

13. Előadás

Előadó: Hajnal Péter

Jegyzetelő: Virágh Zita

2012. május 16.

1. Véletlen algoritmusok: Kielégíthetőség

1. Tétel (Lovász László). *Legyen φ egy CNF formula, minden klózához legfeljebb Δ klóz van, amely közös változót tartalmaz vele. (A közös változó vagy ugyanaz a változó azonos, vagy a negált előfordulása.) A következő paramétereket használjuk: D = változók száma klózonként, m = klózok száma, n = összes változó száma $\leq m \cdot D$.*

Ha $\Delta < \frac{2^D}{4D}$, akkor φ kielégíthető.

A tétel most nem bizonyítjuk, de az eredeti bizonyítás nem konstruktív.

Moser és Tardos Gábor adott egy algoritmust, amely a fenti tételt konstruktíven bizonyítja. Ez egy véletlen algoritmus a tételbeli tulajdonságú φ -hez kielégítő értékadás keresésére.

Az algoritmus azt tudja, hogy:

- (i) vagy talál egy kielégítő értékadást, vagy a végtelenségig fut
- (ii) $\mathbb{E}(\text{futás hossza}) \leq p(|\varphi|)$, ahol p egy polinom.

Speciálisan „kicsi” a valószínűsége annak, hogy sokáig fut. Azaz pozitív a valószínűsége, hogy leáll. Azaz az algoritmus igazolja a tételt.

Leírjuk a Moser—Tardos-algoritmust:

2. Algoritmus. (Moser—Tardos)

1. **Kiinduló lépés:** $x_i \leftarrow r_i$

// független uniform eloszlású bitekkel feltöltjük a változókat

2. **Tesztelő lépés:** Kiértékeljük φ -t.

$\rightarrow 1 \Rightarrow$ STOP, találtunk jó kiértékelést.

$\rightarrow 0 \Rightarrow$ tovább a 3. lépésre.

3. **„Hibakeresés”:**

Keresünk egy klózt, ami hamisan van kiértékelve ($\rightarrow C$)

4. **Újrasorsolás:**

C -beli változókat „újrasorsoljuk” (=új független uniform eloszlású biteket generálunk). Vissza 2. lépéshez.

Tehát az algoritmus futása egy ∞ -futás, ha minden újrasorsolás nem kielégítő értékadáshoz vezet.

3. Tétel (Moser—Tardos). *A fenti algoritmus esetén*

$$\mathbb{E}(\text{újrasorsolások száma}) \leq 20 \cdot Dm.$$

Bizonyítás. A véletlen biteket egy $\infty \times n$ -es táblázatként képzeljük el, ahol az oszlopok a változókkal vannak indexelve, míg a sorok a természetes számokkal. A táblázatban független véletlen bitek találhatók (uniform eloszlásúak).

A táblázat segítségével újraértékeljük az algoritmusunk néhány lépését: 1. lépés az 0. sor elolvasásával ekvivalens. A 4. lépésben x_i újrasorsolása a megfelelő oszlopban az első olvasatlan bit elolvasása

Definiálunk egy eseményt:

$$E_{\geq d \cdot N} := \text{„legalább } d \cdot N \text{ bitet kiolvasunk a futás során a táblázatból”}.$$

Ennek egy részeseménye a következő

$$E = \text{„a } d\text{-edik sorból olvasunk”}.$$

Nézzük az első újrasorsolási lépést, ahol a d -edik sorból kiolvasunk egy bitet. Legyen C_0 a megfelelő klóz. (Persze C_0 -t sokszor újrasorsolhatja az algoritmus. A pontos azonosításhoz hozzátartozik egy t_0 idő is. (t_0, C_0) a pontos algoritmusbeli esemény: a t_0 -edik újrasorsolás során C_0 változóit újraértékeljük.

C_0 újrasorsolása D bit olvasása táblázatunkból. Minden pozíció elolvasásának van egy oka: a megfelelő — mondjuk x — változó C_0 -ben szerepel és az éppen kiolvasott bit felett a táblázatban már olvasott bitek vannak. Miért? Kell lennie (t', C') eseménynek $(t' < t_0)$, ahol x szerepel C' -ben is és a t' -edik újrasorsolás az x oszlopában a $d-1$ -edik sort olvasta el. A (t_0, C_0) eseményben az x újrasorsolásának/ x oszlopában történt olvasásnak egy oka a (t', C') esemény. Az események oksága „visszagöngyölhető”.

