MŰEGYETEM 1782

TYPOTEX

RÓNYAI LAJOS

VÉLETLEN ÉS ALGORITMUSOK

2011

A jegyzet elsősorban a BME Villamosmérnöki és Informatikai Karának informatikus MSc hallgatói számára készült, a Felsőbb matematika D című tárgyhoz.

A véletlen választások alkalmazása átszövi az egész számítógépes világot, jelen van az alapvető protokolloktól a szoftvertechnológiáig szinte minden nagyobb részterületen. Hogyan lehet hatékony véletlen módszereket kapni? Mikor van létjogosultságuk az ilyen megoldásoknak, és mikor érdemes inkább mással próbálkozni? Milyen általános elveket követnek ezek a módszerek? Ezekre a kérdésekre legegyszerűbben talán a *véletlent használó számítási módszerek*, másként mondva a *randomizált algoritmusok* tanulmányozásával kereshetjük a választ. Célunk, hogy megismerkedjünk a legfontosabb ilyen módszerekkel. Eddig nem volt magyar nyelven elérhető ilyen tárgyú jegyzet vagy tankönyv.

Az első fejezetben összefoglaljuk a valószínűséggel kapcsolatos alapfogalmakat. A második fejezetet néhány nevezetes és fontos randomizált algoritmus bemutatásának szenteljük. A *gyorsrendezés* a kiindulópontunk. Ezután érdekes és jellegzetes eljárásokat tárgyalunk geometriai/grafikai, aritmetikai és algebrai feladatokra. Bemutatunk két erőteljes, a véletlent érdemben használó adatszerkezetet (falom, univerzális hashelés). A fejezet Karger, Klein és Tarjan minimális költségű feszítőfát számító algoritmusának bemutatásával zárul.

A harmadik fejezetben a *véletlen módszer* matematikai gyökereivel foglalkozunk. Vezérfonalunk Erdős Pál óriási horderejű felismerése: véletlen választások segítségével érdekes matematikai struktúrák létezése igazolható. A nevezetes példák (hipergráf-színezés, Ramsey-számok, Turán-számok) tárgyalásakor ismételten hangsúlyozzuk, hogy ezek a tiszta létezést bizonyító érvelések igen gyakran vezetnek hatékony randomizált algoritmushoz. Itt foglalkozunk az algoritmusainkban felhasznált véletlen csökkentésének, a *derandomizálásnak* a problémakörével is. Lovász lokális lemmája és annak a nemrég felfedezett briliáns algoritmikus változata (Moser–Tardos) zárja a fejezetet.

A negyedik fejezetben azzal a kérdéssel foglalkozunk, hogy a véletlent használó algoritmusok miként jelennek meg a bonyolultsági osztályok térképén. Megismerkedünk az RP, Las Vegas, és a BPP feladatosztályokkal, és vázoljuk a korábban megismert nagy osztályokhoz való viszonyukat, pontosabban azt, amit ma tudunk ezekről. Lesz szó a BPP=P? kérdésről, ami azt feszegeti, hogy tud-e a véletlen *nagyot* segíteni? Másként fogalmazva: van-e olyan feladat, amely a véletlent segítségül híva polinom időben megoldható, véletlen nélkül viszont nem? A véletlen és az együttes munka (interakció) ötvözetét leíró *interaktív bizonyítások* is itt kaptak helyet; fontos gyakorlati alkalmazása ennek a gondolatkörnek a *nulla ismeretű bizonyítás*, amit széles körben használnak a titkos adatközlés területén.

Az utolsó fejezetben gráfokat és véletlent alkalmazó modelleké a főszerep. Először véges Markov-láncokkal foglalkozunk. Az algoritmikus alkalmazások közül tárgyalunk néhány fontosabbat: elérhetőség irányítatlan gráfokban, a PageRank algoritmus, Metropolis-algoritmus. A fejezet második részében a nagy, bonyolult szerkezetű hálózatok modellezésére alkalmas véletlen gráfokkal foglalkozunk. Az Erdős–Rényi-gráfok, a Watts–Strogatz-, és Albert–Barabási-gráfok rövid ismertetése után Kleinberg modelljével zárul az anyag.

Kulcsszavak: véletlen, algoritmus, randomizálás, rendezés, keresés, bonyolultsági osztályok, komplex hálózatok.

Támogatás:

Készült a TÁMOP-4.1.2-08/2/A/KMR-2009-0028 számú, a „Természettudományos (matematika és fizika) képzés a műszaki és informatikai felsőoktatásban” című projekt keretében.

Készült:

a BME TTK Matematika Intézet gondozásában

Szakmai felelős vezető:

Ferenczi Miklós

Lektorálta:

Benczur András

Az elektronikus kiadást előkészítette:

Vető Bálint

Címlap grafikai terve:

Csépány Gergely László, Tóth Norbert

ISBN: 978-963-279-451-8

Copyright: © 2011–2016, Rónyai Lajos, BME

„A © terminusai: A szerző nevének feltüntetése mellett nem kereskedelmi céllal szabadon másolható, terjeszthető, megjelentethető és előadható, de nem módosítható.”

Tartalomjegyzék

Nem a szükségszerűség, hanem a véletlen van teli varázzsal. Ahhoz, hogy a szerelem felejthetetlen legyen, úgy kell röpködnie körülötte az első pillanattól a véletleneknek, mint a madaraknak assisi Szent Ferenc vállánál.

(Milan Kundera: A lét elviselhetetlen könnyűsége)

Előszó

A jegyzet elsősorban a BME Villamosmérnöki és Informatikai Karának informatikus MSc hallgatói számára készült, a Felsőbb matematika D című tárgyhoz. A 2009-es előadásom óravázlatait Domboróczky Attila tette LaTeX-be, ennek bővítésével és csiszolásával alakult ki a jelenlegi anyag.

A véletlen, a randomizálás a kezdetektől jelen van a számítógépes módszerek világában. Az első nevezetes algoritmikus alkalmazás egy szimulációs program volt az egyesült államokbeli Los Alamosban, a második világháború idején, ahol az atombomba létrehozásán dolgoztak. A maghasadáskor felszabaduló neutronok különböző anyagokban való terjedését elemezték ilyen módszerrel. A hagyomány szerint[1] Neumann János a *Monte Carlo* kódnevet adta a titkos projektnek. Az első széles körben ismert Monte Carlo-módszer a Metropolis-algoritmus lett, amiről a jegyzetben részletesebben is lesz szó. Az utóbbi évtizedekben a *Monte Carlo-algoritmus* a véletlen választásokat használó algoritmus szinonimája lett.

A véletlen választások alkalmazása átszövi az egész számítógépes világot, jelen van az alapvető protokolloktól a szoftvertechnológiáig szinte minden nagyobb részterületen. Hogyan lehet hatékony véletlen módszereket kapni? Mikor van létjogosultságuk az ilyen megoldásoknak, és mikor érdemes inkább mással próbálkozni? Milyen általános elveket követnek ezek a módszerek? Ezekre a kérdésekre legegyszerűbben talán a *véletlent használó számítási módszerek*, másként mondva a *randomizált algoritmusok* tanulmányozásával kereshetjük a választ. Elsődleges célunk, hogy megismerkedjünk a legfontosabb ilyen módszerekkel, tervezésük, elemzésük legegyszerűbb kérdéseivel.

A hazai informatikusképzés tanterveivel összhangban feltételezzük, hogy az Olvasó már ismeri egy bevezető valószínűségszámítási és egy algoritmuselméleti témájú egyetemi tárgy anyagát.

A jegyzet első fejezetében igen röviden összefoglaljuk azokat a valószínűséggel kapcsolatos alapfogalmakat, tételeket, amelyeket később gyakran használunk. A második fejezet a leghosszabb; ezt néhány nevezetes és fontos randomizált algoritmus bemutatásának szenteljük. A ma már klasszikusnak számító és igen hatékony *gyorsrendezés* lesz a kiindulópontunk, ennek alaposabb tanulmányozása során már előkerülnek a véletlent használó módszerek elemzésének és tervezésének fő problémái. Ezt követően érdekes és jellegzetes eljárásokat tárgyalunk geometriai/grafikai, aritmetikai és algebrai jellegű feladatokra. Itt mutatunk be két erőteljes, a véletlent érdemben használó adatszerkezetet is (falom, univerzális hashelés). A fejezet egy összetettebb, a véletlen választásokat rendkívül elegánsan és bámulatosan hatékonyan alkalmazó eljárás, Karger, Klein és Tarjan minimális költségű feszítőfát számító algoritmusának bemutatásával zárul.

A harmadik fejezetben a *véletlen módszer* matematikai gyökereivel foglalkozunk. A vezérfonalunk Erdős Pál óriási horderejű felismerése: véletlen választások segítségével érdekes matematikai struktúrák létezése igazolható. A nevezetes példák (hipergráf-színezés, Ramsey-számok, Turán-számok) tárgyalásakor ismételten hangsúlyozzuk, hogy ezek a tiszta létezést bizonyító érvelések igen gyakran vezetnek hatékony randomizált algoritmushoz. Itt foglalkozunk az algoritmusainkban felhasznált véletlen csökkentésének, a *derandomizálásnak* a problémakörével is. Egy haladó technika, a Lovász lokális lemmája, és annak a nemrég felfedezett briliáns algoritmikus változata (Moser–Tardos) zárja ezt a fejezetet.

[1] [GS], 11. oldal.

A negyedik fejezetben azzal a kérdéssel foglalkozunk, hogy a véletlent használó algoritmusok miként jelennek meg a bonyolultsági osztályok térképén. Megismerkedünk az RP, Las Vegas, és a BPP feladatosztályokkal, és vázoljuk a korábban megismert nagy osztályokhoz való viszonyukat, pontosabban azt, amit ma tudunk ezekről. Lesz szó a BPP=P? kérdésről, ami talán a terület legfontosabb nyitott problémája; azt feszegeti, hogy tud-e a véletlen *nagyot* segíteni? Másként fogalmazva: van-e olyan feladat, amely a véletlent segítségül híva polinom időben megoldható, véletlen nélkül viszont nem? A véletlen és az együttes munka (interakció) ötvözetét leíró *interaktív bizonyítások* is itt kaptak helyet; fontos gyakorlati alkalmazása ennek a gondolatkörnek a *nulla ismeretű bizonyítás*, amit széles körben használnak a titkos adatközlés területén.

Az utolsó fejezetben gráfokat és véletlent alkalmazó modelleké a főszerep. Először véges Markov-láncokkal foglalkozunk. Néhány alapfogalom bevezetése után az algoritmikus alkalmazások közül tárgyalunk néhány fontosabbat: elérhetőség irányítatlan gráfokban, a PageRank algoritmus, Metropolis-algoritmus. A fejezet második részében a nagy, bonyolult szerkezetű hálózatok modellezésére alkalmas véletlen gráfokkal foglalkozunk. Az Erdős–Rényi-gráfok, a Watts–Strogatz-, és Albert–Barabási-gráfok rövid ismertetése után Kleinberg elegáns, a kis világ jelenség algoritmikus változatát mutató modelljével zárul az anyag.

A feldolgozott ismeretekkel kapcsolatos további olvasnivalókra a lábjegyzetekben utalunk. Gyakran hivatkozunk a [RISz] tankönyv egyes részeire.

A jegyzetben a log e alapú logaritmust jelöl. Az ettől eltérő alapszámot a szokásos módon tüntetjük fel (pl. $\log_2$).

A címlapon Monte Carlo tulipánok láthatók.

Köszönetet mondok Györfi Lászlónak és Kós Gézának az anyaggal kapcsolatos értékes észrevételeikért, Krähling Editnek a kézirat nyelvi lektorálásáért. Hálásan köszönöm Benczúr Andrásnak, hogy olyan sokat megosztott velem a tárggyal kapcsolatos egészen kivételes tudásából; ezen túl is egy sor hasznos észrevétellel segített a jegyzet írásában.

Budapest, 2011. április 18.

1. fejezet

Alapfogalmak és tételek valószínűségszámításból

Itt felsorolunk néhány olyan fogalmat, eredményt a valószínűségszámítás köréből, amit gyakran használunk a jegyzetben. Feltételezzük, hogy az Olvasó már találkozott velük. A részletek a legtöbb bevezető valószínűségszámítási jegyzetben és tankönyvben megtalálhatók. Ilyenek például: [BT], [Ke], [R], [F], [P].

Gyakran lesz dolgunk a *véges valószínűségi tér* fogalmával: ez egy $\Omega = \{\omega_1, \dots, \omega_n\}$ halmaz az elemeihez rendelt nemnegatív $p_1, \dots, p_n$ számokkal, a valószínűségekkel, amelyekre $p_1 + \dots + p_n = 1$ teljesül. Ritkábban, de lesz szó végtelen (diszkrét) valószínűségi terekről is. Ekkor $\Omega = \{\omega_1, \dots, \omega_n, \dots\}$ és a nemnegatív p_i számokra $\sum_{i=1}^{\infty} p_i = 1$ a követelmény.

Az Ω részhalmazai az *események*. Az $A \subseteq \Omega$ esemény $\mathbf{P}(A)$ valószínűsége azon p_i valószínűségek összege, melyeknél $\omega_i \in A$. Érvényes az *unió-korlát*: tetszőleges $A_1, \dots, A_m$ eseményekre

$$\mathbf{P}(A_1 \cup A_2 \cup \dots \cup A_m) \le \mathbf{P}(A_1) + \dots + \mathbf{P}(A_m).$$

Egyenlőség csak akkor lehetséges, ha minden $i \neq j$ párra $\mathbf{P}(A_i \cap A_j) = 0$.

Az Ω halmazon értelmezett valós értékű függvények a *valószínűségi változók*. A ξ valószínűségi változó várható értékét $\mathbf{E}(\xi)$ jelöli. Mi itt csak diszkrét valószínűségi változókkal foglalkozunk, amelyek értékkészlete a nemnegatív egészek $\mathbb{Z}^+$ halmazából való. Ekkor a *várható érték* a következő egyszerű összeggel fejezhető ki:

$$\mathbf{E}(\xi) = \sum_{i=0}^{\infty} k \cdot \mathbf{P}(\xi = k).$$

A várható érték lineáris: ha ξ, η valószínűségi változók, amelyeknek van várható értéke, a, b pedig valós számok, akkor

$$\mathbf{E}(a\xi + b\eta) = a\mathbf{E}(\xi) + b(E\eta).$$

Legyen A egy esemény egy valószínűségi térből. Az A *indikátora* az a $\xi = \xi_A$ valószínűségi változó, amelynek értéke 1 az A elemein, másutt pedig 0. Ekkor $\mathbf{E}(\xi) = \mathbf{P}(\xi = 1)$. A ξ_A indikátort p paraméterű Bernoulli-változónak is mondjuk, ha $\mathbf{P}(A) = p$.

Az $A_1, \dots, A_n$ *teljes eseményrendszer*, ha $A_1 \cup A_2 \cup \dots \cup A_n = \Omega$, és $i \neq j$ esetén $A_i \cap A_j = \emptyset$.

Legyenek $A, B \subseteq \Omega$ események és $\mathbf{P}(B) > 0$. Ekkor

$$\mathbf{P}(A|B) = \frac{\mathbf{P}(A \cap B)}{\mathbf{P}(B)}$$

az A esemény B feltétel melletti *feltételes valószínűsége*.

Legyen ξ egy diszkrét valószínűségi változó, A egy pozitív valószínűségű esemény. A ξ változónak az A feltételre vonatkozó feltételes várható értéke a következő összeg

$$\mathbf{E}(\xi|A) = \sum_{i=0}^{\infty} k \cdot \mathbf{P}(\xi = k|A),$$

amennyiben az összeg létezik. Ez biztosan teljesül, ha az $\mathbf{E}(\xi)$ várható érték létezik.

1. Tétel (Teljes várható érték tétele)**.** *Legyen $A_1, A_2, \dots, A_n$ teljes eseményrendszer, ξ pedig diszkrét valószínűségi változó, amelynek létezik az $\mathbf{E}(\xi)$ várható értéke. Ekkor*

$$\mathbf{E}(\xi) = \mathbf{P}(A_1)\mathbf{E}(\xi|A_1) + \mathbf{P}(A_2)\mathbf{E}(\xi|A_2) + \dots + \mathbf{P}(A_n)\mathbf{E}(\xi|A_n).$$

A következő két nevezetes egyenlőtlenség a ξ változónak a várható értékétől való (jelentősebb) eltérésének valószínűségére ad korlátot.

2. Tétel (Markov-egyenlőtlenség)**.** *Legyen ξ nemnegatív értékű valószínűségi változó, amelynek létezik az $\mathbf{E}(\xi)$ várható értéke. Legyen λ pozitív valós szám. Ekkor*

$$\mathbf{P}(\xi > \lambda\mathbf{E}(\xi)) \le \frac{1}{\lambda}.$$

A ξ valószínűségi változó szórása a $\mathbf{D}(\xi) = \sqrt{\mathbf{E}((\xi - \mathbf{E}\xi)^2)}$ mennyiség.

3. Tétel (Csebisev-egyenlőtlenség)**.** *Legyen ξ valószínűségi változó, amelynek létezik az $\mathbf{E}(\xi)$ várható értéke és a $\mathbf{D}(\xi)$ szórása is, ami véges és pozitív. Legyen $\lambda > 0$. Ekkor*

$$\mathbf{P}(|\xi - \mathbf{E}(\xi)| \ge \lambda\mathbf{D}(\xi)) \le \frac{1}{\lambda^2}.$$

Az $A_1, \dots, A_n \subseteq \Omega$ események *teljesen függetlenek*, ha minden $1 \le i_1 < \dots < i_k \le n$ esetén

$$\mathbf{P}(A_{i_1} \cap A_{i_2} \cap \dots \cap A_{i_k}) = \mathbf{P}(A_{i_1}) \cdot \mathbf{P}(A_{i_2}) \cdot \dots \cdot \mathbf{P}(A_{i_k}).$$

Az $A_1, \dots, A_n \subseteq \Omega$ események *páronként függetlenek*, ha az előző egyenlőségeket csak $k = 2$-re követeljük meg.

A $\xi_1, \dots, \xi_n$ diszkrét valószínűségi változók *teljesen függetlenek*, ha minden nemnegatív egészekből álló $k_1, \dots, k_n$ sorozatra

$$\mathbf{P}(\xi_1 = k_1, \dots, \xi_n = k_n) = \mathbf{P}(\xi_1 = k_1) \cdot \dots \cdot \mathbf{P}(\xi_n = k_n).$$

A $\xi_1, \dots, \xi_n$ diszkrét valószínűségi változók *páronként függetlenek*, ha minden $i \ne j$ indexpárra ξ_i és ξ_j függetlenek.

Legyenek $\xi_1, \xi_2, \dots, \xi_n$ teljesen független p paraméterű Bernoulli-eloszlású valószínűségi változók. Ekkor a $\xi = \xi_1 + \xi_2 + \dots \xi_n$ összeg (n, p) paraméterű binomiális eloszlású valószínűségi változó:

$$\mathbf{P}(\xi = k) = \binom{n}{k} p^k (1-p)^{n-k}, \qquad k = 0, 1, \dots$$

Legyen $\lambda > 0$ valós szám. Az η valószínűségi változó λ paraméterű *Poisson-eloszlást* követ, ha

$$\mathbf{P}(\eta = k) = \frac{\lambda^k}{k!} e^{-\lambda}, \qquad k = 0, 1, \dots$$

Ekkor $\mathbf{E}(\eta) = \mathbf{D}^2(\eta) = \lambda$.

A következő tételt szokás a *ritka események törvényének* is nevezni:

4. Tétel. *Legyen $\lambda > 0$ és legyen ζ_n egy $(n, \frac{\lambda}{n})$ paraméterű binomiális eloszlású valószínűségi változó ($n = 1, 2, \ldots$). Ekkor tetszőleges k nemnegatív egészre*

$$\lim_{n\to\infty} \mathbf{P}(\zeta_n = k) = \frac{\lambda^k}{k!} e^{-\lambda}.$$

A ζ_n változók eloszlása tehát nagy n-re Poisson-eloszláshoz közelít.

Bizonyítás.

$$\lim_{n\to\infty} \mathbf{P}(\zeta_n = k) = \lim_{n\to\infty} \binom{n}{k} \left(\frac{\lambda}{n}\right)^k \left(1 - \frac{\lambda}{n}\right)^{n-k} =$$

$$= \lim_{n\to\infty} \left(\frac{n!}{n^k (n-k)!}\right) \left(\frac{\lambda^k}{k!}\right) \left(1 - \frac{\lambda}{n}\right)^n \left(1 - \frac{\lambda}{n}\right)^{-k}.$$

Itt az utolsó tényező 1-hez tart, az utolsó előtti tényező határértéke pedig $e^{-\lambda}$. Elég tehát belátni, hogy az első tényező is 1-hez tart:

$$\lim_{n\to\infty} \frac{n!}{n^k (n-k)!} = \lim_{n\to\infty} \frac{n(n-1)\cdots(n-k+1)}{n^k} = \lim_{n\to\infty} 1 \cdot \left(1 - \frac{1}{n}\right) \cdots \left(1 - \frac{k-1}{n}\right) = 1,$$

mert a jobb oldalon csupán konstans sok (nevezetesen k) tényező van, és ezek mindegyike 1-hez tart. □

1. Feladat. *Mutassuk meg, hogy a tétel állítása akkor is érvényben marad, ha csak annyit teszünk fel, hogy ζ_n eloszlása (n, p_n) paraméterű binomiális, és $\lim_{n\to\infty} np_n = \lambda$.*

A következő eredmény egy ún. nagy eltérés típusú egyenlőtlenség. Sok ilyen jellegű, független összegekre vonatkozó becslés ismeretes. (Lásd pl. [GyKKW] A.2. függelékét, ahol egységesen tárgyalnak több, független változók összegére vonatkozó becslést, így a Chernoff-korlát mellett Bernstein és Hoeffding egyenlőtlenségeit is.) Mi az egyik legegyszerűbbet fogjuk használni:

5. Tétel (Chernoff-egyenlőtlenség)**.** *Legyenek az $X_1, \ldots, X_n$ teljesen független p paraméterű Bernoulli-változók. Legyen $S_n = X_1 + \ldots + X_n$, és legyen $1 \geq \epsilon \geq 0$. Ekkor*

$$\mathbf{P}(S_n - np \geq \epsilon np) \leq e^{\frac{-(\epsilon^2 np)}{3}},$$

és

$$\mathbf{P}(S_n - np \leq -\epsilon np) \leq e^{\frac{-(\epsilon^2 np)}{3}}.$$

Érdemes összevetni a Chernoff-egyenlőtlenséget a 2. és a 3. tétellel: a Chernoff-egyenlőtlenség sokkal erősebb korlátot ad egy igen fontos speciális esetben, amikor ξ teljesen független, azonos paraméterű Bernoulli-változók összege.

2. fejezet

Randomizált algoritmusok

Ebben a fejezetben néhány alapvető véletlent használó algoritmust tárgyalunk. Ezeket a módszereket több területről választottuk, az adatrendezéstől a keresőfa-szerkezeteken át a grafikáig és a prímkeresésig. Amellett, hogy önmagukban is fontosak, e módszerek használható első benyomást nyújtanak arról, hogy a véletlen miként fogható munkába hatékony algoritmusok tervezésére.

2.1 Egy klasszikus algoritmus: a gyorsrendezés

1. Számítási feladat. *Adott egy U rendezett halmaz elemeiből való $b_1, b_2, \ldots, b_n$ sorozat. Rendezzük át a sorozatot növekvő (pontosabban: nem csökkenő) $e_1 \leq e_2 \leq \ldots \leq e_n$ sorrendbe.*

A célunk itt, hogy ezt a feladatot *összehasonlítás alapú* rendező módszerrel oldjuk meg. Egy összehasonlítás alapú rendező algoritmus csak $b_i ? b_j$ alakú kérdésekkel szerezhet információt a bemenő adatokról. Egy ilyen összehasonlításnak kétféle kimenetele lehet: $b_i \leq b_j$, vagy $b_i > b_j$. A rendezés *költsége* legyen az összehasonlítások száma. A célunk tehát a rendezési feladat megoldása minél kevesebb összehasonlítással.[1]

Ismert ([RISz], 2.2.3.), hogy egy jó összehasonlítás alapú rendező algoritmus n hosszú input esetén legalább $\log_2 n!$ összehasonlítást végez.[2] A Stirling-formulából adódó

$$\log_2 n! \approx n(\log_2 n - 1,442) + O(n)$$

közelítéssel számolva legalább mintegy $n \log_2 n$ összehasonlításra biztosan szükség van. Az igazán hatékony módszerek (ilyen pl. az összefésüléses rendezés és a kupacos rendezés) $O(n \log n)$ összehasonlítással megoldják a rendezési feladatot.

A *gyorsrendezés* (Hoare, 1962) legrosszabb esetben $O(n^2)$ összehasonlítással dolgozik. Mint azt később látni fogjuk, a gyorsrendezés várható költsége $O(n \log n)$ összehasonlítás. Gyakorlati szempontból is igen jó módszernek számít, valójában az egyszerűbb általános módszerek közül ezt tekintik a bajnoknak. Az algoritmus vázlata abban az esetben, amikor az input sorozat az $A[1, n]$ tömbben van:

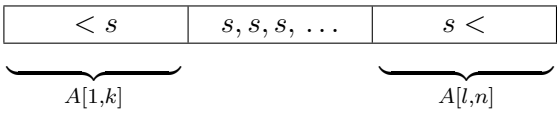
GYORSREND($A[1, n]$)
  PARTÍCIÓ(s)
  GYORSREND($A[1, k]$)

[1] Megjegyezzük, hogy az adatrendezési módszerek és más rokon eljárások esetén jó hatékonysági mérőszám az összehasonlítások száma; a többi költség általában arányos ezzel.

[2] A korlát a legrosszabb esetre vonatkozik. Később itt foglalkozunk az átlagos esetre vonatkozó korlátokkal.

GYORSREND($A[l,n]$)
vége

Az algoritmus lelke a PARTÍCIÓ(s) eljárás, amely először kiválaszt egy véletlen tömbelemet (az input A tömb mindegyik eleme egyenlően valószínű). Legyen ez az elem s. Ezután A-t három részre osztja fel. Az első részbe kerülnek az s-nél kisebb, a középsőbe az s-sel egyenlő, a harmadikba az s-nél nagyobb A-beli elemek:



PARTÍCIÓ(s) megvalósítható $n-1$ kulcs-összehasonlítással, egyszerűen minden más elemet össze kell hasonlítani a kiválasztott s elemmel. Az alábbi vázlatot követve tehetjük ezt meg hatékonyan:

PARTÍCIÓ(s)
  $i=1, j=n$
  ciklus amíg $i<j$
    ha $A[i]<s \to i++$
    ha $A[j]\geq s \to j--$
    ha $A[i]\geq s$ és $A[j]<s \to A[i]$ és $A[j]$ cseréje, $i++, j--$
  ciklus vége
vége

A gyorsrendezés várható költsége

A továbbiakban feltesszük, hogy az $A[i]$ elemek mind különbözők. Legyen e_i a tömb nagyság szerint i-edik eleme ($i=1,2,\ldots,n$).

Az algoritmus egy végrehajtásának a költségén a felmerülő kulcs-összehasonlítások számát értjük. Ez a mennyiség tekinthető egy ξ valószínűségi változónak, ami az s (véletlen) particionáló elemek választásától függ. Az $\mathbf{E}(\xi)$ várható érték a várható költség. Az egyenletes eloszlás szerinti választások miatt ezt úgy is tekintjük, hogy az összes lehetséges futások költségének átlagát kell vennünk.

Szemlélhetjük másképp is az algoritmust: gondolkodhatunk úgy, hogy az $a_1, \ldots, a_n$ elemek véletlen bemeneti sorrendjére nézve – ahol minden sorrend egyenlően valószínű – keressük a költség várható értékét. Ebben az esetben a particionáló s elem mindig a kérdéses résztömb legelső eleme.

Legyen $C(n)$ a várható költség $A[1,n]$-re, továbbá legyen $C(n,i)$ a várható költség abban az esetben, amikor az e_i-t, az i-edik legnagyobb elemet választottuk első s-nek. Úgy is fogalmazhatunk a fentiek alapján, hogy $C(n)=\mathbf{E}(\xi)$ és $C(n,i)=\mathbf{E}(\xi|A_i)$, ahol A_i jelöli azt az eseményt, hogy e_i lesz az első particionáló elem.

Ekkor teljesülnek a következő összefüggések:

$$C(n)=\frac{1}{n}\Big(C(n,1)+C(n,2)+\ldots+C(n,n)\Big), \tag{2.1}$$

$$C(n,i)=\underbrace{n-1}_{\text{PARTÍCIÓ}(s)}+\underbrace{C(i-1)}_{\text{GYORSREND}(A[1,k])}+\underbrace{C(n-i)}_{\text{GYORSREND}(A[l,n])}, \tag{2.2}$$

$$C(0)=C(1)=0.$$

Az első egyenlőség azért igaz, mert, minden elem $\frac{1}{n}$ valószínűséggel lesz első particionáló elem, a második pedig a gyorsrendezés rekurziójából, és particionálás algoritmusából olvasható ki.

Ezután (2.1)-be (2.2)-t sokszor beírjuk:

$$C(n) = n - 1 + \frac{2}{n}\Big(C(n-1) + C(n-2) + \ldots + C(1)\Big),$$

amiből n-nel való szorzás után kapjuk, hogy

$$nC(n) = n(n-1) + 2\Big(C(n-1) + C(n-2) + \ldots + C(1)\Big), \tag{2.3}$$

majd ugyanezt n helyett $n-1$-re is felírjuk:

$$(n-1)C(n-1) = (n-1)(n-2) + 2\Big(C(n-2) + C(n-3) + \ldots + C(1)\Big). \tag{2.4}$$

Ezután (2.3)-ból kivonjuk (2.4)-et:

$$\begin{aligned} nC(n) &= 2(n-1) + (n+1)C(n-1), \\ \frac{C(n)}{n+1} &= \frac{2(n-1)}{n(n+1)} + \frac{C(n-1)}{n}, \\ \frac{C(n)}{n+1} &< \frac{2}{n} + \frac{C(n-1)}{n}. \end{aligned} \tag{2.5}$$

Ezt ismételten önmagába helyettesítjük:

$$\frac{C(n)}{n+1} < \frac{2}{n} + \frac{C(n-1)}{n} < \frac{2}{n} + \frac{2}{n-1} + \frac{C(n-2)}{n-1} < \ldots .$$

Végül azt kapjuk, hogy

$$\frac{C(n)}{n+1} < 2\left(\frac{1}{n} + \frac{1}{n-1} + \frac{1}{n-2} + \ldots + \frac{1}{2} + 1\right) = 2H_n,$$

ahol

$$H_n = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n}$$

az n-edik *harmonikus szám.*

2. Feladat. *Mutassuk meg, hogy $H_n \le \log n + 1$! (Tekintsük a H_n számot az $\int_1^n \frac{1}{x}\,\mathrm{d}x$ integrál közelítő összegének.)*

Megjegyezzük, hogy a feladatban foglalt állításnál erősebb is igaz: H_n közelíthető az $\log n + \gamma$ kifejezéssel, ahol γ egy 0,5 és 0,6 közötti konstans. A feladat becslését alkalmazva:

$$\begin{aligned} C(n) < 2(n+1)H_n &\le 2(n+1)(\log n + 1) \\ &= 2n\log n + O(n) \\ &\approx 1,39n\log_2 n + O(n). \end{aligned}$$

$C(n)$-re ezzel egy igen kedvező felső korlátot kaptunk.

Nézzünk egy másféle levezetést, amely a valószínűségszámítás néhány egyszerű eszközével közelíti meg a problémát. Emlékeztetünk rá, hogy a ξ valószínűségi változó értéke az összehasonlítások száma $A[1,n]$ rendezésekor (a véletlen az s elem választásában van). A $C(n)$ költség

a ξ várható értékével lesz egyenlő: $C(n) = \mathbf{E}(\xi)$. Definiáljuk az X_{ij} indikátorváltozókat $i < j$-re a következőképpen:

$$X_{ij} = \begin{cases} 1, & \text{ha a rendezés során valamikor } e_i\text{-t és } e_j\text{-t összehasonlítjuk;} \\ 0, & \text{egyébként.} \end{cases}$$

A ξ felírható az X_{ij} változók összegeként:

$$\xi = \sum_{i<j} X_{ij},$$

ebből a várható értékre

$$\mathbf{E}(\xi) = \sum_{i<j} \mathbf{E}(X_{ij})$$

adódik. Itt kihasználtuk, hogy *a várhatóérték-operátor lineáris.*

1. Állítás. *Tegyük fel, hogy $i < j$. Ekkor $X_{ij} = 1$ pontosan akkor lesz igaz, ha a*

$$H = \{e_i, e_{i+1}, \ldots, e_{j-1}, e_j\}$$

halmazból e_i vagy e_j lesz a legelső particionáló elem.

Bizonyítás. Az algoritmus futása során mindig az aktuális particionáló elemet hasonlítjuk össze az aktuális résztömb összes többi elemével. Amíg tehát a $H = \{e_i, e_{i+1}, \ldots, e_j\}$ halmazból nem választottunk particionáló elemet, addig nem is volt ezen elemek közti összehasonlítás. Ha ezek közül először e_i-t választjuk, akkor ezzel a H összes többi elemét, így e_j-t is összehasonlítjuk. Hasonlót mondhatunk, ha e_j az első particionáló elem. Ha viszont először egy e_l elemmel particionálunk, ahol l nem az i, j valamelyike, akkor ennél a vágásnál e_i és e_j két különböző résztömbbe kerül, és ezért nem fogjuk őket ezután sem összehasonlítani. □

Az előző állításból következik, hogy

$$\mathbf{P}(X_{ij} = 1) = \frac{2}{j - i + 1},$$

mivel H-ból minden elemet ugyanakkora eséllyel választunk ki először particionáló elemnek. Vegyük észre, hogy egy 0-1 lehetséges értékű η valószínűségi változónak a várható értéke $\mathbf{E}(\eta) = \mathbf{P}(\eta = 0) \cdot 0 + \mathbf{P}(\eta = 1) \cdot 1 = \mathbf{P}(\eta = 1)$. Az X_{ij} változók is ilyenek, tehát

$$\mathbf{E}(X_{ij}) = \frac{2}{j - i + 1},$$

innen:

$$\mathbf{E}(\xi) = \sum_{i<j} \frac{2}{j - i + 1} = \sum_{i=1}^{n} \sum_{\ell=1}^{n-i+1} \frac{2}{\ell} \leq 2nH_n.$$

Ugyanazt a felső korlátot kaptuk, mint a korábbi módszerrel.

3. Feladat. *Adjunk felső becslést annak az eseménynek a valószínűségére, hogy $\xi > 4nH_n$.*

Megjegyzések

1. Megfigyelhetjük, hogy (2.1)-ben a teljes várható érték tétele jelenik meg a $C(n) = \mathbf{E}(\xi)$, és $\mathbf{E}(\xi|A_i) = C(n,i)$ helyettesítések után (ahol az A_i esemény az, hogy e_i az első particionáló elem).

2. A véletlen választás általánosságban akkor előnyös, ha a választási tartományban sok olyan elem van, amilyet keresünk. Nem jó a véletlen választás, ha a kívánatos elemek száma kicsi (ha például keresünk valakit a telefonkönyvben, akkor nem az lesz a jó módszer, hogy véletlenül választunk egy nevet, és megnézzük, hogy ő volt-e a keresett személy).

3. A gyorsrendezésnél az olyan s a jó particionáló elem, ami nincs nagyon a rendezett sorozat szélén. Olyan s kell, amire mindkét keletkező részfeladat „elég nagy" lesz. Nem túl szélső elemből sok van, pl. $a_{\frac{n}{4}}$ és $a_{\frac{3n}{4}}$ között kb. $\frac{n}{2}$ elem van, amelyek jó partíciót adnak.

4. Megemlítjük, hogy létezik $O(n)$ idejű mediánkereső determinisztikus algoritmus, vagyis az $e_{\lfloor \frac{n}{2} \rfloor}$ középső elemet lineáris számú összehasonlítással meg tudjuk találni (lásd pl. *Algoritmusok*, 2.2.7.). Az így választott s is $O(n \log n)$-es futási időt eredményez (a legrosszabb esetben is), de a gyakorlatban ez nem versenyképes a gyorsrendezéssel szemben.

5. Érdekes tapasztalati tény, hogy többnyire az *álvéletlen választásokkal*, vagyis a számítógépek véletlenszám-generátoraival dolgozó algoritmusok is igen jó eredményeket adnak a randomizált eljárások futtatásakor.

Erősebb felső korlát a gyorsrendezés idejére

Az előzőekben a gyorsrendezés költségének várható értékét vizsgáltuk, és egy jó felső korlátot adtunk. A költség várható értéke gyakran elegendő információ egy randomizált algoritmus működéséről. Vannak azonban kritikus alkalmazások, amikor ezen felül többet szeretnénk, például valamiféle garanciát arra, hogy a költség nem lesz túl gyakran sokkal nagyobb a várható értéknél. Ennek pontosabb megfogalmazására hasznosnak bizonyult a következő definíció:

1. Definíció. *Legyen $\mathcal{A}$ egy véletlent használó algoritmus. Tegyük fel, hogy az $\mathcal{A}$ költségének a várható értéke az n hosszú inputokon legfeljebb $f(n)$. Azt mondjuk, hogy az $\mathcal{A}$ költsége* nagy valószínűséggel $O(f(n))$, *ha vannak olyan $c, d > 0$ számok, hogy minden n hosszú bemeneten*

$$\mathbf{P}\left(\mathcal{A} \text{ költsége } > cf(n)\right) \leq \frac{1}{n^d}.$$

A gyorsrendezés ebben az erősebb értelemben is szépen viselkedik. Részben ez magyarázza a gyakorlatban tapasztalt hatékonyságát. Érvényes a következő:

6. Tétel. *A gyorsrendezés költsége nagy valószínűséggel $O(n \log n)$.*

A tételt nem bizonyítjuk, elsősorban a benne foglalt számolások nehézkessége miatt, viszont vázolunk egy lehetséges gondolatmenetet. (Itt olvasható részletesebb bizonyítás: [J]. Még erősebb eredményt található itt: [DH].)

Vizsgáljuk a gyorsrendezés futását n hosszú bemeneteken. Képzeljük el, hogy a programot rekurzív hívások alkalmazásával írtuk meg. A program futása egy bináris fával írható le, amelynek a csúcsaiban a rendezendő b_i elemek vannak. A gyökerében az első particionáló elem s foglal helyet, a bal részfa felel meg az alsó résztömbnek, a jobb részfa pedig az s-nél nagyobb elemeket

tartalmazó résztömbnek. Egy részfa gyökerében levő tömbelem a részfát jelentő résztömb particionálására választott (véletlen) elem. Így szemlélve a módszert elegendő belátni, hogy alkalmas $c, d > 0$ állandókkal igaz lesz, hogy a fa szintjeinek száma csak legfeljebb $\frac{1}{n^d}$ valószínűséggel lehet nagyobb, mint $c \log n$. Ebből már adódik a korlát az összköltségre, hiszen egy szinten az összes munka $O(n)$.

Elég tehát a szintszám korlátozásával foglalkozni. A fa egy csúcsát nevezzük *szerencsésnek*, ha az ottani s' particionáló elem a csúcshoz tartozó S kulcshalmazt úgy osztja fel S_1 és S_2 részekre, hogy $|S|/4 \leq |S_1| \leq 3|S|/4$ és $|S|/4 \leq |S_2| \leq 3|S|/4$ is teljesül. Ez éppen akkor történik így, ha S-ben van legalább $|S|/4$ elem, ami nagyobb s'-nél, és van legalább $|S|/4$ elem S-ben, ami kisebb s'-nél. Annak a valószínűsége tehát, hogy egy csúcs nem szerencsés, legfeljebb $\frac{1}{2}$.

Legyen x a fa egy tetszőleges csúcsa. Hány szerencsés csúcs lehet a fa gyökerétől az x-ig vezető úton? Ha M ilyen csúcs van, akkor $M \leq \log_{4/3} n \leq 4 \log n$, hiszen egy szerencsés csúcsnál a résztömbök legfeljebb 3/4-szeresükre zsugorodnak.

Legyenek $x_1, \ldots, x_m$ különböző csúcsok az x-től a fa gyökeréig vezető úton. Az x_i csúcshoz rendeljük az X_i valószínűségi változót, amelynek értéke 1, ha x_i nem szerencsés, és 0, ha x_i szerencsés. Az X_i változók teljesen függetlenek, és $\mathbf{P}(X_i = 1) \leq \frac{1}{2}$. Az X_i változók összegét egyszerűbben áttekinthető Y_i változók segítségével vizsgáljuk:

4. Feladat. *Mutassuk meg, hogy vannak olyan teljesen független Y_i valószínűségi változók $i = 1, \ldots, m$, amelyekre $\mathbf{P}(Y_i = 0) = \mathbf{P}(Y_i = 1) = \frac{1}{2}$, $X_i \leq Y_i$. Következésképpen minden r valós számra $\mathbf{P}(\sum X_i \geq r) \leq \mathbf{P}(\sum Y_i \geq r)$.*

Az előző feladatbeli Y_i változók $\frac{1}{2}$ paraméterű, teljesen független Bernoulli-változók. Az összegükre alkalmazható a Chernoff-egyenlőtlenség ($\epsilon = \frac{2}{3}$):

5. Feladat. *Mutassuk meg, hogy*

$$\mathbf{P}\left(\sum_{i=1}^{m} Y_i \geq \frac{5}{6}m\right) \leq \frac{1}{e^{\frac{2m}{27}}}.$$

Ezek után beláthatjuk, hogy legfeljebb $\frac{1}{n^{16/9}}$ annak a valószínűsége, hogy az x csúcs mélysége (a gyökérig vezető úton a csúcsok száma) legalább $m = \lceil 24 \log n \rceil$. Ugyanis az úton levő $x_1, \ldots, x_m$ csúcsok közül legfeljebb $4 \log n$ lehet szerencsés, amiből $\sum_{i=1}^{m} X_i \geq m - 4 \log n \geq \frac{5}{6}m$. A feladatok állításait alkalmazva

$$\mathbf{P}\left(\sum_{i=1}^{m} X_i \geq \frac{5}{6}m\right) \leq \mathbf{P}\left(\sum_{i=1}^{m} Y_i \geq \frac{5}{6}m\right) \leq \frac{1}{n^{\frac{16}{9}}}.$$

Annak a valószínűsége, hogy a fában van $\lceil 24 \log n \rceil$ mélységű csúcs, legfeljebb $n \cdot \frac{1}{n^{\frac{16}{9}}} = \frac{1}{n^{\frac{7}{9}}}$. Legfeljebb ennyi a valószínűsége, hogy a gyorsrendezés költsége $\geq n\lceil 24 \log n \rceil$.

Bináris keresőfa naiv beszúrásokkal

Ha egy bináris fának l szintje van, akkor a csúcsok számára az $n \leq 1 + 2 + 2^2 + \cdots + 2^{l-1} = 2^l - 1$ becslés adódik, amiből $l \geq \log_2(n+1)$. A keresés szempontjából tehát a legjobb – azaz legkisebb – szintszám, amit n-pontú fánál elérhetünk, körülbelül $\log_2 n$.

