ALGORITMUSOK ÉS BONYOLULTSÁGELMÉLET

Matematika MSc hallgatók számára

11. Előadás: $\mathcal{BPP}$, nem-uniform $\mathcal{P}$ és a hierarchia

Előadó: Hajnal Péter 2015. tavasz

1. Nem-uniform polinomiális idő

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 $\ulcorner\omega\urcorner$), továbbá $\omega \in L$ akkor és csakis akkor, ha $C_n(\ulcorner\omega\urcorner) = 1$, 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 csakis 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.

$$\begin{array}{ll} \mathcal{S} & \supseteq \mathcal{D} \supseteq \mathcal{EXPSPACE} \supseteq \ldots \supseteq \mathcal{NP} \supseteq \mathcal{P} \ldots \\ \not\in & \\ \mathcal{P}^{\text{nem-uniform}} & \end{array}$$

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.

Mielőtt az osztályt körüljárnánk nézzük $\mathcal{P}^{\text{nem-uniform}}$ ekvivalens leírásait.

1. Lemma. *A következők ekvivalensek:*

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

(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 csakis 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). Formálsian van olyan $q(n)$ polinom, hogy $|S \cap \Sigma^n| \leq q(n)$.

Definíció (Turing-redukció). $L \preceq_{\mathcal{P}}^{\text{Turing}} S$ 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)⇒(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_ν hálózat kódja. A T Turing-gép a tanú-szalag tartalmából kiolvassa C_ν-t, az input-szalag tartalmát $0-1$ bitekkel kódolja, majd a C_ν-t ennek tartalmán kiértékeli. Ha 1-et kap ELFOGAD, ha 0-t ELVET állapotba kerül.

(ii)⇒(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: $\underbrace{1\ 1\ 1\ \ldots\ 1}_{k \text{ db}}$; „$t_k$ 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 kérdezéssel $t_{|\omega|}$ kiszámítható, aminek ismerete megoldja az L-hez tartozás kérdését.*

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

$$1^{|\omega|};\quad \sigma_1 \overset{?}{\in} S$$
$$\sigma_2 \overset{?}{\in} S$$
$$\vdots$$
$$\sigma_k \overset{?}{\in} S$$

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)⇒(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

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

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. $\mathcal{BPP}$ helye a korábbi osztályok között

4. Tétel (Adlemann).

$$\mathcal{BPP} \in \mathcal{P}^{nem-uniform}.$$

Bizonyítás. Legyen $L \in \mathcal{BPP}$. Ekkor (az észrevételeket felhasználva) létezik T véletlen Turing-gép úgy, hogy bármely ω-ra („T hibázik") $\leq 1/2^{q}(n)$, ahol $q(n)$ egy általunk választott polinom, n pedig ω hosszát jelenti.

Ekkor jelöljük a rossz ρ-kat az alábbi módon:

$$R_\omega = \{\rho : T(\omega, \rho) \text{ futás eredménye hibás}\}.$$

Ha összegezve az összes ω-ra az „$r \in R_\omega$" események valószínűségét egy 1-nél határozottan kisebb számot kapunk, akkor létezik olyan ρ_0 véletlen szalagtartalom, melyre igaz, hogy egyik R_ω-hoz sem tartozik hozzá, azaz $T(\omega, \rho_0)$ futás bármely ω-ra korrekt. Ez könnyen elérhető, hiszen $|\Sigma|^n$ darab n hosszú input van és „$r \in R_\omega$" valószínűsége $1/2^{n^2}$-nél kisebbé tehető.

A $T(\omega, \rho)$ futás számolását egy polinom méretű hálózattal leírhatjuk/kódolhatjuk. Tudunk olyan hálózatot csinálni, ami veszi az ω és ρ bitkódját és kiszámolja a futás végeredményét, ami 0 esetén ELVET, 1 esetén pedig ELFOGAD.

Az (ω, ρ) inputot olvasó hálózat polinom méretű és uniform.