Az okság visszagöngyölése során a (t, C) eseményeket egy gyökeres fa-struktúrában ábrázoljuk:

- A csúcsok a visszagöngyölés során elért (t, C) események.
- A (t, C) esemény miatt megjelenő csúcs címkéje C . Azaz az oksági fa csak arra „emlékszik”, hogy mely klóz újrasorsolása történt, arra nem, hogy pontosan hanyadik újrasorsolásról van szó.
- A gyökér a kiinduló (t_0, C_0) eseménynek megfelelő csúcs. Ennek választása úgy történt, hogy vettünk egy „mélyen” fekvő elolvasott bitet. A gyökért „mélyre” rajzoljuk, a fa felfelé nő. Azaz egy apa/fiú viszonynál, az apa mélyebben fekszik, a fiú egy szinttel magasabban lesz mint apja.
- Egy él egy C -vel címkézett csúcs és egy felette lévő C' -vel címkézett csúcsot köt össze. Az él megléte azt jelenti, hogy lenni-e kell a mélyebb csúcs eseménye során egy olyan olvasott bitnek, aminek oka (a felette lévő bit elolvasása) a felette lévő szomszéd esemény során történt meg.

Ha egy aktuális esemény vizsgálatával bővíteni szeretnénk a fánkat, akkor egy ághajtást hajtunk végre. A fenti leírtak több lehetőséget is megengednek az új eseménynek a fába történő berakására (ami egy „apa” választás). Mi a legmagasabb lehetséges apához kötjük be a csúcst.

A fenti tulajdonságok már meghatározzák a kiinduló (t, C) esemény oksági fáját: A visszagöngyöltett eseményeket a valódi idő szerint visszafelé haladva a fent leírt „minél magasabbra” szabály szerint a fába illesztjük.

A fent leírt fa (T_0, C_0) esemény oksági fája $\xrightarrow{\text{jelölés}} O_{t_0, C_0}$.

Észrevétel. O_{t_0, C_0} -ből kiolvasható minden csúcshoz, hogy mely biteket olvassuk ki a táblázatból.

Az egyik legfelső levél csúcs esetén ez nyilvánvaló. A címke klózában szereplő minden változója először lett újrasorsolva (1 indexű sorból jön, közvetlen a 0 indexű sor alatt (amely a kezdeti értékadást tartalmazza)). Valóban, ha ez nem így lenne, akkor lenne egy korábbi esemény az okság visszagöngyöltése során, ami szabályunk szerint magasabbra lenne beszúrva a fába. Az indoklás befejezése: magasság szerinti csökkenő sorrendű indukció/rekurzió.

Legyen T egy gyökeres fa klózzal címkézve. Legyen E_T az a valószínűségszámítási esemény, hogy

„ volt egy olyan (t, C) algoritmikus esemény, amelyre $O_{t, C} = T$ ”,

azaz valamelyik újrasorsolás oksági fája a T lesz.

Ezek után elkezdhetjük „sok újrasorsolás” esemény valószínűségének becslését:

$$\mathbb{P}(E_{\geq d, N}) \leq \mathbb{P}(E) \leq \mathbb{P}(„ \exists \text{ legalább } d \text{ pontú } T, \text{ hogy } E_T \text{ beköv. } ”) \leq \sum_{\substack{T \geq d \\ \text{pontú}}} \mathbb{P}(E_T).$$

A szumma tagjait, illetve a szumma tagjainak számát a következő két lemma becsüli:

4. Lemma.

$$\mathbb{P}(E_T) \leq \left(\frac{1}{2^n}\right)^{|V(T)|}$$

Bizonyítás. $|V(T)|$ darab újrasorsolás volt és mindegyiket elindította egy korábbi véletlen bitekkel történő értékadás, ami a megfelelő klózt hamissá tette. De a 2^D lehetőség közt egyetlen egy rossz kimenetel van (annak kellett megvalósulni a véletlen választás során). ■

5. Lemma. *A v pontú gyökeres, klózzal címkézett T fák száma, amelyek megvalósulhatnak oksági faként legfeljebb*

(i) $m\binom{v\Delta}{v-1}$,

(ii) $m(e\Delta)^v$.

Bizonyítás. (ii) elemi becslésekkel adódik (i)-ből.