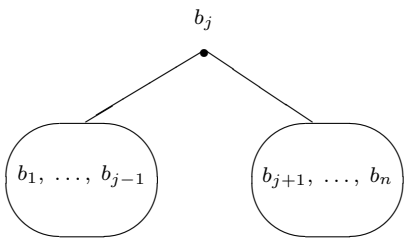
A következőkben érvet mutatunk amellett, hogy a naiv beszúrásokkal épített fák átlagos értelemben nem rosszak; egy beillesztés átlagosan $O(\log_2 n)$ összehasonlításba kerül. A pontos

modell a következő: üres fával kezdjük az algoritmust, a $b_1 < b_2 < \cdots < b_n$ kulcsok egy *véletlen* $a_1, a_2, \ldots, a_n$ sorrendben jönnek; ezeket kell beszúrni naiv módon, azaz minden új elemet levélbe teszünk, és az eddigi fát módosítatlanul hagyjuk (így pl. a_1 lesz mindig a gyökérben). A költségnek most is a kulcs-összehasonlítások számát tekintjük, és a várható (másként mondva: átlagos) költség érdekel bennünket. Az előbbi rendezési feladatra adott elemzésünk itt is használhatónak bizonyul.

Az $n!$ lehetséges sorrendre vett átlagos költséget jelöljük $T(n)$-nel, és $T(n,j)$-vel az olyan fa átlagos költségét, ahol $a_1 = b_j$. $T(n)$-re és $T(n,j)$-re ugyanazok az összefüggések érvényesek, mint $C(n)$-re és $C(n,j)$-re:

$$\begin{aligned} T(n) &= \frac{1}{n}\Big(T(n,1) + T(n,2) + \ldots + T(n,n)\Big), \\ T(n,j) &= n - 1 + T(j-1) + T(n-j), \\ T(0) &= T(1) = 0. \end{aligned}$$

Az első összefüggés azért lesz igaz, mert mindegyik b_j ugyanazzal az $\frac{1}{n}$ valószínűséggel lesz a_1 (vagyis a fa gyökéreleme). Ha pedig $a_1 = b_j$, akkor a bal részfába $j-1$ csúcs kerül, a jobb részfába pedig $n-j$. Innen kapjuk a második formulát.



A korábbi, a $C(n)$-re adott érvelés alapján a $1,39n \log_2 n + O(n)$ felső korlát igaz $T(n)$-re is. Elmondhatjuk tehát, hogy az egy beszúrásra eső átlagos költség $O(\log n)$. Azt a következtetést vonhatjuk le, hogy átlagos fára a naiv beszúrásos algoritmus is jó futási idővel rendelkezik.

2.2 Alsó becslések rendező algoritmusokra

Ahogy már utaltunk rá, egy összehasonlítás alapú determinisztikus rendező módszer n elemből álló bemenet esetén a legrosszabb esetben legalább $\log_2 n!$ összehasonlítást végez. Ennél lényegesen erősebbet állít a következő:

7. Tétel. *Egy determinisztikus összehasonlítás alapú $\mathcal{A}$ rendező algoritmus n elemből álló bemenet esetén átlagosan legalább $\lfloor \log_2 n! \rfloor$ összehasonlítást végez.*

Az állítás úgy értendő, hogy az $e_1, \ldots, e_n$ elemek minden egyes sorrendjére mint bemenetre (összesen $n!$ ilyen sorrend van) nézzük az algoritmus összehasonlításainak a számát, és vesszük ezeknek a számoknak az átlagát.

Bizonyítás. Építsünk bináris fát,[3] amelynek csúcsai az $\mathcal{A}$ algoritmus összehasonlításainak felelnek meg. A fa gyökere az $\mathcal{A}$-ban szereplő első összehasonlítás. Az *igen* válasz felel meg a bal részfának, a *nem* a jobb részfának. A bal részfa gyökere második összehasonlítás azon futások esetén, amelyeknél az első eredménye *igen*, és így tovább. A fa leveleihez nem tartozik összehasonlítás. A fa minden egyes x csúcsához bemeneti sorrendek halmazát rendelhetjük: azokat a

[3] Ez egy ún. *döntési fa*, amely az $\mathcal{A}$ algoritmus futása során fellépő döntéseket írja le.

sorrendeket, amelyekkel indulva az algoritmus eljut az x csúcsba. Így a gyökérhez még mind az $n!$ sorrend hozzárendelhető. A bal fiához már csak azok, ahol az első összehasonlításra a válasz *igen*. A fából törölhetők azok a csúcsok, amelyekhez nem tartozik input permutáció.

A fának ekkor minden leveléhez más, a többi levélétől különböző inputsorrend tartozik. Igazolásul nézzük meg, mi történne ellenkező esetben, ha egy levélhez a $\pi_1 \neq \pi_2$ sorrendek tartoznának. Mivel az algoritmus pontosan ugyanazt az információt tudja a két sorozatról, a növekvő sorrendbe való átrendezés lépései is megegyeznének. Ez pedig nem lehet, hiszen a két bemeneti sorozat különböző.

A fának tehát $n!$ levele van. Egy π bemenet esetén a költség éppen a gyökértől a π címkéjű levélig vezető út éleinek a száma. Az átlagos költség így a gyökér-levél utak hosszösszegének az $\frac{1}{n!}$-szorosa.

Belátjuk most, hogy adott k levélszám mellett a hosszösszeg akkor minimális, ha a fa teljes: minden nem levél csúcsnak két fia van, és a levelek a szomszédos $\lfloor \log_2 k \rfloor$. és $\lceil \log_2 k \rceil$. szinteken helyezkednek el. Ugyanis ha egy belső csúcsnak csak egy fia van, akkor ez a fiú törölhető, és egyetlen részfájának a gyökere tehető a helyére. Ha pedig vannak levelek a d. és egy $\geq d+2$. szinten is, akkor erről az utóbbi szintről egy levél átköthető a d. szintre, ami nem növeli a hosszösszeget. Ha egy teljes fa legalsó szintjének a sorszáma t (a gyökér a nulladik szint), akkor a leveleinek k számára igaz, hogy $2^t \geq k > 2^{t-1}$, ahonnan $t = \lceil \log_2 k \rceil$.

Az $\mathcal{A}$ algoritmusból kapott fa esetén a hosszösszeg tehát legalább $n!\lfloor \log_2 n! \rfloor$, ezért az átlagos hossz legalább $\lfloor \log_2 n! \rfloor$. □

A tétel általánosítható randomizált rendező algoritmusokra, amilyen a gyorsrendezés is.

8. Tétel. *Egy véletlent használó összehasonlítás alapú $\mathcal{A}$ rendező algoritmus várható lépésszámának az átlaga az n elemből álló bemenetekre legalább $\lfloor \log_2 n! \rfloor$.*

A tétel állítása úgy értendő, hogy minden rögzített π inputsorrendre vesszük az összehasonlítások számának várható értékét, majd ezek átlagát képezzük az összes lehetséges $n!$ bemeneti sorrendre. A tétel szerint a gyorsrendezés költsége konstans szorzó erejéig optimális a randomizált módszerek körében is.

Bizonyítás. A bizonyítás lényege, hogy az $\mathcal{A}$ randomizált algoritmust úgy foghatjuk fel, mint egy valószínűségeloszlást determinisztikus rendező algoritmusokon. $\mathcal{A}$-t olyan determinisztikus módszernek tekintjük, amelynek a természetes inputján kívül van még egy bemenete, ami egy w bitsorozat. Ha $\mathcal{A}$ a futása során egy véletlen bitet szeretne, akkor innen veszi a következő, még fel nem használt bitet. Az $\mathcal{A}$ minden egyes futása felfogható egy adott w bitvektort használó $\mathcal{A}_w$ determinisztikus rendező algoritmus futásának. Úgy szemlélhetjük a dolgot, hogy a w-t beépítettük az algoritmusba. A π inputon az algoritmus várható költsége ezek után

$$\sum_w \mathbf{P}(w)(\text{az } \mathcal{A}_w \text{ költsége } \pi\text{-n}).$$

Itt az összegezés az $\mathcal{A}$ futása során előálló összes lehetséges véletlen bitsorozatra történik. Ezeket a mennyiségeket átlagolni kell a π bemenetekre:

$$\frac{1}{n!}\sum_\pi \sum_w \mathbf{P}(w)(\text{az } \mathcal{A}_w \text{ költsége } \pi\text{-n}) = \frac{1}{n!}\sum_w \sum_\pi \mathbf{P}(w)(\text{az } \mathcal{A}_w \text{ költsége } \pi\text{-n}) =$$

$$= \sum_w \mathbf{P}(w) \sum_\pi \frac{1}{n!}(\text{az } \mathcal{A}_w \text{ költsége } \pi\text{-n}) \geq \sum_w \mathbf{P}(w)\lfloor \log_2 n! \rfloor = \lfloor \log_2 n! \rfloor.$$

Az egyenlőtlenségnél a determinisztikus $\mathcal{A}_w$ algoritmusokra alkalmaztuk az előző tételt. □

2.3 Nagy prímszám keresése

A következő feladat fontos szerepet játszik a kriptográfiában.[4] Egyebek között a nevezetes RSA-kódolás egyik alapjának tekinthető.

2. Számítási feladat. *Adott az n pozitív egész szám. Találjunk n bites prímszámot.*

A praktikus alkalmazásoknál az n értéke több száz is lehet. Az ötlet egyszerű: válasszunk egy véletlen n-bites egészet – egyenletes eloszlás szerint –, és vizsgáljuk meg, hogy prím-e; szokásos szakkifejezéssel: vessük alá prímtesztnek. A módszer gyakorlati alkalmazhatósága szempontjából fontos, hogy a prímtalálás valószínűsége ne legyen túl kicsi. Ennek becslésével foglalkozunk a következőkben.

Alapvető eszköz itt a következő híres (és nehéz) számelméleti tétel.[5] Jelölje $\pi(x)$ az $[1, x]$ intervallumban a prímek számát, ahol x pozitív valós szám.

9. Tétel (Prímszámtétel)**.**

$$\lim_{x\to\infty} \frac{\pi(x)}{\frac{x}{\ln x}} = 1.$$

A tételt közelítő számolás során úgy alkalmazzuk, hogy az $\frac{x}{\ln x}$ értéket $\pi(x)$ közelítésének tekintjük:

$$\pi(x) \approx \frac{x}{\ln x}.$$

Az n bites természetes számok éppen a $[2^{n-1}, 2^n - 1]$ intervallum egészei. A prímek száma ezen intervallumban közelítőleg:

$$\begin{aligned}
&\pi(2^n) - \pi(2^{n-1}) \approx \\
&\approx \frac{2^n}{\ln 2^n} - \frac{2^{n-1}}{\ln 2^{n-1}} = \frac{2^{n-1}}{\ln 2}\left(\frac{2}{n} - \frac{1}{n-1}\right) = \frac{2^{n-1}}{\ln 2} \cdot \frac{2n-2-n}{n(n-1)} = \frac{2^{n-1}}{\ln 2} \cdot \frac{n-2}{n(n-1)} \approx \\
&\approx \frac{2^{n-1}}{\ln 2} \cdot \frac{1}{n}.
\end{aligned}$$

A véletlen prímtalálás empirikus valószínűsége tehát:

$$\frac{\text{kedvező esetek}}{\text{összes eset}} = \frac{\pi(2^n) - \pi(2^{n-1})}{2^{n-1}} \approx \frac{1}{n \ln 2}.$$

Például az innen adódó valószínűség $n = 300$-ra hozzávetőlegesen $\frac{1}{0{,}6932 \cdot 300} \approx \frac{1}{208}$, várhatóan tehát 208 próbálkozásból kapunk prímet. Általában igaz, hogy $[1, m]$-ben körülbelül $\frac{1}{\ln m}$ eséllyel találunk prímet.

Ezek után a számítási feladatot megoldó eljárás igen egyszerű: választunk egy véletlen m egész számot a $[2^{n-1}, 2^n - 1]$ intervallumból, majd prímtesztet[6] végzünk rajta.

A prímszámtétel szerint a prímek elég sűrűn vannak, ezért a módszer várhatóan n-ben lineáris számú m választásával talál prímet.

[4] A kriptográfia a [BV] műben olvashatunk.

[5] A [Z] dolgozatban viszonylag rövid bizonyítást és történeti áttekintést is találhatunk. A számelmélet alapjait illetően lásd [FGy].

[6] A prímteszt olyan algoritmus, amely ellenőrzi, hogy a bemenete prímszám-e. A prímtesztelés témájával később még találkozunk a bonyolultsági osztályokról szóló fejezetben a Rabin–Miller-algoritmus kapcsán.

A számolásunk során elhanyagoltuk a Prímszámtételben foglalt közelítés hibáját. Vannak a $\pi(x)$-re vonatkozó egyszerű egyenlőtlenségek, amelyekkel a prímtalálás valószínűségére bizonyosan igaz (bár aszimptotikus értelemben kevésbé pontos) korlátok kaphatók. Ilyen például az alábbi két egyenlőtlenség:[7]

$$\pi(x) > \frac{x}{\ln x}, \quad \text{ha } x \geq 17,$$
$$\pi(x) < 1,26\frac{x}{\ln x}, \quad \text{ha } x > 1.$$

2.4 Ponthalmaz konvex burkának számítása

Itt a számítógépes grafika egyik alapfeladatával, a konvex burok számításával foglalkozunk. Emlékeztetünk rá, hogy $\mathbb{R}^n$ jelöli az n dimenziós valós teret, így $\mathbb{R}^2$ a síkot, $\mathbb{R}^3$ pedig a teret.

A $P \in \mathbb{R}^n$ pontokat n komponensű valós vektoroknak tekinthetjük. Ha P és Q két pont, akkor a $[P,Q]$ összekötő szakaszuk pontjait a két vektor megfelelő lineáris kombinációi adják: $[P,Q] = \{t \cdot P + (1-t) \cdot Q, \quad \text{ahol } t \in [0,1]\}$.

2. Definíció. *A $H \subseteq \mathbb{R}^n$ konvex halmaz, ha $P, Q \in H$ esetén a H a $[P,Q]$ szakaszt is tartalmazza.*

Ilyenek például a körlemez, a gömb, a kocka, a félsík.[8] A definíció közvetlen és hasznos következménye az alábbi tény:

2. Állítás. *Konvex halmazok metszete is konvex.*

3. Definíció. *Tetszőleges $H \subseteq \mathbb{R}^n$-re a H konvex burka a H-t tartalmazó $\mathbb{R}^n$-beli konvex halmazok metszete.*

Az előző észrevétel szerint a konvex burok konvex halmaz. Érvényes a következő (nem bizonyítjuk):

3. Állítás. *Zárt $H \subseteq \mathbb{R}^2$ konvex burka a H-t tartalmazó zárt félsíkok metszete.*

Az állítás könnyen adódik a következő tényből:

6. Feladat. *Legyen $H \subset \mathbb{R}^2$ egy zárt halmaz, és P egy pont, ami nincs H-ban. Ekkor van olyan ℓ egyenes, hogy H az ℓ által határolt egyik nyílt félsíkban van, P pedig a másikban.*

Az állítás és a feladat könnyen általánosítható magasabb dimenziókra. Például a térbeli állításban az egyenes helyett sík, a félsík helyett pedig féltér szerepel. Mi most csak síkbeli alakzatokkal foglalkozunk.

3. Számítási feladat. *Adott egy véges $H = \{P_1, P_2, \ldots, P_n\}$ ponthalmaz $\mathbb{R}^2$-ből, ami általános helyzetű (nincs köztük 3 pont, ami egy egyenesre esik). Keressük a H ponthalmaz konvex burkát.*

[7] Más hasonló formulákkal együtt ezek is megtalálhatók a [BS] monográfiában.

[8] Félsíkon a $P \in \mathbb{R}^2$; $\ell(P) \geq 0$ alakú alakzatokat értjük, ahol ℓ kétváltozós, nem azonosan 0 lineáris függvény. Ez a félsík azon két tartomány egyike, amelyekre az $\ell(P) = 0$ egyenletű egyenes osztja a síkot.

Az előző állítást használva látjuk, hogy a keresett burok egy olyan sokszöglemez, amelynek a határát a P_i pontok közül bizonyosakat összekötő szakaszok alkotják. Szemléletesen szólva ez a határ a pontokat bekerítő legrövidebb kerítés lesz. A feladat megoldásához elég megadni a határon levő P_i pontokat, a határ egy végigjárásának sorrendjében.

A naiv megoldó algoritmus (vázlatos) menete: tegyük fel, hogy a $\{P_1, \dots P_{i-1}\}$ ponthalmazra már megoldottuk a feladatot. Ekkor megvizsgáljuk, hogy a P_i pont az eddigi kerítés mely szakaszainak van a rossz oldalán. Ha a $[P, Q]$ szakasz éle a kerítésnek, és $\ell(x, y) = 0$ a P és Q pontokon átmenő egyenes egyenlete, akkor a P_i pontosan akkor van a $[P, Q]$ szakasz rossz oldalán (a kerítésen kívüli oldalán), ha $\ell(P_i)$ előjele különbözik az $\ell(P_j)$ előjelétől, ahol $1 \leq j < i$ és $P_j \notin [P, Q]$. Ha P_i a $[P, Q]$ szakasz rossz oldalán van, akkor a $[P, Q]$ szakasz nem lesz része az új kerítésnek. Az ilyen szakaszok helyére alkalmas P' és P'' pontokkal a $[P', P_i]$ és $[P_i, P'']$ szakaszok kerülnek. Egy ilyen növelés $O(n)$ aritmetikai művelettel (és összehasonlítással) megoldható, így az összes költség $O(n^2)$ lesz. Megemlítjük, hogy létezik $O(n \log n)$ idejű determinisztikus algoritmus is.

Egy $O(n \log n)$ várható idejű randomizált módszer

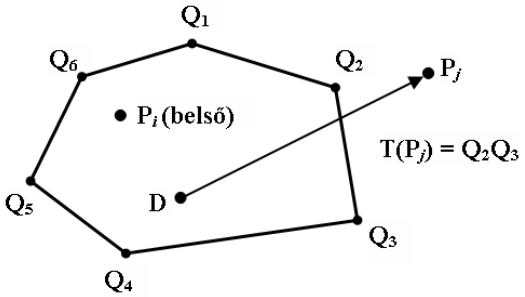
Itt egy hatékony, véletlent használó algoritmust mutatunk be a konvex burok számítására. Kiindulásul veszünk H-ből 3 pontot véletlenül (legyenek ezek P_1, P_2, és P_3). Jelölje Δ a $P_1P_2P_3$ háromszöget. Felveszünk e mellé még egy $D \in \Delta$ pontot. Megadunk egy a H-n értelmezett T leképezést, amit az algoritmus futása során karbantartunk, az inverzével együtt. A $P_j \in H$ ponton legyen

$$T(P_j) = \begin{cases} \text{„belső”}, & \text{ha } P_j \in \Delta \\ \Delta \text{ azon éle, melyet a } [D, P_j] \text{ szakasz metsz}, & \text{ha } P_j \notin \Delta \end{cases}$$

A T inverze a Δ egy e élére megadja azon $P_j \in H$ pontokat, amelyekre $T[P_j] = e$.

7. Feladat. *Mutassuk meg, hogy adott $P \in H$ pontra a $T[P_j]$ konstans sok aritmetikai művelettel meghatározható. (A naiv algoritmus kapcsán említett gondolatok használhatók.)*

Tekintsük mármost az általános helyzetet, amikor a H egy H' részhalmazát már feldolgoztuk. A H' konvex burka egy sokszög, aminek csúcspontjai a $\Delta' = \{Q_1, Q_2, \dots, Q_l\}$ ponthalmaz elemei, a burok határát (vagyis a H'-t befoglaló legrövidebb kerítést) a $Q_1Q_2, Q_2Q_3, \dots, Q_{l-1}Q_l$, Q_lQ_1 szakaszok képezik. Tegyük fel továbbá, hogy a T leképezést és az inverzét is meghatároztuk a Δ helyett már erre a nagyobb Δ' halmazra: $T(P_j) =$ „belső”, ha $P_j \in \Delta'$, különben $T(P_j) = Q_iQ_{i+1}$, ahol a $[D, P_j]$ szakasz metszi Q_iQ_{i+1}-et (itt az i index modulo l értendő) (lásd 2.1. ábra).

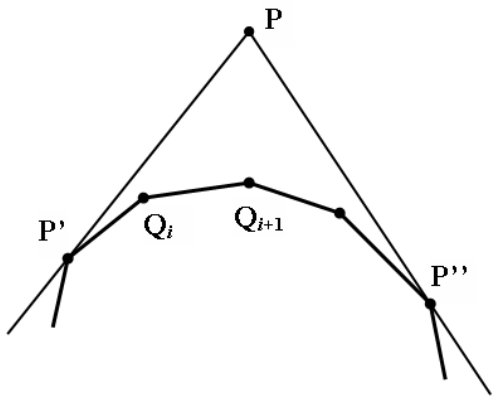


2.1. ábra.

Az algoritmus fő lépése (azaz a H' bővítése, a kerítés, valamint T és T^{-1} újraszámolása): legyen P véletlen pont (egyenletes eloszlás szerint) a $H \setminus H'$-ből.

1. Ha $T(P) =$ „belső”, akkor P-t hozzáadjuk H'-höz és a lépés itt véget ér.

2. Ha $T(P) = e$, ahol e egy éle Δ'-nek, akkor a kerítés mentén haladunk e-től jobbra és balra is addig az utolsó Q pontig, ami még P-ből látszik. A két ilyen utolsó pont legyen P' és P''. A P'-től P''-ig terjedő kerítésszakaszt kicseréljük a $P'P$ és PP'' szakaszokra, ahogyan azt a 2.2. ábra is mutatja.



2.2. ábra.

   Ezt követően P-t hozzáadjuk H'-höz, végül a $H \setminus H'$-beli M pontokra újraszámoljuk T-t és inverzét, de csak azokra az M-ekre, amelyekre a $T(M)$ a $P'P''$ töröttvonal egyik szakasza volt.

A teljes futás alatt keletkező „kerítés”-élek száma $\leq 3 + (n-3) \cdot 2$, mert egy bővítő lépésben legfeljebb 2 új él keletkezik, és $n-3$ bővítő lépés van. Tehát legfeljebb ennyi élet járunk be és törlünk. A régi élek bejárása, törlése, az újak hozzávétele ezért összesen $O(n)$ költséget jelent.

Az 1. lépések összköltsége $O(n)$, a 2. lépésekből azon $T(P)$ újraszámolások összköltsége, ahol az újraszámolt $T(P)$ „belső” lesz, szintén $O(n)$ (mert egy P maximum egyszer lesz ilyen). Az összmunka ennek következtében:

$$O(n) + (T(P) \text{ számolások költsége akkor, amikor } T(P) \text{ értéke másik él lesz}).$$

Visszafelé elemzés

A költség elemzésének alapgondolata, hogy lejátsszuk *visszafelé* a folyamatot, és megnézzük, hogy a j-edik iteráció utáni helyzetből egyet visszalépve mit látunk. Úgy képzelhetjük, hogy az algoritmus futásának filmjét visszafelé, a végétől az eleje felé forgatva nézzük. Mindenekelőtt vezessük be Z_{ij} indikátorváltozókat $(i, j = 4, \ldots, n)$:

$$Z_{ij} = \begin{cases} 1, & \text{ha } T(P_i) \text{ egy másik él lett a } j\text{-edik iteráció után} \\ 0, & \text{különben} \end{cases}$$

A várható összmunka a Z_{ij} valószínűségi változókkal kifejezve:

$$O\Big(n + \mathbf{E}\big(\sum_{i,j=4}^{n} Z_{ij}\big)\Big).$$

Igaz továbbá, hogy

$$\mathbf{E}\Big(\sum_{i,j} Z_{ij}\Big) = \sum_{i,j} \mathbf{E}(Z_{ij}) = \sum_{i,j} \mathbf{P}(Z_{ij} = 1).$$

A j-edik iteráció után visszalépve a meglevő H' halmazból törlünk egy véletlen pontot, mégpedig *egyenletes eloszlás szerint (!)* a már feldolgozott j db pontból. Az algoritmusnál alkalmazott véletlen választások miatt valóban a j pont bármelyikét $\frac{1}{j}$ valószínűséggel töröljük a fordított filmen. A $T(P)$ akkor és csak akkor lesz egy másik él a visszalépés során, ha vagy Q_i, vagy Q_l a törölt csúcs, ahol $T(P) = Q_iQ_l$ a helyzet a j. iteráció után. Az egyenletes választás miatt tehát

$$\mathbf{P}(Z_{ij} = 1) \le \frac{2}{j}.$$

(A Z_{ij} biztosan 0, ha a j. iteráció végére P_i már a kerítésen belül van.) Visszatérve a költség becsléséhez:

$$\sum_{i,j=4}^{n} \mathbf{E}(Z_{ij}) \le \sum_{i,j=4}^{n} \frac{2}{j} \le 2nH_n \le 2n(\log n + 1).$$

Összegezve, az algoritmus várható futási ideje:

$$O\Big(n + \mathbf{E}(\sum_{ij} Z_{ij})\Big) = O(n + n\log n) = O(n\log n).$$

2.5 Egy algebrai probléma

Itt egy igen fontos algebrai természetű feladattal foglalkozunk. Legyen $\mathbb{F}$ egy test.[9] A felmerülő alkalmazások során $\mathbb{F}$ igen gyakran $\mathbb{Q}$, a racionális számok teste, vagy a q elemű $\mathbb{F}_q$ véges test. Emellett előfordul még $\mathbb{F} = \mathbb{R}$ (a valós számok) és $\mathbb{F} = \mathbb{C}$ (a komplex számok) is.

Polinomnak, közelebbről n változós polinomnak nevezzük az olyan algebrai kifejezéseket, melyeket az $x_1, x_2, \ldots, x_n$ változókból, és $\mathbb{F}$-beli elemekből (konstansokból) építünk fel az összeadás, a kivonás és a szorzás alkalmazásával. Az $\mathbb{F}$ test feletti polinomok összességét

$$\mathbb{F}[x_1, x_2, \ldots, x_n]$$

jelöli.

Például $\mathbb{R}[x]$ az egyváltozós valós polinomok halmaza, $x^2+2x-3 \in \mathbb{R}[x]$, vagy $x_1^2-2x_2^3+3ix_4 \in \mathbb{C}[x_1, x_2, x_3, x_4]$.

Legyen $f(x_1, \ldots, x_n)$ egy polinom $\mathbb{F}[x_1, \ldots, x_n]$-ből. Ekkor az $f(x_1, \ldots, x_n) = 0$ egyenletet kielégítő $(a_1, a_2, \ldots, a_n) \in \mathbb{F}^n$ pontok egy hiperfelületet alkotnak, amit V_f-fel jelölünk. A V_f hiperfelület $n = 2$ esetén görbe, $n = 3$ esetén pedig felület. Például, $f(x_1, x_2) = x_1^2 + x_2^2 - 1$ esetén a V_f halmaz éppen az origó-középpontú egységkörvonal.

4. Számítási feladat. *Adott az $f(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n]$ polinom. Találjunk $\alpha = (\alpha_1, \ldots, \alpha_n)$ vektort, melyre $f(\alpha_1, \ldots, \alpha_n) \neq 0$.*

Másként fogalmazva olyan $\alpha \in \mathbb{F}^n$ pontot keresünk, ami *nincs rajta* a V_f hiperfelületen. Ha az $\mathbb{F}$ egy végtelen test, akkor egy véletlen α pont igen nagy eséllyel jó megoldást ad. Ezzel az egyszerű észrevétellel meglepő, már-már ellentmondásos viszonyban van az a tény, hogy nem

[9] Olyan algebrai struktúra, amelyben van négy művelet, az összeadás (+), a kivonás (−), a szorzás (*) és az osztás (/), amelyekre érvényesek a valós számok körében megszokott szabályok; bővebben lásd pl. [KRSz].

ismeretes olyan determinisztikus algoritmus, amely hatékonyan (polinom időben) találna egy ilyen α pontot.

A továbbiakban szeretnénk a véletlen választást használó eljárás egy diszkrét változatát bemutatni, amelyben az α vektor α_i komponenseit egy alkalmas véges halmazból vesszük.

4. Definíció. *Az f polinom foka a kifejtés után a tagjaiban előforduló legnagyobb változószám. Pl. x_1^2 foka 2, $1 + x_1x_2^3x_3$ foka 5. A $0 \neq c \in \mathbb{F}$ konstansok foka nulla, a $0 \in \mathbb{F}$ konstansnak nincs foka.*

Szükségünk lesz a következő tényre (nem bizonyítjuk):

4. Állítás. *Legyen $f(x) \in \mathbb{F}[x]$ egyváltozós, nem azonosan 0 polinom. Ekkor az $f(x) = 0$ egyenlet megoldásainak száma nem több, mint az f foka.*

A bizonyításhoz hasznos a következő:

8. Feladat. *Legyen $f(x) \in \mathbb{F}[x]$ egyváltozós polinom, $a \in \mathbb{F}$ amelyre $f(a) = 0$. Ekkor van olyan $g(x) \in \mathbb{F}[x]$ polinom, amellyel $f(x) = g(x)(x - a)$.*

Most már bizonyítani tudjuk a számítási feladatot megoldó randomizált algoritmus elvi alapjául szolgáló tételt:

10. Tétel (Schwartz–Zippel)**.** *Legyen $f(x_1, \ldots, x_n) \in \mathbb{F}[x_1, \ldots, x_n]$ nem azonosan 0 polinom, aminek a foka d. Legyen továbbá $T \subseteq \mathbb{F}$, $|T| = N$. Legyenek $\alpha_1, \ldots, \alpha_n$ a T-ből egyenletes eloszlás szerint teljesen függetlenül választott elemek (minden i-re az előzőektől teljesen függetlenül, $\frac{1}{N}$ valószínűséggel sorsoljunk α_i-t). Ekkor $\mathbf{P}(f(\alpha_1, \ldots, \alpha_n) = 0) \leq \frac{d}{N}$.*

Az α választására vonatkozó feltétel úgy is fogalmazható, hogy α az egyenletes eloszlás szerint választott véletlen eleme a T^n halmaznak, vagyis bármely $\alpha \in T^n$ valószínűsége $\frac{1}{N^n}$.

Bizonyítás. Nyilván feltehető, hogy $\frac{d}{N} < 1$. Teljes indukciót alkalmazunk n szerint:

- $n = 1$ eset:

$$\mathbf{P}(f(\alpha_1) = 0) = \frac{\text{kedvező esetek}}{\text{összes eset}} \leq \frac{d}{N}.$$

  Itt használtuk az előző állítást, amely szerint a kedvező esetek száma $\leq d$.

- $n > 1$ eset: tegyük fel, hogy $n - 1$-ig igaz a tétel állítása (indukciós feltevés). Fejtsük ki f-et x_1 szerint:

$$f(x_1, \ldots, x_n) = h_0(\mathbf{y}) + h_1(\mathbf{y})x_1 + \ldots + h_k(\mathbf{y})x_1^k,$$
$$\mathbf{y} = (x_2, \ldots, x_n).$$

  A fentiek alapján $k \leq d$, valamint h_k foka $\leq d - k$ (tételezzük fel, hogy $h_k \not\equiv 0$). A bizonyítás további menetéhez vezessük be az alábbi eseményeket:

$$B = \{f(\alpha_1, \ldots, \alpha_n) = 0\},$$
$$A_1 = \{h_k(\alpha_2, \ldots, \alpha_n) = 0\},$$
$$A_2 = \{h_k(\alpha_2, \ldots, \alpha_n) \neq 0\}.$$

A_1, A_2 teljes eseményrendszert alkot, és $\mathbf{P}(A_2) > 0$ az indukciós feltevés szerint. Ekkor

$$\begin{aligned}\mathbf{P}(B) &= \mathbf{P}(B \cap A_1) + \mathbf{P}(B \cap A_2) \\ &\leq \mathbf{P}(A_1) + \mathbf{P}(B|A_2)\mathbf{P}(A_2) \\ &\leq \mathbf{P}(A_1) + \mathbf{P}(B|A_2).\end{aligned}$$

Az indukciós feltevésből következik, hogy $\mathbf{P}(A_1) \leq \frac{d-k}{N}$, az $n = 1$ esetből pedig következik, hogy $\mathbf{P}(B|A_2) \leq \frac{k}{N}$, így

$$\mathbf{P}(B) \leq \mathbf{P}(A_1) + \mathbf{P}(B|A_2) \leq \frac{d-k}{N} + \frac{k}{N} = \frac{d}{N}.$$

□

A tételből adódik, hogy $N = 2d$ esetén $\mathbf{P}(f(\alpha_1, \ldots, \alpha_n) = 0) \leq \frac{1}{2}$. Ezzel az N értékkel, és az $\alpha \in T^n$ véletlen választásával, majd pedig az $f(\alpha)$ kiszámításával olyan algoritmust kapunk, amely legalább $\frac{1}{2}$ valószínűséggel talál V_f-en kívüli pontot, ha f nem azonosan nulla.

Az $(\alpha_1, \ldots, \alpha_n)$ választását t-szer függetlenül megismételve, annak valószínűsége, hogy mindig 0-t kapunk, kisebb lesz, mint $\frac{1}{2^t}$.

Két alkalmazási példa

1. Teljes párosítás keresése páros gráfban

5. Számítási feladat. *Adott a $G = (L, U; E)$ páros gráf, amelyben L és U alkotja a két csúcshalmazt, $|U| = |L| = n$, és él csak U és L között mehet. Állapítsuk meg, hogy létezik-e G-ben teljes párosítás (n db független él).*

Tegyük fel, $L = \{l_1, l_2, \ldots, l_n\}$ és $U = \{u_1, u_2, \ldots, u_n\}$. Egy $n \times n$-es M mátrixot készítünk: $M = (m_{ij})_{i,j=1}^n$.

$$m_{ij} = \begin{cases} x_{ij} \text{ változó}, & \text{ha } (l_i, u_j) \in E, \\ 0, & \text{különben.} \end{cases}$$

11. Tétel. *A G páros gráfban létezik teljes párosítás $\iff \det M \not\equiv 0$ ($\det M$ nem azonosan 0 polinomja az x_{ij} változóknak).*

Bizonyítás. M determinánsa $\pm x_{1i_1} x_{2i_2} \ldots x_{ni_n}$ alakú tagok összege, ahol $i_1, \ldots, i_n$ csupa különböző egész. Ha $\det M \not\equiv 0$, akkor van egy

$$\pm\, x_{1i_1} \ldots x_{ni_n} \tag{2.6}$$

alakú nem nulla tag. Ekkor az (l_1, u_{i_1}), (l_2, u_{i_2}), …, (l_n, u_{i_n}) élek egy teljes párosítást adnak.

A fordított irány: ha G-ben létezik teljes párosítás, pl. $(l_1, u_{i_1}), (l_2, u_{i_2}), \ldots, (l_n, u_{i_n})$, akkor $\det M \neq 0$, mert a (2.6) tag megmarad kifejtés után. Minden élhez egyedi, az összes többitől különböző változót rendeltünk, ezért a kifejtési tagok nem ejthetik ki egymást. □

A teljes párosítás létezése esetén M determinánsa az x_{ij} változók $\mathbb{Q}$ feletti polinomja, a foka $\leq n$. Az előző tételnek van egy hasznos következménye. Legyen $\mathbb{F} = \mathbb{Q}$, és tekintsük a $\det M$ polinomot $\mathbb{Q}$ felett. Írjunk a determinánsban a változók helyébe véletlen és függetlenül választott számokat a $T = \{1, 2, \ldots, 2n\}$ halmazból. Az ezekből a véletlen értékekből képzett vektort

jelöljük α-val. Ekkor, ha G-ben van teljes párosítás, akkor Schwartz–Zippel-tételt alkalmazva $\mathbf{P}(\det M(\alpha) = 0) \leq \frac{n}{2n} = \frac{1}{2}$.

Ezzel egy randomizált algoritmust nyerünk a teljes párosítás létezésének eldöntésére. Mennyi a költsége a számításnak? A determinánsnak mint polinomnak sok tagja lehet (maximum $n!$), de numerikusan (a változókba számokat írva) $O(n^3)$ aritmetikai művelettel kiszámolható, lényegében a lineáris egyenletrendszerek megoldása során használt Gauss-eliminációt alkalmazva. Egy ilyen randomizált párosítástesztet tehát $O(n^3)$ művelettel megvalósítható. Az így nyert módszer lassúbb a kombinatorikus alapú párosítástesztnél, viszont igen erőteljesen párhuzamosítható, mégpedig azért, mert a determináns számolása hatékonyan párhuzamosítható.[10]

Kiszámítás kontra tesztelés

Gyakran könnyebb egy számítás helyességét ellenőrizni, mint újra elvégezni azt. Ebben az összefüggésben is jól használható a Schwartz–Zippel-tétel.

6. Számítási feladat. *Adottak az A, B, C $n \times n$-es mátrixok valamilyen $\mathbb{F}$ testbeli elemekkel. Adjunk választ arra a kérdésre, hogy $AB \stackrel{?}{=} C$.*

AB számítása a definíció szerint kb. $2n^3$ művelet $\mathbb{F}$-ben. Egyelőre nyitott kérdés, hogy valójában mennyi művelet kell a mátrixszorzáshoz. Érdekességként említjük, hogy a jelenlegi rekord $O(n^\omega)$ művelet, ahol ω értéke $2,37$ körül van, viszont ezzel a módszerrel nem praktikus számolni, ha az n nem túl nagy.[11]

Az $AB \stackrel{?}{=} C$ tesztelésére lehetőség van $O(n^2)$ művelet elvégzésével is, randomizált teszt alkalmazásával. Az alapötlet: vegyünk egy $T \subseteq \mathbb{F}$ halmazt, melynek elemszáma legyen n. Válasszunk véletlen $(\alpha_1, \ldots, \alpha_n)^t = \alpha$ oszlopvektort T-beli komponensekkel. Nézzük meg, hogy $A(B\alpha) \stackrel{?}{=} C\alpha$ teljesül-e. Ha $AB = C$, akkor itt mindig igent kell kapnunk. Ennek a tesztnek a költsége 3 db mátrix-vektor szorzás (először kiszámítjuk a $\beta = B\alpha$ vektort, majd pedig az $A\beta$ és a $C\alpha$ vektorokat), ami összesen is $O(n^2)$ idő.

Határozzuk meg a hiba valószínűségét, pontosabban annak a valószínűségét, hogy $AB \neq C$, de $A(B\alpha) = C\alpha$. Legyen $\mathbf{x} = (x_1, \ldots, x_n)^t$ egy változókból álló oszlopvektor. Ha $AB \neq C$, akkor az $AB\mathbf{x} - C\mathbf{x} = \mathbf{y}$ vektor koordinátái nem mind 0 lineáris polinomjai az $x_1, \ldots, x_n$ változóknak. Tegyük fel, hogy az eredmény j-edik komponense, $y_j \not\equiv 0$. Ekkor az $y_j(x_1, \ldots, x_n)$ lineáris polinom foka $= 1$. A Schwartz–Zippel-tétel szerint $\leq \frac{1}{n}$ a valószínűsége, hogy $y_j(\alpha) = 0$, tehát legfeljebb $\frac{1}{n}$ a valószínűsége annak, hogy a teszt *nem ad* tanút.

2.6 Hashelés

Univerzális hashelés

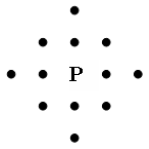
A hash-szervezés igen fontos adatszerkezethez vezet, ami széles körben használható, erőteljes megoldásokat ad. A véletlen többféleképpen is kapcsolatba hozható ezzel a szervezéssel. Ezek egyikét, az univerzális hashelést vesszük alaposabban szemügyre. Törekvésünk motiválására tekintsük a következő algoritmikus feladatot:

7. Számítási feladat. *Adott n db egész koordinátájú pont a síkon. Állapítsuk meg, hogy van-e köztük $P \neq Q$ úgy, hogy $d(P, Q) \leq 2$.*

[10] Lásd pl.: [Cs], és [Be].

[11] A számítógépes aritmetika egyik legfontosabb nyitott kérdése, hogy kiszámítható-e két $n \times n$-es mátrix szorzata $O(n^{2+\epsilon})$ aritmetikai művelettel.

Ha naiv megközelítéssel minden pontpár távolságát meghatározzuk, akkor $O(n^2)$ (uniform) költségű megoldáshoz jutunk. Ennél hatékonyabb, $O(n \log n)$ uniform költségű megoldást kaphatunk kiegyensúlyozott keresőfák (Pl. piros-fekete fa, AVL-fa, 2-3-fa) alkalmazásával. Az ötlet az, hogy építsünk keresőfát az adott pontokkal mint kulcsokkal. Ennek ára $O(n \log n)$. Ezt követően minden P pontra nézzük meg, hogy a P ponthoz közeli 12 db egész koordinátájú R pont (ezeket szemlélteti a következő ábra) közül valamelyik benne van-e a fában. A művelet költsége: $12n \cdot O(\log n) = O(n \log n)$, így az összes munka költsége is $O(n \log n)$.



Meg fogjuk mutatni, hogy a véletlenre támaszkodva, univerzális hasheléssel a feladat lineáris várható időben is megoldható. Ki fog derülni, hogy egy alkalmasan választott hash-adatszerkezetben a feladatban felmerülő műveletek (beszúrás, keresés) várható ideje konstans korlát alatt marad, szemben a keresőfákéval, ahol csak logaritmikus korlát érvényes.

A továbbiakban X egy adott (nagy) kulcshalmaz, $|X| = r$, és $[M] = \{0, 1, 2, \ldots, M-1\}$ a hash-címek[12] tartománya, ahol M pozitív egész, végül H legyen egy $X \mapsto [M]$ függvényekből álló halmaz. Ez utóbbit tekintjük a lehetséges hash-függvényeink halmazának.

5. Definíció. *A fenti jelölések mellett a H egy* 2-univerzális család, *ha $\forall x \neq y \in X$-re és véletlen (egyenletesen választott) $h \in H$-ra igaz, hogy $\mathbf{P}(h(x) = h(y)) \leq \frac{1}{M}$.*

Nézzünk mindjárt egy példát: legyen H az összes $X \mapsto [M]$ függvényből álló halmaz, következésképpen $|H| = M^r$. Ha $x \neq y$, akkor éppen M^{r-1} darab h ad ütközést (ennyiszer lesz $h(x) = h(y)$), vagyis $\mathbf{P}(h(x) = h(y)) = \frac{1}{M}$. Ez a H tehát 2-univerzális. Az alkalmazások szempontjából súlyos gond viszont, hogy ez a H túl nagy, r-ben exponenciális számú eleme van. Itt mutatunk majd egy olyan 2-univerzális H^* függvényhalmazt, amelynek kevesebb, mint $4r^2$ eleme van, és ezért hatékonyan kezelhető például abban az értelemben, hogy egyszerűen és gyorsan tudunk belőle véletlen h-t választani. Az is teljesülni fog a $h \in H^*$ függvényekre, hogy a $h(x)$ hash-címek gyorsan és kényelmesen számíthatók.

Először körvonalazzuk, hogy miként használható egy 2-univerzális család hash-szerkezetek megvalósítására. Kiindulásként vezessük be a $\delta(x, y, h)$ indikátorváltozókat, ahol $x, y \in X$ és $h \in H$. A $\delta(x, y, h)$ értéket így definiáljuk:

$$\delta(x, y, h) = \begin{cases} 1, & \text{ha } x \neq y, \text{ és } h(x) = h(y) \text{ (vagyis ütközés adódik } h\text{-val)} \\ 0, & \text{különben.} \end{cases}$$

A δ függvény segítségével kényelmesen kifejezhető a 2-univerzalitás. A H függvényhalmaz pontosan akkor lesz 2-univerzális, ha bármely $x \neq y$ esetén

$$\delta(x, y, h_1) + \ldots + \delta(x, y, h_k) \leq \frac{k}{M}, \tag{2.7}$$

ahol $H = \{h_1, h_2, \ldots, h_k\}$. Valóban, az egyenlőtlenség szerint a H elemeinek legfeljebb az M-ed része okoz ütközést. Ez egyenértékű azzal, hogy véletlen $h \in H$ függvényt választva $\leq \frac{1}{M}$ lesz a

[12] Lásd pl. [RISz], 4. fejezet.