A tételt bizonyító hálózat ugyanez a hálózat lesz, de $0, 1$-ek fognak szerepelni azon változók helyén, amelyek a véletlen biteket kódolják. (Így csak az ω inputot kódoló változók lesznek inputkapuk a hálózatban.) A véletlen bitek lerögzítése/behuzalozása úgy történik, hogy a ρ_0 véletleszszalag-tartalmat kódolja.

Az új hálózat már nem uniform, hiszen ρ_0 megtalálására nincs eljárásunk. Egy összeszámolási indoklás alapján tudjuk létét. Másrészt hálózatunk n hoszú inputokon T Turing-gépet számolását végzi el, azaz L-et dönti el. ■

5. Tétel (Gács Péter—Sipser).

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

Bizonyítás. (Lautemann) $\mathcal{BPP}$ zárt a komplementálásra, így elég csak belátnunk, hogy $\Sigma_2\mathcal{P}$-ben benne van. A tétel bizonyítása előtt először definiáljunk két fogalmat és igazoljuk az alábbi lemmát.

Definíció. Legyen $R \subset \Sigma^N$.

Az R halmaz akkor és csak akkor ritka, ha $|R| < 1/N|\Sigma^N|$.

Az R halmaz akkor és csak akkor sűrű, ha $|R| > (1 - 1/N)|\Sigma^N|$.

Fontos észrevennünk, hogy ezen fogalmak NEM egymás komplementerei.

6. Lemma (Főlemma). *Legyen $N > |\Sigma|$ tetszőleges. Ekkor*

i) Ha R ritka, akkor tetszőleges $c_1, \ldots, c_N \in \Sigma^N$ esetén $\cup_i (R + c_i) \subsetneq \Sigma^N$

ii) Ha R sűrű, akkor van olyan $c_1, \ldots, c_N \in \Sigma^N$, hogy $\cup_i (R + c_i) = \Sigma^N$.

A lemma bizonyítása: Az i) állítás triviális, hiszen a $R + c_i$ halmaz elemszáma megegyezik R elemszámával (a c_i hozzáadása olyan, mintha a számegyenesen eltolnám). Így $\cup_{i=1}^N (R + c_i)$-ben kevesebb elem van, mint $|\Sigma^N|$, ugyanis R ritka.

ii) rész bizonyításához valószínűségszámítási módszert használunk. $c_1, \ldots, c_N$ véletlen, uniform eloszlású Σ^N-beli elemek. Elég belátni, hogy

$$(\cup_{i=1}^N (R + c_i) = \Sigma^N) > 0.$$

Azaz ha véletlen c_i-ket választok, akkor nem lehetetlen esemény, hogy az eltólt R-ek együtt kiadják Σ^N-et. Az „$\cup_{i=1}^N (R + c_i) = \Sigma^N$” eseményünk egy átírása:

„bármely Σ^N-beli x-re létezik olyan i index, melyre $x \in c_i + R$”,

azaz

$$\text{„mindegyik } x \in \Sigma^N\text{-re } (x - R) \cap \{c_1, \ldots, c_N\} \neq \emptyset\text{”},$$

azaz

$$\overline{\text{„van olyan } x \in \Sigma^N\text{, hogy } (x - R) \cap \{c_1, \ldots, c_N\} = \emptyset\text{”}} = \overline{B}.$$

Tudjuk, hogy az $x - R$ halmaz mérete legalább $(1 - 1/N)|\Sigma^N|$, mivel R sűrű. Ekkor kapjuk, hogy konkrét $x \in \Sigma^N$ esetén

$$((x - R) \cap \{c_i\} = \emptyset) \leq \frac{1}{N},$$

továbbá a c_i-k független választása miatt

$$((x - R) \cap \{c_1, \ldots, c_N\} = \emptyset) \leq \left(\frac{1}{N}\right)^N,$$

Jelöljük B_x-szel az x-hez rossz c_i-k halmazát. Kaptuk, hogy B_x valószínűsége kisebb, mint $(1/N)^N$, továbbá felhasználva, hogy x-ekből $|\Sigma|^N$ darab van, kapjuk hogy $(\cup B_x) = (B) < 1$. Ez adja a lemma állítását. ■

