsilon. \end{array}$$

Bevezetjük SAT egy nehezítését:

Definíció.

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

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

Vegyük észre, hogy $SAT^* \in \mathcal{NP}$, így $SAT^* \mathcal{NP}$ -teljes. Ezek után újrafogalmazhatjuk Mahaney tételét:

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

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

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

- (i) Az L_i lista elemei i hosszú 0-1 sorozatok. Azaz $L_i \subseteq \{0, 1\}^i$ $i \in \{0, 1, \dots, n\}$
- (ii) L_i hossza polinomiális. Azaz $\exists q$ polinom, hogy $|L_i| < q(n)$.
- (iii) $\forall i \quad \exists \ell_i \in L_i$: ha φ kielégíthető, akkor legyen a olyan kielégítése, amely $\leq \ell_i$.

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

Kérdések:

1. Mi lesz L_0 ?
2. L_i ismeretében hogyan számolható ki L_{i+1} ?
3. Miért korrekt az algoritmus?

Válaszok:

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

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

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 annyiban 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 lexikografikusan 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 (R polinomiális algoritmus) és az ilyen hosszú S -beli lehetőségek számát is becsülhetjük S ritkasága miatt. Legyen \bar{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:

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

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

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

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

Ezzel látjuk, hogy a listáinkat hogyan generáljuk. Az algoritmus végen L_n elemeiről ellenőrizzük, hogy kielégítik-e φ -t. Ha valamelyik igen, akkor elfogadjuk az inputot (és ebben az esetben válaszuk 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 leeg'íthető 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 $k_{i+1} \in L_{i+1}^+$. Azt kell igazolnunk, hogy k_{i+1} az összes ritkítást túléli.

Az első ritkítésnél ez stimmel, hiszen $\{0, 1\}^i$

L_i -t meghatározhatjuk, és mivel k_{i+1} a k_i egy kiterjesztése, így $k_{i+1} \in L_i^+$.

Kiszámítjuk ezek után L_i^+ -t, amelynek szintén eleme k_{i+1} . Ennek során lesz egy olyan halmaz, amely tartalmazza k_i -t, és ugyanarra az S -beli elemre redukálódik. Ebből k_i lesz a lexikografikusan első elem, így azt fogjuk meghagyni. ■