$h(x) = h(y)$ ütközés valószínűsége. A (2.7) egyenlőtlenség bal oldalát röviden $\delta(x, y, H)$-nak is írhatjuk. Ezt a jelölést általánosítva $X', Y' \subseteq X$; $H' \subseteq H$ esetén legyen

$$\delta(X', Y', H') = \sum_{x \in X'} \sum_{y \in Y'} \sum_{h \in H'} \delta(x, y, h).$$

Ezt a tömörebb jelölést alkalmazva érvényes a következő egyszerű, de igen hasznos tény:

5. Állítás. *Legyen $x \in X$, $S \subseteq X$, továbbá $h \in H$ egy véletlen függvény (egyenletes eloszlás szerint) a H 2-univerzális családból. Ekkor*

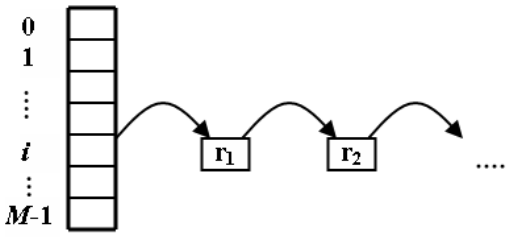
$$\mathbf{E}(\delta(x, S, h)) \leq \frac{|S|}{M}.$$

Bizonyítás. A várható érték definíciója és a (2.7) egyenlőtlenség segítségével könnyen kapjuk az állítást:

$$\begin{aligned}\mathbf{E}(\delta(x, S, h)) &= \sum_{h \in H} \frac{1}{|H|} \delta(x, S, h) = \frac{1}{|H|} \sum_{h \in H} \delta(x, S, h) = \frac{1}{|H|} \sum_{h \in H} \sum_{s \in S} \delta(x, s, h) \\ &= \frac{1}{|H|} \sum_{s \in S} \delta(x, s, H) \leq \frac{1}{|H|} \sum_{s \in S} \frac{|H|}{M} = \sum_{s \in S} \frac{1}{M} = \frac{|S|}{M}.\end{aligned}$$

□

Tegyük fel, hogy X egy részhalmazát akarjuk tárolni véletlenül választott $h \in H$ hash-függvény segítségével, ahol H egy 2-univerzális család. Az ütközéseket rekordok láncolásával oldjuk fel, ahogy az ábra szemlélteti.[13]



Az r_j kulcsokra igaz, hogy $h(r_j) = i$. Ha S a tárolt kulcsok halmaza, és egy művelet a {keres, beszúr, töröl, módosít} halmazból az $x \in X$-et érinti, akkor a művelet költsége arányos lesz az x-hez tartozó lánc hosszával, ami legfeljebb $1 + \delta(x, S, h)$.

12. Tétel. *Tegyük fel, hogy kezdetben egy üres struktúránk van. Végrehajtunk egy $t_1, \ldots, t_n$ műveletsort, ahol $t_i \in \{$beszúr, keres, töröl, módosít$\}$. Ezek közül a beszúrások száma legyen $k \leq M$. Ekkor a sorozat összköltsége (uniform költség) véletlen $h \in H$ alkalmazása esetén várhatóan*

$$O\left(n\left(1 + \frac{k}{M}\right)\right) = O(n). \tag{2.8}$$

[13] A szerkezet lényegében ugyanaz, mint a [RISz] 4.1.1. szakaszában leírt vödrös módszernél, azzal az eltéréssel, hogy itt rekordokat és nem lapokat láncolunk.

Bizonyítás. Tegyük fel, hogy a t_i művelet előtt az $S_i \subseteq X$ kulcshalmaz van a szerkezetben, és a művelet az $x_i \in X$ kulcsot érinti (ezt illesztjük be vagy keressük, stb.). A költség arányos a mutatókövetések számával, ami ebben az esetben legfeljebb $\leq 1 + \delta(x_i, S_i, h)$. Ennek a várható értéke az előző állítás szerint nem több, mint $\leq 1 + \frac{k}{M}$. Az összes művelet várható költségére a várható érték additivitása miatt így a $O(n(1 + \frac{k}{M}))$ korlátot nyerjük. Végül $k \leq M$ miatt $1 + \frac{k}{M} \leq 2$, tehát az összmunka $O(n)$. □

Az iménti érvelésből az is kitűnik, hogy egyetlen t művelet várható költsége konstans.

Térjünk vissza a most kiinduló feladathoz, a közeli pontok problémájához. Keresőfa helyett használjunk inkább 2-univerzális hashelést úgy, hogy a szerkezet méretére $M = n$ teljesüljön. Ekkor $k = M$ és az n pont tárolása $O(n)$ várható költségű (n db beszúr művelet szükséges). A pontok tárolása után még $12n$ db keres művelet következik, melynek várható összköltsége szintén $O(n)$. Ez a költségbecslés akkor tekinthető realisztikusnak, ha a pontok koordinátái nem túl nagyok.

2-univerzális hash-függvénycsalád konstrukciója

Egy viszonylag kis méretű 2-univerzális függvényhalmazt adunk meg. Legyen p prímszám, $p \geq r = |X|$. Feltesszük, hogy $X \subseteq \mathbb{F}_p$, továbbá, hogy $\mathbb{F}_p = \{0, 1, \ldots, p-1\}$. Egy 2-univerzális $\mathbb{F}_p \to [M]$ függvényhalmazt adunk meg. Ha a p nem sokkal nagyobb, mint r, akkor ez a család hatékonyan használható az X kulcshalmaz kezelésére is. Feltesszük tehát, hogy $X = \mathbb{F}_p$. Bevezetjük a következő függvényeket:

- Legyen $f_{a,b} : \mathbb{F}_p \mapsto \mathbb{F}_p$, $f_{a,b}(x) := ax + b \mod p$, ahol $a, b \in \mathbb{F}_p$.
- Legyen $g : \mathbb{F}_p \mapsto [M]$, $g(x) := x \mod M$.
- Legyen $h_{a,b} : \mathbb{F}_p \mapsto [M]$, $h_{a,b}(x) := g(f_{a,b}(x))$.

Végül legyen $H^* = \{h_{a,b} | a \neq 0;\ a, b \in \mathbb{F}_p\}$. Ekkor nyilván $|H^*| = p(p-1)$. Azt is szeretnénk elérni, hogy $|H^*|$ minél kisebb legyen. Ismert, hogy $r > 1$ esetén $(r, 2r)$-ben van p prímszám (Csebisev tétele[14]). Elérhető tehát, hogy $p < 2r$, amikor is $|H^*| < 4r^2$. Ekkor a H^* egy véletlen eleme leírható kb. $2 \log_2 r$ bittel. Az is igaz, hogy a $h_{a,b}(c)$ értékek gyorsan számolhatók ($c \in \mathbb{F}_p$).

13. Tétel. *H^* egy 2-univerzális függvénycsalád.*

Bizonyítás. Először megmutatjuk, hogy $x \neq y \in \mathbb{F}_p$ esetén $\delta(x, y, H^*) = \delta(\mathbb{F}_p, \mathbb{F}_p, g)$ (vagyis, hogy $\delta(x, y, H^*)$ független x, y-tól). Ehhez tegyük fel, hogy $h_{a,b}(x) = h_{a,b}(y)$ (azaz ütközés van). A következőket írhatjuk fel ekkor:

$$ax + b = u \mod p \tag{2.9}$$
$$ay + b = v \mod p \tag{2.10}$$
$$u \equiv v \mod M$$

(2.9) és (2.10) lineáris egyenletrendszert ad a, b-re (x, y, u, és v fix), amelynek a determinánsa

$$\det \begin{pmatrix} x & 1 \\ y & 1 \end{pmatrix} = x - y \neq 0,$$

tehát $x \neq y$ esetén egyértelműen oldható meg az egyenletrendszer.

[14] Lásd pl. [FGy] 5.5.3. tétel.

A fentiek alapján azok az (a,b) párok, ahol $a \neq 0$, és azok az (u,v) párok, ahol $u \neq v$, kölcsönösen megfeleltethetők egymásnak. A $h_{a,b}(x) = h_{a,b}(y)$ ütközések száma ezért éppen azon u,v párok száma lesz, amelyekre $u \neq v$ és $u \equiv v \mod M$. Az utóbbi mennyiség pedig definíció szerint pontosan $\delta(\mathbb{F}_p, \mathbb{F}_p, g)$.

Másodszor igazoljuk, hogy $\delta(\mathbb{F}_p, \mathbb{F}_p, g) \leq \frac{p(p-1)}{M}$. Azon (u,v) párok számát keressük, melyekre $\delta(u,v,g) = 1$. Az u értékét p-féleképpen választhatjuk meg. Egy rögzített u-hoz olyan v kell, melyre $u \equiv v \mod M$, $0 \leq v \leq p-1$, $u \neq v$. Az ilyen v-k száma legfeljebb

$$\left\lceil \frac{p}{M} \right\rceil - 1 \leq \frac{p}{M} + \frac{M-1}{M} - \frac{M}{M} = \frac{p-1}{M},$$

azaz a párok száma $\leq \frac{p(p-1)}{M}$. Innen a tétel bizonyítása azonnal adódik:

$$\delta(x,y,H^*) = \delta(\mathbb{F}_p, \mathbb{F}_p, g) \leq \frac{p(p-1)}{M} = \frac{|H^*|}{M},$$

tehát H^* 2-univerzális. □

2.7 Egy jó randomizált keresőfa: a „falom”

A falom[15] egy randomizált keresőfa-konstrukció jó műveleti időkkel, és nagyon egyszerű algoritmusokkal.

6. Definíció. *Az F falom egy bináris fa, aminek a csúcsaiban (k,p) alakú párok vannak, ahol k kulcs, a p szám pedig egyedi prioritás, $p \in [0,1]$. Az F a k kulcsok szerint bináris keresőfa, a p prioritások szerint pedig max-kupac, azaz minden csúcsban lévő p nagyobb, mint az ő részfájában levő többi prioritásérték.*

A felhasználó célja a rendezett U halmazból való k kulcsok, pontosabban a velük azonosított rekordok hatékony tárolása, kezelése. Ezek a kulcsok tehát a bemenet részét képezik az F használatakor. A p prioritásokat viszont az F-et kezelő algoritmusok állítják elő azzal a céllal, hogy a hagyományos keresőfa-műveleteket hatékonnyá, gyorssá tegyék. Minden beszúrásnál a beillesztendő k kulcshoz véletlen p-t sorsolunk. A $[0,1]$ intervallumból egyenletes eloszlás szerint és csúcsonként egymástól teljesen függetlenül választjuk (sorsoljuk) a p prioritásokat. A p prioritások egyediek: különböző csúcsokhoz különböző p-k tartoznak. Mi most az egyszerűbb tárgyalás kedvéért azt is feltesszük, hogy az F-beli kulcsok is mind különbözők.

6. Állítás. *A falom alakját egyértelműen meghatározzák a benne lévő (k,p) párok.*

Bizonyítás. A gyökérben a max-kupac-tulajdonság miatt csak az a (k,p) pár lehet, amelyikben p maximális. A gyökér bal részfájában pedig azok a (k',p') párok vannak a keresőfa-tulajdonság miatt, amelyekre $k' < k$. A jobb részfájába azok a (k'',p'') párok kerülnek, amelyekre $k'' > k$. Az eddigi gondolatmenet megismételhető ezekre a részfákra. □

A szokásos keresőfa-algoritmusokat (keres(k), beszúr(k), töröl(k), min, max) szeretnénk megvalósítani. A keresés, a min és a max a bináris keresőfáknál tanult módon megy itt is.

Nézzük meg a beszúr(k) műveletet: k helyét a naiv beszúrás logikáját követve megkeressük; ez egy új levél lesz, majd sorsolunk mellé egy véletlen $p \in [0,1]$ prioritást (az eddigi sorsolásoktól

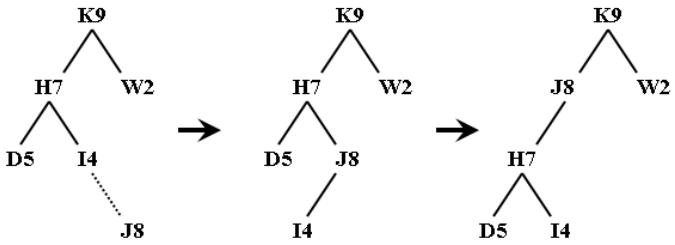
[15] A *falom* szó a *fa* és *halom* szavak összevonásából keletkezett, ahogy a szerkezet eredeti angol neve, a *treap* a *tree* és *heap* szavakból.

teljesen függetlenül). Utána a (k, p) csúcsot forgatásokkal felfelé mozgatjuk annyira, hogy a max-kupac-tulajdonság teljesüljön a prioritásokra nézve.[16]

Törlésnél forgatásokkal levisszük az adott elemet addig, amíg levélpozícióba nem kerül, és onnan egyszerűen kitöröljük. A forgatások nem rontják el a keresőfa-tulajdonságot, csak a kupac-tulajdonságot befolyásolják.

9. Feladat. *A törlendő (k, p) csúcsot elvben a jobb részfájába és a bal részfájába is forgathatnánk. Hogyan célszerű választani a két lehetőség közül?*

1. Példa. *Szúrjuk be a J kulcsot az ábra bal oldalán lévő falomba, legyen a J-hez sorsolt prioritás $p = 0,8$. A prioritásoknak csak a tizedesvessző utáni első jegyét adjuk meg, így az új csúcs jelölése: J8. A bal oldalon a kiinduló F szerepel, de már feltüntettük az új levél helyét is. A következő két fa pedig az első és a második (egyben utolsó) forgatás utáni helyzetet mutatja.*



A gyakorlatban a p prioritásoknak csak véges sok bitjét sorsoljuk. Egy új (k, p) pár beszúrásakor – miután már megtaláltuk az új levél helyét – csak annyi bitet sorsolunk p-hez és az apjában levő p' prioritáshoz, hogy egyik se legyen prefixe a másiknak. Ha egy forgatás után p és az új apjának p' értéke közül egyik prefixe a másiknak, akkor mindkettőhöz újabb bitet (esetleg biteket) sorsolunk egészen addig, amíg valamelyik prefixe a másiknak.

Az algoritmusokat azzal a (kissé idealizált) feltételezésel elemezzük, hogy a prioritások egymástól teljesen független, a $[0, 1]$ intervallumban egyenletes eloszlású számok. Hasznos lesz a következő feladat, amely a prioritások szerepét segít megérteni:

10. Feladat. *Legyenek $p_1, p_2, \ldots, p_d$ a $[0, 1]$ intervallumon egyenletes eloszlású, teljesen független valószínűségi változók. Ekkor*

(a) $\mathbf{P}(p_1 > p_2) = \mathbf{P}(p_2 > p_1) = \frac{1}{2}$,

(b) $\mathbf{P}(p_{i_1} > p_{i_2} > \ldots > p_{i_d}) = \frac{1}{d!}$, *ahol $i_1, i_2, \ldots, i_d$ az $1, 2, \ldots, d$ indexek egy tetszőleges permutációja,*

(c) $\mathbf{P}(p_i = \max\{p_1, \ldots, p_d\}) = \frac{1}{d}$ *tetszőleges $1 \le i \le d$ esetén,*

(d) $\mathbf{P}(p_1 > p_d$ *és* $p_d = \max\{p_2, \ldots, p_d\}) = \frac{1}{d(d-1)}$.

14. Tétel. *Legyen F egy falom, csúcsainak száma $|F| = n$, x pedig egy csúcsa. A gyökértől az x-ig menő út hossza várhatóan $\le 2 \log n + 2$.*

1. Következmény. *Az F-ben való sikeres keresés várható költsége $O(\log n)$.*

[16] Jól ismert – lásd pl. [RISz] 3.5. szakasz –, hogy a forgatások megtartják a keresőfa-tulajdonságot, így elég arra ügyelni, hogy az új p ne sértse meg a kupac-tulajdonságot. Ez pedig elérhető úgy, hogy a friss csúcsot szükség szerint felfelé forgatjuk.

Bizonyítás. Legyenek $k_1 < k_2 < \ldots < k_n$ az F-ben tárolt kulcsok, és vezessük be az X_{ij} indikátorváltozókat:

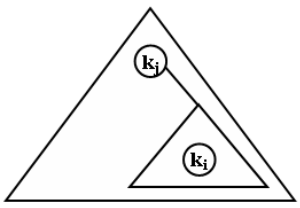
$$X_{ij} = \begin{cases} 1, & \text{ha a kulcsok } [k_i, k_j] \text{ intervallumában } k_j \text{ csúcsa a legnagyobb prioritású} \\ 0, & \text{különben.} \end{cases}$$

Hozzátesszük, hogy itt $[k_i, k_j]$ és $[k_j, k_i]$ ugyanazt az intervallumot jelentik: az összes olyan kulcsot, amely k_i és k_j között helyezkedik el a kulcsok rendezésére nézve.

7. Állítás. *Legyen*

$$X_i = \sum_{j=1}^{n} X_{ij}.$$

Ekkor X_i éppen a csúcsok száma a gyökértől a k_i kulcsot tartalmazó csúcsig vezető úton.



Bizonyítás. Tegyük fel, hogy k_j őse k_i-nek. Ez esetben $X_{ij} = 1$, mert a fenti ábrán látható helyzetben a $[k_j, k_i]$ intervallum benne van k_j jobb részfájában. Tehát a $[k_j, k_i]$ intervallumba eső kulcsok közül a k_j prioritása a legnagyobb, hiszen a részfa minden elemének prioritásánál nagyobb. Hasonlóan érvelhetnénk, ha k_i a k_j bal részfájában lenne. Meg kell még mutatnunk, hogy ha k_j nem őse k_i-nek, akkor $X_{ij} = 0$. Ez azért igaz, mert ekkor létezik egy k_j-től különböző k_l közös ősük. Legyen a k_l kulcsot tartalmazó csúcs a legalacsonyabb közös ős. Ekkor k_l benne van a $[k_i, k_j]$ intervallumban, és a prioritása nagyobb, mint k_j prioritása. (Itt megengedjük azt a lehetőséget is, hogy $k_i = k_l$.) □

Visszatérve a tétel bizonyításához, a gyökértől k_i-ig menő úton a csúcsok várható száma ezek alapján:

$$\mathbf{E}(X_i) = \sum_{j=1}^{n} \mathbf{E}(X_{ij}) = \sum_{j=1}^{n} \mathbf{P}(X_{ij} = 1).$$

Annak a valószínűsége, hogy k_j kapja a $[k_i, k_j]$ intervallum elemei közül a legnagyobb prioritást (feltéve, hogy $i < j$) a feladat (c) részét alkalmazva (ekkor $d = j - i + 1$):

$$\frac{1}{j - i + 1}.$$

Hasonlóan, ha $i > j$, akkor ez a valószínűség

$$\frac{1}{i - j + 1}.$$

Ebből következően

$$\mathbf{P}(X_{ij} = 1) = \frac{1}{|j - i| + 1}.$$

A $\sum_{j=1}^{n} \mathbf{P}(X_{ij} = 1)$ kifejezésben egy fix $|j - i|$ maximum kétszer fordulhat elő, innen

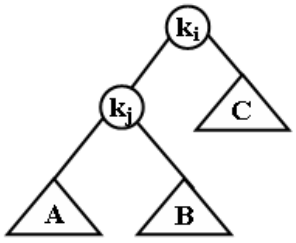
$$\sum_{j=1}^{n} \mathbf{P}(X_{ij} = 1) \leq \sum_{l=1}^{n} \frac{2}{l} = 2H_n \leq 2(\log n + 1).$$

□

A forgatások számának elemzése

Arra vagyunk itt kíváncsiak, hogy várhatóan hány forgatásra van szükség, amikor egy k kulcsot törlünk az n elemű F falomból. Az ehhez szükséges műveletsort időben megfordítva is nézhetjük, amiből arra jutunk, hogy ugyanennyi lesz a forgatások száma, amikor k-t beillesztjük az $n - 1$ elemű F'-be. Legyen

$$Y_{ij} = \begin{cases} 1, & \text{ha } k_i \text{ prioritása a legnagyobb a } [k_i, k_j]\text{-ben, és } k_j \text{ prioritása} \\ & \text{a második legnagyobb a fenti intervallumban,} \\ 0, & \text{különben.} \end{cases}$$



Például a fenti ábra által mutatott helyzetben, amikor is k_i-t lefelé forgatjuk k_j-nél, teljesül,[17] hogy

$$[k_j, k_i] \subseteq \{k_j, k_i\} \cup \{B \text{ részfa kulcsai}\}.$$

Ezek között nyilván k_i prioritása a legnagyobb, k_j prioritása pedig a második legnagyobb, azaz $Y_{ij} = 1$. Hasonló mondható, ha k_j jobb oldali gyermeke k_i-nek. Ezzel lényegében bizonyítottuk a következőt:

15. Tétel. *A k_i kulcs törlésekor a forgatások száma nem több, mint $\sum_{j=1}^{n} Y_{ij}$.*

Legyen ezután

$$Y_i = \sum_{j=1}^{n} Y_{ij},$$

[17] Ennek meggondolásához hasznos az a tény, hogy egy bináris keresőfa kulcsait az inorder bejárás növekvő sorrendben adja meg.

és adjunk felső korlátot a várható értékére:

$$\mathbf{E}(Y_i) = \sum_{j=1}^{n} \mathbf{E}(Y_{ij}) = \sum_{j=1}^{n} \mathbf{P}(Y_{ij} = 1), \tag{2.11}$$
$$\mathbf{P}(Y_{ij} = 1) = \frac{1}{(j-i+1)(j-i)}, \text{ ha } i < j,$$
$$\mathbf{P}(Y_{ij} = 1) = \frac{1}{(i-j+1)(i-j)}, \text{ ha } i > j.$$

Itt használtuk a feladat (d) részét $d = j - i + 1$, illetve $d = i - j + 1$ választással. Innen, a (2.11) sort folytatva:

$$\sum_{j=1}^{n} \mathbf{P}(Y_{ij} = 1) \le \sum_{l=2}^{n} \frac{2}{l(l-1)} = \sum_{l=2}^{n} 2\underbrace{\Big(\frac{1}{l-1} - \frac{1}{l}\Big)}_{\substack{\text{teleszkópos} \\ \text{összeg}}} \le 2.$$

Tehát a forgatások számának várható értéke törléskor ≤ 2.

2. Következmény. *Az n elemű falom esetén egy elem törlésének és beillesztésének a várható költsége is $O(\log n)$.*

Bizonyítás. A törlés a törlendő elem kereséséből, majd pedig annak lefelé mozgatásából áll. Az első részfeladat várható költsége $O(\log n)$, a másodiké a 15. tétel szerint $O(1)$, az összes költség tehát $O(\log n)$.

A k beillesztésének összköltsége arányos lesz a beillesztett, már F-ben levő k keresésének úthosszával, plusz a k törlésekor fellépő forgatások számával. Két tételünk szerint ezek összege is várhatóan $O(\log n)$. □

1. Megjegyzés. *Pontosabb elemzéssel megmutatható*[a]*, hogy tetszőleges $c > e$-re annak a valószínűsége, hogy az F fa magassága nagyobb, mint $2c \log n$, legfeljebb $n \left(\frac{n}{e}\right)^{-c \log(\frac{c}{e})}$, amiből az is következik, hogy a fa várható magassága is $O(\log n)$.*

[a] Lásd pl. [SA] 4.8. Lemma.

11. Feladat. *Mutassuk meg, hogy az F falom alakja megegyezik annak az F^* bináris keresőfának az alakjával, amelybe az F-beli kulcsokat illesztettük be az $F^* = \emptyset$ fából indulva naiv beszúrásokkal, mégpedig az F-beli prioritások szerint csökkenő sorrendben. (Tehát pl. ugyanaz a k kulcs van a két fa gyökerében, ugyanaz a k_1 és k_2 kulcs a gyökér bal, illetve jobb fiában, stb.)*

A korábbi feladat (b) részére gondolva az F úgy is tekinthető, mint egy naiv beszúrásokkal véletlen bemeneti sorrend szerint épített bináris keresőfa. Érvényesek tehát rá az 2.1. szakaszbeli megállapítások.

2.8 Síkbeli autopartíció

Most ismét egy geometriai/grafikai hátterű feladatot fogunk tárgyalni. Itt *tartomány*on olyan síkbeli konvex halmazt értünk, amelyet véges sok szakasz határol. Szokás még ezeket konvex sokszögeknek, esetleg konvex sokszöglemezeknek is nevezni.

8. Számítási feladat. *Adott egy T téglalap a síkon, benne n egymást nem metsző $s_1, \ldots, s_n$ szakasz. Bontsuk fel a T téglalapot minél kevesebb tartományra úgy, hogy az s_i szakaszok csak tartományok határán legyenek, azok belsejében ne legyen pontjuk.*

Ez a feladat fontos több grafikus alkalmazásnál, meghatározó szerepet játszik, egyebek között a láthatóság számítására szolgáló *Painter's algoritmusnál.*[18] A cél az, hogy lehetőleg kevés tartományt kapjunk, és viszonylag gyorsan.

Az RA-algoritmus

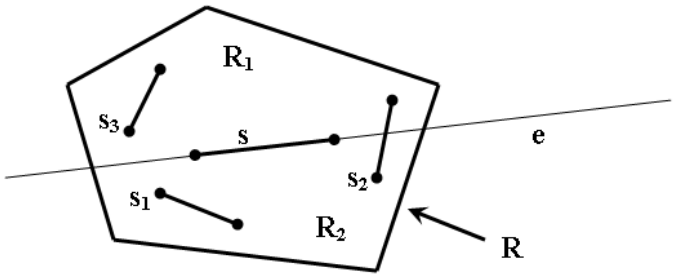
Az RA-algoritmus (randomised autopartitioning) rekurzív módon egy bináris fát hoz létre, melynek csúcsai (R, L) alakúak, ahol R egy tartomány, L pedig az R belsejében levő szakaszok egy listája. Az R tartományok a kiinduló T téglalap részei, az L listákon levő $s \in L$ szakaszok pedig valamelyik s_i bemeneti szakasznak részei.

A fa gyökere (T, L) lesz, ahol $L = \{s_1, \ldots, s_n\}$, az input szakaszok listája. Az RA-eljárás általános lépése a következő: tegyük fel, hogy éppen az (R, L) csúcsnál tartunk (kezdetben ez a csúcs a gyökér lesz). Az L az R belsejében levő szakaszok egy listája.

1. Ha $|L| = 0$, azaz R-ben már nincs szakaszunk, akkor az (R, L) feldolgozása befejeződik. (Ez a rekurzió megállási feltétele.)

2. Különben legyen s az L egy véletlen eleme (egyenletes eloszlás szerint).

3. Vágjuk ketté az R tartományt az s szakasz e egyenesével. Az így kapott tartományok R_1 és R_2.



   Készítsük el az R_i tartományokba eső szakaszok L_i listáit. Ennek a menetét az ábrával szemléltetjük. Itt s_1 az L_2-be kerül, s_3 az L_1-be, s_2-t kettévágjuk s_4 és s_5 szakaszdarabokká, s_4 az L_1-be, s_5 az L_2-be kerül. (R_1, L_1) és (R_2, L_2) legyenek az (R, L) fiai.

4. Hívjuk meg rekurzívan ugyanezt az eljárást az (R_1, L_1), majd pedig az (R_1, L_1) csúcsra.

Az RA-algoritmus elemzése

Határozzuk meg, hogy hány tartományunk lesz az RA-algoritmus lefutása után! Tudjuk, hogy egy szakaszdarabka 2 tartomány határán van, és a T-t kivéve minden, az eljárás futása során keletkező tartományt határol L-ből származó szakasz egy darabja. Ezért ha $n \geq 1$, akkor a végső (levélhez tartozó) tartományok száma legfeljebb a vágásokkal keletkező szakaszdarabkák számának kétszerese. Tehát a keletkező szakaszdarabkák számát kell megbecsülnünk. Becsülhetjük a szakaszdarabok száma helyett az s_i szakaszokon levő osztópontok számát is, hiszen a kettő között szoros összefüggés van:

$$\text{az összes szakaszdarabok száma } = n + \text{ osztópontok száma},$$

[18] Lásd pl. [A] 7.3. szakasz.

mert kezdetben n db szakasz van, és minden további osztópont eggyel növeli a szakaszdarabok számát. Legyen mármost az input szakaszlistánk $\{s_1, \ldots, s_n\}$. Vezessük be a szükséges indikátorváltozókat:

$$Y_{ij} = \begin{cases} 1, & \text{ha } s_i \text{ egyenese az algoritmus futása során vágja az } s_j \text{ szakaszt} \\ & \text{(csak az éppen feldolgozott szakaszdarab tartományán belüli metszés számít),} \\ 0, & \text{különben.} \end{cases}$$

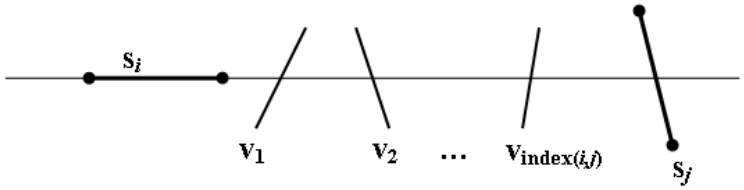
Legyen Y a következő valószínűségi változó:

$$Y = \sum_{i=1}^{n} \sum_{j=1}^{n} Y_{ij}.$$

Y értéke az algoritmus során az s_i szakaszokon keletkező osztópontok száma. Vizsgáljuk ennek a várható értékét a már megszokott módon:

$$\mathbf{E}(Y) = \mathbf{E}\left(\sum_{i=1}^{n} \sum_{j=1}^{n} Y_{ij}\right) = \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{E}(Y_{ij}) = \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{P}(Y_{ij} = 1).$$

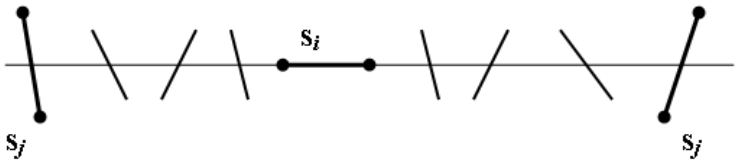
Jelöljük index(i, j)-vel azon s_l szakaszok számát, amelyeket s_i egyenese metsz, mielőtt elérné s_j-t (feltéve, hogy $Y_{ij} = 1$).



Ekkor $Y_{ij} = 1$ csak akkor fordulhat elő, ha az RA-algoritmus az $s_i, s_j, v_1, \ldots, v_{\text{index}(i,j)}$ és még esetleg további szakaszok közül (ez összesen legalább index(i, j)+2 szakasz) először s_i-t választja ki particionáló szakasznak. Tehát

$$\mathbf{P}(Y_{ij} = 1) = \frac{\text{kedvező esetek}}{\text{összes eset}} \leq \frac{1}{\text{index}(i, j) + 2}.$$

Próbáljuk mármost megválaszolni, hogy adott i-re és k-ra hány olyan j létezhet, amelyre index$(i, j) = k$. Például $k = 3$ esetén – a következő ábrán is láthatóan – 2 olyan s_j szakaszt találhatunk, melyre az index$(i, j) = 3$ fennáll.



Ennél több alkalmas s_j nyilván nem lehetséges, hiszen s_i-től indulva mindkét irányban legfeljebb 1 ilyen metszéspont létezhet. Ugyanezen okból a válasz a kérdésre általában, tetszőleges k-ra: legfeljebb 2. Az osztópontok várható száma ezt is figyelembe véve a következőképpen alakul:

$$\mathbf{E}(Y) = \sum_{i=1}^{n} \sum_{j=1}^{n} \mathbf{P}(Y_{ij} = 1) \leq \sum_{i=1}^{n} \sum_{j=1}^{n} \frac{1}{\text{index}(i, j) + 2} \leq \sum_{i=1}^{n} \sum_{k=1}^{n} \frac{2}{k} = 2nH_n.$$

Várhatóan tehát $O(n \log n)$ tartományt kapunk. Megjegyezzük, hogy már az sem magától értetődő, hogy *létezik* ennyi tartományból álló autopartíció. Az algoritmus érdekes valószínűségi hátterű bizonyítást nyújt ilyen partíció létezésére.

Ha nem csak a várható érték érdekel bennünket, hanem például az $Y > \lambda 2nH_n$ esemény valószínűsége, ahol $\lambda > 0$, akkor ezt becsülhetjük a Markov-egyenlőtlenség segítségével. A $\lambda = 4$ választással a Markov-egyenlőtlenség szerint

$$\mathbf{P}(Y > 8nH_n) \leq \frac{1}{4}.$$

2.9 Az ujjlenyomat-módszer

– *Nézze meg a nagyítójával, Mr. Holmes!*
– *Azt teszem.*
– *Ugye tisztában van vele, hogy nincs két egyforma ujjlenyomat?*
– *Igen, hallottam már róla.*
Sir Arthur Conan Doyle: A norwoodi tűzeset

Több érdekes algoritmikus feladat részeként találkozhatunk a következő típusú igénnyel: *adott egy nagy U halmaz két eleme, x és y. Döntsük el, hogy $x = y$ igaz-e.*

Egy lehetséges, és gyakran erőteljes megoldást adó megközelítés ujjlenyomat képzésén alapszik. Az *ujjlenyomat-módszer* egy $f : U \mapsto V$ függvényt alkalmaz, ahol $|V|$ általában sokkal kisebb, mint $|U|$. Az $x \in U$ elem ujjlenyomata $f(x)$. Az $x \stackrel{?}{=} y$ kérdést úgy próbáljuk megválaszolni, hogy kiszámítjuk az $f(x)$ és $f(y)$ ujjlenyomatokat, majd ezek egyenlőségét vizsgáljuk. Ha $f(x) \neq f(y)$, akkor biztosan $x \neq y$. Ha viszont $f(x) = f(y)$, akkor még nem bizonyos, hogy $x = y$. Az $f(x) = f(y)$-ból az $x = y$-ra következtetés[19] lehet hibás, de gyakran elérhető, hogy ennek az esélye csekély legyen.

A hashelés maga is tekinthető az ujjlenyomat-módszer alkalmazásának. A $h : U \to [M]$ hash-függvények felfoghatók ujjlenyomat-számító függvényeknek. A következőkben további két alkalmazásból adunk ízelítőt.

Egy feladat a kommunikációs bonyolultság köréből

9. Számítási feladat. *A és B egymástól távol levő személyek. A-nál van az $x = (a_1, \ldots, a_n) \in \{0,1\}^n$ bitsorozat, B-nél pedig az $y = (b_1, \ldots, b_n) \in \{0,1\}^n$ bitsorozat. Céljuk annak eldöntése, hogy $x \stackrel{?}{=} y$, és ezt minél kevesebb bit elküldésével szeretnék megtudni (a végén mindkettőjüknek tudnia kell a helyes választ, nem elég csak az egyik félnek).*

Ismert[20], hogy ez teljes bizonyossággal csak úgy oldható meg, ha legalább $n+1$ bitet elküldenek a felek. Ha pl. A elküldi B-hez a teljes x szót (ez n bit), akkor B egyedül is ellenőrizni tudja, hogy $x = y$ teljesül-e. Ha igen, akkor 1-est küld A-nak, ha nem, akkor nullát. A feladat megoldásához tehát $n+1$ bit elküldése elegendő. Megmutatható, hogy ennyi kell is, ezt itt nem részletezzük.

[19] Ilyen következtetést alkalmaznak a személyazonosításnál, ahol U az emberek összességét jelenti, f pedig az igazi ujjlenyomat képzése. A következtetés nem teljesen hibátlan. A 2004-es madridi robbantások helyszínén talált egyik ujjlenyomat olyan mértékű egyezést mutatott egy oregoni férfiéval, hogy az ártatlanságára utaló temérdek bizonyíték ellenére is egészen addig fogva tartották, amíg az ujjlenyomat igazi gazdáját meg nem találta a spanyol rendőrség.

[20] [L] 9.1.3. Példa.

A véletlen segítségével, az ujjlenyomat-módszert használva sokkal kevesebb bit elküldésével is megoldható a feladat. Cserébe viszont az algoritmus (kis valószínűséggel) hibázhat. Pontosabban fogalmazva, olyan megoldást adunk, amelynél az elküldött bitek száma $O(\log n)$, a hibázás valószínűsége pedig $O(\frac{1}{n})$. Az ötlet az, hogy x helyett A az $f(x)$ ujjlenyomatot küldi el, ennek birtokában B teszteli az $f(x) = f(y)$ egyenlőséget.

Az ujjlenyomatot képező $f : U \mapsto V$ függvény értelmezési tartománya az n hosszú bitsorozatok $U = \{0,1\}^n$ halmaza. Az V értékkészlet pedig az $\mathbb{F}_p$ test lesz, ahol p egy véletlen p prímszám (egyenletes eloszlás szerint) az $[1,\tau]$ intervallumból. A τ értékét később határozzuk meg. Az f-et megadó összefüggés pedig

$$f(x) = f_p(x) = f_p(a_1, \ldots, a_n) = (a_1 2^{n-1} + a_2 2^{n-2} + \ldots + a_{n-1}2 + a_n) \mod p.$$

Másképpen mondva, x-et binárisan ábrázolt egésznek tekintjük, és ennek vesszük a p-vel való osztási maradékát. A fentebb vázolt protokoll akkor hibázik, ha $x \neq y$, de $f_p(x) = f_p(y)$. Megmutatjuk, hogy ennek a valószínűségére a következő korlát érvényes:

$$\mathbf{P}(f_p(x) = f_p(y) | x \neq y) \leq \frac{n}{\pi(\tau)},$$

ahol $\pi(\tau)$ a prímek száma az $[1,\tau]$ intervallumban. Ugyanis $f_p(x) = f_p(y)$ pont akkor teljesül, ha $\alpha = x_{val} - y_{val}$ osztható p-vel, ahol

$$x_{val} = a_1 2^{n-1} + a_2 2^{n-2} + \ldots + a_{n-1}2 + a_n,$$
$$y_{val} = b_1 2^{n-1} + b_2 2^{n-2} + \ldots + b_{n-1}2 + b_n.$$

Mivel $|\alpha| \leq 2^n$, α-nak legfeljebb n db prímosztója lehet, azaz csak legfeljebb n db p esetén fordulhat elő, hogy $f_p(x) = f_p(y)$. Az esemény bekövetkeztéhez vezető p prímek száma legfeljebb n, az összes eset száma pedig $\pi(\tau)$; az egyenlőtlenség tehát fennáll. Innen, a $\tau = (n \log n)^2$ választással elég nagy n esetén alkalmas $c > 0$ konstanssal érvényes a következő:

$$\mathbf{P}(\text{a protokoll hibázik}) = \mathbf{P}(f_p(x) = f_p(y) | x \neq y) \leq \frac{n}{\pi(\tau)},$$

amiből a Prímszámtétel alapján a hiba valószínűsége

$$\leq \frac{n \log \tau}{\tau} = \frac{2n(\log n + \log\log n)}{n^2 \log^2 n} \leq \frac{n}{n^2} c = \frac{c}{n} = O\left(\frac{1}{n}\right). \tag{2.12}$$

Itt használtuk, hogy $\frac{(\log n + \log\log n)}{\log^2 n}$ korlátos marad, amint n tart a végtelenhez.

A teljes protokoll tehát a következő: A választ egy véletlen p prímet az $[1,\tau]$ intervallumból. Kiszámítja az $f_p(x)$-et, majd a p-t és $f_p(x)$-et átküldi B-nek (pontosabban e számok bináris kódját küldi el). A p ismeretében a B kiszámítja az $f_p(y)$ értéket, majd megvizsgálja, hogy $f_p(x) = f_p(y)$ igaz-e, és az eredménytől függő bitet küld A-nak. Ez összesen legfeljebb mintegy $2\log_2 \tau$ bit elküldését jelenti. Mivel $\tau = (n \log n)^2$,

$$\log_2 \tau = \log_2 (n \log n)^2 = 2(\log_2 n + \log_2 \log n) \leq 4 \log_2 n,$$

vagyis mintegy $8 \log_2 n$ bit kommunikációjával kivitelezhető a protokoll.

Randomizált mintaillesztés

A szövegszerkesztő programok egyik igen gyakran használt funkciója a keresés: egy hosszabb s szövegben nézzük, hogy előfordul-e benne az m szó. Ennek itt azt az esetét tekintjük, amikor a szöveg és a keresett minta is bitekből áll.

Emlékeztetünk, hogy az s szónak *részszava* az m szó, ha vannak olyan (esetleg üres) y, z szavak, hogy $s = ymz$.

10. Számítási feladat. *Adottak az $s = s_1 \dots s_n$, és $m = m_1 \dots m_d$ bitsorozatok ($s_i, m_j \in \{0,1\}$). Keressük az m (minta) első részszóként való előfordulását s-ben.*

A feladat természetesen csak akkor érdekes, ha $d \le n$. A naiv algoritmus: m-et sorban összehasonlítjuk az s szó d bitből álló $s(j) = s_j s_{j+1} \cdots s_{j+d-1}$ részszavaival, $j = 1, 2, \dots, n-d+1$-re, illetve az első olyan i indexig, amelyre $m = s(i)$. Ennek a módszernek a költsége bitösszehasonlításokban mérve a legrosszabb esetben $d(n-d+1)$, vagyis ez négyzetes költségű algoritmus. A feladat megoldható determinisztikus lineáris időben is, mégpedig legfeljebb $2(n+d)$ betű-összehasonlítással.[21] Mi itt egy randomizált módszert ismerünk meg, a Rabin–Karp-algoritmust. Az algoritmus meglepően egyszerű, lényegében a naiv algoritmus logikáját ötvözi az előző szakaszban tárgyalt ujjlenyomatos egyenlőségteszttel. A Rabin–Karp-algoritmus várhatóan gyorsan megoldást ad, és az ötlete jól működik bizonyos kétdimenziós minták esetén is.

Legyen $\tau = d(n \log n)^2$. A Rabin–Karp-algoritmus menete nagy vonalakban a következő:

1. Válasszunk egy véletlen p prímszámot az $[1, \tau]$ intervallumból (egyenletes eloszlás szerint). Legyen kezdetben $j = 1$.
2. Számítsuk ki az $f_p(m)$ és $f_p(s(j))$ értékeket, majd ellenőrizzük, hogy $f_p(m) = f_p(s(j))$ teljesül-e (itt f_p az előző szakaszban bemutatott ujjlenyomatképző függvény).
   (a) Ha nincs egyenlőség, és még nem értük el s végét, akkor j-t eggyel növeljük, majd folytatjuk a 2. lépésnél.
   (b) Ha egyenlőség van, akkor illeszkedést jelzünk.