A tétel bizonyítása: Legyen $\Sigma = \{0, 1, \ldots, |\Sigma| - 1\}$, additív mod $|\Sigma|$ aritmetikával. Legyen T egy $L \in \mathcal{BPP}$-t bizonyító Turing-gép. Σ^N-beli ω esetén definiáljuk a következő halmazt:

$$J_\omega = \{\omega : \text{ amelyekkel } T \text{ ELFOGAD állapotba kerül } \omega\text{-n}\}.$$

Ha $\omega \in L$, akkor J_ω sűrű, míg ellenkező esetben, J_ω ritka. A Főlemmát alkalmazva könnyen kapjuk a Tételbeli állítást. $\omega \in L$ akkor és csak akkor, ha léteznek a Főlemmabeli c_i-k úgy, hogy bármely Σ^N-beli x esetén $c_i + J_\omega$ lefedi x-et valamely i indexre, azaz kapjuk, hogy $c_i - x$ J_ω-beli, $c_i - x$ véletlen szalagtartalom esetén ELFOGAD-ó futása van a Turing-gépnek.

A tétel bizonyításához már csak azt kell belátnunk, hogy a fenti leírás L egy $\Sigma_2\mathcal{P}$-beli nyelvként való jellemzése. Valóban:

$$\omega \in L \Longleftrightarrow \exists c_1, \ldots, c_N \forall x\text{-re } (T(\omega, c_i - x) = ELFOGAD),$$

ahol a zárójeles kifejezés egy determinisztikus, polinomiális Turing-géppel eldönthető. Így a bizonyítás már teljes. ■

3. 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. $L \in \Sigma_i\mathcal{P} \overset{\text{def}}{\Longleftrightarrow} \exists T$ $|\omega|$-ban polinomiális tanúszalagos Turing-gép úgy, hogy a plusz tanúszalag tartalma $\exists\tau_1\forall\tau_2 \ldots Q\tau_i$, ahol Q egy kvantor. Azaz i-szer váltakoznak a kvantorjelek $\exists$-kel kezdődően. Továbbá

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

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: $\forall\tau_1\exists\tau_2 \ldots Q\tau_i$, és

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

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

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

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

Ezekután lássuk a főtételünket:

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

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

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

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

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

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

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

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

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 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:

8. Lemma. *Létezik olyan polinomiális TG, hogy ami egy halózatból (igazából halózat kódjából) egy másik halózatot 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 $L \in \Sigma_i\mathcal{P}$, 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 $\widetilde{C}_n$-okat rekurzívan, input a $\langle\varphi(x_1, \ldots, x_k)\rangle$. 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:

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

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 halózat 1-et számolt ki. Vesszük a következő inputot:

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

ahol ε_k az előző „sikeres” 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 $\varphi(\varepsilon_1,\ldots,\varepsilon_k)$-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 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

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

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”.

4. Mahaney-tétel

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

9. 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.

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

A bizonyítás előtt átfogalmazzuk a tételt.

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

Feltesszük, hogy Σ rendezett (karaktereink között van egy rendezés, ábécé sorrend): $(\Sigma^{\leq n}, \leq)$. 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 $\omega_1 = a_1\ldots a_k, \omega_2 = b_1\ldots b_\ell \in \Sigma^{\leq n}$ 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 (feltüntettük, hogy a soorendet melyik szabály határozza meg):

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

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

Bevezetjük SAT egy nehezítését:

Definíció.

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

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

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

11. 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: $\text{SAT}^* \preceq S$, 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 $L_0, L_1, \dots, L_n$ listasorozatot számol ki polinom időben, amelyre a következők teljesüljenek:

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

(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

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

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 (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:

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

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 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. Az algoritmus végen L_n elemeiről ellenórizzük, 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, akko r φ nem is elégíthető ki.

Indirekten bizonyítunk. Tegyük fel, hogy L_n-ben nincs kielégító 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.

Az első ritkításnál 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 lexikongrafikusan első elem, így azt fogjuk meghagyni.

■