(i)-hez azt kell látni, hogy T felépítéséhez meg kell adni a gyökér címkéjét (m lehetőség), majd azt, hogy milyen $v - 1$ ághajtás történik. A $v - 1$ ághajtás mindegyike leírható úgy, hogy megmondjuk melyik csúcshoz (v -vel becsülhető lehetőségszám), milyen címkéjű (az ághajtás alapcsúcának címkéjét „metszeni” kell a címkeklóznak, ami legfeljebb Δ lehetőség) csúcsot adunk hozzá. A $v\Delta$ lehetőségből a $v - 1$ ághajtás különböző lesz és sorrendjük lényegtelen (egy élhalmazt írunk le). ■

Folytathatjuk a becslést

$$\begin{aligned} \mathbb{P}(E_{\geq d \cdot N}) &\leq \sum_{\delta=d}^{\infty} \sum_{\substack{T: \\ \delta \text{ csúcstű oksági fa}}} \mathbb{P}(E_T) \leq \sum_{\delta=d}^{\infty} m(e\Delta)^\delta \cdot \left(\frac{1}{2^D}\right)^\delta = m \cdot \sum_{\delta=d}^{\infty} \left(\frac{e\Delta}{2^D}\right)^\delta \\ &= m \cdot \left(\frac{e\Delta}{2^D}\right)^d \cdot \frac{1}{1 - \frac{e\Delta}{2^D}}. \end{aligned}$$

A tétel feltételeiből $\Delta \leq \frac{2^D}{4D}$, azaz $\frac{e\Delta}{2^D} < \frac{1}{D}$ ebből már következik, hogy az utolsó tényező legfeljebb 2 és így

$$\mathbb{P}(E_{\geq d \cdot N}) \leq 2m \frac{1}{D^d}$$

A tétel bizonyításának befejezése standard valószínűségszámítási becslés. ■

2. Véletlen algoritmusok: Elérhetőség

Először definiáljuk a fejezet alapproblémáját.

Definíció. Az ELÉRHETŐSÉG a következő probléma:

Input: G gráf, $s, t \in V(G)$, Output: $\exists?$ st út G -ben.

Azaz ELÉRHETŐSÉG a következő nyelv:

$$\{(G, s, t) : \text{van } st \text{ út } G\text{-ben}\}$$

A probléma irányított változata is fontos:

Definíció. Amennyiben gráfunk irányított és irányított st út létezését teszteljük $\overrightarrow{\text{ELÉRHETŐSÉG}}$ problémáról/nyelvről beszélünk.

Nyilvánvaló, hogy ELÉRHETŐSÉG, $\overrightarrow{\text{ELÉRHETŐSÉG}}$ \mathcal{NL} -beli nyelv, azaz nem-determinisztikus logaritmikusan megoldható. Tudjuk, hogy ELÉRHETŐSÉG, $\overrightarrow{\text{ELÉRHETŐSÉG}}$ $\mathcal{SPACE}(\log^2)$ -beli nyelv, azaz log-négyzet tárban determinisztikusan megoldható (Savitch-tétel). Ennél több $\overrightarrow{\text{ELÉRHETŐSÉG}}$ nyelvről nem is nagyon ismert. Minden előrelépés hatalmas áttörés lenne, hiszen a teljes \mathcal{NL} nyelvostárról tudnánk meg valami lényegeset.

Az ELÉRHETŐSÉG esetén azonban további eredmények ismertek.

6. Tétel (Aleliunas–Karp–Lipton–Lovász–Rackoff, 1979). *Létezik olyan ELÉRHETŐSÉG-et kiszámító véletlen algoritmus, amely*

- (i) *polinomiális és LOGSPACE-ben megvalósítható,*
- (ii) $(G, s, t) \in \text{ELÉRHETŐSÉG}$ esetén $\mathbb{P}(\text{ELFOGAD eredmény } (G, s, t)\text{-n}) > \frac{1}{2}$.
Továbbá $(G, s, t) \notin \text{ELÉRHETŐSÉG}$ esetén $\mathbb{P}(\text{ELVET eredmény } (G, s, t)\text{-n}) = 0$.

A tételt az alábbi, viszonylag egyszerű algoritmus igazolja:

7. Algoritmus. (*Aleliunas–Karp–Lipton–Lovász–Rackoff*)

Véletlen sétát végzünk s -ből kiindulva $10 \cdot n^3$ lépésen keresztül.

Minden lépés után tetszeljük t elérését:

ha elérjük: BIZTOS-JÓ

ha nem érjük el: séta folytatása

Ha ELFOGAD állapot lérése nélkül ért véget a séta akkor VALÓSZÍNŰLEG-ROSSZ állapottal állunk le.

Az algoritmus analízise sem nehéz, ha a Markov-láncok elméletének alapjait ismerjük. Az analízist elhagyjuk.

A tételben szereplő feltételek melletti algoritmussal eldönthető nyelvek osztályát \mathcal{RL} -lel jelöljük. A tétel tömören összefoglalva $\text{ELÉRHETŐSÉG} \in \mathcal{RL}$.

* * *

A véletlen szerepe kiküszöbölhető. Ez a „derandomizáció” egy hosszú kutatás koronája. Csak megemlítjük a végső tételt.

8. Tétel (Reingold, 2008). $\text{ELÉRHETŐSÉG} \in \mathcal{L}$.