Az algoritmus a 2.(b) lépésnél hibázhat. Az ujjlenyomatok egyezéséből nem következik, hogy $m = s(j)$. A hiba valószínűsége a 2.(b) lépésnél

$$\mathbf{P}(f_p(m) = f_p(s(j)) | m \neq s(j)) \le \frac{d}{\pi(\tau)}, \tag{2.13}$$

az előző alkalmazási példánál látottakhoz hasonló meggondolás szerint. Annak valószínűsége, hogy *valahol* hibázunk, legfeljebb a (2.13) valószínűség n-szerese

$$\mathbf{P}(\text{valahol hibázunk}) \le \frac{nd}{\pi(\tau)}.$$

Használva, hogy $\tau = dn^2 \log^2 n$, a korábbi számolásunkhoz hasonlóan kapjuk, hogy

$$\mathbf{P}(\text{valahol hibázunk}) \le \frac{nd \log \tau}{\tau} = \frac{nd \log \tau}{dn^2 \log^2 n} = \frac{1}{n} \cdot \frac{\log \tau}{\log^2 n} = O\left(\frac{1}{n}\right),$$

mert $\frac{\log \tau}{\log^2 \tau}$ korlátos marad. A hibázás valószínűsége tehát $O(\frac{1}{n})$. A Rabin–Karp-algoritmus 2.(b) lépésénél két dolgot tehetünk:

- Nem teszünk semmit, megállunk „igen” válasszal. Ez egy ún. *Monte Carlo-típusú* algoritmus. A válasza esetleg hibás, a hiba valószínűsége csekély: $O(\frac{1}{n})$.
- A másik lehetőség, hogy innentől átállunk (teljesen) a naiv algoritmusra az első tényleges illeszkedésig (amikor $s(\ell) = m$), illetve ilyen híján az s végéig. Ennek a fázisnak a költsége $O(dn)$. Az így adódó módszer egy ún. *Las Vegas-típusú* algoritmus: olyan randomizált módszer, amely sosem ad téves választ.

[21] Nevezetes ilyen módszer (tetszőleges véges abc feletti s és m esetére) a Knuth–Morris–Pratt-algoritmus; lásd [RISz], 323–327.

Mit mondhatunk a kétféle eljárás költségéről? Uniform költségszámítás esetén egy $f_p(m) = f_p(s(j))$ összehasonlítás költsége egységnyi. Használva, hogy

$$f_p(s(j+1)) = 2f_p(s(j)) - 2^d s_j + s_{j+d} \mod p,$$

az $f_p(s(j+1))$ gyorsan számolható $f_p(s(j))$-ből. Így az összes f_p számolások költsége $O(n+d)$. Ebből következik, hogy a Monte Carlo-változat összköltsége $O(n+d)$.

Hasonló igaz a Las Vegas-változatra is: jelölje A azt az eseményt, hogy az algoritmus hibázik, B pedig a komplementerét, és legyen ξ az algoritmus uniform költsége.[22] A feltételes várható érték tétele szerint alkalmas $c > 0$ konstanssal

$$\mathbf{E}(\xi) = \mathbf{P}(B)\mathbf{E}(\xi|B) + \mathbf{P}(A)\mathbf{E}(\xi|A) \le c(n+d) + c\frac{1}{n}nd = O(n+d),$$

hiszen $\mathbf{E}(\xi|A) = O(nd)$ és $\mathbf{E}(\xi|B) = O(n+d)$. Az uniform költségszámítás reálisnak mondható, ha p ábrázolható egy gépi szóban.

12. Feladat. *Mutassuk meg, hogy mindkét változat várható logaritmikus költsége $O((n+d)\log n)$.*

13. Feladat. *Mutassuk meg, hogy a Rabin–Karp-algoritmus kiterjeszthető a lineáris költségkorlát megtartásával arra az esetre, amikor s és m betűi kettőnél nagyobb elemszámú I abc-ből valók.*

Ismert, hogy a Rabin–Karp-algoritmus kevésbé hatékony a gyakorlatban, mint a Knuth–Morris–Pratt-algoritmus, ha egyetlen m mintát keresünk s-ben. Ha viszont több ugyanolyan d hosszúságú $m^1, \ldots, m^k$ mintánk van, és arra vagyunk kíváncsiak, hogy ezek közül melyek fordulnak elő részszóként s-ben, akkor a Rabin–Karp-módszer igen erősnek bizonyul. Az ötlet az, hogy az m^i mintákat egy H hash-szerkezetbe szervezzük, mégpedig a $h = f_p$ mint hash-függvény segítségével. Utána az $s(j)$ szavakat keressük H-ban. A részleteket az olvasóra bízzuk:

14. Feladat. *Mutassuk meg, hogy az előbb megfogalmazott többszörös keresési feladat megoldható $O(n+kd)$ uniform költséggel.*

Polinomok ujjlenyomata

A 2.5. szakaszban tárgyaltak egy része is felfogható az ujjlenyomat-módszer alkalmazásának. Tekintsük most azt az esetet, amikor $U = U_{n,d}$ az $\mathbb{F}$ test feletti legfeljebb d fokú $f(x_1, \ldots, x_n)$ polinomok halmaza. A következő feladat a polinomazonosság ellenőrzésének egy lehetséges megfogalmazása.

11. Számítási feladat. *Adottak az $f(x_1, \ldots, x_n), g(x_1, \ldots, x_n) \in U_{n,d}$ polinomok. Igaz-e, hogy $f \equiv g$?*

Itt az *adott* azt jelenti, hogy $\alpha = (\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$ vektor esetén hatékonyan ki tudjuk számítani az $f(\alpha)$ és $g(\alpha)$ értékeket.

Legyen $T \subseteq \mathbb{F}$, $|T| = 4d$. A Schwartz–Zippel-lemma szerint ha $\alpha \in T^n$ véletlen vektor, és $f \not\equiv g$, akkor legalább $\frac{3}{4}$ valószínűséggel igaz lesz, hogy $f(\alpha) \neq g(\alpha)$. Ebben az értelemben a T-beli értékekből képzett véletlen behelyettesítés tekinthető jó ujjlenyomatnak az $U_{n,d}$ halmazra nézve.

[22] Emlékeztetünk rá (pl. [RISz] 7.10.), hogy egy algoritmus *uniform költsége* a végrehajtott elemi utasítások száma. A *logaritmikus költségszámítás* szerint egy elemi utasítás ára az utasítás bemenő adatainak az összhossza.

2.10 Minimális költségű feszítőfa keresése

– *Dobjatok ki minden súlyos holmit... mindent!*
Jules Verne: A rejtelmes sziget[23]

Legyen $G = (V, E, c)$ valós számokkal súlyozott élű, összefüggő irányítatlan gráf (továbbra is használjuk az $n = |V|$ és $e = |E|$ jelöléseket). Az $f \in E$ él költségét (súlyát) $c(f)$ jelöli. A $F \subseteq G$ részfa feszítőfája G-nek, ha F tartalmazza a G összes csúcsát. Az F feszítőfa költsége a $c(f_1) + c(f_2) + \cdots + c(f_{n-1})$ összeg, ahol $f_1, \ldots, f_{n-1}$ az F élei.

A minimális feszítőfa keresésének feladata: adott a súlyozott élű, összefüggő irányítatlan G gráf. Keressünk benne olyan F feszítőfát, amelynek a költsége a lehető legkisebb.

Kruskal és Prim klasszikus algoritmusai[24] $O((n + e) \log n)$ lépésszámú (uniform költségű) megoldást adnak állistával megadott G esetében. Nevezetes nyitott kérdés, hogy létezik-e lineáris idejű, vagyis $O(n + e)$ költségű módszer.

A véletlen segítségével elérhető a bűvös határ: 1994-ben David R. Karger, Philip N. Klein és Robert E. Tarjan [KKT] lineáris várható idejű randomizált módszert javasoltak. Megoldásuk a véletlen mintavételezés kivételesen szép és hatékony algoritmikus alkalmazása.

Az eljárás ismertetéséhez szükségünk lesz bizonyos előkészületekre. A tárgyalás egyszerűsítésére feltesszük, hogy a bemeneti G gráfban nincs két azonos súlyú él. Célszerű lesz megengednünk, hogy G nem feltétlenül összefüggő. Ekkor természetesen minimális költségű feszítőerdőt (röviden mkf-et) keresünk; ennek részfái minimális költségű feszítőfák a G megfelelő összefüggő komponenseiben.

A Borůvka-algoritmus

Az eljárás bemenete az állistával adott, nem feltétlenül összefüggő, súlyozott élű $G = (V, E, c)$ gráf. Az eredmény egy $F \subseteq G$ mkf éleinek a listája. Kezdetben $F = \emptyset$. A fő lépések a következők:

1. Minden $v \in V$ csúcshoz keressük meg a legkisebb súlyú $(v, w) \in E$ élet; ezt színezzük kékre. Az izolált pontokat töröljük G-ből.
2. A kék éleket adjuk F-hez.
3. Húzzuk össze a G kék éleit, az így kapott gráf[25] legyen $G_1 = (V_1, E_1, c)$.
4. Ha $|V_1| \leq 1$, akkor megállunk, ha pedig $|V_1| \geq 1$, akkor legyen $G := G_1$ és menjünk vissza az 1. lépéshez.

A piros-kék algoritmus segítségével megmutatható[26], hogy végül az F-beli élek egy mkf-et alkotnak. Az 1–3. lépések egy végrehajtását egy *Borůvka-menetnek* nevezzük.

8. Állítás. *Az első Borůvka-menet végén G_1 csúcsainak száma legfeljebb $\frac{n}{2}$.*

Bizonyítás. Legyen a menetben kapott kék fák csúcsainak száma $k_1, \ldots, k_s$. Itt $k_2 \geq 2$, mert az izolált pontokat töröltük. Tehát

$$n \geq k_1 + k_2 + \cdots + k_s \geq 2s,$$

amiből $s \leq \frac{n}{2}$ adódik. □

[23] Fordította: Majtényi Zoltán.
[24] Lásd pl. [RISz] 6.6.
[25] G_1 csúcsai a G kék fái lesznek; az élei pedig a különböző kék fák között menő G-beli élek. Hurokéleket nem engedünk meg, és a párhuzamos élek közül csak a minimális súlyút tartjuk meg. A megtartott éleknél megjegyezzük, hogy melyik E-beli élből származnak.
[26] Lásd pl. [RISz] 155–156.

15. Feladat. *Mutassuk meg, hogy egy Borůvka-menet megvalósítható $O(n+e)$ költséggel.*

Nehéz élek és mkf-verifikálás

Legyen $F \subseteq G$ egy rögzített erdő, és $u \neq v$ a G két csúcsa. Legyen

$$w_F(u,v) := \begin{cases} \infty, \text{ ha } u \text{ és } v \text{ nincsenek az } F \text{ ugyanazon komponensében,} \\ \text{különben pedig a maximális élsúly az } u \text{ és } v \text{ közötti egyetlen } F\text{-beli úton.} \end{cases}$$

7. Definíció. *Legyen $G = (V, E, c)$ súlyozott élű irányítatlan gráf, F egy erdő G-ben. Az $(u,v) \in E$ él F-nehéz, ha $c(u,v) > w_F(u,v)$. Az $(u,v) \in E$ él F-könnyű, ha $c(u,v) \leq w_F(u,v)$.*

Megjegyezzük, hogy maguk az F élei nyilván F-könnyűek, hiszen esetükben $c(u,v) = w_F(u,v)$.

9. Állítás. *Legyen F tetszőleges G-beli erdő. Ekkor a G F-nehéz élei nincsenek benne a G egyetlen minimális költségű feszítőerdejében sem.*

Bizonyítás. Indirekt gondolkodva legyen $f = (u,v)$ egy F-nehéz él, és H egy mkf, aminek f az éle. Az f F-nehéz él, tehát F-ben létezik egy t út u és v között, és ennek minden f' élére $c(f') < c(f)$. Az u és v a $H \setminus \{f\}$ két különböző komponensében van, mivel a H erdő. A t viszont összeköti az u, v csúcsokat, tehát van t-nek olyan f' éle, amelyre $(H \setminus \{f\}) \cup \{f'\}$ is feszítőerdő. Az új feszítőerdő költsége kisebb a régiénél, ami ellentmond H minimalitásának. □

16. Tétel. *Legyen F a $G = (V, E, c)$ gráf egy feszítőerdeje. Az F akkor és csak akkor lesz mkf, ha F megegyezik a a G-beli F-könnyű élek H halmazával.*

Itt is használnunk kell azt a feltevésünket, hogy nincs két azonos súlyú él.

Bizonyítás. Az előző állítás mutatja, hogy ha $H = F$, akkor F szükségképpen mkf, hiszen $G \setminus F$ élei nem lehetnek benne egyetlen mkf-ben sem.

Nézzük ezután a fordított irányt! $H \supseteq F$ mindig igaz. Elég tehát azt belátni, hogy ha F mkf és $f = (u,v) \in H$, akkor $f \in F$ is teljesül. Ellenkező esetben az u és v közötti F-beli úton van olyan f' él, amelyre $c(f') > c(f)$. Valóban, mivel f F-könnyű, ezért van olyan f' él az úton, melyre $c(f') \geq c(f)$, másfelől pedig $f \neq f'$, és emiatt $c(f) \neq c(f')$. Mármost $(F \setminus \{f'\}) \cup \{f\}$ az F-nél kisebb költségű feszítőerdő, ami ellentmondást jelent. A tétel bizonyítása ezzel teljes. □

A következő igen fontos eredményt Valerie King és Komlós János [Ki] és [Ko] találták az 1980-as évek közepén. A bizonyítást és az algoritmus részleteit nem tárgyaljuk.

17. Tétel. *Tegyük fel, hogy éllistával adott a $G = (V, E, c)$ súlyozott élű gráf és abban egy $F \subseteq G$ erdő. Ekkor $O(n+e)$ uniform költségű determinisztikus algoritmussal megadható a G-beli F-könnyű élek listája.*

Az előző tételt is figyelembe véve látjuk, hogy a King–Komlós-algoritmus segítségével determinisztikusan lineáris időben ellenőrizhető, hogy egy adott F feszítőerdő mkf-e.

Mintavételezés G-ből

A randomizált számítások szemszögéből ez a Karger–Klein–Tarjan-algoritmus legérdekesebb része. Legyen $0 < p < 1$, és $G = (V, E, c)$ a bemeneti gráf. Utóbbiból élek véletlen kiválasztásával (szokásos szakkifejezést használva: véletlen mintavételezéssel) kapjuk a $G(p)$ gráfot. Ennek csúcshalmaza V. Az $f \in E$ élet illetően sorsolunk – a többi éltől teljesen függetlenül: p valószínűséggel f éle lesz $G(p)$-nek, $1-p$ valószínűséggel pedig nem lesz éle $G(p)$-nek. A $G(p)$ tehát egy véletlen részgráfja G-nek, és az éleinek várható száma $e \cdot p$.

A Karger–Klein–Tarjan-algoritmus elemzéséhez szükségünk lesz még két nevezetes valószínűségeloszlásra is:

8. Definíció. *Legyen $0 < p < 1$. Az X valószínűségi változót p paraméterű* geometriai eloszlású *valószínűségi változónak nevezzük, ha a $k = 1, 2, \ldots$ értékeket veszi fel, és*

$$\mathbf{P}(X = k) = p \cdot (1-p)^{k-1}.$$

A geometriai eloszlású X-re úgy gondolhatunk, hogy olyan pénzdarabot dobálunk fel teljesen függetlenül a korábbi dobásoktól, amelyiknél p a *fej* valószínűsége; X értéke éppen az első k, amelynél *fej*et dobunk. A p paraméterű geometriai eloszlású X várható értéke:

$$\mathbf{E}(X) = \sum_{k=1}^{\infty} k \cdot \mathbf{P}(X = k) = \sum_{k=1}^{\infty} k \cdot p(1-p)^{k-1} = \sum_{k=1}^{\infty} \sum_{j=k}^{\infty} p(1-p)^{j-1} =$$

$$= \sum_{k=1}^{\infty} p(1-p)^{k-1} \sum_{j=0}^{\infty} (1-p)^{j} = \sum_{k=1}^{\infty} p(1-p)^{k-1} \cdot \frac{1}{p} = \frac{1}{p}.$$

9. Definíció. *Az Y valószínűségi változót (n, p)* paraméterű negatív binomiális eloszlású *változónak nevezzük, ha $Y = X_1 + X_2 + \cdots + X_n$, ahol az X_i valószínűségi változók teljesen független p paraméterű geometriai eloszlású változók.*

Ekkor nyilvánvaló, hogy

$$\mathbf{E}(Y) = \sum_{i=1}^{n} \mathbf{E}(X_i) = \frac{n}{p}.$$

Az Y éppen az előzőekben említett éremmel végzett (teljesen független) dobások száma az n-edik fejig. Ebből a szemléletből közvetlenül adódik, hogy

$$\mathbf{P}(Y = k) = \binom{k-1}{n-1} p^n (1-p)^{k-n}, \qquad k = n, n+1, \ldots$$

Az itt következő tétel nyújtja az alapötletet a Karger–Klein–Tarjan-algoritmus véletlent használó részéhez. Lehetőséget kínál arra, hogy gyorsan eldobhassunk G-ből sok olyan élet (nevezetesen az F-nehéz éleket), amely nem lehet eleme mkf-nek.

18. Tétel. *Legyen $G = (V, E, c)$ súlyozott élű irányítatlan gráf, $|V| = n$, amelyben nincs két azonos élsúly. Legyen $0 < p < 1$, és F a $G(p)$ gráf minimális költségű feszítőerdeje. Jelölje ξ a G gráf F-könnyű éleinek számát. Ekkor $\mathbf{E}(\xi) \leq \frac{n}{p}$.*

Bizonyítás. Legyenek $f_1, \ldots, f_e$ a G élei. Nyilván feltehető (esetleg a számozás megváltoztatásával), hogy $c(f_1) < c(f_2) < \cdots < c(f_e)$. Azt is feltehetjük, hogy a $G(p)$ gráfot ebben a sorrendben építjük: először f_1 kapcsán sorsolunk, utána f_2 jön, stb. Közben a Kruskal-algoritmus alkalmazásával kapjuk az F feszítőerdőt. Kezdetben F üres.

Tegyük fel, hogy az $f_1, f_2, \ldots, f_{i-1}$ éleket már feldolgoztuk. Tekintsük f_i-t. Az f_i él pontosan akkor lesz F-könnyű, ha a végpontjai az eddig kialakult F különböző komponenseiből valók. Különben nyilván F-nehéz lesz, hiszen minden korábbi élnél nagyobb a súlya. Megjegyezzük, hogy az f_i könnyű vagy nehéz volta az $i-1$. sorsolás után már nem változik: ha nehéz volt közvetlenül a sorsolás után, akkor az is marad, hiszen F-ből nem törlünk élet. Ha pedig F-könnyű volt, akkor ezt a további élek már nem tudják megváltoztatni, mert a súlyuk nagyobb, mint $c(f_i)$.

Az i-edik sorsolás előtt tehát már világos, hogy f_i könnyű-e. Az f_i pontosan akkor lesz éle $G(p)$-nek, ha a sorsolás eredménye *fej*. Ha pedig emellett még F-könnyű is, akkor bekerül F-be, különben pedig nem.

Nevezzünk egy sorsolást *fontosnak*, ha F-könnyű éllel kapcsolatos. A fontos sorsolások közül legfeljebb $n-1$ lehet *fej*, ugyanis ezek adják az F éleit, és egy G-beli erdőnek legfeljebb $n-1$ éle lehet. A sorsolások végeztével sorsoljunk még annyit – ezek a *végső* sorsolások –, hogy a fontos és a végső sorsolások összesen n fejet adjanak. Legyen a fontos és végső sorsolások száma Y. Az Y definíció szerint (n,p) paraméterű negatív binomiális eloszlású valószínűségi változó, hiszen egy $p, 1-p$ éremmel dobálunk egymástól teljesen függetlenül az n-edik *fej*ig. Nyilvánvaló másfelől, hogy $\xi \le Y$, ugyanis ξ éppen a fontos dobások száma. Innen várható értékekre térve kapjuk a tételt:

$$\mathbf{E}(\xi) \le \mathbf{E}(Y) = \frac{n}{p}.$$

□

A Karger–Klein–Tarjan-algoritmus és elemzése

A bemenet a súlyozott élű, irányítatlan $G = (V, E, c)$ gráf; a kimenet a G-beli F mkf élhalmaza. Kezdetben F üres. A módszer Borůvka-menetek és mintavételi fázisok váltogatásával oldja meg a feladatot. Az előbbi típusú lépés a csúcsok számát csökkenti, a második pedig (várható értékben) az élszámot fogyasztja erőteljesen. Az algoritmus fő lépései a következők:

MKF-algoritmus

1. Alkalmazzunk 3 egymás utáni Borůvka-menetet. Legyen a kék élek összehúzásával kapott gráf G_1, a kék élek halmaza C. Ha G_1 üres, akkor $F := C$, és vége.

2. Képezzük a $G_2 := G_1\left(\frac{1}{2}\right)$ gráfot.

3. Alkalmazzuk (rekurzívan) az MKF-algoritmust G_2-re, ennek a minimális költségű feszítőerdeje legyen F_2.

4. King–Komlós algoritmussal töröljük G_1-ből az F_2-nehéz éleket; az így kapott gráf legyen G_3.

5. (Rekurzívan) MKF-fel számítsuk ki G_3 minimális költségű feszítőerdejét, ez (mint élhalmaz) legyen F_3.

6. Legyen végezetül $F := C \cup F_3$.

Helyesség: A Borůvka-algoritmus működése (helyessége) miatt a G-beli mkf $C \cup F^*$ alakú, ahol F^* mkf G_1-ben. Másfelől pedig $F^* = F_3$, mivel G_1-ből csupa nehéz élek eldobásával kaptuk G_3-at; a nehéz élek pedig nem lehetnek benne egyetlen mkf-ben sem.

A módszer *időigényének* elemzéséhez legyen $T(n,e)$ az algoritmus várható költségének maximuma legfeljebb n pontú és legfeljebb e élű input gráfok esetén. Vegyük ezután sorra a módszer egyes lépéseit:

Az első lépés költsége a Borůvka-menettel kapcsolatos feladatra tekintettel $O(n+e)$. G_1-nek legfeljebb $\frac{n}{8}$ csúcsa és legfeljebb e éle van.

A második lépés uniform költsége $O(n+e)$, G_2-nek legfeljebb $\frac{n}{8}$ csúcsa és várhatóan legfeljebb $\frac{e}{2}$ éle van.

A harmadik lépést illetően legyen p_i annak a valószínűsége, hogy a G_2 gráfnak éppen i éle van. A lépés költsége a teljes várható érték tétele alapján legfeljebb $\sum_{i=0}^{e} p_i T(\frac{n}{8}, i)$.

A negyedik lépés költsége (King–Komlós-algoritmus) $O(n+e)$.

Az ötödik lépés vizsgálatához legyen q_j annak a valószínűsége, hogy G_3-nak j éle van. Ezzel a jelöléssel élve az ötödik lépés ára legfeljebb $\sum_{j=0}^{e} q_j T(\frac{n}{8}, j)$.

A hatodik lépés költsége nyilvánvalóan $O(n)$. Az egyes lépéseinkre kapott becsléseket összeadva kapjuk, hogy alkalmas $c > 0$ állandóval

$$T(n,e) \le \sum_{i=0}^{e} p_i T\left(\frac{n}{8}, i\right) + \sum_{j=0}^{e} q_j T\left(\frac{n}{8}, j\right) + c(n+e).$$

Innen $n+e$ szerinti indukcióval megmutatjuk, hogy $T(n,e) \le 2c(n+e)$, feltéve, hogy a c állandót elég nagynak választottuk ahhoz, hogy a korlát kis n, e értékekre igaz legyen. Ekkor ugyanis elég csak az indukciós lépéssel foglalkozni. Mármost az indukciós feltevés szerint

$$T(n,e) \le 2c \sum_{i=0}^{e} p_i \left(\frac{n}{8} + i\right) + 2c \sum_{j=0}^{e} q_j \left(\frac{n}{8} + j\right) + c(n+e).$$

Vegyük észre, hogy $\sum_i p_i = 1$, $\sum_i i p_i \le \frac{e}{2}$, továbbá, hogy $\sum_j q_j = 1$ és $\sum_j j q_j \le \frac{n}{4}$ (itt az előző tételt alkalmaztuk a G_1 gráfra és a $p = \frac{1}{2}$ valószínűségre). Az egyenlőtlenség jobb oldala így becsülhető tovább:

$$\le c\frac{n}{4} + ce + c\frac{n}{4} + c\frac{n}{2} + c(n+e) = cn + ce + c(n+e) = 2c(n+e).$$

A módszer várható költsége valóban lineáris. Indoklás nélkül megjegyezzük, hogy erősebb állítás is igaz: a költség *nagy valószínűséggel* lineáris lesz.

3. fejezet

Véletlen és létezés

Itt a diszkrét matematika egyik alapvető fontosságú bizonyítási technikáját, a *véletlen módszert* ismerjük meg. Ezt *Erdős-módszernek* is szokás nevezni: a világhírű magyar matematikus, Erdős Pál fedezte fel,[1] és tőle származik néhány a legerősebb alkalmazásai közül. Dióhéjban úgy fogalmazhatunk, hogy a véletlen módszerrel különböző struktúrák létezését bizonyítjuk a véletlen segítségével. Alkalmas valószínűségi térben megmutatjuk, hogy nem nulla a valószínűsége annak, hogy a kérdéses struktúra létezik. Innen következik a struktúra létezése. Az állításokban magukban nincs szerepe a véletlennek, az csupán a bizonyítás eszközéül szolgál. A módszer tekinthető egyfelől tisztán létezést bizonyító matematikai technikának. Érdekessége és igen hasznos vonása másfelől, hogy közel van a hatékony algoritmusokhoz is: gyakran előfordul, hogy a tiszta létezést biztosító gondolatmenet kis módosítása hasznos randomizált algoritmust ad ahhoz, hogy megfelelő tulajdonságú objektumot találjunk.

Az előző fejezetben megismert RA-algoritmust tekinthetjük első példának. A véletlenre támaszkodva megmutattuk, hogy n egymást nem metsző szakaszhoz (a síkon) létezik olyan autopartíció, amely mindössze $O(n \log n)$ tartományból áll. A következőkben a véletlen módszer néhány jellegzetes alkalmazását mutatjuk be, mindenütt utalva az algoritmikus vonatkozáskora is.

3.1 Hipergráfok 2-színezése

19. Tétel (Erdős)**.** *Legyenek $H_1, \dots, H_m$ az U halmaz r elemű részhalmazai. Tegyük fel, hogy $m \le 2^{r-1}$. Ekkor U elemei megszínezhetők 2 színnel (piros, fehér) úgy, hogy egyetlen H_i sem lesz egyszínű.*

Bizonyítás. Színezzük az $u \in U$ pontokat $\frac{1}{2} - \frac{1}{2}$ valószínűséggel piros, vagy fehér színűre, egymástól teljesen függetlenül. Jelöljük R_i-vel azt az eseményt, hogy „H_i egyszínű”. Ez akkor következik be, ha H_i minden pontja piros vagy, ha a H_i minden pontja fehér. Innen

$$\mathbf{P}(R_i) = \frac{1}{2^r} + \frac{1}{2^r} = \frac{1}{2^{r-1}},$$

$$\mathbf{P}(\text{van egyszínű } H_i) = \mathbf{P}(\bigcup_{j=1}^{m} R_j) < \sum_{j=1}^{m} \mathbf{P}(R_j) = \frac{m}{2^{r-1}} \le 1.$$

[1] Erdőssel lényegében egyidőben Claude Shannon, az információelmélet alapító atyja is eljutott ehhez a gondolathoz: híres-nevezetes csatornakódolási tételét véletlen kódválasztással bizonyította. Lásd: [GyGyV], [Gy].

A szigorú egyenlőtlenség azért igaz, mert az R_j események nem egymást kizárók. Összességében azt kaptuk, hogy $\mathbf{P}(\text{van egyszínű } H_i) < 1$, vagyis van olyan 2-színezése U-nak, amelynél nincs egyszínű H_i. Létezik tehát a kívánt tulajdonságú színezés. □

Szeretnénk itt is hangsúlyozni, hogy az állításban magában nyoma sincs a véletlennek. A bizonyítás során a véletlen mégis kényelmes és hatékony eszköznek[2] bizonyult.

Megjegyezzük, hogy ha m nagyon közel van 2^{r-1}-hez, akkor a véletlenül választott színezés esetleg csak kis eséllyel lesz jó színezés, mert a $\mathbf{P}(\textit{van egyszínű } H_i)$ valószínűségre adott korlátunk közel van 1-hez. Ha viszont az $\frac{m}{2^{r-1}}$ korlát jóval kisebb, mint 1, akkor az előbbi valószínűség kicsi, és ezért a véletlen színezés komoly eséllyel jó. Ekkor a véletlen választás értelmes randomizált algoritmust eredményez.

Számszerű példa: legyen $m = 100000$, és $r = 20$. Annak a valószínűsége, hogy a véletlen színezés rossz lesz:

$$\mathbf{P}(\text{van egyszínű } H_i) < \frac{m}{2^{r-1}} = \frac{100000}{2^{19}} \leq \frac{2^{10} \cdot 2^7}{2^{19}} = \frac{1}{4}.$$

Ebben az esetben praktikusan is érdemes a pontok véletlen színezésével próbálkozni.

Megjegyezzük még, hogy itt egy nevezetes algoritmikus probléma speciális esetével találkoztunk. Ez a hipergráfok 2-színezésének feladata:

12. Számítási feladat. *Legyenek adottak az U alaphalmaz $H_1, \ldots, H_m$ részhalmazai. Döntsük el, hogy U elemei megszínezhetők-e 2 színnel (piros, fehér) úgy, hogy egyetlen H_i se legyen egyszínű.*

Ismert, hogy a feladat NP-teljes, ezt nem bizonyítjuk.

3.2 Ramsey-számok

A kombinatorikában, a logikában és a számítások világában is fontos szerepet játszik a következő fogalom:

10. Definíció. *Az $R(s,t)$* Ramsey-szám *az a legkisebb pozitív egész n, melyre igaz, hogy ha K_n (teljes gráf) éleit megszínezzük piros és fehér színnel, akkor lesz benne vagy csupa piros élű K_s, vagy csupa fehér élű K_t.*

Jól ismert ismert például, hogy $R(3,3) = 6$. Ennek az állításnak az egyik része az, hogy ha K_6 éleit piros és fehér színnel színezzük, akkor vagy lesz benne fehér K_3, vagy piros K_3. Ezt könnyen beláthatjuk: vegyünk a már színezett K_6-ból egy $v \in K_6$ csúcsot. A v-ből induló éleken az egyik szín legalább 3-szor fordul elő. Legyen x, y, z három olyan csúcs a gráfban, melyekből ugyanolyan színű (mondjuk piros) él vezet v-be. Ha az x, y, z közötti élek bármelyike piros, akkor az a v-be menő két piros éllel együtt egy piros K_3-at alkot. Ha viszont mindhárman fehérek, akkor együtt egy fehér K_3-at képeznek. Ezzel beláttuk, hogy $R(3,3) \leq 6$, a másik irányt az olvasóra bízzuk:

16. Feladat. *Színezzük meg a K_5 éleit két színnel úgy, hogy ne legyen benne egyszínű háromszög.*

[2] Azt is mondhatjuk, hogy a valószínűségek nyelvén kényelmes volt megfogalmaznunk egy leszámláláson alapuló érvelést. Az U lehetséges $2^{|U|}$ színezése közül megbecsültük a rosszak számát. Az ilyenek száma kevesebb, mint $2^{|U|} \cdot \frac{m}{2^{r-1}} \leq 2^{|U|}$, tehát nem lehet mind rossz.

Az sem teljesen magától értetődő, hogy $R(s,t)$ létezik egyáltalán. Tisztázza a létezés kérdését és felső korlátot is nyújt a következő tétel:

20. Tétel (Erdős–Szekeres)**.** *Legyenek $s,t \geq 2$ egészek. Ekkor*

$$R(s,t) \leq \binom{s+t-2}{s-1}.$$

Bizonyítás. Csak vázoljuk a gondolatmenetet. Ennek alapja a következő:

17. Feladat. *Legyen $s,t \geq 3$. Ekkor*

$$R(s,t) \leq R(s-1,t) + R(s,t-1)$$

Nyilvánvaló másfelől, hogy $R(2,t) = R(t,2) = t$. Ezután a feladat állítását használva $s+t$ szerinti indukcióval adódik a tétel. □

A tétel következménye, hogy

$$R(t,t) \leq \binom{2t-2}{t-1} < 2^{2t-2} < 2^{2t} = 4^t.$$

Nyitott kérdés, hogy létezik-e $\epsilon > 0$, hogy nagy t-re $R(t,t) < (4-\epsilon)^t$. A másik irránnyal, $R(t,t)$ alsó becslésével foglalkozik Erdős Pál legendás tétele. Az eredmény egyike a valószínűségi módszer első megjelenéseinek:

21. Tétel (Erdős, 1947)**.** *Ha $t \geq 3$, akkor*

$$R(t,t) > 2^{\frac{t}{2}}.$$

Nyitott a kérdés, hogy $c^t \leq R(t,t)$ igaz-e valamely $c > \sqrt{2}$-re. A jelenlegi becslések $R(t,t)$-t tehát $\sqrt{2}^t$ és 4^t közé helyezik.

Bizonyítás. Megmutatjuk, hogy alkalmas, a t függvényében elég nagy N-re a K_N teljes gráf élei színezhetők két színnel úgy, hogy benne minden K_t tarka (nem egyszínű) lesz. Színezzük a K_N éleit pirosra vagy fehérre $\frac{1}{2} - \frac{1}{2}$ valószínűséggel, egymástól teljesen függetlenül. Legyen $X \subseteq V(K_N)$ a csúcsoknak egy $|X| = t$ elemű részhalmaza. Ekkor

$$\mathbf{P}(\text{az } X\text{-en belüli élek egyszínűek}) = \frac{1}{2^{\binom{t}{2}}} + \frac{1}{2^{\binom{t}{2}}} = 2^{1-\binom{t}{2}}. \tag{3.1}$$

Jelölje p annak a valószínűségét, hogy a színezéskor keletkezik csupa egyszínű élekkel rendelkező K_t. Ekkor az előző formulát is használva

$$p = \mathbf{P}(\text{van } X \subseteq V(K_N),\ |X| = t, \text{ hogy az } X\text{-re épülő teljes gráf egyszínű}) <$$
$$< \sum_{\substack{X \subseteq V(K_N) \\ |X|=t}} 2^{1-\binom{t}{2}} = \binom{N}{t} \cdot 2^{1-\binom{t}{2}} \leq \frac{N^t}{t!} \cdot 2^{-\frac{t^2}{2}} \cdot 2^{\frac{t}{2}+1} = 2^{t\log_2 N} \cdot 2^{-\frac{t^2}{2}} \cdot \frac{2^{\frac{t}{2}+1}}{t!} \leq$$
$$\leq 2^{t(\log_2 N - \frac{t}{2})}.$$

Az utolsó egyenlőtlenségnél felhasználtuk, hogy

$$\frac{2^{\frac{t}{2}+1}}{t!} \leq 1,$$

ha $t \geq 3$. Végezetül

$$2^{t(\log_2 N - \frac{t}{2})} \leq 1,$$

ha $\log_2 N - \frac{t}{2} \leq 0$ teljesül, vagyis, ha $\log_2 N \leq \frac{t}{2}$. Legyen $N = \lfloor 2^{\frac{t}{2}} \rfloor$. Erre az N-re a $\log_2 N \leq \frac{t}{2}$ egyenlőtlenség biztosan teljesül, így a $p < 1$ is fennáll. Létezik tehát ekkor a K_N éleinek olyan 2-színezése, ahol minden K_t tarka. Ebből következik, hogy $R(t,t) > N$, amint azt állítottuk. □

Mint ahogyan a hipergráf 2-színezésénél láttuk, itt is igaz, hogy ha a számunkra kedvezőtlen esemény valószínűségére 1-nél sokkal kisebb korlátunk van, akkor véletlent használó algoritmussal jó eséllyel elkerülhetjük ezt az eseményt. A bizonyításban $2\log_2 N \leq t$ esetén tudtuk garantálni a kedvező esemény létét (a csupa tarka színezést). Ha itt a jobb oldal jóval nagyobb, mondjuk $t = 4\log_2 N$, akkor a véletlen színezés jó eséllyel ad csupa tarka színezést. Ekkor a rossz színezés p valószínűségét így becsülhetjük:

$$p < 2^{4\log_2^2 N - 8\log_2^2 N} = 2^{-4\log_2^2 N} \leq 2^{-4\log_2 N} = \frac{1}{N^4},$$

ez pedig kicsi lesz, ha N nagy. Itt is megfigyelhetjük tehát, hogy a véletlen módszeren alapuló tiszta egzisztenciabizonyítás nincs messze a véletlent használó, gyakorlati értelemben is eredményes algoritmustól.

3.3 Egy alsó korlát $\omega(G)$-re, és a Turán-tétel

A G irányítatlan gráf legnagyobb teljes részgráfjának a csúcsszámát $\omega(G)$ jelöli. A következőkben a véletlen segítségével bizonyítunk egy alsó korlátot az $\omega(G)$ értékre. Legyen G csúcshalmaza $V = \{v_1, v_2, \ldots, v_n\}$, a v_i foka pedig d_i.

10. Állítás. $\omega(G) \geq \sum_{i=1}^{n} \frac{1}{n-d_i}$.

Bizonyítás: N. Alon és J. Spencer érvelése [AZ] 32. fejezet. Legyen $\pi = w_1, w_2, \ldots, w_n$ a G csúcsainak egy véletlen permutációja egyenletes eloszlás szerint, vagyis bármely π permutáció valószínűsége $\mathbb{P}(\pi) = \frac{1}{n!}$. A π segítségével definiáljuk a $C_\pi \subseteq V$ csúcshalmazt: legyen $w_1 \in C_\pi$; $i > 1$ esetén a w_i pontosan akkor kerüljön a C_π halmazba, ha (w_i, w_j) éle G-nek minden $j < i$ esetén.

Nyilvánvaló a definícióból, hogy a C_π csúcshalmaz teljes gráfot feszít. Határozzuk meg a C_π méretének a várható értékét! Ismét az indikátormódszer lesz a segítségünkre. Legyen az X_i valószínűségi változó értéke 1, ha $v_i \in C_\pi$, különben pedig legyen az X_i értéke 0. Ekkor nyilvánvaló, hogy

$$|C_\pi| = X_1 + X_2 + \cdots + X_n.$$

Az $X_i = 1$ pontosan akkor következik be, ha a π permutációban v_i a legelső azon $v \in V$ csúcsok közül, amelyekre (v_i, v) nem éle G-nek. Ilyen tulajdonságú v csúcsból $n - d_i$ van, ezek közül pedig mindegyik egyforma eséllyel előzi meg a többit, vagyis $\mathbf{P}(X_i = 1) = \frac{1}{n-d_i}$. Innen

$$\mathbf{E}|C_\pi| = \sum_{i=1}^{n} \mathbf{E}X_i = \sum_{i=1}^{n} \mathbf{P}(X_i = 1) = \sum_{i=1}^{n} \frac{1}{n-d_i}.$$

Van tehát a G-ben legalább ekkora csúcsszámú teljes részgráf. □

Az állítás segítségével egyszerű bizonyítás adódik a klasszikus gráfelmélet egyik legendás tételére:

22. Tétel (Turán). *Legyen $p \geq 2$ egész, és $G = (V, E)$ egy irányítatlan gráf, amely nem tartalmaz p pontú teljes részgráfot. Ekkor*

$$|E| \leq \left(1 - \frac{1}{p-1}\right) \frac{|V|^2}{2}.$$

Bizonyítás. Először emlékeztetünk a Cauchy–Schwarz-egyenlőtlenségre. Legyenek $a_1, \ldots, a_n$ és $b_1, \ldots, b_n$ tetszőleges valós számok, akkor

$$(a_1 b_1 + a_2 b_2 + \cdots + a_n b_n)^2 \leq (a_1^2 + \cdots + a_n^2)(b_1^2 + \cdots + b_n^2).$$

Tegyük fel, hogy $V = \{v_1, \ldots, v_n\}$, és legyen a v_i foka d_i. Legyen ezután $a_i = \sqrt{n - d_i}$, $b_i = \frac{1}{\sqrt{n-d_i}}$, és alkalmazzuk a Cauchy–Schwarz-egyenlőtlenséget. Itt $a_i b_i = 1$, tehát

$$n^2 \leq \sum_{i=1}^{n} (n - d_i) \sum_{i=1}^{n} \frac{1}{n - d_i} \leq \sum_{i=1}^{n} (n - d_i)\omega(G).$$

Jól ismert, hogy $\sum_{i=1}^{n} d_i = 2|E|$, a tétel feltétele szerint pedig $\omega(G) \leq p - 1$. Ezeket használva

$$n^2 \leq (n^2 - 2|E|)(p - 1)$$

adódik, amiből egyszerű átrendezéssel kapjuk a Turán-egyenlőtlenséget. □

3.4 Nagy vágás irányítatlan gráfokban

11. Definíció. *Legyen $G = (V, E)$ irányítatlan gráf. Az $S \subseteq V$ csúcshalmaz a G egy* vágása. *Az S vágás $k(S)$* értéke *azon G-beli élek száma, amelyeknek egyik végpontja S-ben van, a másik $V \setminus S$-ben (ezek az ún. keresztező élek).*

A definícióban az $S = \emptyset$ és $S = V$ is megengedett, ezekben az esetekben $k(S) = 0$. Itt most arra vagyunk kíváncsiak, hogy milyen nagy $k(S)$ érhető el alkalmasan választott $S \subseteq V$ csúcshalmazzal. Ezzel foglalkozik a következő tétel. Legyen $e = |E|$.

23. Tétel. *Van olyan $S \subseteq V$, amelyre $k(S) \geq \frac{e}{2}$*

Bizonyítás. Sorsoljunk véletlen $S \subseteq V$-t. Ezt úgy tesszük, hogy minden $v \in V$ pontot $\frac{1}{2} - \frac{1}{2}$ valószínűséggel választunk S-be, illetve $V \setminus S$-be, a többi ponttól teljesen függetlenül. Ekkor $k(S)$ egy valószínűségi változó. Elég belátni, hogy $\mathbf{E}(k(S)) \geq \frac{e}{2}$, mert ebből következik olyan S létezése, amelyre $k(S) \geq \frac{e}{2}$. Legyen $f \in E$, és vezessünk be a következő indikátorváltozót:

$$Y_f = \begin{cases} 1, & \text{ha } f \text{ keresztező él,} \\ 0, & \text{különben.} \end{cases}$$

Ekkor nyilván

$$k(S) = \sum_{f \in E} Y_f \;\Rightarrow\; \mathbf{E}(k(S)) = \sum_{f \in E} \mathbf{E}(Y_f).$$

Használva, hogy Y_f indikátorváltozó,

$$\mathbf{E}(Y_f) = \mathbf{P}(Y_f = 1) = \frac{\text{kedvező esetek}}{\text{összes eset}} = \frac{2}{4} = \frac{1}{2},$$

mert $Y_f = 1$ a 4 lehetséges eset közül abban a 2 esetben lesz igaz, amikor az $f = (u, v)$ él egyik végpontja S-be kerül, a másik pedig nem. A keresett várható érték tehát

$$\mathbf{E}(k(S)) = \sum_{f \in E} \mathbf{E}(Y_f) = \sum_{f \in E} \frac{1}{2} = \frac{e}{2}.$$

□

A bizonyítás természetes randomizált algoritmust kínál, aminek részeként véletlen $S \subseteq V$ csúcshalmazt sorsolunk. Ezt lefuttatva várhatóan legalább $\frac{e}{2}$ keresztező élet kapunk. Ha itt az $\mathbf{E}(k(S))$ várható érték mellett valószínűségeket is szeretnénk tudni, akkor a Csebisev-egyenlőtlenséget alkalmazhatjuk. Ehhez viszont ismernünk kell a $k(S)$ szórását, vagy legalább tudnunk kell becsülni.

Most abban a szerencsés helyzetben vagyunk, hogy a $\mathbf{D}(k(S))$ szórás pontosan kiszámítható.

18. Feladat. *Mutassuk meg, hogy*

$$\mathbf{D}(k(S)) = \frac{\sqrt{e}}{2}.$$

(A szórás definícióját közvetlenül alkalmazhatjuk, észrevéve, hogy ha $f \neq f'$ két éle G-nek, akkor $\mathbf{E}(Y_f \cdot Y_{f'}) = \frac{1}{4}$.)

A szórás számolásához esetünkben használható a következő tétel is (nem bizonyítjuk):

24. Tétel. *Legyenek $\xi_1, \ldots, \xi_k$ páronként független valószínűségi változók, és tegyük fel, hogy $\mathbf{D}(\xi_i)$ létezik minden i-re. Legyenek $c_1, \ldots, c_k \in \mathbb{R}$. Ekkor*

$$\mathbf{D}^2(c_1\xi_1 + \ldots + c_k\xi_k) = c_1^2\mathbf{D}^2(\xi_1) + \ldots + c_k^2\mathbf{D}^2(\xi_k).$$

19. Feladat. *Mutassuk meg, hogy az Y_f indikátorváltozók páronként függetlenek.*

A feladat szerint a $c_i = 1$ és $\xi_i = Y_f$ választásokkal alkalmazható a tétel:

$$\mathbf{D}^2(Y_f) = \mathbf{E}\left(\left(Y_f - \frac{1}{2}\right)^2\right) = \mathbf{E}\left(Y_f^2 - Y_f + \frac{1}{4}\right) = \mathbf{E}\left(\frac{1}{4}\right) = \frac{1}{4},$$

$$\mathbf{D}^2(k(S)) = \mathbf{D}^2\left(\sum_{f \in E} Y_f\right) = \sum_{f \in E} \mathbf{D}^2(Y_f) = \sum_{f \in E} \frac{1}{4} = \frac{e}{4} \Rightarrow$$

$$\Rightarrow \mathbf{D}(k(S)) = \frac{\sqrt{e}}{2}.$$

A szórás birtokában alkalmazható a Csebisev-egyenlőtlenség. Például a $\lambda = 2$ értékkel:

$$\mathbf{P}\left(\left|k(S) - \frac{e}{2}\right| \geq \sqrt{e}\right) \leq \frac{1}{4}.$$

Az eredményt általában (nagy e esetén) úgy értelmezhetjük, hogy $k(S)$ jó eséllyel közel lesz a várható értékéhez. Ez megfelel a szórással kapcsolatos intuitív elvárásunknak: kis szórás esetén csak kis valószínűséggel lehetünk messze a várható értéktől.

A $k(S)$ maximumának meghatározása (maximális értékű vágás keresésének feladata) NP-nehéz, illetve a feladat eldöntési változata NP-teljes.

3.5 A Max 2SAT-feladat

A logika bámulatosan sok ponton találkozik a számítások világával, a mesterséges intelligenciától az adatbázisokon keresztül egészen az algoritmusokig. Mi itt az utóbbi területre teszünk egy rövid kirándulást.

12. Definíció. *A $\phi = C_1 \wedge C_2 \wedge \ldots \wedge C_m$ egy* 2-CNF formula *(konjunktív normálformájú formula), ha $C_i = l_{i_1} \vee l_{i_2}$ alakú (kéttagú diszjunkció), ahol l_{i_j} literál (Boole-változó, vagy Boole-változó negáltja). A C_i-ket tagoknak nevezzük. A ϕ 2-CNF* élő*, ha minden C_i tagjában pontosan két változó fordul elő (esetleg negálva).*

2. Példa. *A $\phi = (\overline{x}_1 \vee x_3) \wedge (x_2 \vee \overline{x}_3)$ egy élő 2-CNF.*

13. Számítási feladat (2SAT-feladat (2-satisfiability))**.** *Adott a ϕ 2-CNF formula. Döntsük el, hogy van-e a ϕ változóinak olyan kiértékelése (*igaz *és* hamis *értékekkel), amelyre ϕ igaz lesz.*

Ismert és hasznos tény, hogy a 2SAT-feladat polinom időben megoldható.[3] Közeli rokona, a 3SAT, amelyben a tagok $l_{i_1} \vee l_{i_2} \vee l_{i3}$ alakúak lehetnek (vagyis 3 literál szerepelhet tagonként), már NP-teljes. A 2SAT másik közeli rokona a következő, szintén NP-nehéz feladat:

14. Számítási feladat (Max 2SAT)**.** *Adott a $\phi = C_1 \wedge C_2 \wedge \ldots \wedge C_m$ 2-CNF formula. Határozzuk meg a változók egy olyan kiértékelését, amelyre az igaz C_i tagok száma maximális lesz.*

20. Feladat. *Legyen $G = (V, E)$ egy irányítatlan gráf. A $v \in V$ csúcsoknak feleltessük meg az x_v Boole-változókat, és tekintsük azt a ϕ 2-CNF formulát, amelynek tagjai $x_u \vee x_v$ és $\overline{x}_u \vee \overline{x}_v$, ahol $(u, v) \in E$ (ez összesen $2|E|$ tagot jelent). A változók egy adott kiértékelése esetén legyen S azon w csúcsok halmaza, amelyekre x_w igaz. Mutassuk meg, hogy ennél a kiértékelésnél az igaz tagok száma $|E| + k(S)$. A maximális értékű vágás feladata tehát a Max 2SAT speciális esete.*

A feladat a két algoritmikus kérdés közötti szoros kapcsolatra utal. Erre tekintettel nem meglepő a következő tétel:

25. Tétel. *Legyen ϕ élő 2-CNF formula. Ekkor van a ϕ változóinak olyan kiértékelése, amely a tagok legalább $\frac{3}{4}$-edét igazzá teszi.*

Bizonyítás. Legyen $\phi = C_1 \wedge \ldots \wedge C_m$. Vegyünk egy véletlen kiértékelést, azaz a változóknak adjunk $\frac{1}{2} - \frac{1}{2}$ valószínűséggel *igaz*, illetve *hamis* értéket, egymástól teljesen függetlenül. Legyen

$$Y_i = \begin{cases} 1, & \text{ha } C_i \text{ igaz,} \\ 0, & \text{különben.} \end{cases}$$

Ekkor az igaz tagok várható száma:

$$\mathbf{E}\left(\sum_{i=1}^{m} Y_i\right) = \sum_{i=1}^{m} \mathbf{E}(Y_i) = \sum_{i=1}^{m} \mathbf{P}(Y_i = 1).$$

Vizsgáljunk egy $C_i = l_{i_1} \vee l_{i_2}$ tagot. C_i pontosan akkor lesz hamis, ha l_{i_1} és l_{i_2} is hamis. Ez $\frac{1}{2} \cdot \frac{1}{2} = \frac{1}{4}$ valószínűséggel következik be (a változók függetlensége miatt a valószínűségek összeszorozhatók). Tehát a C_i tag $1 - \frac{1}{4} = \frac{3}{4}$ valószínűséggel lesz igaz. Innen pedig

$$\mathbf{E}\left(\sum_{i=1}^{m} Y_i\right) = \sum_{i=1}^{m} \mathbf{P}(Y_i = 1) = \sum_{i=1}^{m} \frac{3}{4} = \frac{3}{4}m.$$

Ebből következően létezik olyan kiértékelés, amelynél a C_i tagok legalább $\frac{3}{4}$-e igaz. □

[3] Lásd pl. [RISz] 8.7.1.

Itt és az előző szakaszban is használtuk azt az egyszerű észrevételt, hogy ha a ξ valószínűségi változóra $\mathbf{E}(\xi) \geq k$ teljesül, akkor van olyan ω pont az eseménytérben, amelyre $\xi(\omega) \geq k$.

3.6 Derandomizálás

Második boszorkány: *Nem oly szerencsés, mégis boldogabb!*
William Shakespeare: Macbeth

A számítások során alkalmazott véletlen választásokat (véletlen biteket) is tekinthetjük erőforrásnak, ugyanúgy mint az időt vagy a tárfelhasználást. Így szemlélve a dolgot, törekedhetünk arra, hogy csökkentsük a módszereink által használt véletlen bitek számát. Ezt a törekvést szokás *derandomizálásnak* nevezni. Célja az, hogy minél kevesebb véletlen felhasználásával oldjunk meg egy adott feladatot. Két fontosabb okot látunk, ami igazolhatja ezt a törekvést.

Az egyik inkább elvi és filozofikus jelentőségű. Érdekes kérdés, hogy valóban szükség van-e az algoritmusok világában a véletlenre? Nem igaz-e, hogy minden, amit a véletlennel hatékonyan meg tudunk oldani, kezelhető nélküle is, esetleg némi hatékonyságvesztés árán? Az eddigi algoritmusaink – a hiperfelületen kívüli pont keresését leszámítva – olyan problémára adtak hatékony megoldást, amelyeket a véletlen nélkül is elég jól lehet kezelni (pl. gyorsrendezés, konvex burok számítása). Határozott válasz a kérdésre jelenleg nem ismert. A *nincs* válasz mindenesetre azt jelentené, hogy minden véletlent alkalmazó algoritmus derandomizálható. Akik ebben az irányban gondolkodnak, aktívan keresik a derandomizálás lehetőségeit.

A másik ok a véletlen biteket használó algoritmusok hibázási lehetősége. Kritikus alkalmazásoknál (pl. adatbiztonság, erőművek irányítása, stb.) ezek a hibalehetőségek nem fogadhatók el, és ezért itt kifejezetten törekednek a véletlen elkerülésére.[4]

A mintatér csökkentése

Térjünk vissza a nagy vágás keresésének feladatához. Legyen $G = (V, E)$, $|V| = n$, és $|E| = e$. Eredetileg minden csúcshoz egy véletlen bitet használtunk fel, és bebizonyítottuk, hogy $\frac{e}{2}$ értékű vágás létezik G-ben. Másképpen mondva, olyan eseménytérben dolgoztunk, ahol 2^n db elemi esemény van, ami óriásinak mondható a bemenet hosszához képest. Megmutatjuk, hogy ennél a feladatnál ezt a nagy értéket levihetjük akár $\approx 2n$-re (tehát elég lesz csupán mintegy $\log_2 n$ véletlen bit). Ezt a mintateret csökkentő módszert tárgyaljuk az alábbiakban.

Legyen $k = \lceil \log_2 n + 1 \rceil$, azaz k az a minimális egész, melyre $2^k > n$. Minden $v \in V$ csúcshoz egy k-hosszú, nem $\vec{0}$ bitvektort rendelünk, különböző csúcsokhoz különböző vektort. Ez a k választása miatt megtehető. A v csúcs vektorát jelöljük $\vec{v}$-vel. A véletlen S vágás konstrukciója során minden $v \in V$ csúcshoz sorsoltunk egy véletlen bitet. Most is rendelni fogunk az v-hez egy X_v bitet, de sokkal kevesebb sorsolással.

Legyenek $\vec{v} = (v_1, \ldots, v_k)$ és $\vec{r} = (r_1, \ldots, r_k)$ k hosszúságú 0,1-vektorok. A $\vec{v} \cdot \vec{r}$ *skalárszorzat* értéke

$$\vec{v} \cdot \vec{r} = v_1 r_1 + v_2 r_2 + \cdots v_k r_k.$$

Az X_v biteket ezek után úgy kapjuk, hogy választunk *egyetlen* k-hosszú, véletlen $\vec{r}$ bitvektort, ennek kiszámítjuk a skalárszorzatát $\vec{v}$-vel, és vesszük az eredmény modulo 2 maradékát:

$$X_v = \vec{v} \cdot \vec{r} (\mathrm{mod}\ 2).$$

Az X_v bit éppen a $\sum_{i=1}^{k} v_i r_i$ összeg paritása lesz. Ezután az S vágás definíciója a régi: legyen $v \in S$, ha $X_v = 1$, és $v \notin S$, ha $X_v = 0$. Ahhoz, hogy $\mathbf{E}(k(S)) \geq \frac{e}{2}$ most is teljesüljön, elég

[4] A [RISz] 9.4. szakaszának elején további idevágó gondolatokat találhat az Olvasó pro és kontra is.

belátni, hogy $\mathbf{P}(Y_f = 1) = \frac{1}{2}$ ebben az eseménytérben is teljesül. Ehhez viszont elegendő, hogy az X_v véletlen bitekre igazak az alábbiak:

(1) $\mathbf{P}(X_v = 1) = \mathbf{P}(X_v = 0) = \frac{1}{2}$,

(2) az X_v valószínűségi változók páronként függetlenek: $\mathbf{P}(X_v = \epsilon \text{ és } X_w = \delta) = \frac{1}{4}$, ha $v \neq w$ csúcsok, és $\epsilon, \delta \in \{0, 1\}$.

Valóban, ha az $f = (v, w)$ élet nézzük, akkor a keresztezés valószínűsége ebben a térben is $\frac{1}{2}$ lesz.

11. Állítás. *Az (1) és (2) tulajdonságok teljesülnek az X_v valószínűségi változókra.*

Bizonyítás. (1) Tegyük fel, hogy $v_1 \neq 0$ (a koordináták átszámozásával ez feltehető). Legyen

$$\vec{r'} = (1 - r_1, r_2, \ldots, r_k).$$

A $\vec{r}$ vektorból úgy kapjuk $\vec{r'}$-t, hogy $\vec{r}$ első bitjét ellentétesre változtatjuk. Ekkor

$$\vec{v} \cdot \vec{r} = 1 \;\Leftrightarrow\; \vec{v} \cdot \vec{r'} = 0,$$

tehát a két esemény ugyanannyi db $\vec{r}$-nél következik be, azaz valószínűségük $\frac{1}{2} - \frac{1}{2}$.

(2) Megmutatható (nem részletezzük, két egyenletből álló mod 2 lineáris egyenletrendszer vizsgálatát jelenti), hogy $X_v = \epsilon$, és $X_w = \delta$ a lehetséges 2^k db $\vec{r}$- vektorból 2^{k-2} esetén teljesül, így

$$\mathbf{P}(X_v = \epsilon \text{ és } X_w = \delta) = \frac{\text{kedvező esetek}}{\text{összes eset}} = \frac{2^{k-2}}{2^k} = \frac{1}{4}.$$

□

Ebben az új valószínűségi térben $2^k \leq 2n$ darab elemi esemény van. Ennyi S polinom időben, véletlen nélkül is végignézhető, és ezek között biztosan lesz olyan, amelyre $k(S) \geq \frac{e}{2}$. A véletlent így teljesen ki tudjuk küszöbölni, miközben az algoritmusunk továbbra is hatékony (polinom idejű) marad. A kis eseménytér választását az tette lehetővé, hogy nem kellett a véletlen bitek teljes függetlensége az algoritmus működéséhez, elég volt csupán a páronkénti függetlenség.

2. Megjegyzés. *1. A Max 2SAT-ra adott véletlent használó közelítő algoritmusunk (ami várhatóan $\frac{3}{4}m$ igaz tagot ad) lényegében ugyanezzel a módszerrel derandomizálható.*

2. A legalább $\frac{e}{2}$ értékű vágás keresésére ismert más hatékony algoritmus is:

21. Feladat. *Mutassuk meg, hogy a következő mohó eljárás polinom idejű módszert ad: kezdetben legyen $S \subseteq V$ tetszőleges. Ha egy csúcsnak több szomszédja van a saját partján, mint a másikon (tehát S-belinek S-ben, $V \setminus S$-belinek $V \setminus S$-ben), akkor tegyük át a másik partra. Ezt addig ismételjük, amíg találunk ilyen áttehető csúcsot.*

A feltételes várható érték módszere

Itt egy másik derandomizáló módszerrel ismerkedünk meg. Elvi alapja a következő egyszerű észrevétel.

12. Állítás. *Legyen $A_1, A_2, \ldots, A_m$ teljes eseményrendszer, és ξ diszkrét valószínűségi változó, amelynek létezik az $\mathbf{E}(\xi)$ várható értéke. Ekkor létezik olyan i, amelyre $\mathbf{E}(\xi) \le \mathbf{E}(\xi|A_i)$.*

Bizonyítás. Legyen i olyan index, amelyre $\mathbf{E}(\xi|A_i)$ a lehető legnagyobb az $\mathbf{E}(\xi|A_j)$ értékek közül $(j = 1, \ldots, m)$. Ekkor a teljes várható érték tétele alapján

$$\mathbf{E}(\xi) = \sum_{j=1}^{m} \mathbf{P}(A_j)\mathbf{E}(\xi|A_j) \le \sum_{j=1}^{m} \mathbf{P}(A_j)\mathbf{E}(\xi|A_i) = \mathbf{E}(\xi|A_i)\sum_{j=1}^{m} \mathbf{P}(A_j) = \mathbf{E}(\xi|A_i).$$

□

Tegyük fel, hogy a ξ egy randomizált algoritmus által számított mennyiség, ami adott bemenet esetén az $s_1, \ldots, s_n$ véletlen választások függvénye. Olyan $\omega = (x_1, \ldots, x_n)$ mintát (itt x_i az s_i választás egy eredménye) szeretnénk, amelyre $\xi(\omega) \ge \mathbf{E}(\xi)$.

Az előző állítás alapján a s_1 véletlen választásnak van olyan x_1 kimenetele, hogy $\mathbf{E}(\xi) \le \mathbf{E}(\xi|s_1 = x_1)$. Itt a teljes eseményrendszert a s_1 lehetséges kimenetelei adják. A gondolatmenetet ξ helyett a $(\xi|s_1 = x_1)$ változóra ismételve kapjuk, hogy alkalmas x_2-vel $\mathbf{E}(\xi|s_1 = x_1) \le \mathbf{E}(\xi|s_1 = x_1, s_2 = x_2)$; általában tetszőleges $x_1, \ldots, x_j$ értékekhez létezik x_{j+1}, hogy

$$\mathbf{E}(\xi|s_1 = x_1, \ldots, s_j = x_j) \le \mathbf{E}(\xi|s_1 = x_1, \ldots, s_{j+1} = x_{j+1}).$$

Ha mármost az algoritmikus feladatunk olyan szerencsés, hogy ilyen x_{j+1} elemet az $x_1, \ldots, x_j$ ismeretében hatékony determinisztikus algoritmussal tudunk kapni, akkor ugyancsak jó determinisztikus módszert kapunk olyan $x_1, \ldots, x_n$ találására, amelyekkel

$$\mathbf{E}(\xi) \le \mathbf{E}(\xi|s_1 = x_1) \le \cdots \le \mathbf{E}(\xi|s_1 = x_1, \ldots, s_n = x_n) = \xi(x_1, \ldots, x_n).$$

Ezen a módon teljesen derandomizáltuk a számítást, az $\omega = (x_1, \ldots, x_n)$ mintára érvényes a $\xi(\omega) \ge \mathbf{E}(\xi)$ egyenlőtlenség.

3. Példa. *Nézzük meg ezt a módszert a nagy vágást kereső randomizált módszer kapcsán. Ekkor az $s_1, \ldots, s_n$ sorsolások a G gráf csúcsaihoz köthetők; az s_i két lehetséges kimenetele $v_i \in S$, illetve $v_i \notin S$, mégpedig mindkettő $\frac{1}{2}$ valószínűséggel következik be. A ξ szerepét a vágás $k = k(S)$ értéke játssza.*

Nézzük meg, mi a helyzet az első $j+1$ sorsolás után: az $\mathbf{E}(k|s_1 = x_1, \ldots, s_{j+1} = x_{j+1})$ értéke $a + \frac{b}{2}$ alakba írható, ahol a azon S-et keresztező élek száma, amelyeknek mindkét végpontja a már sorsolt csúcsok (vagyis $v_1, v_2, \ldots, v_{j+1}$) közül való, b pedig azon élek száma, amelyeknek legalább egy csúcsát még nem sorsoltuk. Az ilyen él a további sorsolások eredményeként $\frac{1}{2}$ valószínűséggel keresztezi S-et; tehát az indikátormódszerrel számolva látjuk, hogy $\frac{1}{2}$-el járul a várható értékhez. Az $a + \frac{b}{2}$ a sorsolások eredményének ismeretében lineáris időben kiszámolható.

Ha az első j csúcs S-be tartozásáról már döntöttünk ($x_1, \ldots, x_j$ már megvan), akkor x_{j+1} a $v_{j+1} \in S$ és a $v_{j+1} \notin S$ lehetőségek közül az legyen, amelyikre $\mathbf{E}(k|s_1 = x_1, \ldots, s_{j+1} = x_{j+1})$ nem kisebb a másik lehetőség választásával kapott értéknél. Egészen pontosan, mivel b mindkét esetben ugyanaz, a két lehetőség közül azt kell választani, amelyik a nem kevesebb $(v_i, v_{j+1}) \in E$ alakú keresztező élet adja, ahol $i \le j$. Ezzel egy másik igen egyszerű mohó algoritmust nyertünk legalább $\frac{e}{2}$ értékű vágás számítására.

22. Feladat. *Mutassuk meg közvetlenül, valószínűségi megfontolások nélkül, hogy az előbb vázolt mohó algoritmus helyes: legalább $\frac{e}{2}$ értékű vágást eredményez.*

23. Feladat. *(a) Mutassuk meg, hogy a K_n teljes gráf élei kiszínezhetők három színnel úgy, hogy az egyszínű háromszögek száma legfeljebb $\binom{n}{3}3^{-2}$ legyen.*

(b) Adjunk n-ben polinom idejű randomizált algoritmust egy (a)-nak eleget tevő színezés megadására.

(c) Derandomizáljuk a (b) algoritmusát a feltételes várható érték módszerével.

3.7 Mintavétel és igazítás

A mintavétel és igazítás (sample and modify) elve, hogy alakítsunk ki véletlen választásokkal egy struktúrát, ami majdnem teljesíti a kívánt feltételeket, utána javítsuk ki az esetlegesen maradó (kis) hibákat. Az elv alkalmazható például gráfokban nem túl kis méretű független csúcshalmaz (benne semelyik két csúcs sincs összekötve) létének az igazolására. Pontosabban fogalmazva, érvényes a következő:

26. Tétel. *Legyen $G = (V, E)$ irányítatlan gráf, $|V| = n$ és $|E| = e$. Tételezzük fel, hogy $e \geq \frac{n}{2}$. Ekkor van G-ben $\frac{n^2}{4e}$ méretű független csúcshalmaz.*

Bizonyítás. Legyen $d = \frac{2e}{n}$ (a G-beli fokok átlaga), és tekintsük a következő véletlenen alapuló kiválasztási folyamatot:

1. A csúcsokat éleikkel együtt töröljük G-ből $1 - \frac{1}{d}$ valószínűséggel, egymástól teljesen függetlenül (egy csúcs így $\frac{1}{d}$ valószínűséggel marad meg a kialakuló részgráfban).
2. A megmaradt éleket töröljük egyik végpontjukkal együtt (ez a végpont tetszőleges).

Az 1. lépésben történt a mintavétel, a másodikban az igazítás. A végül megmaradó csúcshalmaz nyilván független, de mekkora méretű? Legyen X az 1. lépés után maradó csúcsok, Y pedig az élek száma. Ekkor a 2. lépés után maradó ponthalmaz mérete legalább $X - Y$. Írjuk fel ennek várható értékét:

$$\mathbf{E}(X - Y) = \mathbf{E}(X) - \mathbf{E}(Y),$$

ahol $\mathbf{E}(X) = \frac{n}{d}$ (ez például indikátorváltozókkal látható be). Az 1. lépés után a G egy f éle pontosan akkor marad meg, ha mindkét végpontja megmarad. Ennek valószínűsége $\frac{1}{d^2}$. Innen az élek indikátorait használva

$$\mathbf{E}(Y) = e\mathbf{P}(f \text{ megmarad az 1. lépés után}) = \frac{e}{d^2} = e\frac{n^2}{4e^2} = \frac{n^2}{4e} = \frac{n}{2d}.$$

Így a végül megmaradó csúcshalmaz méretének várható értéke legalább

$$\mathbf{E}(X - Y) = \frac{n}{d} - \frac{n}{2d} = \frac{n}{2d} = \frac{n^2}{4e}.$$

Ekkora független halmaz nyilván van a gráfban. □

Például, ha a G gráfra igaz, hogy $e = 20n$, akkor biztosan lesz benne legalább $\frac{n^2}{80n} = \frac{n}{80}$ elemű független csúcshalmaz.

3.8 Lovász László lokális lemmája (LLL)

Amit a török meghagyott, elvitte a tatár, amit a tatár ott felejtett, elhordta a német... Nem maradt ott egyéb egy hosszú, kétélű acélpengénél...

Móra Ferenc: Hunyadi kardja

A következő alaphelyzetből indulunk ki: legyenek $E_1, \dots, E_n$ rossz (számunkra nem kívánatos) események egy valószínűségi térből. Szeretnénk feltételt kapni arra, hogy pozitív valószínűséggel egyik rossz esemény se következzen be. Olyan egyszerűen ellenőrizhető és jól alkalmazható feltételt szeretnénk, amelyből következik, hogy

$$\mathbf{P}(\bigcap \bar{E}_i) = \mathbf{P}(\bar{E}_1 \cap \ldots \cap \bar{E}_n) > 0.$$

Másként mondva: arra kellene kényelmes feltétel, hogy a rossz események (török, tatár, német) együttesen 1-nél kisebb valószínűséget adjanak. Ha az E_i események egymástól teljesen függetlenek és $\mathbf{P}(E_i) < 1$ igaz minden i-re, akkor természetesen $\mathbf{P}(\bar{E}_i) > 0$ minden i-re, és

$$\mathbf{P}(\bigcap \bar{E}_i) = \mathbf{P}(\bar{E}_1) \cdot \ldots \cdot \mathbf{P}(\bar{E}_n) > 0.$$

Ha tehát az E_i események teljesen függetlenek, akkor a kézenfekvő szükséges feltétel (hogy ne legyen E_i, amelyre $\mathbf{P}(E_i) = 1$) mindjárt elegendő is.

A fenti észrevétel sokszor nem alkalmazható, mert a rossz események nem teljesen függetlenek. Gyakran előfordul viszont, hogy ugyan az E_i események nem teljesen függetlenek, de csak „kevés” függés van közöttük. Ezzel a helyzettel foglalkozik a lokális lemma.

13. Definíció. *Az F esemény teljesen független a B_i, $i \in \mathcal{I}$ eseményektől, ha*

$$\mathbf{P}(F | \bigcap_{j \in \mathcal{J}} B_j) = \mathbf{P}(F) \quad \textit{teljesül minden} \quad \mathcal{J} \subseteq \mathcal{I} \quad \textit{esetén.}$$

Gyakran elég jó képünk van arról, hogy az E_i esemény mely másik E_j eseményektől függhet. Az ilyen természetű információ leírására alkalmas a függetlenségi gráf.

14. Definíció. *Legyenek $E_1, \dots, E_n$ események egy valószínűségi térből. A G gráf egy függetlenségi gráf, ha csúcshalmaza $\{1, 2, \dots, n\}$ és minden i-re az E_i teljesen független az*

$$\{E_j : \ j \neq i, \ (j,i) \notin E(G)\}$$

eseményektől.

27. Tétel (LLL egy változata)**.** *Legyenek $E_1, \dots, E_n$ események egy valószínűségi térből, legyen G függetlenségi gráf és $d \geq 1$. Tegyük fel továbbá, hogy igazak a következők:*

- $\mathbf{P}(E_i) \leq p$, $i = 1, \dots, n$.
- *G-ben a csúcsok foka $\leq d$.*
- $4dp \leq 1$.

Ekkor $\mathbf{P}(\bar{E}_1 \cap \ldots \cap \bar{E}_n) > 0$.

Bizonyítás. Indukciót alkalmazunk n szerint a következő két egyenlőtlenség bizonyítására:

$$\mathbf{P}(E_1|\bar{E}_2\cdots\bar{E}_n)\le\frac{1}{2d},\tag{3.2}$$

$$\mathbf{P}(\bar{E}_1\bar{E}_2\cdots\bar{E}_n)>0.$$

Elég a (3.2) egyenlőtlenséget belátni, ugyanis belőle és az indukciós feltevésből kapjuk, hogy

$$\mathbf{P}(\bar{E}_1\bar{E}_2\cdots\bar{E}_n)=\mathbf{P}(\bar{E}_2\cdots\bar{E}_n)\mathbf{P}(\bar{E}_1|\bar{E}_2\cdots\bar{E}_n)\ge\mathbf{P}(\bar{E}_2\cdots\bar{E}_n)\left(1-\frac{1}{2d}\right)>0.$$

Az $n=1,2$ esetek egyszerűek[5] meggondolásukat az Olvasóra bízzuk. Ezután feltesszük, hogy $n>2$. Az E_i események esetleges átszámozásával elérhető, hogy G-ben az 1 csúcs szomszédai $2,\dots,h$, ahol $h\ge 2$. A feltételes valószínűség definícióját és az indukciós feltevést alkalmazva adódik a következő:

$$\mathbf{P}(E_1|\bar{E}_2\cdots\bar{E}_n)=\frac{\mathbf{P}(E_1\bar{E}_2\cdots\bar{E}_n)}{\mathbf{P}(\bar{E}_2\cdots\bar{E}_n)}=\frac{\frac{\mathbf{P}(E_1\bar{E}_2\cdots\bar{E}_n)}{\mathbf{P}(\bar{E}_{h+1}\cdots\bar{E}_n)}}{\frac{\mathbf{P}(\bar{E}_2\cdots\bar{E}_n)}{\mathbf{P}(\bar{E}_{h+1}\cdots\bar{E}_n)}}=\frac{\mathbf{P}(E_1\bar{E}_2\cdots\bar{E}_h|\bar{E}_{h+1}\cdots\bar{E}_n)}{\mathbf{P}(\bar{E}_2\cdots\bar{E}_h|\bar{E}_{h+1}\cdots\bar{E}_n)}.$$

A jobb oldali tört számlálójának felső becslése:

$$\mathbf{P}(E_1\bar{E}_2\cdots\bar{E}_h|\bar{E}_{h+1}\cdots\bar{E}_n)\le\mathbf{P}(E_1|\bar{E}_{h+1}\cdots\bar{E}_n)=\mathbf{P}(E_1)\le\frac{1}{4d}.$$

Itt használtuk, hogy E_1 teljesen független az $E_{h+1},\dots,E_n$ eseményektől.

A nevező alsó becslése:

$$\mathbf{P}(\bar{E}_2\cdots\bar{E}_h|\bar{E}_{h+1}\cdots\bar{E}_n)=1-\mathbf{P}(E_2\cup\cdots\cup E_h|\bar{E}_{h+1}\cdots\bar{E}_n)\ge 1-\sum_{j=2}^{h}\mathbf{P}(E_j|\bar{E}_{h+1}\cdots\bar{E}_n).$$

Alkalmazható az indukciós feltevés a $\{j,h+1,\dots,n\}$ indexű eseményekre (ezek száma kisebb, mint n) és az ezen indexek által feszített részgráfra: $\mathbf{P}(E_j|\bar{E}_{h+1}\cdots\bar{E}_n)\le\frac{1}{2d}$, ami szerint a nevező legalább $1-\frac{h-1}{2d}\ge\frac{1}{2}$, mert $h-1\le d$.

Az eddigieket összerakva:

$$\mathbf{P}(E_1|\bar{E}_2\cdots\bar{E}_n)=\frac{\mathbf{P}(E_1\bar{E}_2\cdots\bar{E}_h|\bar{E}_{h+1}\cdots\bar{E}_n)}{\mathbf{P}(\bar{E}_2\cdots\bar{E}_h|\bar{E}_{h+1}\cdots\bar{E}_n)}\le\frac{\frac{1}{4d}}{\frac{1}{2}}=\frac{1}{2d}.$$

□

Az LLL két alkalmazása

A Beck-tétel

Visszatérünk az uniform hipergráfok 2-színezésének témájához. A következő tétel jelentősen tágítja az Erdős-tétel (19. Tétel) által mutatott horizontot: itt nincs felső korlát a H_i halmazok m számára.

28. Tétel (Beck)**.** *Legyenek $H_1,\dots,H_m$ az U halmaz r elemű részhalmazai. Tegyük fel, hogy egy H_i csak legfeljebb 2^{r-3} másikat metsz. Ekkor van U-nak olyan 2-színezése, amelynél egyik H_i sem egyszínű.*

[5] Az első egyenlőtlenség csak $n>1$ esetén értelmes.

Bizonyítás. Színezzük U elemeit $\frac{1}{2}-\frac{1}{2}$ valószínűséggel pirosra, illetve fehérre, egymástól teljesen függetlenül (ahogyan korábban az Erdős-tétel bizonyításánál tettük). Jelölje E_i azt az eseményt, hogy H_i *egyszínű*. Ekkor nyilván

$$p = \mathbf{P}(E_i) = \frac{1}{2^{r-1}}.$$

Az $u \in U$ pontok színének egymástól teljesen független választása miatt kijelenthetjük, hogy E_i független minden E_j-től, amelyre $H_i \cap H_j = \emptyset$. Ennek következtében a függetlenségi gráfban a fokszámokra $d \leq 2^{r-3}$ adódik. Mivel

$$4dp \leq 4 \cdot 2^{r-3} \cdot \frac{1}{2^{r-1}} = 1,$$

az LLL alkalmazható, vagyis $\mathbf{P}(\bar{E_1} \cap \ldots \cap \bar{E_n}) > 0$. Létezik tehát csupa tarka 2-színezés. □

Ritka $k-SAT$-formulák kielégíthetősége

Legyen $\phi = C_1 \wedge \ldots \wedge C_m$ egy élő $k-CNF$ formula. Ebben tehát minden C_i tag alakja $l_{i_1} \vee \ldots \vee l_{i_k}$, ahol l_{i_j} literál. Minden C_i tagban pontosan k különböző változó szerepel (esetleg negáltan).

A $k-CNF$ formulák kielégíthetőségének a kérdése $k \geq 3$ esetén már NP-teljes. Ennek tükrében különösen érdekes a következő állítás:

13. Állítás. *Legyen $\phi = C_1 \wedge \ldots \wedge C_m$ egy élő $k-CNF$ formula. Tegyük fel, hogy egy változó legfeljebb $T = \frac{2^k}{4k}$ darab C_i tagban szerepelhet. Ekkor létezik a ϕ-t igazzá tevő kiértékelés.*

Bizonyítás. Legyenek $x_1, \ldots, x_n$ a ϕ változói. Ezeknek $\frac{1}{2}-\frac{1}{2}$ valószínűséggel adjunk *igaz*, illetve *hamis* értéket, egymástól teljesen függetlenül. Legyen $E_i = \{C_i \text{ hamis}\}$. Alkalmazzuk a lokális lemmát. Az E_i események valószínűsége

$$p = \mathbf{P}(E_i) = \frac{1}{2^k},$$

ugyanis a C_i tagban előforduló k változó összes kiértékelése közül egy ad hamis értéket. Csak akkor lehetséges függés az E_i és az E_j események között, ha C_i-ben és C_j-ben van közös x_ℓ változó. Legyen Y_i azon C_j tagok száma, melyeknek van C_i-vel közös változója. Ekkor minden i-re igaz, hogy

$$Y_i \leq kT = \frac{k2^k}{4k} = 2^{k-2},$$

mert a C_i-ben k változó van. Tehát egy E_i legfeljebb 2^{k-2} másik E_j-től függhet. A Lovász-lemma alkalmazhatósági feltétele teljesül:

$$4pd = 4 \cdot 2^{-k} \cdot 2^{k-2} = 1.$$

A lemma alapján $\mathbf{P}(\bigcap \bar{E_i}) > 0$, azaz létezik jó kiértékelés. □

4. Példa. *Legyen $k = 16$ (16 literál lehet tagonként), ekkor $T = \frac{2^{16}}{4 \cdot 16} = 1024$, azaz egy x_i változó maximum 1024-szer fordul elő ϕ-ben. Az ilyen ϕ $16-CNF$ formulának a tétel szerint létezik kielégítő kiértékelése.*

Az LLL algoritmikus változata

A lokális lemma elegáns és egyszerű algoritmikus változatát dolgozta ki 2009-ben Robin A. Moser és Tardos Gábor. Módszerük az LLL legtöbb ismert alkalmazásánál használható.

Legyen $\mathcal{P}$ teljesen független valószínűségi változók egy véges halmaza, amelyek az Ω valószínűségi téren értelmezettek. Tegyük fel, hogy az E_i eseményeket ezekből a valószínűségi változókból származtatjuk úgy, hogy minden i-re van egy P_i részhalmaza $\mathcal{P}$-nek, hogy E_i bekövetkezése csak a P_i-beli változók értékeitől függ. Tegyük fel továbbá, hogy (i,j) pontosan akkor éle a G függetlenségi gráfnak, ha $i \neq j$ és $P_i \cap P_j \neq \emptyset$, vagyis van olyan $\xi \in \mathcal{P}$ valószínűségi változó, amitől E_i és E_j is függhet. Vegyük észre, hogy a kapott G tényleg függetlenségi gráf lesz az E_i eseményekre nézve. Végül tegyük fel, hogy az $E_1, \ldots, E_n$ eseményekre és G gráfra teljesülnek a lokális lemma feltételei. Moser és Tardos a következő egyszerű algoritmust javasolják:

1. Értékeljük ki a $\xi \in \mathcal{P}$ változókat egy véletlen $\omega \in \Omega$ helyen, és nézzük ennél a kiértékelésnél az E_i eseményeket. Ha egyikük sem következik be, akkor ω egy elemi eseményt ad a $\cap_{i=1}^{n} \bar{E}_i$ metszetből, és ezzel befejeztük az eljárást. Ha nem, akkor

2. válasszunk egy tetszőleges E_j eseményt, ami bekövetkezett. A P_j-beli ξ valószínűségi változókat értékeljük ki újra, a saját eloszlásuk szerint, egymástól teljesen függetlenül (a $\mathcal{P} \setminus P_j$-belieket változatlanul hagyjuk). Ha ezt követően egyik E_i sem következik be, akkor készen vagyunk, ellenkező esetben ismételjük a 2. lépést.

Moser és Tardos bizonyították (ennek a tárgyalását itt mellőzzük), hogy a 2. lépést várhatóan legfeljebb $\frac{n}{d-1}$ alkalommal kell elvégeznünk (itt n és d az LLL kimondásában szereplő értékek).

5. Példa. *Nézzük meg mindezt egy konkrét példán keresztül, a Beck-tétel esetében! Ekkor az Ω valószínűségi tér elemi lehetnek az $|U|$ hosszúságú éremfeldobás-sorozatok lehetséges eredményei. A térnek tehát $2^{|U|}$ eleme van, minden egyes sorozat bekövetkezési valószínűsége $\frac{1}{2^{|U|}}$. A $\xi_1, \xi_2, \ldots, \xi_{|U|}$ az U elemeihez rendelt valószínűségi változók. A $\xi_i : \Omega \to \{$piros, fehér$\}$ az i-edik elem színét mondja meg a véletlen színezésnél: ez* piros*, ha az i-edik éremfeldobás értéke* írás*, ellenkező esetben az i-edik elem színe* fehér*. Az E_j eseményhez tartozó változók P_j halmazába azon $\xi \in \mathcal{P}$ változók tartoznak, amelyek H_j-beli pontot színeznek.*

A Moser–Tardos-algoritmus 1. lépésében egy véletlen $\omega \in \Omega$ pontot (azaz éremfeldobás-sorozatot) választunk és ennek megfelelően színezzük U elemeit. A 2. lépést akkor kell elvégeznünk, ha van egyszínű H_j. Ekkor egy ilyen H_j elemeit kell újraszíneznünk az algoritmus szerint: az $u \in H_j$ elemek esetén újra elvégezzük az éremfeldobást – egymástól teljesen függetlenül –, és ezeket az elemeket az új dobások szerint színezzük át.

24. Feladat. *Mik lesznek az Ω, a $\mathcal{P}$ és a P_i halmazok a ritka $k-SAT$-feladat esetében?*

4. fejezet

Véletlen és bonyolultsági osztályok

Az algoritmikus problémákat a számításelmélet a kiszámítási (megoldási) nehézségük szerint osztályozza. Az így kapott ún. bonyolultsági osztályok (pl. P, NP, EXPTIME) gyakorlati szempontból is hasznos iránytűt adnak a felmerülő algoritmikus feladatok kezeléséhez. A következőkben azzal foglalkozunk, hogy a véletlen miként jelenik meg a bonyolultsági osztályok térképén, milyen érdekes feladatosztályokat kapunk, ha a véletlent is bevonjuk a számítás megengedett eszközei közé.

Ahogy a számításelméletben szokásos, elsősorban eldöntési feladatokra összpontosítunk: az L algoritmikus feladat kapcsán az x bemenetről el kell döntenünk, hogy a feladatbeli kérdésre *igen*, vagy *nem* lesz a válasz. Például ha L a gráfok 3-színezésének a feladata (szokásos jelöléssel $3SZÍN$), akkor az input G gráfokra azt kell eldönteni, hogy G színezhető-e 3 színnel.

A bonyolultsági osztályokat sokszor a megoldó algoritmusok lépésszáma segítségével definiáljuk. Ennek a kivitelezéséhez szükséges, hogy pontosan tudjunk beszélni a bemenet hosszáról. Hasznos lesz ezért az eldöntési feladatokat mint *nyelveket* megadni.

Tekintsük az $\mathcal{I} = \{0, 1\}$ ábécét. A lehetséges bemenetek az $x \in \mathcal{I}^*$ szavak. Az $x = x_1 \dots x_n$ szó az $x_i \in \mathcal{I}$ jelek (bitek) sorozata. Az x bitjeinek n száma az x szó *hossza*, szokásos jelöléssel $|x| = n$. Az $L \subseteq \mathcal{I}^*$ halmazokat nyelveknek nevezzük. Az L nyelv tekinthető igen-nem feladatnak a következőképpen: az $x \in I^*$ bemenetre azt kérdezzük, hogy $x \in L$ teljesül-e. $x \in L$ esetén a válasz „igen”, $x \notin L$ esetén „nem”. Fordítva, az eldöntendő számítási feladatok az input bináris kódolása révén felfoghatók *nyelvfelismerési feladat*oknak abban az értelemben, hogy az x inputról el kell dönteni, hogy bele tartozik-e az *igen* választ adó szavak nyelvébe.

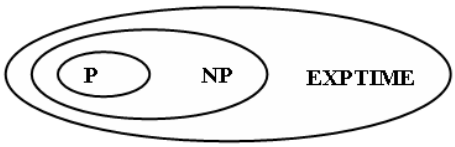
Így szemlélve például a $3SZÍN$-feladat a 3 színnel színezhető gráfok leírásaiból álló nyelv. Egy gráf leírása lehet pl. az illeszkedési mátrixa (mondjuk a mátrix sorfolytonos leírásának bináris kódja).

4.1 Néhány nevezetes bonyolultsági osztály felidézése

A P bonyolultsági osztályt a polinom időben felismerhető nyelvek alkotják: az L nyelv akkor van P-ben, ha van olyan algoritmus az L felismerésére, amely n hosszú $x \in I^*$ input szó esetén legfeljebb cn^d lépésben eldönti, hogy $x \in L$ teljesül-e. Itt c, d csak az L nyelvtől (feladattól) függő pozitív konstansok. Az algoritmus *lépéseinek* fogalmát itt most nem részletezzük. Szemléletesen egy programra gondolhat az Olvasó, és az általa elvégzett bit-műveletek számára.[1] A P és a többi itt említésre kerülő bonyolultsági osztály is *robusztus* abban az értelemben, hogy nagy mértékben független a definíciót megalapozó gépmodelltől. A P-re szokás úgy tekinteni,

[1] A [RISz] 7.1., 7.2., 8.1. és 8.2. szakaszaiban található árnyaltabb leírás ezekről a kérdésekről.

mint a determinisztikus algoritmussal hatékonyan (polinom időben) kezelhető eldöntési feladatok összességére.



Az L nyelv definíció szerint akkor van az NP nyelvosztályban, ha létezik egy $L_1 \in \mathrm{P}$ nyelv, melynek bemenete párokból áll, és létezik $c > 0$ konstans úgy, hogy $x \in \mathcal{I}^*$ esetén

$$x \in L \;\Leftrightarrow\; \exists y \in \mathcal{I}^* : \; |y| \leq |x|^c, \text{ amellyel } (x,y) \in L_1.$$

Az előző meghatározásban szereplő y szó az x bemenet L-be tartozásának *tanúja*. Úgy is fogalmazhatunk, hogy az L pontosan akkor lesz NP-beli, ha az L-be tartozásnak van rövid és hatékonyan ellenőrizhető tanúja.

6. Példa. *Legyen $L = 3SZÍN$ a 3-színezhető gráfok nyelve. Legyen $x \in \mathcal{I}^*$ egy G gráf leírása. Tegyük fel, hogy G 3-színezhető. Ekkor y szerepére alkalmas egy 3-színezés leírása (minden csúcsnak megmondjuk a színét). Az L_1 nyelv ekkor (G,y) párokból áll, y 3-színezése G-nek. Az $L = 3SZÍN$ nyelv NP-ben van, mert létezik olyan c (pl. $c = 1$), hogy az x leírású G gráf pontosan akkor van L-ben, ha létezik hozzá y színezés $|y| \leq |x|^c$, úgy, hogy $(G,y) \in L_1$. Itt $L_1 \in P$ valóban teljesül, mert egy (G,y) párról gyorsan (polinom időben) el tudjuk dönteni, hogy y 3-színezése-e G-nek.*

Az EXPTIME nyelvosztályba azok az L nyelvek tartoznak, amelyekhez van olyan algoritmus, amely az n hosszú x inputra az $x \in L$ kérdést legfeljebb $d2^{n^k}$ lépésben megválaszolja. Itt d és k csak L-től függő pozitív egészek. EXPTIME az exponenciális időben felismerhető nyelvek osztálya.

Az PSPACE osztályba azok az L nyelvek tartoznak, amelyekhez van legfeljebb polinomiális számú tárcellát használó algoritmus. Pontosabban fogalmazva, vannak olyan (csak L-től függő) c, d pozitív állandók, hogy az algoritmus az n hosszú x inputra az $x \in L$ kérdést legfeljebb cn^d tárcella használatával megválaszolja.

25. Feladat. *Mutassuk meg, hogy érvényesek az ábráról leolvasható tartalmazási viszonyok:* P⊆NP⊆EXPTIME.

4.2 Az RP nyelvosztály

Az elnevezés a *random* és a *polynomial* szavak kezdőbetűiből származik.

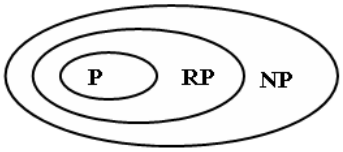
15. Definíció. *Legyen $L \subseteq \mathcal{I}^*$. Az $L \in$ RP pontosan akkor teljesül, ha létezik $L_1 \in \mathrm{P}$, és $c > 0$, hogy*

1. $L = \{x \in \mathcal{I}^* : \; \exists y \in \mathcal{I}^*, \; |y| = |x|^c, \quad \text{amelyre } (x,y) \in L_1\}$.
2. *$x \in L$ esetén az $|x|^c$-hosszú $y \in \mathcal{I}^*$ szavak legalább felére igaz, hogy $(x,y) \in L_1$.*

A definíció első pontja azt fogalmazza meg, hogy $L \in$ NP. A második pont szerint, ha x-hez létezik tanú, akkor mindjárt rengeteg tanú létezik.

26. Feladat. *Bizonyítsuk be, hogy az* RP *nyelvosztály nem változik meg, ha a definícióban a* felére *helyett d-szeresére szerepel, ahol $0 < d < 1$ egy tetszőleges állandó.*

Az alábbi ábra az RP osztály helyét szemlélteti a bonyolultsági osztályok között. A tudomány mai állása mellett nem tudjuk, hogy a tartalmazások valódiak-e.



Itt az RP$\subseteq$ NP tartalmazás a definíció 1. pontja miatt igaz. A P$\subseteq$RP nyilvánvaló: ha $L \in$P, akkor például $L_1 = \{(x, 1) : \ x \in L\}$ és $c = 0$ megfelelő lesz.

Az RP-teszt

Az RP-beli nyelvek felismerésére hatékony (polinom idejű) véletlent használó módszer adható. Legyen $L \in$ RP, és legyen L_1 a hozzá tartozó párokból álló nyelv, *Alg* pedig az L_1-et felismerő (determinisztikus) polinom idejű algoritmus. Az RP-*teszt* hatékony, véletlen választást alkalmazó módszer annak megválaszolására, hogy az $x \in \mathcal{I}^*$ input esetén $x \in L$ teljesül-e. A teszt a következő két lépésből áll:

- Válasszunk egy véletlen $y \in \mathcal{I}^*$ szót, amelyre $|y| = |x|^c$.
- Az *Alg* algoritmus segítségével nézzük meg, hogy $(x, y) \in L_1$ teljesül-e. Ha *igen* a válasz, akkor arra következtetünk, hogy $x \in L$. Ha *nem* a válasz, akkor azzal a következtetéssel zárjuk a tesztet, hogy *valószínűleg* $x \notin L$.

Az RP definíciójában szereplő 1. feltétel miatt az igenlő válasz mindig helyes. Ha van tanú x-hez, akkor x bizonyosan az L nyelvbe tartozik. A nemleges válasz lehet hibás. Megeshet, hogy $x \in L$ teljesül ugyan, de rossz y tanújelöltet választottunk. A definíció 2. feltétele szerint azonban ennek a balszerencsének a valószínűsége legfeljebb 1/2. A hibázás valószínűsége tetszőlegesen kicsivé tehető a jól ismert módon: t darab függetlenül választott y kipróbálása után legfeljebb $\frac{1}{2^t}$ a téves következtetés valószínűsége. Mivel $L_1 \in$P, és y hossza az x hosszában polinomiális, egy RP-teszt polinom időben megvalósítható. Tehát a kérdés, hogy $x \in L$ igaz-e, várhatóan polinom időben megválaszolható.

A Rabin–Miller-prímteszt

Itt részben az előzőek szemléltetésére vázlatosan megmutatjuk, hogy az összetett egészek nyelve

$$\ddot{O} = \{m \in \mathbb{Z}^+ : m \text{ nem prím}\} \in \text{RP}.$$

16. Definíció. *Legyen $m - 1 = 2^k \cdot n$, ahol k, n pozitív egészek, és n páratlan. Az a egész szám $(1 \le a < m)$ az m összetettségének* Rabin–Miller-tanúja, *ha az $a^n - 1$, $a^n + 1$, $a^{2n} + 1$, $a^{4n} + 1$, …, $a^{2^{k-1} \cdot n} + 1$ egészek egyike sem osztható m-mel.*

14. Állítás. *Ha m prímszám, akkor m-hez nem létezik Rabin–Miller-tanú.*

Bizonyítás. Tekintsük az alábbi azonosságot tetszőleges a egész számmal, melyre $1 \le a < m$:

$$a^{m-1} - 1 = \underbrace{\underbrace{\underbrace{\underbrace{(a^n-1)}_{a^{2n}-1}(a^n+1)}_{a^{4n}-1}(a^{2n}+1)}_{a^{8n}-1}(a^{4n}+1)\dots}_{a^{2^k\cdot n}-1}(a^{2^{k-1}\cdot n}+1). \tag{4.1}$$

Ha az m szám prím, akkor a kis Fermat-tétel szerint m osztja az $a^{m-1}-1$ egészet. Ennélfogva m osztja (4.1) jobb oldalát és m prím volta miatt annak egyik tényezőjét is. Ezért a nem lehet Rabin–Miller-tanú. □

A következő állítás szerint a páratlan összetett számokhoz sok Rabin–Miller-tanú létezik. A bizonyítást itt nem tárgyaljuk.[2]

15. Állítás. *Ha az $m > 2$ páratlan egész szám nem prím, akkor az $1 \le a < m$ egészek legalább fele Rabin-Miller tanú.*

Az előző két állítás alapján kimondhatjuk a következő tételt:

29. Tétel. Ö $\in$ RP, *vagyis az összetett számok nyelve* RP*-ben van.*

Bizonyítás. Legyen

$$L_1 = \left\{ (m,a) \;\middle|\; \begin{array}{l} m > 2,\ 1 \le a < m \text{ egészek, és ha } m \text{ páratlan,} \\ \text{akkor } a \text{ egy } m\text{-hez tartozó Rabin–Miller-tanú} \end{array} \right\},$$

továbbá $c = 1$. Ezekkel a választásokkal $L =$Ö-re teljesülnek az RP-be tartozás 1. és 2. feltételei, utóbbi a $d = (1/4)$ tényezővel. Legyen ugyanis n az m bitjeinek száma. Az előző állításból következik, hogy a legfeljebb n bittel leírható egészek legalább 1/4-e Rabin–Miller-tanúja lesz m összetettségének, amennyiben m páratlan összetett szám. Ha $m > 2$ és m páros, akkor az utolsó bitje mutatja az összetettségét. Ha pedig az m prím, akkor nincs hozzá Rabin–Miller-tanú.

Meg kell még mutatnunk, hogy $L_1 \in$P. Más szóval egy adott a-ról hatékonyan kell ellenőriznünk, hogy a Rabin–Miller-tanú-e. Az ötlet az, hogy *gyors hatványozással*[3] számolhatjuk az $a^{2^i n}$ alakú számok osztási maradékát modulo m. Megmutatható, hogy egy (m,a) pár ellenőrzése ezt az utat követve $O(\log^3 m)$ bitművelettel megvalósítható. □

A tételbeli L_1 nyelvhez tartozó RP-tesztet Rabin–Miller-prímtesztnek is nevezik, és az egyik legnépszerűbb prímteszt a gyakorlatban, futási ideje $O(\log^3 m)$. Ha m-et összetettnek találja, akkor ez a válasza biztosan helyes. Hiba a másik esetben lehetséges, amikor a teszt eredménye: *m valószínűleg prím.* A hiba valószínűsége a teszt független ismétlésével tetszőlegesen kicsivé tehető. A gyakorlatban a kriptográfusok akkor tekintenek egy nagy m egészet prímnek, ha a teszt többszöri ismétlése sem bizonyította az összetettségét.

2002-ben tudományos szenzációnak számított, hogy M. Agrawal, N. Kayal és N. Saxena determinisztikus polinom idejű algoritmust [AKS] adtak az összetett számok felismerésére. Más szóval, a tétel állításánál erősebb Ö$\in$P is igaz. Az AKS-teszt – bár polinom idejű – a gyakorlatban eddig nem tudott versenyezni a jóval gyorsabb randomizált tesztekkel, mint amilyen a Rabin–Miller-teszt is.

[2] Lásd pl. [BS], Vol. 1, Theorem 9.4.5.
[3] Lásd pl. [RISz] 265. old.

A Las Vegas nyelvosztály

Igen érdekesek azok az RP-beli L nyelvek (algoritmikus eldöntési feladatok), amelyekre a komplementer nyelv $I^* \setminus L$ is RP-be tartozik.

17. Definíció. *Az L nyelv a* Las Vegas *nyelvosztályba tartozik, ha $L \in$ RP és $I^* \setminus L \in$ RP.*

Az RP-teszt mintájára bevezetjük a *Las Vegas-tesztet* annak eldöntésére, hogy adott $L \in$ Las Vegas és $x \in I^*$ esetén $x \in L$ teljesül-e. A Las Vegas-teszt két RP tesztből áll, az egyik az $x \stackrel{?}{\in} L$, a másik pedig az $x \stackrel{?}{\in} I^* \setminus L$ kérdés megválaszolására végzett RP-teszt. Ha a kettő közül valamelyik teszt *igen* választ ad, akkor x abba a nyelvbe tartozik, ahol az igent kaptuk. Tudjuk, hogy az *igen* válasz az RP-teszt esetében mindig helyes. Ekkor tehát pontosan tisztáztuk L és x viszonyát. Annak valószínűsége, hogy egyik tesztnél sem kapunk igent, legfeljebb $\frac{1}{2}$, hiszen annál a tesztnél, amelyiknél az *igen* várható, ennek a válasznak a valószínűsége legalább $\frac{1}{2}$. Ha egyik tesztnél sem kaptunk igent, azt úgy értelmezhetjük, hogy nem sikerült a kérdést megválaszolnunk. Ennek a valószínűsége a Las Vegas-teszt független ismétlésével gyorsan lecsökkenthető. A lényeges különbség a szimpla RP-teszthez képest, hogy a Las Vegas-tesztnél sosem kapunk helytelen eredményt. Ha egyik teszt sem adott igenlő választ, akkor úgy vehetjük, hogy tovább kell próbálkoznunk az igazság kiderítésével. Az teszt becsületes abban az értelemben, hogy sosem ad hamis választ a kérdésre. A Las Vegas-teszt jellemzőit így összegezhetjük: *gyors (polinom idejű), legalább $\frac{1}{2}$ valószínűséggel válaszol az ($x \in L$?) kérdésre, és a válasza mindig helyes.*

A következő Q (eldöntendő) feladatról ismert, hogy $Q \in$Las Vegas, és nem ismert ugyanakkor, hogy $Q \in$P igaz lenne. Egyszerű szavakkal: Q felismerésére van hatékony, becsületes randomizált algoritmus, de nem ismert hatékony determinisztikus algoritmus.

15. Számítási feladat. *A Q bemenete egy (c, a, p) egész számokból álló hármas, ahol p prímszám, $1 \leq c, a < p$. Az eldöntendő kérdés az, hogy van-e olyan b egész az $[1, c]$ intervallumban, amelyre $b^2 - a$ osztható p-vel.*

Megjegyezzük, hogy Q lényegében a másodfokú egyenletek megoldásának feladata az $\mathbb{F}_p$ testben. Az algoritmust itt nem tárgyaljuk.[4]

27. Feladat. *Mutassuk meg, hogy $Q \in$ NP $\cap$ coNP.*

4.3 A BPP nyelvosztály

A BPP nyelvosztály a legtágabb értelmes feladatosztályt körvonalazza, amelyre létezik polinom idejű véletlent használó algoritmus.

18. Definíció. *Az $L \in \mathcal{I}^*$ a* BPP*-ben van, ha $\exists L_1 \in P$ és $c > 0$, hogy*

1. *ha $x \in L$, akkor az $|x|^c$ hosszú $y \in \mathcal{I}^*$ szavak legalább kétharmadára igaz, hogy $(x, y) \in L_1$,*
2. *ha $x \notin L$, akkor az $|x|^c$ hosszú $y \in \mathcal{I}^*$ szavak legalább kétharmadára igaz, hogy $(x, y) \notin L_1$.*

Megjegyezzük, hogy a definícióban a $\frac{2}{3}$ helyett bármilyen d állhatna, amelyre $\frac{1}{2} < d < 1$. (Miért?)

Itt a tanú fogalma sokat lazult az NP kapcsán tanult fogalomhoz képest. Az L nyelvbe nem tartozó x szavakhoz is létezhet tanú, de ezek markáns kisebbséget kell, hogy alkossanak a

[4] Lásd pl. [I] 2. kötet, 18.3.

tanújelöltek körében. Az RP-hez képest változás van a definíció 1. pontjában is. Itt erőteljes többség szükséges.

A BPP név a *bounded error*, *probabilistic* és *polynomial* kifejezések kezdőbetűiből származik. A definícióból közvetlenül következnek az alábbi tartalmazási viszonyok:

$$\mathrm{P} \subseteq \mathrm{RP} \subseteq \mathrm{BPP} \subseteq \mathrm{EXPTIME}.$$

Talán elég, ha csak a legutolsó tartalmazáshoz fűzünk egy kis magyarázatot. Legyen $L \in$BPP és $x \in I^*$. Az $x \stackrel{?}{\in} L$ kérdést exponenciális időben megválaszolhatjuk úgy, hogy sorra vesszük az $|x|^c$ hosszúságú $y \in I^*$ szavakat, és mindegyikre megnézzük (polinom időben), hogy $(x, y) \in L_1$ teljesül-e. Az *igen* válaszok számának nyilvántartásával meg tudjuk válaszolni az x-re vonatkozó kérdést. A definíció szimmetriájából pedig következik, hogy

$$\mathrm{coBPP} = \mathrm{BPP}.$$

A korábbi randomizált osztályokhoz hasonlóan itt is adódik egy természetes algoritmus. A *BPP-teszt* véletlent használó polinom idejű módszert ad az $L \in$BPP nyelvek felismerésére. Legyen $x \in I^*$. Szeretnénk eldönteni, hogy $x \in L$ igaz-e. A következőt tehetjük: legyen $k > 2$ és válasszunk egyenletesen és egymástól teljesen függetlenül k darab $y^i \in I^*$ szót, amelyek hossza $|y^i| = |x|^c$, $i = 1, \ldots, k$. Ezután nézzük meg mindegyik y^i-re, hogy $(x, y^i) \in L_1$ igaz-e. Az eredeti kérdést illetően többségi döntést hozunk: ha az *igen* fordult elő többször, akkor úgy döntünk, hogy *valószínűleg* $x \in L$, ellenkező esetben pedig arra jutunk, hogy *valószínűleg* $x \notin L$.

A BPP-tesztnél mindkét válasz lehet hibás. A k növelésével a hiba valószínűsége rohamosan csökkenthető.

7. Példa. *Tegyük fel, hogy $k = 3$ és $x \in L$. A helyes többségi döntés q valószínűségére szeretnénk alsó korlátot kapni. A válaszunk helyes lesz, ha a három y^i közül legalább kettő esetén azt kapjuk, hogy $(x, y^i) \in L_1$. Tudjuk, hogy ez minden i-re $\geq \frac{2}{3}$ eséllyel igaz. A választásaink függetlenségét is figyelembe véve tehát*

$$q \geq 1 \cdot \left(\frac{2}{3}\right)^3 + 3 \cdot \frac{1}{3}\left(\frac{2}{3}\right)^2 = \frac{20}{27} = \frac{2}{3} + \frac{2}{27}.$$

Nézzük, mi történi a k növelésével. Legyen továbbra is $x \in L$ (ugyanezen meggondolás a definíció szimmetriája miatt érvényes lesz az $x \notin L$ esetben is), és tegyük fel, hogy a BPP-teszt során az y^i tanújelölteket választottuk. Legyen továbbá

$$X_i = \begin{cases} 1, & \text{ha } (x, y^i) \in L_1, \\ 0, & \text{különben.} \end{cases}$$

Ekkor a Chernoff-egyenlőtlenség (5. tétel) alkalmazható a teljesen független $X_1, \ldots, X_k$ változókra. Ezek $p = \mathbf{P}(X_i = 1) \geq \frac{2}{3}$ paraméterű Bernoulli-változók. Legyen $\epsilon = \frac{2p-1}{2p}$. Ekkor $(1 - \epsilon)p = \frac{1}{2}$, és a Chernoff-egyenlőtlenség alapján a hibás döntés valószínűsége

$$\mathbf{P}\left(S_k \leq \frac{k}{2}\right) = \mathbf{P}\left(S_k \leq (1-\epsilon)pk\right) \leq e^{\frac{-\epsilon^2 kp}{3}} = e^{-c \cdot k},$$

ahol c egy pozitív állandó. A hibavalószínűség tehát a k növelésével gyorsan tart 0-hoz.

A tudomány mai állása szerint nem tudjuk, hogy a BPP osztály hol helyezkedik el a P és az EXPTIME között. Jelenleg egyik véglet sincs kizárva. BPP=P azt jelentené, hogy a véletlen

nem tud *óriásit*[5] segíteni: ami véletlen segítségével polinom időben megoldható, az determinisztikus polinom időben is megoldható. A másik véglet, a BPP=EXPTIME azt jelentené, hogy a véletlennel hatalmas gyorsítás érhető el: a determinisztikusan exponenciális időigényű feladatokat lehetne a véletlen segítségével polinom időben legyűrni.

Korábban az volt a számításelmélet kutatói között az uralkodó nézet (talán sejtésnek is nevezhetjük), hogy BPP≠P. Ez határozottan megváltozott az elmúlt másfél évtizedben. Ma már a szakértők többsége úgy gondolja, hogy BPP=P.[6] A BPP helyének meghatározása jelenleg igen aktív és rendkívül színes kutatási terület. Érdekes kapcsolatokat mutat például a BPP esetleges derandomizálhatósága a kriptográfiából ismert, és ott sokra értékelt egyirányú függvényekkel.[7] A témától búcsúzóban egy ilyen tételt említünk:

30. Tétel (M. Blum, S. Micali, A. Yao)**.** *Ha létezik egyirányú permutáció, akkor*

$$BPP \subseteq \cap_{\delta>0} TIME\left(2^{n^\delta}\right).$$

Ha tehát létezik alkalmas egyirányú függvény, akkor BPP szubexponenciális időben derandomizálható.

4.4 Interaktív bizonyítások

A több résztvevő által végzett együttes számítás, az *interakció* központi motívum több helyen is az informatikában. Az algoritmusok világa sem számít kivételnek. Az interaktív bizonyítások fogalma a 80-as évek közepén alakult ki, és rendkívül gyümölcsözőnek bizonyult egy sor részterületen, köztük egészen váratlanokon is, mint amilyen például a hatékony közelítő módszerek elmélete. Az *interaktív bizonyítás* számítástudományi fogalma lényegében egyidőben, két egymástól független kutatás eredményeként született meg. Az egyik kriptográfiai hátterű: Goldwasser, Micali és Rackoff [GMR] a biztonságos információközlés egy problémáját vizsgálva eljutottak a nulla ismeretű bizonyítás fogalmáig (amivel itt később mi is foglalkozunk). A másik megközelítés Babai Lászlótól[8] származik, aki mátrixcsoportokkal kapcsolatos algoritmikus kérdések tanulmányozására dolgozta ki az Arthur–Merlin-játékokat [B]. A két munkából kibontakozott elmélet ma is egyike a legfontosabb számításelméleti területeknek, sok szép eredménnyel és izgalmas kutatási problémákkal. Kiemelkedő eredményükért Babai, Goldwasser, Micali és Rackoff elnyerték a számításelmélet legmagasabb szakmai kitüntetését, a Gödel-díjat.

Arthur–Merlin-protokollok

Két szereplőnk van, Arthur király és Merlin, a varázsló, akik egymással kommunikálnak. Arthur türelmetlen uralkodó, csak polinom idejű működésre képes: egy x bemenettel maximum $|x|^c$ ideig tud foglalkozni. Ezzel szemben Merlinnek varázsereje van: nincs korlátja a számítási képességeinek.[9] Bármely L nyelv és $x \in I^*$ szó esetén egyetlen lépésben el tudja dönteni, hogy $x \in L$ igaz-e. A két fél együttműködésének, interakciójának célja a következő: adott egy L nyelv és egy $x \in I^*$ szó; Arthur szeretne meggyőződni (bizonyosságot szerezni) arról, hogy

[5] Több szép példát is láttunk már igen gyors véletlent használó algoritmusra olyan feladatok esetén, amelyek determinisztikusan is elég jól kezelhetők. Ilyenek a gyorsrendezés, vagy a konvex burok számítása.

[6] Lásd pl. [AB] 20. fejezet.

[7] Az $f : I^* \to I^*$ függvény egyirányú, ha x-ből $f(x)$ hatékonyan számítható, viszont az inverz feladatra nincs hatékony (randomizált) algoritmus: adott y szóhoz nehéz találni olyan x-et, amelyre $f(x) = y$.

[8] Babai László a BME informatikusképzésének egyik alapító professzora is: ő tanította itt először az *Algoritmusok elméletét*.

[9] Valójában elegendő lenne azt feltenni, hogy Merlin hatékonyan meg tudja oldani a PSPACE-beli feladatokat.

$x \in L$, amennyiben ez igaz. Ennek érdekében számításokat végezhet, és kérdéseket tehet fel Merlinnek. A kérdésekre kapott válaszok és saját számításai alapján dönt arról, hogy az $x \in L$ állítást igaznak tartja-e. Az *igen* döntést elfogadásnak, a *nem* döntést elutasításnak nevezzük. Feltesszük itt, hogy a helyes bizonyítást elfogadja. Az egész információcserére és a számításokra együttesen csak legfeljebb $|x|^c$ idő van, ahol $c > 0$ csak L-től függő állandó. A protokollal[10] szemben két alapvető követelményünk van:

- *Teljesség*: Ha $x \in L$, akkor erről Merlin meg tudja győzni Arthurt, vagyis végül Arthur elfogadja x-et.
- *Helyesség*: Ha $x \notin L$, akkor Merlin nem tudja meggyőzni Arthurt arról, hogy $x \in L$ (még akkor sem, ha csal). Ekkor Arthur elutasítja x-et.

Milyen L nyelvekhez van ilyen Arthur–Merlin-protokoll? Erre válaszol az alábbi tétel és az utána következő feladat.

31. Tétel. *Legyen $L \in$ NP. Ekkor létezik L-hez Arthur–Merlin-protokoll.*

Bizonyítás. Az $L \in$NP miatt létezik $c > 0$ és $L_1 \in$ P, úgy hogy $x \in L$ pontosan akkor igaz, ha létezik hozzá $y \in \mathcal{I}^*$, melyre $|y| \leq |x|^c$, és $(x, y) \in L_1$.

Ezek után $x \in L$ esetén a protokoll a következő lehet:

1. Arthur elküldi az x szót Merlinnek.
2. Válaszul Merlin küld egy $y \in \mathcal{I}^*$ szót, ahol $|y| \leq |x|^c$ és $(x, y) \in L_1$ (ha $x \in L$).
3. Arthur ellenőrzi, hogy $(x, y) \stackrel{?}{\in} L_1$. Ha igen, akkor elfogad (elfogadja, hogy $x \in L$), ha nem, akkor elutasít.

Igazoljuk, hogy a fenti protokoll helyes Arthur–Merlin-protokoll:

- Arthur számítási képességeinek megfelel, mert $(x, y) \stackrel{?}{\in} L_1$ polinom időben ellenőrizhető, hiszen $L_1 \in$ P, és $|y| \leq |x|^c$. A küldött és fogadott üzenetek összhossza is polinomiális.
- Ha $x \in L$, akkor létezik rövid y, mellyel $(x, y) \in L_1$ teljesül. Ilyet Merlin képes találni, és ezzel el tudja érni, hogy Arthur elfogadja az $x \in L$ tényt.
- Ha viszont $x \notin L$, akkor Merlin nem tud jó y-t küldeni, mivel nem létezik jó y. Így nem tudja becsapni Arthurt (meggyőzni arról, hogy $x \in L$), mert Arthur polinom időben rájön, hogy $(x, y) \notin L_1$, bármi legyen is az esetleg elküldött y.

□

28. Feladat. *Mutassuk meg, hogy a tétel megfordítása is igaz: Ha L-hez van Arthur–Merlin-protokoll, akkor $L \in$ NP. (Gondoljuk meg, hogy $x \in L$-hez mi lehet megfelelő y tanú.)*

[10] A protokoll fogalmát a szemléletesség megtartása miatt nem adjuk meg formálisan. Az érdeklődő olvasó itt talál pontos definíciót: [AB] 8. fejezet.

Randomizált interaktív bizonyítások

Az előbbiekben sikerült az NP nyelvosztályt jellemeznünk az interakció segítségével. Felmerül a kérdés, hogy kaphatunk-e itt többet, ha a véletlennek is szerepet juttatunk. Létezhet-e NP-n kívüli nyelvekhez is az előbbihez hasonló értelmű interaktív bizonyítás? Mint látni fogjuk, a válasz igen.[11] Tartsuk meg a két szereplőnket: Arthur királyt, aki kérdez, és Merlin varázslót, aki válaszol. A probléma, amiről kommunikálnak, ugyancsak a régi: adott egy $L \subseteq \mathcal{I}^*$ nyelv és egy $x \in \mathcal{I}^*$ szó. Merlin szeretné Arthurral elfogadtatni, hogy $x \in L$. Merlin itt is varázserővel rendelkezik, amennyiben bármely algoritmikus kérdést képes egy lépésben megoldani. Arthur továbbra is polinom ideig működik, de használhat véletlen választásokat is a számításai során. A két fél párbeszédét leíró protokollal szembeni követelményben is van változás: megjelenik benne a véletlen.

- *Teljesség*:[12] Ha $x \in L$, akkor $\mathbf{P}(\text{Arthur elfogadja, hogy } x \in L)) = 1$.
- *Helyesség*: Ha $x \notin L$, akkor $\mathbf{P}(\text{Arthur elfogadja, hogy } x \in L)) \leq \frac{1}{2}$.

Most is az érdekel bennünket, hogy mely nyelvekhez létezik Arthur–Merlin-protokoll.

29. Feladat. *Mutassuk meg, hogy ha a helyesség definíciójában $\frac{1}{2}$ helyébe 0-t írunk, akkor ismét az* NP *nyelvosztályt kapjuk. Csak ezekhez létezik AM-protokoll.*

Legyen

$$\text{IP} = \{L \subseteq \mathcal{I}^* : L\text{-hez van Arthur–Merlin-protokoll}\}.$$

A következőkben egy olyan nyelvhez adunk Arthur–Merlin-protokollt, amelyről nem ismert, hogy NP-beli lenne.

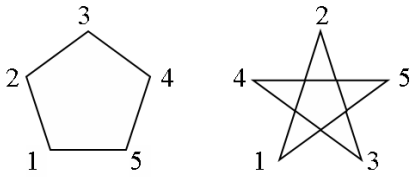
GNI – a nem izomorf gráfok nyelve

19. Definíció. *Legyenek G_1 és G_2 irányítatlan gráfok. A G_1* izomorf *G_2-vel, ha van olyan, kölcsönösen egyértelmű $f : V(G_1) \mapsto V(G_2)$ leképezés, amelyre*

$$(x, y) \in E(G_1) \Leftrightarrow \big(f(x), f(y)\big) \in E(G_2).$$

G_1 és G_2 izomorfiáját $G_1 \cong G_2$ jelöli.

Például az alábbi ábrán látható ötszög és csillag izomorf gráfok, amint azt az $f(x) = x$ leképezés mutatja.



[11] Hozzá kell tennünk, hogy a tudomány mai állása szerint. Jelenleg nem tudjuk bizonyítani, hogy az adódó nyelvosztály tényleg bővebb NP-nél, de széles körben elfogadott sejtés szerint ez igaz.

[12] A szokásos definíció itt csupán azt követeli meg, hogy az elfogadás valószínűsége legalább $\frac{2}{3}$ legyen. Igazolható, hogy ez a követelmény ekvivalens az általunk megadottal, lásd [FGMSZ].

Tekintsük az izomorf gráf-párokból álló nyelvet és a komplementerét:

$$GI = \{(G_1, G_2) : G_1 \cong G_2\},$$
$$GNI = \{(G_1, G_2) : G_1 \ncong G_2\}.$$

Nyilvánvalóan igaz, hogy $GI \in$ NP. A $(G_1, G_2) \in GI$ tanúja egy alkalmas f leképezés lehet. A GI gyakorlati szempontból is fontos feladat. Elméleti szempontból is különleges: olyan régóta ismert NP-beli nyelv, amelyről nem tudjuk, hogy NP-teljes-e, és azt sem, hogy P-ben van-e. A GNI nyelvről nem ismeretes, hogy NP-beli lenne. Igaz viszont a következő:

32. Tétel. $GNI \in$ IP.

Bizonyítás. Alkalmazzuk a következő protokollt:

1. Az Arthurnál levő input legyen a (G_1, G_2) gráfpár, és az egyszerűség kedvéért tegyük fel, hogy
$$V(G_1) = V(G_2) = \{1, 2, \ldots, n\} = V.$$
Arthur választ egy véletlen[13] $i \in \{1, 2\}$ értéket ($\frac{1}{2} - \frac{1}{2}$ valószínűséggel), és egy $f : V \mapsto V$ véletlen permutációt (bármelyik f valószínűsége $\frac{1}{n!}$). Ezt követően képezi a $H = f(G_i)$ gráfot, ezt elküldi Merlinnek, és megkérdezi tőle, hogy mi volt az i.

2. Merlin válasza egy $j \in \{1, 2\}$.

3. Ha $i = j$, akkor Arthur elfogadja, hogy $G_1 \cong G_2$, különben pedig nem.

A protokollt elemezve látható, hogy ha $G_1 \ncong G_2$ (azaz $(G_1, G_2) \in GNI$), akkor a H a G_1 és G_2 közül pontosan az egyikkel izomorf. Erre Merlin rájön, és meg tudja mondani a helyes i értéket. Ha viszont $G_1 \cong G_2$ (azaz $(G_1, G_2) \notin GNI$), akkor H ugyanazzal az $\frac{1}{2}$ valószínűséggel származhat G_1-ből és G_2-ből:

$$\mathbf{P}(i = 1|\text{ Arthur a } H \text{ gráfot küldi}) = \mathbf{P}(i = 2|\text{ Arthur a } H \text{ gráfot küldi}) = \frac{1}{2}.$$

Ez abból adódik, hogy

$$|\{\underbrace{(1, f)}_{\substack{\text{sorsolás}\\ \text{eredménye}}} : \text{ Arthur a } H \text{ gráfot küldi}\}| = |\{\underbrace{(2, f')}_{\substack{\text{sorsolás}\\ \text{eredménye}}} : \text{ Arthur a } H \text{ gráfot küldi}\}|.$$

A két halmaz azért lesz egyenlő méretű, mert ha $\sigma : V \to V$ izomorfizmust ad G_1 és G_2 között, akkor a $(2, f')$ pár pont akkor eredményezi H-t, ha $(1, f'\sigma)$ választásakor H-t kapunk. Másként mondva, σ kölcsönösen egyértelmű megfeleltetést létesít a két halmaz között.

Merlin tehát $G_1 \cong G_2$ esetén nem tud jobbat tenni, mint tippeli i-t. A helyes válaszra, és ezzel Arthur megtévesztésére ezért $\frac{1}{2}$ az esélye. Tekintetbe véve még azt is, hogy az üzenetek összmérete és Arthur számításainak az ideje is polinomiális $|x|$-ben, a protokoll helyességét és a tételt igazoltuk. □

Megjegyezzük, hogy a protokoll független ismétlésével a helytelen elfogadás valószínűsége rohamosan csökkenthető.

Bizonyítás nélkül említjük a következő fontos eredményt.

33. Tétel. IP=PSPACE.

[13] A véletlen választásait Arthur rejtve, Merlin számára nem látható módon teszi.

Helyette a következő, valamivel egyszerűbb tételt tárgyaljuk:

34. Tétel. IP$\supseteq$CoNP.

Bizonyítás. Legyen

$$\#3SAT = \{(\phi, k), \text{a } \phi \text{ egy } 3CNF, \text{aminek éppen } k \text{ kielégítő kiértékelése van}\}.$$

A $\#3SAT$ nyelv nyilván $coNP$-nehéz, hiszen a $k = 0$ esetén éppen a kielégíthetetlen $3CNF$ formulák nyelvét (lényegében az NP-teljes $3SAT$ komplementerét) kapjuk, ami teljes $coNP$-re nézve.

Egy n változós ϕ Boole-formula tekinthető a $\{0,1\}^n$ halmazon értelmezett 0,1-értékű függvénynek. A protokollhoz, amit meg szeretnénk adni, célszerű lesz ezt a függvényt kiterjeszteni $\mathbb{F}^n$-ből $\mathbb{F}$-be vezető függvénnyé, ahol az $\mathbb{F}$ egy alkalmas test (és így nyilván van benne 0 és 1). Ezt a kiterjesztést szokás *aritmetizálásnak* nevezni. A ϕ formula $\mathbb{F}^n$-re való ϕ^* kiterjesztését rekurzióval adjuk meg. Az x_i Boole-változónak az $\mathbb{F}$ feletti X_i változó felel meg, vagyis $x_i^* = X_i$. Az $\overline{x}_i$ negált változó esetén $(\overline{x}_i)^* = 1 - X_i$. Ha ϕ és ψ Boole-formulák, akkor $(\phi \wedge \psi)^* = \phi^* \cdot \psi^*$ és $(\phi \vee \psi)^* = \phi^* + \psi^* - \phi^* \cdot \psi^*$. Ezekkel a szabályokkal tetszőleges ϕ $3CNF$-re definiáltuk az $\mathbb{F}$ feletti ϕ^* polinomot, ami a $\{0,1\}^n$ halmazon megegyezik a ϕ függvénnyel.

Tegyük fel, hogy a ϕ $3CNF$ formula változói $x_1, \ldots, x_n$, és a formula m tagból áll. Ekkor a $\phi \in \mathbb{F}[X_1, \ldots, X_n]$ polinom foka legfeljebb $3m$ lesz. A ϕ ismeretében hatékonyan nyerhetünk egy a ϕ^* polinomot megadó algebrai kifejezést. A kapott algebrai kifejezés hatékony abban az értelemben is, hogy a ϕ méretében (lényegében n-ben és m-ben) polinomiális számú $\mathbb{F}$-beli művelettel kiértékelhető tetszőleges $(\alpha_1, \ldots, \alpha_n) \in \mathbb{F}^n$ helyen.

8. Példa. *Legyen ϕ a következő $2CNF$: $\phi = (x_1 \vee \overline{x}_2) \wedge (\overline{x}_1 \vee x_3)$. Ekkor*

$$\phi^* = (X_1 + (1 - X_2) - X_1 \cdot (1 - X_2)) \cdot (1 - X_1 + X_3 - (1 - X_1) \cdot X_3).$$

Legyen $f(X_1, \ldots, X_n) \in \mathbb{F}[X_1, \ldots, X_n]$ egy polinom, és $0 \leq i \leq n$. Az $f_{(i)}$ polinom legyen a következő:

$$f_{(i)}(X_1, \ldots, X_i) := \sum_{X_{i+1}=0}^{1} \sum_{X_{i+2}=0}^{1} \cdots \sum_{X_n=0}^{1} f(X_1, X_2, \ldots, X_n).$$

Az $x_1, \ldots, x_n$ változókat tartalmazó ϕ kielégítő kiértékeléseinek száma definíció szerint éppen $(\phi^*)_{(0)}$. Érvényesek továbbá a következők:

$$f_{(n)}(X_1, \ldots, X_n) = f(X_1, \ldots, X_n), \tag{4.2}$$

$$f_{(i-1)}(X_1, \ldots, X_{i-1}) = f_{(i)}(X_1, \ldots, X_{i-1}, 0) + f_{(i)}(X_1, \ldots, X_{i-1}, 1). \tag{4.3}$$

Tekintsük a következő H halmazt:

$$H = \{(f, s) |\ f \in \mathbb{F}[X_1, \ldots, X_n] \ \ s \in \mathbb{F}, \text{ és } f_{(0)} = s\}.$$

A tétel bizonyításához elegendő Arthur–Merlin-protokollt adnunk az olyan H-beli (f, s) párokra, amelyekre a $k := \max\{fokf, n, 7\}$ választással $\mathbb{F} = \mathbb{F}_p$, ahol a p egy prímszám, és $2^k < p < 2^{k+1}$. A ϕ jó kiértékeléseinek száma megkapható a $(\phi^*)_{(0)}$ modulo p értékéből, ugyanis az ilyen kiértékelések száma legfeljebb $2^n \leq 2^k < p$.

A protokoll fő lépései a következők:

1. Arthur elkéri Merlintől az $f_{(1)}(X_1)$ polinomot.

2. Merlin válasza a $g(X_1) = g_0 + g_1X_1 + \cdots + g_kX_1^k \in \mathbb{F}[X_1]$ polinom.

3. Arthur ellenőrzi, hogy $g(0) + g(1) = s$ teljesül-e. Ha nem, akkor elutasít.

A következő lépéseivel Arthur azt próbálja kizárni, hogy Merlin hamisat lépett, azaz $f_{(1)}(X_1) \neq g(X_1)$.

4. Arthur választ egy véletlen $r \in \mathbb{F}$ elemet, és kiszámítja az $s' = g(r)$ értéket. (Merlin korrekt válasza esetén $s' = f_{(1)}(r)$.)

5. Ha $n > 1$, akkor Arthur elindítja ugyanezt a protokollt az (f', s') párra, ahol $f' = f(r, X_2, \ldots, X_n)$. Ha $n = 1$, akkor közvetlenül ellenőrzi, hogy $f'_{(0)} = s'$ igaz-e.

6. Arthur végül elfogad, ha eddig sehol sem kellett elutasítania.

A protokoll nyilván hatékony: k-ban polinom időben végrehajtható. A teljessége is közvetlenül látszik. Ha $(f, s) \in H$, akkor Merlin a korrekt válaszokkal elérheti, hogy Arthur elfogad. Marad a helyesség kérdése: korlátot kell adnunk a téves elfogadás valószínűségére. Tegyük fel tehát, hogy $(f, s) \notin H$, azaz $s \neq f_{(0)}$. Jelölje $L(f, s)$ azt az eseményt, hogy $(f, s) \notin H$, de Arthur mégis elfogad, azaz Merlinnnek itt sikerül becsapnia.[14] Érvényes a következő tartalmazás:

$$L(f, s) \subseteq S \cup L(f', s'),$$

ahol

$$S = \{g(r) = f_{(1)}(r), \quad \text{de } g(X_1) \not\equiv f_{(1)}(X_1)\},$$

ugyanis $L(f, s)$ bekövetkezéséhez Merlinnek vagy be kell csapnia Arthurt az (f', s') párral, vagy pedig akkora szerencséjének kell lennie, hogy Arthur pont a $g(X_1)$ és $f_{(1)}(X_1)$ egyenlőséghalmazából választotta az r elemet (ez utóbbi esetben viszont a varázsló könnyen be is tudja csapni a királyt, innen már elég mindig az igazat mondania). Valószínűségekre térve adódik, hogy

$$\mathbf{P}(L(f, s)) \leq \mathbf{P}(S) + \mathbf{P}(L(f', s')),$$

amiből

$$\mathbf{P}(L(f, s)) \leq \frac{k}{2^k} + \mathbf{P}(L(f', s')). \tag{4.4}$$

Itt a $\mathbf{P}(S)$ becslésére a Schwartz–Zippel-tételt használtuk a $T = \mathbb{F}_p$ halmazra és az $f_{(1)}(X_1) - g(X_1)$ polinomra. Utóbbi a feltevésünk szerint nem azonosan nulla, a foka legfeljebb k, és $|T| = p > 2^k$.

A 4.4 egyenlőtlenséget ezek után alkalmazhatjuk az (f', s') párra, stb. Végül azt kapjuk, hogy

$$\mathbf{P}(L(f, s)) \leq \frac{kn}{2^k} \leq \frac{k^2}{2^k} < \frac{1}{2},$$

ami bizonyítja a protokoll helyességét. □

A bizonyításban használtuk, hogy az f bármely $\mathbb{F}_p^n$ beli kiértékelése k-ban polinom időben kiszámítható. Ez teljesül, ha f egy ϕ^* alakú polinom.

A tétel szerint olyan feladatokhoz is kaphatunk (statisztikus értelemben vett) tanút, amelyek nincsenek NP-ben. A tételt, illetve a bizonyítását úgy is értelmezhetjük, hogy viszonylag szerény számítási erővel (Arthur) ellenőrizni tudjuk egy sokkal nagyobb számítási erejű eszköz (Merlin) működésének helyességét. A véletlen fontos szerepet játszik a bizonyításban. Érdekes lehet ennek a gondolatkörnek az alkalmazása a programtesztelésben, ahol Merlin egy nagy teljesítményű, bonyolult program, amit tesztelni kívánunk, Arthur pedig a (szerényebb erőforrásokkal rendelkező) tesztelő környezet.

[14] L itt a *lóvá tesz* első betűje.

Nulla ismeretű bizonyítások

Hozzon ajándékot is, mégpedig úgy, hogy mégse hozzon,
amikor pedig belép, köszönjön is, ne is!
Mátyás király és a székely ember lánya
Illyés Gyula: Hetvenhét magyar népmese

Személyazonosítás során bizonyos információkat adunk át a bennünket azonosítónak, amelyekkel bizonyítjuk kilétünket. Ha ezeket az információkat elektronikus formában közöljük, akkor azokat könnyű másolni és megtartani. Komolyan felmerül ezért a veszély, hogy valaki megszerzi az azonosságunkat abban az értelemben, hogy sikeresen azonosíthatja magát helyettünk, a nevünkben. Ennek a problémának a kezelésére született a nulla ismeretű bizonyítás fogalma. Úgy szeretnénk bizonyítani a személyazonosságunkat, hogy eközben semmi olyan információt ne adjunk át, ami birtokában később más tudja a nevünkben azonosítani magát. Első hallásra képtelenségnek tűnik a gondolat, de mégsem az.

A *nulla ismeretű bizonyítás* (zero knowledge proof) Arthur–Merlin-protokoll egy L nyelvhez, amely során $x \in L$ esetén Merlin ennél a ténynél *semmivel sem fed fel több információt* Arthurnak. Merlin tehát Arthurnak bizonyítja az $x \in L$ tényt az AM-protokollok szabályai szerint, de ennél semmivel több érdemi információt nem közöl vele. Ezt úgy kell érteni, hogy Arthurnál a párbeszéd után csupán ismert eloszlású véletlen sorozatok maradnak. Vagyis olyasmi, amit maga is előállíthatott volna, Merlinnel való kommunikáció nélkül.[15]

A következőkben egy nulla ismeretű bizonyítást vázolunk az $L = 3SZÍN$ nyelvhez.

Adott G gráf esetén Merlin képes nulla ismerettel igazolni, hogy G 3-színezhető (ha ez utóbbi állítás igaz). A protokollhoz szükség van egyirányú függvényre. Az egyirányú függvény olyan $f : \mathcal{I}^* \mapsto \mathcal{I}^*$ függvény, amelyre $x \in \mathcal{I}^*$ esetén $f(x)$ hatékonyan (polinom időben) számítható, és nincs hatékony algoritmus (randomizált sem), amely adott $y \in \mathcal{I}^*$ esetén ad x-et, melyre $f(x) = y$). Nem tudjuk biztosan, hogy létezik-e ilyen függvény, de inkább azok a szakértők vannak többségben, akik szerint létezik ilyen. Például ilyennek véljük a jól ismert RSA-kódolást.

Tételezzük fel, hogy adott a G gráf Arthurnál és Merlinnél is, és Merlin bizonyítani szeretné Arthurnak, hogy G 3-színezhető. Merlin lépése a következő: minden $v \in V(G)$ csúcshoz választ egy E_v, D_v kódoló-dekódoló párt. Ez lehet például egy RSA-kódoló, illetve dekódoló függvény. Bármely x bemenetre $E_v(x)$ hatékonyan számítható, ám pusztán $E_v(x)$ ismeretében a D_v nélkül x nem számítható hatékonyan. Legyen a három szín halmaza

$$S = \{\text{piros, fehér, zöld}\}.$$

Merlin először képezi S egy véletlen χ keverését (permutációját): mindegyik sorrend esélye $\frac{1}{3!} = \frac{1}{6}$. Merlin ezután kiválasztja a G egy 3-színezését. A $v \in V(G)$ csúcs színét $sz(v)$ jelöli, $sz(v) \in S$. Végül Merlin minden $v \in V(G)$-re kiszámolja az $y_v = E_v(\chi(sz(v)))$ üzeneteket, és ezeket elküldi Arthurnak. Egyszerű szavakkal: a v csúcs színére alkalmazza a χ keverőfüggvényt, és az eredményt még el is kódolja egy kriptográfiai értelemben biztonságos kódolóval.

Ekkor Arthur választ egy véletlen $(u, w) \in E(G)$ élet (egyenletes eloszlás szerint), és az u, w párt elküldi Merlinnek. Válaszul a varázsló elküldi a D_u-t és D_w dekódoló kulcsokat Arthurnak. Arthur képezi a $D_u(y_u)$ és $D_w(y_w)$ értékeket. Ha ezek S-beliek (színek kódjai) és különbözőek, akkor továbblépünk, ha nem, akkor Arthur elutasítással befejezi a párbeszédet.

Ha továbblépünk, akkor az elejéről kezdődik a protokoll, Merlin új kódoló/dekódolókat, keverőfüggvényt választ, stb. Összesen legfeljebb $|E|$-szer iterálunk, és Arthur az előzőektől teljesen függetlenül választ élet. Arthur végül elfogad, ha egyetlen egy iterációs lépés során sem volt elutasítás.

[15] A pontos, formális definíciótól itt is eltekintünk. Lásd pl. [AB] 9.4, és [BV] 9. fejezet.

Vizsgáljuk meg most a protokoll helyességét. Ha G 3-színezhető, akkor van jó sz 3-színezése. Ha Merlin ilyet használt, akkor nyilván $\chi(sz(u)), \chi(sz(w))$ két különböző S-beli szín lesz. Ebben az esetben Arthur sosem fog elutasítani, vagyis $|E|$ iterációs menet után elfogad. A dialógus és a számítás összideje polinomiális G leírásának hosszában.

Mi a helyzet, ha G nem színezhető 3 színnel? Ekkor nem létezik G-hez jó sz. Azaz minden sz-re igaz ekkor, hogy van olyan (u, w) éle G-nek, amire sz nem jó: esetleg $sz(u)$ vagy $sz(w)$ nem S-beli, vagy ha igen, akkor nem különböznek egymástól. Ilyen (u, w) élet egy iterációban legalább $\frac{1}{|E|}$ valószínűséggel talál Arthur. Annak a valószínűsége, hogy a protokoll során nem talál hibát és ezért tévesen elfogad, legfeljebb $(1 - \frac{1}{|E|})^{|E|} \approx \frac{1}{e}$. Beláttuk[16] ezzel, hogy helyes AM-protokollal van dolgunk.

Miért lesz a protokoll nulla ismeretű? A dekódoló kulcsok nélkül az y_v üzeneteket Arthur nem tudja értelmezni, azok nem adnak érdemi információt G színezéséről. A D_u és D_w kulcsokkal meg tudja fejteni az y_u és y_w üzeneteket, de amit kap, egy véletlen, a korábbiaktól teljesen független $s \neq s' \in S$ elempár. Ilyet maga is tud generálni, a varázsló segítsége nélkül.

Általánosabban szólva érvényes a következő (nem bizonyítjuk):

35. Tétel (Goldreich, Micali, Wigderson)**.** *Egyirányú függvény létezése esetén minden $L \in$ NP feladathoz van nulla ismeretű bizonyítás.*

Kézenfekvően adódik a kérdés: mit mondhatunk akkor, ha esetleg nem létezik egyirányú függvény? A következő tétel szerint az interakció szerepének növelése kiküszöböli ezt a gondot. Ezt is bizonyítás nélkül közöljük.

36. Tétel (Ben-Or, Goldwasser, Kilian, Wigderson)**.** *Minden $L \in$ NP nyelvhez van nulla ismeretű bizonyítás két – egymástól elkülönített – Merlinnel.*

A tétel azt sugallja, hogy két azonosítókártya alkalmazásával lehetséges nulla ismeretű személyazonosítás.

[16] Az is kivédhető, hogy egy rossz szándékú varázsló D_u, D_w helyett olyan hamis kulcsot ad Arthurnak, ami Merlint segíti csalni.

5. fejezet

Gráfok és a véletlen

5.1 Véges Markov-láncok

A Markov-láncok a véletlen folyamatok leírására szolgáló egyszerű, de igen erőteljes és fontos eszközök. Egyszerűségük miatt jól áttekinthetők és elemezhetők, ugyanakkor meglepően sok jelenség modellezésére használhatók. Nagyon népszerűek több területen is, egyebek között a gépi tanulásban, az algoritmusok elemzésében, valamint egyszerűbb fizikai és biológiai folyamatok modellezésében.

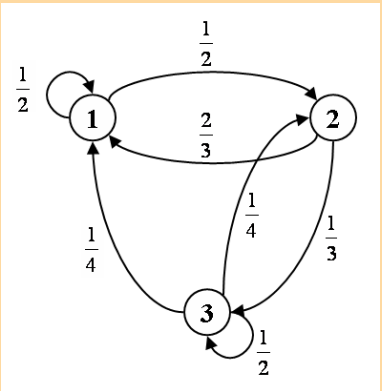
20. Definíció. *A nemnegatív valós számokkal súlyozott élű $G = (V, E, p)$ irányított gráfot* Markov-láncnak *nevezzük, ha minden $i \in V$ csúcsra teljesül, hogy*

$$\sum_{(i,j)\in E} p_{ij} = 1.$$

Itt p_{ij} az (i, j) irányított él súlyát jelöli. A gráf csúcsait egy folyamat állapotainak tekintjük. A folyamatot gyakran bolyongásnak nevezzük, amelynek az állomásai a csúcsok. A folyamat állapotról állapotra lépked. Ha éppen az $i \in V$ csúcsban van, akkor a p_{ij} lesz annak a valószínűsége, hogy a következő lépésben a $j \in V$ csúcsba lép (itt $(i, j) \in E$).

A folyamat fontos sajátossága, hogy a következő lépés valószínűsége csak attól a csúcstól függ, ahol éppen vagyunk, és független az esetleg eddig megtett lépésektől.

9. Példa. *Tekintsük az ábrán látható 3 csomópontú Markov-láncot:*



A gráf minden élére felírtuk a súlyát. Például a $(3, 1)$ él súlya $\frac{1}{4}$.

Tegyük fel, hogy $|V| = n$, és terjesszük ki a p súlyozást a G éleiről az összes lehetséges (i, j) párra a kézenfekvő módon: legyen $p_{ij} = 0$, ha $(i, j) \notin E$. A $G = (V, E, p)$ Markov-lánc mátrixa a következő $n \times n$-es mátrix:

$$\mathbf{P} = (p_{ij})_{i,j=1}^{n}.$$

A lánc definíciójából következik, hogy $\mathbf{P}$ *sorsztochasztikus*, azaz mindegyik sorában az elemek összege 1.

Legyen $\mathbf{p} = (p_1, \ldots, p_n)$ egy valószínűségeloszlás a G csúcsain, és tegyük fel, hogy a folyamat (bolyongás) éppen p_i valószínűséggel tartózkodik az i állapotban. Ekkor, a lánc szabályai szerint egyet lépve a $\mathbf{pP}$ sorvektor adja meg a következő állapot eloszlását. Általánosabban t lépés után $(t = 1, 2, \ldots)$ az aktuális állapot eloszlása $\mathbf{pP}^t$ lesz.

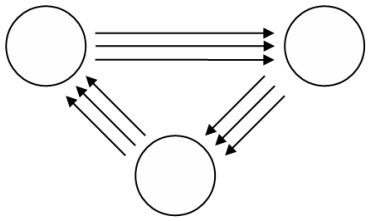
Az előző példa Markov-láncához tartozó $\mathbf{P}$ mátrix és annak négyzete $\mathbf{P}^2$ a következő:

$$\mathbf{P} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{2}{3} & 0 & \frac{1}{3} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \end{pmatrix}, \quad \mathbf{P}^2 = \begin{pmatrix} \frac{7}{12} & \frac{1}{4} & \frac{1}{6} \\ \frac{5}{12} & \frac{5}{12} & \frac{1}{6} \\ \frac{5}{12} & \frac{1}{4} & \frac{1}{3} \end{pmatrix}.$$

Ha mondjuk $\mathbf{p} = (1, 0, 0)$, vagyis az 1 csúcsból indul a bolyongás, akkor $\mathbf{pP}^2 = (\frac{7}{12}, \frac{1}{4}, \frac{1}{6})$, ami szerint az 1 csúcsból indulva $\frac{1}{6}$ a valószínűsége, hogy két lépés után éppen a 3 csúcsba jutunk.

21. Definíció. *A G Markov-lánc* irreducibilis*, ha a G gráf (amelyben csak a pozitív súlyú éleket hagyjuk meg) erősen összefüggő, vagyis bármely csúcsból bármely másikba el lehet jutni irányított úton.*

22. Definíció. *A G Markov-lánc j csúcsa (állapota)* periodikus*, mégpedig $\Delta > 1$ egész periódussal, ha annak valószínűsége, hogy j-ből s lépésben j-be jutunk, mindig 0, kivéve, ha Δ osztója s-nek.*



Az ábrán lévő körök csúcshalmazokat jelölnek, amelyeken belül tegyük fel, hogy nincs él, továbbá élek csak a nyilak irányában futnak. Ebben egy periodikus csúcshoz tartozó Δ periódus csak a 3 többszöröse lehet.

23. Definíció. *A G Markov-lánc* aperiodikus*, ha nincs periodikus csúcsa.*

24. Definíció. *Legyen $\pi = (\pi_1, \ldots, \pi_n)$ egy valószínűségeloszlás. A π stacionárius eloszlása G-nek, ha $\pi\mathbf{P} = \pi$.*

Ha egy π stacionárius eloszlás adja meg annak a valószínűségeit, hogy milyen állapotban van a folyamat (annak a valószínűsége, hogy az i csúcsban vagyunk, éppen π_i), akkor ugyanez a π

marad az állapotok aktuális eloszlása, ha lépünk egyet a G Markov-lánc szabálya szerint. A példában már vizsgált

$$\mathbf{P} = \begin{pmatrix} \frac{1}{2} & \frac{1}{2} & 0 \\ \frac{2}{3} & 0 & \frac{1}{3} \\ \frac{1}{4} & \frac{1}{4} & \frac{1}{2} \end{pmatrix}$$

mátrixú Markov-láncnak a

$$\pi = \begin{pmatrix} \frac{1}{2}, & \frac{3}{10}, & \frac{1}{5} \end{pmatrix}$$

vektor stacionárius eloszlása.

Szükségünk lesz egy további jelölésre. Legyen

$$\mathbf{P}^t = (p_{ij}^{(t)})_{i,j=1}^n, \ (t = 1, 2, \ldots),$$

azaz a $p_{ij}^{(t)}$ a $\mathbf{P}$ mátrix t-edik hatványának (i, j) helyzetű eleme. A következő tétel alapvető fontosságú a Markov-láncok elméletében.

37. Tétel. *Legyen $G = (V, E, p)$ egy véges, irreducibilis és aperiodikus Markov-lánc.*
1. A G Markov-láncnak pontosan egy $\pi = (\pi_1, \ldots, \pi_n)$ stacionárius eloszlása van. A π_j értékek mind pozitívak.
2. Minden $1 \le i, j \le n$ pár esetén létezik a $\lim_{t\to\infty} p_{ij}^{(t)}$ határérték, és független i-től.
3. Érvényesek a $\pi_j = \lim_{t\to\infty} p_{ij}^{(t)}$ egyenlőségek $(j = 1, \ldots, n)$.
4. $h_{ii} := \frac{1}{\pi_i}$ az i-ből indulva i-be való első visszatérés lépésszámának a várható értéke.

3. Megjegyzés. *A tétel első és negyedik állítása igaz marad akkor is, ha pusztán csak irreducibilis a G lánc. Az utóbbit nem bizonyítjuk, az előbbit viszont külön feladatként is megfogalmazzuk:*

30. Feladat. *Legyen $G = (V, E, p)$ véges, irreducibilis Markov-lánc, a mátrixa $\mathbf{P}$. Legyen I az n-szer n-es egységmátrix $(n = |V|)$. Igazak a következők:*

(a) A $\mathbf{Q} = \frac{1}{2}(I + \mathbf{P})$ mátrix egy irreducibilis és aperiodikus Markov-lánc mátrixa.

(b) A $\mathbf{Q}$ és $\mathbf{P}$ mátrixú Markov-láncoknak ugyanazok a stacionárius eloszlásai.

(c) A G-nek pontosan egy stacionárius eloszlása van.

31. Feladat. *Mutassuk meg, hogy a 37. tétel 2. és 3. állítása nem marad érvényben, ha csak annyit teszünk fel, hogy a G Markov-lánc irreducibilis.*

A 37. tételt nem bizonyítjuk.[1] Irreducibilis és aperiodikus G Markov-lánc esetén tekintsük a $\mathbf{\Pi}$ a $\lim_{t\to\infty} \mathbf{P}^t$ mátrixot. A tétel szerint

$$\mathbf{\Pi} = \begin{pmatrix} \pi_1 & \pi_2 & \ldots & \pi_n \\ \vdots & \vdots & & \vdots \\ \pi_1 & \pi_2 & \ldots & \pi_n \end{pmatrix}.$$

Legyen $\mathbf{p}$ egy tetszőleges valószínűségeloszlás G csúcsain. Ha $t \to \infty$, akkor a tétel alapján $\mathbf{p}P^t \to \mathbf{p}\Pi = \pi$, tehát az aktuális állapot eloszlása tart a π eloszláshoz, bármi volt is a kiinduló

[1] Bizonyítás található a [GyGyP] munkában.

állapot eloszlása. Ebben az értelemben a lánc hosszú idő alatt lényegében elfelejti a kezdő eloszlást, és sok lépés után közelítőleg π_i valószínűséggel lesz az i állapotban.

Az eddig szemlélt példánkban $\pi_3 = \frac{1}{5}$, vagyis sok-sok lépés után a lánc $\frac{1}{5}$ valószínűséggel lesz a 3-as állapotban. A tétel utolsó állítása szerint a 3-as állapotból indulva várhatóan 5 lépés múlva fog a folyamat először visszatérni ugyanebbe az állapotba.

Véletlen séta irányítatlan gráfokon

Legyen $G = (V, E)$ összefüggő, irányítatlan gráf, melyben nincs hurokél. A gráf élei mentén véletlen sétát tehetünk úgy, hogy minden $i \in V$ csúcsnál egyenletes eloszlás szerint választunk az i-hez csatlakozó élek közül, és az így kiválasztott (i, j) él j végpontjába lépünk. A folyamat egy Markov-láncnak tekinthető a következő módon: készítsük el G-ből a $G' = (V', E', p)$ Markov-láncot, ahol $V' = V$, és minden $(i, j) \in E$ élhez E'-ben két irányított él tartozzon, az egyik i-ből j-be, a másik j-ből i-be mutasson. Ha G-ben az i csúcs foka d_i, akkor legyen $p_{ij} = \frac{1}{d_i}$ a G'-beli (i, j) élek súlya. Ezzel nyilvánvalóan Markov-láncot kapunk, hiszen az egy csúcsból kiinduló élek összsúlya 1. A G' irreducibilis lesz, mert a G összefüggősége miatt G' erősen összefüggő.

Mi mondható a G' periodicitásáról? Bármely i csúcs és $(i, j) \in E'$ esetén az $i \to j \to i$ pozitív valószínűségű 2 hosszú séta, tehát a Δ periódus (ha létezik egyáltalán) csak 2 lehet. Ha a G páros, akkor $\Delta = 2$ valóban periódusa lesz G'-nek. Ha viszont G nem páros gráf, akkor G' bármely csúcsán át vezet páratlan hosszú séta is, tehát a G' lánc aperiodikus.

A G' lánc a G összefüggősége miatt irreducibilis, így a 37. tételhez fűzött megjegyzés szerint pontosan egy $\pi = (\pi_1, \ldots, \pi_n)$ stacionárius eloszlása van.

16. Állítás. *A G' stacionárius eloszlása*

$$\pi_i = \frac{d_i}{2|E|}, \quad i = 1, 2, \ldots, n.$$

Bizonyítás. A stacionárius eloszlás egyértelműsége miatt elég belátni, hogy az állításban leírt π vektorra $\pi\mathbf{P} = \pi$ teljesül. Valóban,

$$(\pi\mathbf{P})_i = \sum_{j:(j,i)\in E} \pi_j p_{ji} = \sum_{j:(j,i)\in E} \frac{d_j}{2|E|} \cdot \frac{1}{d_j} = \sum_{j:(j,i)\in E} \frac{1}{2|E|} = \frac{d_i}{2|E|} = \pi_i.$$

□

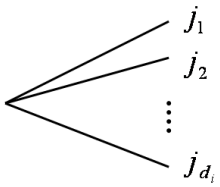
Az előbb idézett megjegyzés szerint

$$h_{ii} = \frac{1}{\pi_i} = \frac{2|E|}{d_i}$$

lesz az i-ből i-be való első visszatérésig a lépésszám várható értéke.

A következőkben arra keressük a választ, hogy hány lépésben jut el a véletlen séta i-ből j-be, ahol $i, j \in V$ tetszőleges csúcsok. Évégből vezessük be a ξ_{ij} valószínűségi változókat: ξ_{ij} értéke legyen l, ha az i-ből induló bolyongás l lépés után jut először j-be. A ξ_{ij} diszkrét valószínűségi változó, melynek lehetséges értékei: $1, 2, \ldots$. Legyen továbbá $h_{ij} = \mathbf{E}(\xi_{ij})$. A célunk értelmes felső korlátot találni a h_{ij} mennyiségekre. Ennek első lépéseként bizonyítjuk a következő állítást:

17. Állítás. *Ha $(i, j) \in E$, akkor $h_{ji} \leq 2|E|$.*



Bizonyítás. Legyenek az $i \in V$ csúcs szomszédai $j_1, j_2, \ldots, j_{d_i}$: Legyen A_ℓ az az esemény, hogy i-ből először a j_ℓ csúcsba lépünk. Alkalmazzuk a teljes várható érték tételét az $A_1, A_2, \ldots, A_{d_i}$ eseményekre és a $\xi = \xi_{ii}$ valószínűségi változóra:

$$h_{ii} = \mathbf{E}(\xi) = \frac{1}{d_i} \cdot (\mathbf{E}(\xi|A_1) + \cdots + \mathbf{E}(\xi|A_{d_i})) = (1 + h_{j_1 i}) + (1 + h_{j_2 i}) + \cdots + (1 + h_{j_{d_i} i}).$$

Innen a h_{ii} értékét behelyettesítve, majd d_i-vel szorozva kapjuk, hogy

$$2|E| = d_i + \sum_{i:(j,i)\in E} h_{ji}.$$

A jobb oldali tagok nemnegatív valós számok, így tetszőleges $(i,j) \in E$ esetén $h_{ji} \le 2|E|$. □

Az előző fontos speciális eset után nézzünk, mi mondható tetszőleges $i, j \in V$ párra:

38. Tétel. *Legyen $G = (V, E)$ összefüggő, irányítatlan gráf, amelyre $|V| > 1$. Legyen továbbá $i, j \in V$ tetszőleges csúcspár. Ekkor a G-n való véletlen sétánál a j-ből az első i-be érkezés lépésszámának h_{ji} várható értékére fennáll a $h_{ji} \le |V| \cdot 2|E|$ egyenlőtlenség.*

Bizonyítás. A G összefüggő, van tehát (nem feltétlenül egyszerű) út j-ből i-be. Ennek csúcsai legyenek $j = j_0, j_1, \ldots, j_k = i$, és itt $(j_l, j_{l+1}) \in E$ $(l = 0, \ldots, k-1)$. Nyilván feltehető, hogy $k \le |V|$.

A ξ_{ji} valószínűségi változó – a j ből induló séta első i-be érésének lépésszáma – nem nagyobb mint mint

$$\xi_{j_0 j_1} + \xi_{j_1 j_2} + \cdots + \xi_{j_{k-1} j_k},$$

hiszen utóbbi az olyan j-ből induló G-beli véletlen séta lépésszáma, amely eléri a $j_1, j_2, \ldots, j_k = i$ csúcsokat, mégpedig ebben a sorrendben, tehát végül az $i = j_k$ csúcsot is. Innen a várható értékekre azt kapjuk, hogy

$$h_{ji} \le h_{j_0 j_1} + h_{j_1 j_2} + \cdots + h_{j_{k-1} j_k}.$$

A jobb oldal tagjaira alkalmazható az előző állítás korlátja:

$$h_{ji} \le k \cdot 2|E| \le 2|V| \cdot |E|.$$

□

Ha a tételbeli G gráf esetében $|V| = n$, akkor $|E| \le \frac{n^2}{2}$, és így a tétel szerint $h_{ji} \le n^3$. A Markov-egyenlőtlenséget alkalmazva látjuk, hogy legfeljebb $2n^3$ lépésben legalább $\frac{1}{2}$ valószínűséggel eljutunk j-ből i-be:

$$\mathbf{P}(\xi_{ji} \ge 2\mathbf{E}(\xi_{ji})) = \mathbf{P}(\xi_{ji} \ge 2n^3) \le \frac{1}{2}. \tag{5.1}$$

A h_{ij} mennyiségeket egyszerűbb G gráfokra pontosan is ki tudjuk számolni. Legyen például U_n az n hosszú út. Ennek csúcsai $1, \ldots, n$, az élei pedig $(i, i+1)$, $(i = 1, \ldots n-1)$.

18. Állítás. *Az U_n gráfban $1 \le i < j \le n$ esetén $h_{ij} = (j-1)^2 - (i-1)^2$.*

Bizonyítás. A 37. tételhez fűzött megjegyzés alapján $h_{11} = h_{nn} = 2(n-1)$, és $h_{ii} = n-1$, ha $1 < i < n$. Az U_j úton a j-bel való első visszatérés várható ideje ennek alapján $2(j-1)$. Ez nyilvánvalóan eggyel több, mint az U_j úton a $j-1$-ből j-be érés várható ideje: $h_{j-1,j}+1 = 2j-2$, ahonnan $h_{j-1,j} = 2j-3$. Vegyük észre, hogy $h_{j-1,j}$ ugyanaz lesz U_n-ben, mint U_j-ben, ha $n > j$, mert $j-1$-ből az első j-be érkezést illetően a magasabb sorszámú csúcsok nem játszanak szerepet.

Ha $1 \le i < j-1 < n$, akkor az i-ből j-be érkezéshez először el kell jutni $j-1$-be, majd onnan j-be. Innen a következő összefüggés adódik:

$$h_{ij} = h_{i,j-1} + 2j - 3,$$

amiből

$$h_{ij} = (2i-1) + (2i+1) + \cdots + (2j-3) = (j-1)^2 - (i-1)^2.$$

□

32. Feladat. *Legyen C_n az n hosszú kör, amit U_n-ből kapunk az $(n,1)$ él hozzáadásával. Mutassuk, meg, hogy $h_{12} = n-1$ és $h_{13} = 2(n-2)$, ha $n \ge 4$. (Az utóbbihoz lássuk be, hogy $h_{11} = 1 + \frac{1}{4}(2+h_{13}) + \frac{1}{4}(2+h_{n-2,1})$.)*

33. Feladat. *Mutassuk, meg, hogy ha G a K_n teljes gráf, akkor $h_{ij} = n-1$, ha $i \neq j$.*

Elérhetőség irányítatlan gráfokban

Itt most talán egy kissé szokatlan szemszögből nézünk egy jól ismert, és az időigényét tekintve nagyszerűen kezelhető feladatra.

16. Számítási feladat. *Adott egy $G = (V,E)$ irányítatlan gráf, és az $s,t \in V$ csúcsai. Állapítsuk meg, hogy van-e út G-ben s-ből t-be.*

Jól ismert, hogy éllistával megadott G esetében a feladat megoldható lineáris időben, vagyis $O(|V|+|E|)$ uniform költséggel, mégpedig a szélességi vagy a mélységi bejárás alkalmazásával. A G-ben pontosan akkor van út az s-ből t-be, ha az s gyökerű szélességi, vagy mélységi fának a t csúcsa. Ezek a bejárások a gyakorlatban is igen gyorsak.

De mit mondhatunk, ha az *idő* helyett a *tár* használatában mérjük a hatékonyságot? Megoldható-e a feladat (n-ben polinomiális időben) úgy, hogy legfeljebb $O(\log n)$ bit nagyságú tárat használunk fel a G csak olvasható éllistáján kívül, ahol $n = |V|$?

A válasz igen:[2] indítsunk véletlen sétát a G csúcsain a G' Markov-lánc szabályai szerint, és tegyünk meg legfeljebb $2n^3$ lépést. Ha elérjük t-t, akkor a válasz *igen*, különben pedig *nem*. Az igenlő válasz mindig helyes, a nemleges lehet hibás, ám a hiba valószínűsége (5.1) miatt legfeljebb $\frac{1}{2}$ lehet. A bolyongás lebonyolításához csupán $O(\log n)$ bites írható-olvasható tár elegendő: kell egy számláló a lépésszám nyilvántartásához, valamint egy kis tár az aktuális $u \in V$ csúcs nyilvántartásához, ahol éppen vagyunk, illetve az u-ból való véletlen továbblépés megoldásához.

A 2005-ös esztendő egyik tudományos szenzációja volt, hogy Omer Reingold[3] $O(\log n)$ tárfelhasználású *determinisztikus* algoritmust adott a feladat megoldására. Nevezetes nyitott kérdés, hogy létezik-e a feladatra $O(\log n)$ tárat használó determinisztikus algoritmus abban az esetben, ha G *irányított gráf*.

[2] Lásd: [AKLLR], illetve [AB] 7.19. tétel.
[3] Lásd pl. a 21.21. tételt Arora és Barak [AB] művében.

A PageRank algoritmus

A Google cég viharsebes megerősödésének, a versenytársak közüli kiemelkedésének egyik okaként az internetes keresőjében alkalmazott igen jó rangsoroló algoritmusát szokás említeni.[4] A keresőrendszer egy adott k kérdésre netlapok egy H_k halmazát adja meg, amelyek a k kérdéssel kapcsolatosan érdemi információt tartalmaznak. Gyakori eset, hogy a H_k halmaz nagy, túl sok eleme van ahhoz, hogy egy felhasználó mindegyikkel foglalkozzon. Ez a gond vezet el a rangsorolás igényéhez: a H_k elemeit jó lenne valamilyen automatikus eljárással fontossági sorrendbe rendezni, és aztán ebben a sorrendben tárni a lapokat a felhasználó elé. Erre szolgáló módszert javasolt Sergei Brin és Larry Page 1998-ban. A fontosság definiálására egy $H \supseteq H_k$ netlaphalmaz elemei közötti mutatókat használja.

Legyen $|H| = n$. Az n érték lehet igen nagy, akár több milliárd is. Definiáljuk az n-szer n-es $\mathbf{A} = (a_{ij})$ mátrixot a következőképpen: a mátrix sorait és oszlopait H elemeivel indexeljük. Tegyük fel, hogy az $i \in H$ lapról d_i lapra van mutató a H halmazból, és az egyszerűség kedvéért azt is, hogy $d_i \geq 1$ igaz minden i-re. Ezek után legyen $a_{ij} = \frac{1}{d_i}$, ha az i lapról van mutató j-re, különben pedig legyen $a_{ij} = 0$. Az $\mathbf{A}$ mátrix sorsztochasztikus, tehát egy H csúcshalmazú Markov-lánc mátrixának tekinthető.

34. Feladat. *Legyen* $\mathbf{j} = (\frac{1}{n}, \ldots, \frac{1}{n})$ *az az* n *komponensű sorvektor, amelynek minden eleme* $\frac{1}{n}$. *Tegyük fel, hogy* $\mathbf{p} = \lim_{t\to\infty} \mathbf{j}\mathbf{A}^t$ *létezik. Mutassuk meg, hogy ekkor* $\mathbf{p}$ *stacionárius eloszlása* $\mathbf{A}$*-nak:* $\mathbf{pA} = \mathbf{p}$.

Első kísérletként definiáljuk a lapok fontosságát a feladatbeli $\mathbf{p}$ vektor segítségével: az i lap fontossága legyen p_i, a $\mathbf{p}$ vektor i-edik komponense. Intuitív indoklásként képzeljük el a *sztochasztikus szörfölőt*, akit az $\mathbf{A}$ mátrix leír. Egyenletes eloszlás szerint választ kiindulópontot az n lap közül; majd arról a j lapról, ahol éppen tartózkodik, egyenletes eloszlás szerint választva ugrik az egy mutató mentén elérhető d_j lap valamelyikére. Közelítőleg p_i valószínűséggel lesz sok lépés után az i lapon. Ez a szemlélet a fontosság egy rekurzív definícióját sugallja: az a lap lesz fontos, amelyre *sok fontos* lap mutat.

Az elképzelés hátulütője, hogy a limesz nem feltétlenül létezik, amint azt a következő egyszerű, zsákutca jellegű példa mutatja. Legyen az $\mathbf{A}$ mátrix

$$\mathbf{A} = \begin{pmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 1 & 0 & 0 \end{pmatrix}.$$

Ekkor $(1,1,1)\mathbf{A} = (2,1,0)$, $(2,1,0)\mathbf{A} = (1,2,0)$, $(1,2,0)\mathbf{A} = (2,1,0)$, amiből világos, hogy az $(1,1,1)\mathbf{A}^t$ sorozat nem konvergens. Az $\mathbf{A}$ mátrixra, illetve a neki megfelelő Markov-láncra nem teljesülnek a 37. tétel feltételei.

Végezzünk el egy egészen kis korrekciót: $\mathbf{A}$ helyett tekintsük a következő $\mathbf{B}$ mátrixot:

$$\mathbf{B} := 0,8 \cdot \mathbf{A} + 0,2 \cdot \mathbf{N},$$

ahol az n-szer n-es $\mathbf{N}$ mátrix minden eleme $\frac{1}{n}$. Közvetlenül látszik, hogy $\mathbf{B}$ is sorsztochasztikus, tehát Markov-lánc mátrixa. A B mátrix minden eleme pozitív (a gráfjában él vezet bármely i csúcsból bármely j-be), ezért az utóbbi Markov-lánc irreducibilis és aperiodikus is. A 37. tétel alapján létezik a csupa pozitív komponensű $\pi = (\pi_1, \ldots, \pi_n)$ stacionárius eloszlása, és

$$\pi = \lim_{t\to\infty} \mathbf{j}\mathbf{B}^t.$$

[4] Érdekes adalék ehhez, hogy a Stanford University, a PageRank algoritmus szabadalmának tulajdonosa, 2005-ben 336 millió dollárért adta el a módszer használatának jogáért kapott Google-részvényeit. Ez drámai módon hangsúlyozza azt is, hogy egy jó számítási eljárás komoly gyakorlati értéket jelenthet.

A H-beli lapok fontossága ezek után definiálható a π vektor segítségével: az i lap fontossága legyen π_i. A π vektort elég közelítőleg ismerni, hiszen csak a komponensei közötti nagyság szerinti sorrend érdekel bennünket, nem maguk a komponensek. A $\mathbf{jB}^t$ vektorok hatékonyan számíthatók iterációval a $\mathbf{jB}^t = (\mathbf{jB}^{t-1})\mathbf{B}$ azonosság alapján, és a konvergencia sebessége miatt elég csak néhány iterációs lépést tenni.

Szemléletesebben szólva a módosított algoritmus és $\mathbf{B}$ a *szeszélyes sztochasztikus szörfölőnek* felel meg, aki minden csúcsnál 0,8 valószínűséggel pontosan úgy viselkedik, mint a sztochasztikus szörfölő. A fennmaradó 0,2 a valószínűsége, hogy rájön a szeszély, ekkor egyenletesen választ egyet az összes H-beli lap közül.

Alább közvetlen, a 37. tételt nem használó bizonyítást adunk a $\mathbf{jB}^t$ sorozat konvergenciájára. A gondolatmenet mutatja azt is, hogy a konvergencia valóban igen gyors.

39. Tétel. *Legyen $0 < c < 1$ egy valós szám. Ekkor tetszőleges π kezdőeloszlás esetén a*

$$\pi\left((1-c)\mathbf{A} + c\mathbf{N}\right)^t$$

sorozat konvergál egy csupa pozitív elemű $\mathbf{p}$ stacionárius eloszláshoz, ahogy t tart a végtelenhez. A $\mathbf{p}$ független a π kezdőeloszlástól.

Bizonyítás.[5] A hatványozásnál a zárójeleket felbontva

$$\left((1-c)\mathbf{A} + c\mathbf{N}\right)^t = \sum d\mathbf{X}_t\mathbf{X}_{t-1}\cdots\mathbf{X}_2\mathbf{X}_1$$

adódik, ahol az összegezés az összes (számszerűen 2^t) olyan t hosszúságú $\mathbf{X}_t\cdots\mathbf{X}_1$ szorzatra értendő, amelynek minden tényezője $\mathbf{A}$ vagy $\mathbf{U}$. A d együttható pedig $c^k(1-c)^{t-k}$, ha éppen k darab $\mathbf{U}$ van a sorozatban. Úgy is szemlélhetjük ezt, hogy egy $(c, 1-c)$-érmét dobunk fel teljesen függetlenül t-szer egymás után; ha az i-edik dobás eredménye *fej*, akkor $\mathbf{X}_i = \mathbf{U}$, különben pedig $\mathbf{X}_i = \mathbf{A}$. Az $\mathbf{X}_t\cdots\mathbf{X}_1$ együtthatója éppen *az előfordulásának a valószínűsége lesz.*

Mivel $\mathbf{A}$ egy sorsztochasztikus mátrix, $\mathbf{AU} = \mathbf{UU} = \mathbf{U}$ teljesül. Ha tehát az $\mathbf{X}_t\cdots\mathbf{X}_1$ szorzatnál $j+1$ a legkisebb index, melyre $\mathbf{X}_{j+1} = \mathbf{U}$, akkor $\mathbf{X}_t\cdots\mathbf{X}_1 = \mathbf{UA}^j$. Ezt figyelembe véve

$$\left((1-c)\mathbf{A} + c\mathbf{N}\right)^t = (1-c)^t\mathbf{A}^t + \sum_{j=0}^{t-1} d_j\mathbf{UA}^j$$

alakba írható alkalmas valós d_j együtthatókkal. Mi lesz d_j értéke? Az éremfeldobós szemlélet szerint ez annak a valószínűsége, hogy éppen a j-edik alkalommal dobunk először *fej*et. Ez a valószínűség pedig nyilvánvalóan $d_j = c(1-c)^j$ (geometriai eloszlás). Használva továbbá, hogy $\pi\mathbf{U} = \mathbf{j}$,

$$\pi\left((1-c)\mathbf{A} + c\mathbf{N}\right)^t = \pi(1-c)^t\mathbf{A}^t + \sum_{j=0}^{t-1} c(1-c)^j\mathbf{jA}^j.$$

[5]Ezzel a gondolatmenettel Benczúr András ismertetett meg. Egy másik – Kós Gézától származó – bizonyítás valamivel több matematikai eszközt használ, mégis igen szemléletes üzenetet hordoz. A lényege a következő: tekintsük $\mathbb{R}^n$-ben azon $\mathbf{x} = (x_1, \ldots, x_n)$ pontok T halmazát, amelyekre $x_1 + \cdots + x_n = 1$. Világos, hogy ha $\mathbf{X}$ sorsztochasztikus mátrix, akkor $T\mathbf{X} \subseteq T$, és $\mathbf{xU} = \mathbf{j}$ minden $\mathbf{x} \in T$ vektorra. A T metrikus tér az $||\cdot||_1$ távolsággal. Megmutatjuk, hogy ennek a térnek

$$\mathbf{B} = (1-c)\mathbf{A} + c\mathbf{N}$$

kontrakciója. Legyen ugyanis $\mathbf{x}, \mathbf{y} \in T$. Ekkor

$$||\mathbf{x}\left((1-c)\mathbf{A} + c\mathbf{N}\right) - \mathbf{y}\left((1-c)\mathbf{A} + c\mathbf{N}\right)||_1 =$$

$$= ||(1-c)(\mathbf{x}-\mathbf{y})\mathbf{A} + c(\mathbf{xN} - \mathbf{yN})||_1 = (1-c)||(\mathbf{x}-\mathbf{y})\mathbf{A}||_1 \leq (1-c)||\mathbf{x}-\mathbf{y}||_1.$$

A $\mathbf{B}$ valóban kontrakció. A Banach-fixponttétel szerint pontosan egy fixpontja van, és minden rögzített $\mathbf{x} \in T$ esetén az $\mathbf{xB}^t$ sorozat a fixponthoz tart, mégpedig exponenciális sebességgel.

Itt a jobb oldal első tagja a nullvektorhoz tart, ha $t \to \infty$ mert $\pi\mathbf{A}^t$ egy eloszlásvektor, és ezért minden komponense legfeljebb 1. A $\pi\left((1-c)\mathbf{A}+c\mathbf{N}\right)^t$ tehát tart a $\sum_{j=0}^{\infty} c(1-c)^j\mathbf{j}\mathbf{A}^j$ sorösszeghez. Ez egyfelől létezik, hiszen komponensenként majorálható a $\sum_{j=0}^{\infty} c(1-c)^j$ konvergens számsorral. Másrészt az összegvektor minden komponense legalább $\frac{c}{n}$ az első tag miatt. Ezzel igazoltuk, hogy a határérték létezik, a kezdőeloszlástól független, és pozitív. □

A bizonyításból kiderül még, hogy a konvergencia gyors; legalább olyan, mint az $1-c$ hányadosú geometriai soré. Ennek megfelelően nagyobb c értékre gyorsabb konvergencia várható.

A módosított algoritmus által kialakított rangsorrend, illetve $\mathbf{p}$ stacionárius eloszlás úgy is értelmezhető a

$$\mathbf{p} = \lim_{t\to\infty} \pi\left((1-c)\mathbf{A}+c\mathbf{N}\right)^t = \sum_{j=0}^{\infty} c(1-c)^j\mathbf{j}\mathbf{A}^j$$

összefüggés alapján, hogy az egyenletes eloszlásból indulva c paraméterű geometriai eloszlású lépést teszünk az $\mathbf{A}$ lánc szabályai szerint (vagyis $c(1-c)^j$ annak a valószínűsége, hogy j-t lépünk így), és nézzük a helyünk így kialakuló eloszlását.

A Google jelenleg használatos rangsoroló algoritmusát a cég titokban tartja. A PageRank csupán egyike a sok eljárásnak, amelyet a fontossági sorrend kialakításához alkalmaznak.[6] Ezek alapján lehetőség van személyre szabott sorrend kialakítására is; ennek a módszerei figyelembe veszik a felhasználó eddigi kereséseit és a helyet is, ahonnan kérdez.

A Metropolis-algoritmus

For me, great algorithms are the poetry of computation. Just like verse, they can be terse, allusive, dense, and even mysterious.

Francis Sullivan: The Joy of algorithms; *Computing in Science and Engineering*, January/February 2000.

Az idézett folyóiratszámban érdekes listát tettek közzé a XX. század legjobb tíz algoritmusáról. Azokról, amelyek a szerzők véleménye szerint a legnagyobb hatással voltak a tudományos-technikai fejlődésre. Az előkelő gyűjteményben helyet kapott a gyorsrendezés, a lineáris programozási feladatok megoldására szolgáló szimplex módszer, a gyors Fourier-transzformáció, és az itt bemutatásra kerülő Metropolis-algoritmus. Az utóbbi módszernek fontos magyar vonatkozása is van: egyik szerzője a magyar származású fizikus, Teller Ede [MRRTT].

Egy sor érdekes alkalmazásnál hasznos, ha egy előre megadott (és messze nem uniform) eloszlás szerint tudunk véletlen elemet kapni. Tegyük fel például, hogy a $G=(V,E)$ gráf csúcsain értelmezett, pozitív értékű $f: V \to \mathbb{R}_+$ függvény maximumhelyét keressük. Itt is feltesszük, hogy $V=\{1,\dots,n\}$. Ha a gráf csúcsain egy olyan π valószínűségeloszlásunk van, amely erősen koncentrálódik a maximumhelyekre, akkor a π szerint választott véletlen csúcsok jó eséllyel maximumhelyek lesznek. Adott f esetében ilyen eloszlást kaphatunk, ha a csúcsok valószínűségét $g(v)=e^{f(v)/T}$-vel arányosnak választjuk, ahol T egy kis pozitív szám.

A kérdés ezek után, hogy miként tudunk a $\pi=(\pi_1,\dots,\pi_n)$ eloszlás szerinti véletlen elemet választani, ahol

$$\pi_i = \frac{g(i)}{g(1)+\dots+g(n)}.$$

Erre ad meglepően elegáns közelítő módszert a Metropolis-algoritmus. Az egyszerűség kedvéért tegyük fel, hogy G irányítatlan, hurokélek nélküli, összefüggő gráf, amelyben minden csúcs foka d. Az algoritmus egy véletlen séta a következő szabályok szerint. Tegyük fel, hogy az $i \in V$ csúcsban vagyunk éppen. Ekkor válasszunk véletlenül – egyenletes eloszlás szerint – az i

[6] Lásd pl. http://www.google.com/intl/en/corporate/tech.html.

szomszédai közül egy j csúcsot. Ha $g(j) \geq g(i)$, akkor i-ből j-be lépünk. Ha $g(j) < g(i)$, akkor $\frac{g(j)}{g(i)}$ valószínűséggel lépünk át i-ből j-be, és $1 - \frac{g(j)}{g(i)}$ valószínűséggel maradunk i-ben. Szokás szerint az összes véletlen választást teljesen függetlennek tételezzük fel.

Mutassuk meg először, hogy az így kapott véletlen séta Markov-láncot ad. Elég belátni, hogy minden $i, j \in V$ párra létezik a csak i-től és j-től függő p_{ij} átmenetvalószínűség, és $\sum_j p_{ij} = 1$ igaz minden i-re. Valóban, ha $i \neq j$ és $(i,j) \notin E$, akkor $p_{ij} = 0$. Ha $(i,j) \in E$ és $g(j) \geq g(i)$, akkor $p_{ij} = \frac{1}{d}$. Ha $(i,j) \in E$ és $g(j) < g(i)$, akkor $p_{ij} = \frac{1}{d}\frac{g(j)}{g(i)}$. Ami a hurkokat illeti, $p_{ii} = \frac{1}{d}\sum_{j\in H}(1 - \frac{g(j)}{g(i)})$, ahol a H halmaznak azok a j csúcsok az elemei, amelyekre $(i,j) \in E$, továbbá $g(j) < g(i)$. Legyen H^* azon j csúcsok halmaza, amelyekre $(i,j) \in E$ és $j \notin H$ (pont ezekre az j indexekre lesz $p_{ij} = \frac{1}{d}$). Végezetül

$$\sum_j p_{ij} = \sum_{j\in H^*} p_{ij} + \sum_{j\in H} p_{ij} + p_{ii} = \sum_{j\in H^*} \frac{1}{d} + \sum_{j\in H} \frac{1}{d}\cdot\frac{g(j)}{g(i)} + \frac{1}{d}\sum_{j\in H}\left(1 - \frac{g(j)}{g(i)}\right) =$$

$$= \sum_{j\in H^*} \frac{1}{d} + \sum_{j\in H} \frac{1}{d} = 1,$$

mivel a két összegben összesen d tag van.

Mik lesznek a az így kapott Markov-lánc G' gráfjának az élei? Ezek az $i \to j$ és $j \to i$ élek, ahol $(i,j) \in G$, továbbá hurokélek azoknál az i csúcsoknál, ahol H nem üres.

40. Tétel. *Legyen $G = (V,E)$ irányítatlan, hurokélek nélküli, összefüggő gráf, amelyben minden csúcs foka d. Legyen továbbá $g : V \to \mathbb{R}_+$ a G csúcsain értelmezett, pozitív értékű, nem konstans függvény. Ekkor az előzőekben definiált Markov-lánc irreduciblis és aperiodikus. Az egyetlen stacionárius eloszlása $\pi = (\pi_1, \ldots, \pi_n)$, ahol*

$$\pi_i = \frac{g(i)}{g(1) + \cdots + g(n)}.$$

Bizonyítás. A lánc irreducibilis, mert G összefüggő. A G' gráfban biztosan lesz hurokél, mert g nem konstans, és így az irreducibilitását is figyelembe véve látjuk, hogy aperiodikus. A 37. tétel szerint egyetlen stacionárius eloszlása létezik. Elég tehát megmutatni, hogy a megadott π eloszlás stacionárius. Ehhez pedig nyilván elegendő belátni, hogy ha $\mathbf{g} = (g(1), \ldots, g(n))$, és $\mathbf{P}$ a lánc mátrixa, akkor $\mathbf{gP} = \mathbf{g}$ teljesül. Ezt egyfelől megtehetjük közvetlen számolással. Másfelől viszont kényelmesebben célt érünk, ha észrevesszük, hogy minden i, j csúcspárra teljesül az alábbi, egyszerűen ellenőrizhető egyenlőség:

$$g(j)p_{ji} = g(i)p_{ij}.$$

Jelölje mármost $(\mathbf{gP})_i$ a $\mathbf{gP}$ vektor i-edik komponensét. Ekkor

$$(\mathbf{gP})_i = \sum_{j=1}^{n} g(j)p_{ji} = \sum_{j=1}^{n} g(i)p_{ij} = g(i)\sum_{j=1}^{n} p_{ij} = g(i).$$

A π eloszlás valóban stacionárius eloszlása a láncnak. □

A 37. tétel következményeként a Metropolis-algoritmus Markov-lánca *bármilyen* kezdő eloszlásból indulva konvergál a π eloszláshoz.

Az algoritmusnak seregnyi érdekes alkalmazása van;[7] egyáltalán nem véletlenül szerepel az elmúlt század legjobbjai között. Nem szabad azonban ezt a módszert sem mindent megoldó csodaszernek tekinteni. A stacionárius eloszláshoz való konvergencia sebessége gyakran igen lassú, és általában nehezen elemezhető.

10. Példa. *Azt szeretnénk itt szemléltetni, hogy a módszer igen lassú is lehet. Tegyük fel, hogy a B_n Boole-kockán értelmezett f függvény maximumát szeretnénk megtalálni. A B_n pontjai az n-hosszú $0,1$-vektorok. Tegyük fel, hogy f értéke mindenütt 1, kivéve egyetlen $\mathbf{v}$ vektort, ahol az értéke egy nagy pozitív szám. A $\mathbf{v}$ megkeresésére alkalmazhatjuk a Metropolis-algoritmust. Kiinduló Markov-láncként vehetjük a gyakran alkalmazott Hamming-gráfon való véletlen sétát. A Hamming-gráf csúcshalmaza B_n, és két vektor között pontosan akkor van él, ha egyetlen koordinátában különböznek egymástól. A Hamming-gráf összefüggő, reguláris, a csúcsainak száma 2^n, a csúcsainak foka n. Legyen G a Hamming-gráf véletlen sétájának Markov-lánca. Erre alkalmazható a 37. tétel utáni megjegyzés, ami alapján $h_{\mathbf{vv}} = 2^n$.*

35. Feladat. *Mutassuk meg, hogy ha $\mathbf{w}$ a G-ben a $\mathbf{v}$-től 1 távolságra levő csúcs, akkor $h_{\mathbf{wv}} = 2^n - 1$.*

A feladat azt mondja, hogy várhatóan n-ben exponenciálisan[8] sokat lép a G lánc, mire $\mathbf{w}$-ből $\mathbf{v}$-be érkezik. Másfelől vegyük észre, hogy a G-re épített Metropolis-algoritmus a $\mathbf{v}$ csúcsot kivéve minden csúcsból pontosan a G szabálya szerint lép, ezért a Metropolis-algoritmus várhatóan igen sokat lép, mire eltalál $\mathbf{v}$-be. Ha tehát f nem nyújt fogódzót arról, hogy merre vannak a maximumhelyei, akkor a Metropolis-algoritmus sem fog hatékonyan működni.

Algoritmus elemzése véletlen sétával

Térjünk vissza egy kis időre a jól ismert $2SAT$-feladathoz. Ennek bemenete egy $\phi = C_1 \wedge C_2 \wedge \ldots \wedge C_m$ alakú formula, amelynek a C_i tagjai $C_i = l_{i_1} \vee l_{i_2}$ alakúak, ahol l_{i_j} literál (Boole-változó, vagy Boole-változó negáltja).

Ha adott egy ilyen ϕ formula, akkor determinisztikus polinom időben meg tudjuk találni a benne szereplő $x_1, \ldots, x_n$ változók egy kielégítő kiértékelését, amennyiben ilyen egyáltalán létezik. Ilyen kielégítő kiértékelés keresésére szolgál a következő egyszerű, véletlent használó módszer is:

1. Válasszuk az $x_1, \ldots, x_n$ változóknak egy tetszőleges K kiértékelését, és nézzük ennél a ϕ értékét. Ha ez *igaz*, akkor készen vagyunk, és befejeztük a munkát. Ha nem, akkor

2. Válasszunk a ϕ-ből egy tetszőleges C_s tagot, aminek az értéke *hamis*. Ezután válasszunk $\frac{1}{2} - \frac{1}{2}$ valószínűséggel, a korábbi választásoktól teljesen függetlenül egy véletlen l literált C_s-ből. Az ebben szereplő x_j változó értékét változtassuk meg (ha $l = x_j$ akkor legyen $x_j = igaz$, ha pedig $l = \overline{x}_j$, akkor legyen $x_j = hamis$).

3. Számítsuk ki ϕ igazságértékét az így megváltoztatott kiértékelésnél. Ha ez *igaz*, akkor készen vagyunk. Ellenkező esetben menjünk vissza a 2. lépéshez.

[7] P. Diaconis [D] dolgozata a Markov-láncok és a Metropolis-algoritmus több érdekes alkalmazási körét érinti. Ismerteti például a rendkívül erőteljes és látványos megoldást betűhelyettesítéses (titkos) kódok megfejtésére. A 2.2. szakaszban leírja az általános Metropolis-algoritmust, amelyben a reguláris gráfon való véletlen séta helyett jóval általánosabb Markov-láncokra épül a módszer.

[8] Ez a lépésszám a B_n méretében viszont polinomiális; a mostani fejtegetésünk abban az esetben érdekes, ha az eredeti feladatunk inputhossza n-ben polinomkorlátos.

A módszer esetleg kiegészíthető azzal, hogy valamilyen k lépésszám elérésével fejezzük be a munkát akkor is, ha addig nem kaptunk jó kiértékelést, mégpedig azzal a következtetéssel, hogy a ϕ formulának valószínűleg nincs kielégítő kiértékelése.

Tegyük fel, hogy létezik kielégítő kiértékelése ϕ-nek. Mit mondhatunk ekkor az algoritmus működéséről? Mennyi idő alatt, hány iterációs lépésben találunk jó kiértékelést? Erre ad választ a következő tétel:

41. Tétel. *Ha a ϕ formulának van kielégítő kiértékelése, akkor az előző algoritmus várhatóan legfeljebb n^2 iterációs lépésben talál ilyen kiértékelést.*

A tétel bizonyításához csatolt véletlen sétákat fogunk használni.

Bizonyítás. Legyen K^* a ϕ formula egy rögzített kielégítő kiértékelése. Egy tetszőleges másik K kiértékelés *közelsége* legyen i, ha K és K^* éppen i változó értékében egyezik meg. A közelség egy 0 és n közötti érték. A közelség segítségével algoritmusunkat tekinthetjük véletlen bolyongásnak a $[0, n]$ egészein. Az i *állapot* annak felel meg, hogy az éppen szemlélt kiértékelés közelsége i. Hogyan lépünk ebben a folyamatban? Ha n-ben vagyunk, akkor nem lépünk tovább, hiszen a K^* jó kiértékelés. Ha 0-ban vagyunk, (és lépünk egyáltalán), akkor biztosan 1-re lépünk, mert ekkor a változtatás közelebb visz K^*-hoz. Ha i-ben vagyunk, ahol $0 < i < n$, és lépünk egyáltalán, akkor $p \geq \frac{1}{2}$ valószínűséggel $i+1$-re lépünk, és $q = 1-p$ valószínűséggel $i-1$ lesz az új állapot. Ez azért igaz, mert a C_s két változója közül legalább az egyik (de esetleg mindkettő) értéke különbözik a K^*-beli értéktől, így ilyet legalább $\frac{1}{2}$ eséllyel találunk. A valószínűségek itt nyilvánvalóan *nem* csak az aktuális állapottól függenek, tehát ez a rendszer nem Markov-lánc.

Hogyan tudunk felső korlátot adni az n-be érés várható idejére? Az ötlet az, hogy definiálunk a $[0, n]$ egészein egy másik G bolyongást, ami egyrészt jól elemezhető Markov-lánc, másfelől az aktuális helyzete *soha sincs jobbra* az algoritmus bolyongásának aktuális állapotától (közelségétől). Ebből már következik, hogy a G láncra vonatkozó h_{0n} érték felső korlát lesz az algoritmus várható iterációszámára. A G bolyongást a következőképpen definiáljuk: a G induljon ugyanabból az $i < n$ pontból, ahonnan az algoritmus. Tegyük fel, hogy a t-edik lépésben az algoritmus a C_s tagot választotta, és a G bolyongás a $0 < j < n$ helyzetben van. Két eset lehetséges:
(a) A C_s kiértékelése mindkét változóban eltér a K^*-tól. Ekkor az algoritmus biztosan jobbra lép. A G-t illetően ekkor sorsolunk: a G az összes korábbi sorsolástól teljesen függetlenül $\frac{1}{2} - \frac{1}{2}$, valószínűséggel lép jobbra, illetve balra.
(b) A C_s kiértékelése egy változóban tér el K^*-tól. Ekkor G lépjen ugyanúgy, mint az algoritmus.

A két szélső pontból G mindig lépjen visszafelé. Könnyű indukcióval adódik, hogy az első n-be érkezés előtt G sosem tartózkodhat jobbra a másik bolyongótól. A G bolyongás pedig nem más, mint *a véletlen séta az U_{n+1} úton* (itt nullával kezdjük a csúcsok számozását). Ehhez elég azt észrevenni, hogy a (b) esetben az algoritmus és G is $\frac{1}{2}$ valószínűséggel lép jobbra, illetve balra, minden más lépéstől teljesen függetlenül – az l literál választása miatt. Ez biztosítja, hogy a G átmenet-valószínűségei tényleg az U_{n+1} úton való véletlen séta átmenet-valószínűségei. Az úton való véletlen sétáról tanultak alapján $h_{0n} = n^2$. A tételt ezzel bizonyítottuk. □

Ha a k megállási időt $2n^2$-nek választjuk, akkor a tétel és a Markov-egyenlőtlenség alapján legfeljebb $\frac{1}{2}$ a téves válasz valószínűsége.

Az itt látott gondolat, az eredetihez csatolt másik folyamat alkalmazása igen fontos eszköz a véletlen folyamatok elméletében.

5.2 Komplex hálózatok

A nehezebb feladatot: egy szögecselő munkást a Ford-művek műhelyéből, ezek után magam vállaltam, és négy láncszemmel szerencsésen meg is oldottam. A munkás ismeri műhelyfőnökét,

műhelyfőnöke magát Fordot, Ford jóban van a Hearst-lapok vezérigazgatójával, a Hearst-lapok vezérigazgatójával tavaly alaposan összeismerkedett Pásztor Árpád úr, aki nekem nemcsak ismerősöm, de tudtommal kitűnő barátom – csak egy szavamba kerül, hogy sürgönyözzön a vezérigazgatónak, hogy szóljon Fordnak, hogy Ford szóljon a műhelyfőnöknek, hogy az a szögecselő munkás sürgősen szögecseljen nekem össze egy autót, éppen szükségem lenne rá.

Karinthy Frigyes: Láncszemek

A nagy méretű, bonyolult szerkezetű hálózatok vizsgálata jelentős szerepet játszik több fontos alkalmazási területen. Ilyen hálózatnak tekinthető például a társadalom (a csúcsok az emberek, az élek az ismeretségek), a web (csúcsok a weblapok, a köztük menő mutatók adják az éleket), a szakértő közösségek (itt akkor van él két csúcs között, ha a csúcsokat jelentő személyek dolgoztak együtt közös munkán), vagy egy sejt mint kémiai hálózat (a csúcsok a kémiai összetevők, két csúcs között él van, ha közvetlen kapcsolat áll fenn közöttük), és még igen sok más példa említhető. Több tudományterületen is megszületett a felismerés, hogy pusztán e hálózatok struktúrája, gráfszerkezete egy sor értékes információt ad a szóban forgó jelenségkörről. A megnövekedett érdeklődést támogatja a számítástechnika és számítástudomány erőteljes fejlődése. A nagy erejű számítógépek, a megnövekedett adattárolási kapacitások, és a gyors (gyakran külső táras adatszerkezetekkel dolgozó, adatbányászati filozófiát követő[9]) algoritmusok kiváló lehetőségeket nyújtanak az ilyen hálózatok elemzésére, empirikus tanulmányozására. Az utóbbi pár évtizedben kialakult a *hálózatok tudománya*, amely több hasznos fogalmat vezetett be. A terület egyik legjelesebb kutatója Barabási Albert-László erdélyi származású fizikus.[10]

A hálózatok felépülésben, szerkezetében gyakran szerepet játszanak bizonyos determinisztikus mechanizmusok és véletlenszerű (vagy legalábbis ilyennek gondolt) jelenségek. Megértésüket segítik a gráfmodellek. Ezekből itt röviden ismertetünk hármat.

Erdős–Rényi-gráfok

Az Erdős–Rényi-modell egy véletlen gráfok előállítására szolgáló modell, amely igen hasznosnak bizonyult egyebek között a gráfokkal kapcsolatos kombinatorikai és algoritmikus kérdések tanulmányozásában.

25. Definíció. *Legyen n pozitív egész, $0 \leq p \leq 1$. Az $ER(n,p)$ Erdős–Rényi-gráf egy irányítatlan gráf, amelynek csúcspontjai $1, 2, \ldots, n$. Az (i,j) pár $(i \neq j)$ esetében sorsolunk: p valószínűséggel lesz (i,j) éle a gráfnak. Ezek a sorsolások egymástól teljesen függetlenek.*

Az $\mathrm{ER}(n,p)$ tulajdonképpen egy valószínűségi mező az olyan irányítatlan gráfok halmazán, amelyek csúcsai $1, \ldots, n$. Ebben a térben tehát az elemi események gráfok; az m élű G gráf valószínűsége

$$p^m (1-p)^{\binom{n}{2}-m}.$$

36. Feladat. *Mennyi lesz az $ER(n,p)$ gráfban a háromszögek (K_3 részgráfok) számának várható értéke?*

Az Erdős–Rényi-gráfokkal tulajdonképpen már találkoztunk: az $R(t,t)$ Ramsey-szám alsó becslése során (3. fejezet, 21. tétel) valójában az $ER(N, \frac{1}{2})$ véletlen gráfot vizsgáltuk (a piros

[9] Ami némi egyszerűsítéssel azt mondja, hogy csak a legfeljebb lineáris idejű számítások azok, amelyek igazán nagy adathalmazokon elvégezhetők.

[10] [Ba] olvasmányos bevezetőt nyújt a hálózatok világába. Érdemes továbbá megnézni, illetve elolvasni Barabási ilyen tárgyú előadását a Mindentudás Egyetemén: http://www.mindentudas.hu/barabasialbertlaszlo/index.html.

élek által alkotott gráf), és arra kerestünk feltételt, hogy pozitív valószínűséggel ne legyen sem benne, sem a komplementerében K_t teljes részgráf.

Legyen ℓ_{ij} az (i,j) él indikátorváltozója, azaz legyen $\ell_{ij}=1$, ha $(i,j)\in ER(n,p)$, és $\ell_{ij}=0$ egyébként. Az ℓ_{ij} valószínűségi változók teljesen független p paraméterű Bernoulli-változók $(1\le i<j\le n)$. Az élek száma $m=\sum_{i,j}\ell_{ij}$, ennek várható értéke $\mathbf{E}(m)=\binom{n}{2}p$. Alkalmazható a Chernoff-egyenlőtlenség: tetszőleges $1\ge\epsilon\ge 0$ esetén a

$$\mathbf{P}\left(\left|m-\binom{n}{2}p\right|>\epsilon\binom{n}{2}p\right)$$

valószínűség exponenciális sebességgel tart nullához, ha $n\to\infty$. Az élek száma tehát igen erősen koncentrálódik a várható érték körül.

Jelölje D az $\mathrm{ER}(n,p)$ gráfban egy csúcs (mondjuk az 1) fokát. A D egy valószínűségi változó, amelynek könnyű megadni az eloszlását:

$$\mathbf{P}(D=k)=\binom{n-1}{k}p^k(1-p)^{n-1-k},\quad k=0,1,\dots,n-1.$$

Ebből, vagy akár az indikátormódszerrel látjuk, hogy $\mathbf{E}(D)=(n-1)p$. Ezúttal is teljesen független Bernoulli-összegről van szó, így alkalmazható a Chernoff-egyenlőtlenség. Bármely $1\ge\epsilon\ge 0$ esetén

$$\mathbf{P}\left(|D-(n-1)p|>\epsilon(n-1)p\right)\le 2e^{-\frac{\epsilon^2(n-1)p}{3}}.$$

Itt is erős koncentráció mutatkozik a várható érték körül.

Erdős Pál és Rényi Alfréd $ER(n,p)$-vel kapcsolatos híres dolgozatainak talán a legfontosabb eredménye a küszöbfüggvények fogalma, és annak bizonyítása, hogy egy sor érdekes tulajdonságnak van küszöbfüggvénye. Legyen A egy gráftulajdonság (pl. A lehet az, hogy *a gráfban van háromszög*).

26. Definíció. *A $k(n):\mathbb{Z}^+\to\mathbb{R}^+$ függvény az A tulajdonság küszöbfüggvénye, ha bármely olyan $p(n)$ valószínűségek esetén, amelyekre $\lim_{n\to\infty}p(n)/k(n)=0$, teljesül, hogy*

$$\lim_{n\to\infty}\mathbf{P}(ER(n,p(n))\text{-ben igaz } A)=0;$$

továbbá, ha bármely olyan $q(n)$ valószínűségek esetén, amelyekre $\lim_{n\to\infty}q(n)/k(n)=\infty$, teljesül, hogy

$$\lim_{n\to\infty}\mathbf{P}(ER(n,q(n))\text{-ben igaz } A)=1.$$

A $k(n)$ küszöbfüggvény létezése azt jelenti, hogy van egy viszonylag éles határ, ami alatti élsűrűségnél az ER-gráfban szinte biztosan biztosan nem teljesül A, a határ feletti élsűrűség mellett pedig szinte biztosan igaz lesz A.

11. Példa. *Megmutatjuk, hogy $k(n)=\frac{1}{n^2}$ küszöbfüggvénye annak az A tulajdonságnak, hogy* a gráfnak van éle. *Tegyük fel először, hogy a $p(n)$ valószínűségekre $\lim_{n\to\infty}p(n)/k(n)=\lim_{n\to\infty}p(n)n^2=0$. Az $ER(n,p(n))$ gráfban az élszám várható értéke $\mathbf{E}(m)=\binom{n}{2}p(n)<n^2p(n)\to 0$, ha $n\to\infty$, ahonnan $\mathbf{P}(m>0)\to 0$.*

Legyenek másfelől a $q(n)$ valószínűségek olyanok, hogy $q(n)/k(n)=q(n)n^2\to\infty$, ha $n\to\infty$. Megjegyezzük, hogy ekkor nyilván $q(n)\binom{n}{2}\to\infty$ is igaz. Mármost

$$\mathbf{P}(m=0)=(1-q(n))^{\binom{n}{2}}=\left((1-q(n))^{\frac{1}{q(n)}}\right)^{q(n)\binom{n}{2}}\le\left(e^{-1}\right)^{q(n)\binom{n}{2}}\to 0,$$

ha $n \to \infty$. Tehát $\mathbf{P}(m > 0) \to 1$. Az egyenlőtlenségnél a tetszőleges valós x-re érvényes $(1 - x) \leq e^{-x}$ becslést alkalmaztuk.

Bizonyítás nélkül megemlítünk néhány érdekes gráftulajdonságot, a küszöbfüggvényével együtt:

- *Van a gráfban legalább másodfokú csúcs:* $k(n) = \frac{1}{n^{3/2}}$.
- *A gráfban megjelenik bármely rögzített d pontú részfa (pl. d pontú csillag):* $k(n) = \frac{1}{n^{\frac{d}{d-1}}}$.
- *Van a gráfban d pontú kör:* $k(n) = \frac{1}{n}$.
- *A gráf összefüggő:* $k(n) = \frac{\log n}{n}$.

Érdemes olykor a $k(n)$ küszöböt *időnek* tekinteni, ahogy az azonosan nulla függvény felől az azonosan 1 függvény felé növekszik, és ennek megfelelően az Erdős–Rényi-gráfok változását folyamatként vizsgálni, amint p megy 0-tól 1-ig. Az $ER(n, p)$ gráfok tulajdonságai ekkor (fordulatos[11]) fejlődési történetként jelennek meg: $\frac{1}{n^2}$-nél az első él, majd $\frac{1}{n^{3/2}}$ időben két csatlakozó él, stb.

Az Erdős–Rényi-gráfoknak $p(n) \geq \frac{\log n}{n}$ esetén $O(\log n)$ az átmérője, vagyis bármely két csúcs között van rövid út. Ez az ún. *kis világ*-jelenség, amit több érdekes hálózatban megfigyeltek, és amit Karinthy Frigyes már 1929-ben leírt az emberi ismeretségi kapcsolatok gráfjáról.

Az $ER(n+1, \frac{\lambda}{n})$ gráfokban tetszőleges csúcs fokszámának az eloszlása $(n, \frac{\lambda}{n})$ paraméterű binomiális eloszlás, ami a 4. tétel szerint nagy n-re jól közelíthető λ paraméterű Poisson-eloszlással. Az utóbbiról pedig látszik, hogy a $\mathbf{P}(\eta = k)$ valószínűségek exponenciálisan csökkennek a k növekedtével. A fokszámok eloszlása tehát erősen koncentrálódik a várható érték körül.

A Watts–Strogatz-modell

Egy sor érdekes hálózatban megfigyelhető a *csomósodás* jelensége: egy csúcs szomszédai között több kapcsolat van, mint más hasonló méretű csúcshalmazokon belül. Gondoljuk például arra a gráfra, amelynek emberek a csúcspontjai, és két csúcs akkor van összekötve, ha az illetők ismerik egymást. Egy ember ismerősei közül sokan jó eséllyel szintén ismerősei lesznek egymásnak. Az $ER(n, p)$ gráfoknál nem tapasztalunk ilyesmit; bármely csúcshalmazon belül körülbelül a lehetséges élek p-szerese lesz tényleg él. Mint látni fogjuk, a Watts–Strogatz-modellnek alaptulajdonsága a csomósodás.

A $WS(n, k, p)$ Watts–Strogatz-gráf csúcsai $1, 2, \ldots, n$ (itt $n \gg k$ pozitív egészek, $0 \leq p \leq 1$ egy valószínűség). A gráf élei kezdetben $(1, 2), (2, 3), \ldots, (n-1, n), (n, 1)$, vagyis egy körünk van. A következő lépésben kössük össze éllel a kör mentén egymástól legfeljebb k távolságra levő csúcsokat. Egy csúcs foka ekkor $2k$ lesz, és az élek száma kn. A következő menetben az eddig kapott összes él esetén sorsolunk. Az élet $1 - p$ valószínűséggel meghagyjuk, p valószínűséggel teljesen újat sorsolunk helyette: egyenletesen választunk az összes (i, j) pár közül. Végül töröljük az esetleges hurokéleket és többszörös éleket. Az összes véletlen választásunk teljesen független a többitől.

A megmaradó régi élek *csomósodást* biztosítanak, az új élek pedig *kis világot*: viszonylag rövid utakat bármely két csúcs között. A WS-gráfok kis p esetén a determinisztikus lépésekkel megadott szabályos gráfra hasonlítanak; a p növekedésével egyre inkább olyanokká válnak, mint az Erdős–Rényi-gráfok.

[11] Különös, első látásra váratlan fordulat például, hogy egy nagy fa előbb jelenik meg, mint bármilyen kicsi kör.

Albert–Barabási-gráfok

Mert akinek van, annak adatik, és bővelkedik...
Máté Evangéliuma, 13,10–17

Vannak olyan nagy hálózatok, amelyekben a csúcsok fokszámának az eloszlása nem annyira koncentrálódik a várható érték körül, mint az ER vagy a WS gráfoké. Több érdekes gráf esetén sikerült kimérni, hogy a fokszámok eloszlása sokkal lassabban cseng le, pontosabban *hatványeloszlást* követ. Más szóval, ha $P(k)$ jelöli annak a valószínűségét, hogy egy csúcs foka k akkor a $P(k)$ sorozat $k = 0, 1, \ldots, n-1$ nagyjából arányos a $k^{-\gamma}$ sorozattal, ahol $\gamma > 0$ konstans. A nevezetes példáknál γ leggyakrabban 2 és 3 közé esik. Például, ha

- a gráf csúcsai: amerikai filmszínészek, él akkor van közöttük, ha játszottak közös filmben. Ekkor $\gamma \approx 2{,}3 \pm 0{,}1$.
- A gráf csúcsai matematikusok, él akkor van közöttük, ha írtak közös tudományos dolgozatot. Itt $\gamma = 2{,}1$.
- A gráf csúcsai telefonszámok. Él akkor fut közöttük, ha egy adott nap volt közöttük beszélgetés. Itt is $\gamma = 2{,}1$.
- A csúcsok az internettartományok (domain a szokásos elnevezés). Két csúcs között akkor van él, ha van egyikről mutató a másikra. Ekkor $\gamma \approx 2{,}15 - 2{,}20$.

Hogyan modellezhetők az ilyen gráfok? Milyen egyszerű konstrukciós szabályok adnak hatványeloszlást a fokszámokra? Az első és máig legfontosabb ilyen tulajdonságú gráfmodellt Albert Réka és Barabási Albert-László javasolta. Legyenek m és t pozitív egészek. $AB(m, t)$ Albert–Barabási-gráfnak t csúcsa és mt éle van. Tulajdonképpen természetes és egyszerű rekurzív folyamat eredményeként jön létre az $AB(m, 1), AB(m, 2), \ldots, AB(m, t)$ gráfsorozat.

1. Egyetlen v_1 csúcs és m hurokél alkotja az $AB(m, 1)$ gráfot.
2. Tegyük fel, hogy $AB(m, t-1)$ már elkészült, t csúcsa és $m(t-1)$ éle van. $AB(m, t)$ innen egy új v_t csúcs hozzáadásával keletkezik A v_t csúcshoz illeszkedően m élet konstruálunk a véletlen segítségével. A v_t-ből induló l-edik élet ($l = 1, \ldots, m$) a gráf véletlenül választott v_j ponthoz kötjük (a véletlen választások itt is teljesen függetlenek egymástól). A v_j-hez kötés valószínűsége d_j-vel arányos. Pontosabban fogalmazva, ha a kötés előtt a csúcsok fokai rendre $d_1, d_2, \ldots, d_t$, akkor a v_j csúcshoz kötés valószínűsége

$$\frac{d_j}{d_1 + d_2 + \cdots + d_t}.$$

Az Albert–Barabási-gráfok épülésekor érvényesül a *preferenciális kötődés* elve. Egy új csúcs nagyobb eséllyel kötődik kapcsolatokban már eddig is gazdagabb csúcshoz. Az ismeretségi gráfok esetén is jól megfigyelhető ez a jelenség.[12] Ezáltal a hálózatban lesz néhány az átlagosnál sokkal magasabb fokú csúcs is. Megmutatható, hogy a nem túl nagy k értékekre a fokszámok $P(k)$ eloszlása az AB-gráfokban hatványeloszlást követ, mégpedig $\gamma = 3$ kitevővel.

[12] Az iwiw-kapcsolatok gráfjában 2006-os adatok szerint az átlagos fok valamivel 100 alatt volt, ugyanakkor előfordult több tízezres fokú csúcs is.

Algoritmikus kis világ

Az emberi kapcsolatok gráfjának vizsgálatában meghatározó szerepet játszottak Stanley Milgram 60-as években végzett kísérletei. Ezek egyikében egy találomra választott nebraskai személy (az s forrás) megkapta egy massachusetts-i személy (a t célszemély) címét és még néhány alapvető információt (pl. t foglalkozását). Az s feladata az volt, hogy próbáljon egy üzenetet eljuttani t-hez meghatározott szabály szerint: az üzenetet küldje el egy olyan s_1 személynek, aki egyfelől közeli ismerőse (keresztnevükön szólítják egymást), másfelől s vélekedése szerint esetleg közelebb van t-hez, abban az értelemben, hogy ugyanezen szabály szerint rövidebb úton tudja eljuttatni az üzenetet t-hez. Tulajdonképpen azt kérték tőle, hogy továbbítsa a küldeményt ahhoz a szomszédjához az emberi kapcsolatok gráfjában, aki szerinte a szomszédai közül a legközelebb van t-hez ebben a gráfban. Az s_1 személy ugyanezeket az instrukciókat kapta, és a többiek is, akikhez a folyamat során eljutott az üzenet. Sok ilyen kísérletet végeztek el. Gyakran megesett, hogy az üzenet egyáltalán nem érkezett meg a végső címzett t-hez. Nagyjából az üzenetek harmada ért célba, ezek viszont meglepően rövid úton. Sok kísérletre vetítve a gráfban megtett lépések számának átlaga hat alatt maradt! Ez ékesen mutatja a kis világ-jelenséget a kapcsolatok gráfjában.

Jon Kleinberg amerikai számítástudós egy további fontos üzenetet olvasott ki a kísérlet eredményéből: ez nem pusztán rövid utak létezését mutatta ki, hanem egy igen erőteljes, egyszerű és hatékony *elosztott működésű algoritmus* létét is, amivel az üzenet rövid úton eljuttatható a címzetthez. Itt az elosztott működés lényeges többlet. Tudjuk, hogy a rövid utak jól feltérképezhetők a szélességi keresés segítségével. Ez azonban sokkal több információt használ fel a gráfról, mint ami a Milgram-kísérletek résztvevőinek megengedett volt. Kleinberg a kísérlet eredményeit elemezve arra jutott, hogy a rövid utak mellett a kapcsolatok gráfja még további, a decentralizált üzenetküldés szempontjából hasznos információkat is tartalmaz. Kleinberg ezután kidolgozott egy modellcsaládot, ahol kimutatható az *algoritmikus kis világ*-jelenség: egyszerű és hatékony elosztott algoritmussal lehet rövid utat találni két csúcs között. Ezek közül a legegyszerűbbel ismerkedünk meg itt.[13]

A Kl_n Kleinberg-gráf építésekor a C_n körből indulunk ki, ahol $n > 2$ egész. Ennek csúcsai $1, \dots, n$ az élei $(i, i+1)$, ha $1 \le i < n$, továbbá $(1, n)$. Ezek az élek adják a csúcsok *helyi ismerőseit*. Minden csúcsnak választunk pontosan egy *távoli ismerőst*. Jelentse $d(u, v)$ az u és v csúcsoknak a C_n körön mért távolságát. Ez egy 0 és $\frac{n}{2}$ közötti egész. Az u csúcs távoli ismerősét a többi csúcsétől teljesen függetlenül véletlenül választjuk. Annak a valószínűsége, hogy v-t választjuk, legyen arányos $\frac{1}{d(u,v)}$-vel. Vegyük észre, hogy az így kapott él irányított. Tipikusan az u távoli ismerősének a távoli ismerőse valaki más lesz, nem u. A v választásának valószínűsége

$$\frac{d(u,v)^{-1}}{d(u,v_1)^{-1} + \dots + d(u,v_{n-1})^{-1}}$$

lesz, ahol $v_1, \dots, v_{n-1}$ a gráf u-tól különböző csúcsai. A későbbi gondolatmenetünkhöz érdemes lesz felső becslést adni a nevezőben szereplő összegre:

$$\sum_{w \neq u} \frac{1}{d(u,w)} \le 2\left(1 + \frac{1}{2} + \dots + \frac{1}{\lfloor \frac{n}{2} \rfloor}\right) = 2H_{\lfloor \frac{n}{2} \rfloor} \le 2\left(1 + \log \lfloor \frac{n}{2} \rfloor\right) \le$$

$$\le 2\left(1 + \log \frac{n}{2}\right) \le 2\left(1 + \log_2 \frac{n}{2}\right) = 2\log_2 n.$$

Az első egyenlőtlenségnél az használtuk, hogy u-tól adott d távolságra legfeljebb két csúcs van; a következőnél a harmonikus szám ismert felső korlátja van, végül pedig áttértünk 2-es alapú

[13] A témáról sok érdekeset olvashatunk [EK] 20. fejezetében.

logaritmusra. A becslésből azt nyerjük, hogy ha $u \neq v$, akkor annak a valószínűsége, hogy v lesz az u távoli ismerőse, legalább

$$\frac{1}{2d(u,v)\log_2 n}.$$

Megmutatjuk, hogy a Kl_n gráfban érvényes az algoritmikus kis világ. Másként fogalmazva: nem csak annyi igaz, hogy két csúcs között várhatóan van rövid út, hanem ilyen utat hatékony, pusztán lokális választásokon alapuló algoritmussal kaphatunk is. Az eljárás bemenete az $s, t \in V(Kl_n)$ csúcspár. A cél eljutni s-ből t-be.

A rövidlátó algoritmus első lépésében az s-től ahhoz az u ismerőséhez lépünk, amelyre $d(u,t)$ a legkisebb. Ha $u \neq t$, akkor innen ugyanezen szabály szerint lépünk tovább. Általában, az út során minden további érintett csúcsnál ugyanezzel a mohó választással élünk.

A résztvevő csúcsoknak csak a saját szomszédságukat kell ismerni, illetve tetszőleges t csúcsra meg kell tudni állapítaniuk annak távolságát az ismerőseiktől. Ennél több információra nincs szükségük a gráfról.

42. Tétel. *Legyenek s, t a Kl_n gráf véletlenül választott csúcsai. A rövidlátó algoritmus várhatóan legfeljebb $3(\log_2 n)^2$ lépésben eljut s-ből t-be.*

Itt egy *lépés* egy élen való áthaladást jelent.

Bizonyítás. A t megközelítését fázisokra osztjuk. Azt mondjuk, hogy a j-edik fázisban vagyunk, ha az utunk során éppen meglátogatott u csúcsra $2^j \leq d(u,t) < 2^{j+1}$ igaz. Jelölje X_j a j-edik fázisból induló lépések számát. X_j nyilván egy valószínűségi változó, és az algoritmus X teljes lépésszámára igaz a következő:

$$X \leq 1 + X_1 + X_2 + \cdots + X_{\lfloor \log_2 n \rfloor}.$$

Az $\mathbf{E}X$ felső becsléséhez elég tehát felső becslést adni az $\mathbf{E}X_j$ várható értékekre.

Evégből tegyük fel, hogy éppen az u csúcsnál tartunk a j-edik fázisban, és $d(u,t) = d$. Biztosan kilépünk ebből a fázisból, ha az u távoli ismerőse legfeljebb $\frac{d}{2}$ lépésnyire van t-től. Mekkora az esély erre? Legyen I a t-től legfeljebb $\frac{d}{2}$ távolságra levő csúcsok halmaza. Legalább d csúcs van I-ben, és ezek közül az u-hoz legtávolabbinak az u-tól való távolsága legfeljebb $d + \frac{d}{2} = \frac{3d}{2}$. Annak a valószínűsége, hogy a $w \in I$ lesz az u távoli ismerőse, az eddigiek alapján legalább

$$\frac{1}{2d(u,w)\log_2 n} \geq \frac{1}{3d/2} \cdot \frac{1}{2\log_2 n} = \frac{1}{3d\log_2 n}.$$

Mivel az I-ben legalább d csúcs van, annak a valószínűsége, hogy u távoli ismerőse I-beli, legalább

$$d \cdot \frac{1}{3d\log_2 n} = \frac{1}{3\log_2 n}.$$

Az algoritmus minden lépéssel közelebb jut t-hez, ezért feltehetjük, hogy az u távoli ismerősét éppen akkor sorsoljuk, amikor u-ból lépni készülünk.

Annak az esélye, hogy legalább i lépést teszünk a j-edik fázisban, legfeljebb

$$\left(1 - \frac{1}{3\log_2 n}\right)^{i-1},$$

amiből

$$\mathbf{E}X_j = \sum_{i=1}^{\infty} \mathbf{P}(X_j \geq i) \leq \sum_{i=1}^{\infty} \left(1 - \frac{1}{3\log_2 n}\right)^{i-1} = 3\log_2 n.$$

Végezetül innen

$$\mathbf{E}X \le 1 + \mathbf{E}X_1 + \mathbf{E}X_2 + \cdots + \mathbf{E}X_{\lfloor \log_2 n \rfloor} \le \log_2 n \cdot 3 \log_2 n = 3(\log_2 n)^2.$$

Itt az előző becslést használtuk, amikor $j \ge 2$, másrészt pedig világos, hogy $1+X_1 \le 3 \le 3\log_2 n$, hiszen legfeljebb 3 távolságból a közeli ismerősök mentén is elég 3 lépés, és $n > 1$. A bizonyítás ezzel teljes. □

Irodalomjegyzék

[A] M. K. Agoston: *Computer Graphics and Geometric Modelling*, Springer-Verlag, 2005

[AKS] M. Agrawal, N. Kayal, N. Saxena: PRIMES is in P, *Annals of Mathematics*, 160(2004), 781–793, http://www.cse.iitk.ac.in/users/manindra/algebra/primality_v6.pdf

[AZ] M. Aigner, G. M. Ziegler: *Bizonyítások a könyvből*, Typotex, 2009

[AKLLR] R. Aleliunas, R. M. Karp, R. Lipton, L. Lovász, C. Rackoff: Random walks, universal traversal sequences and the complexity of maze problems, *Proc. IEEE FOCS'79*; 1979, 29–31

[AB] S. Arora, B. Barak: *Computational Complexity*, Cambridge University Press, 2009

[B] Babai, L.: Trading Group Theory for Randomness, *Proceedings of 17th ACM Symposium on the Theory of Computation*, Providence, RI, 1985

[BS] E. Bach, J. Shallit: *Algorithmic number theory*, MIT Press, 1996

[BT] Balázs M., Tóth B.: *Valószínűségszámítás I.*, elektronikus jegyzet

[Ba] Barabási Albert-László: (*Behálózva: A hálózatok új tudománya*, Magyar Könyvklub, 2003)

[Be] S. J. Berkowitz: On computing the determinant in small parallel time using a small number of processors, *Information Processing Letters*, 18(1984), 147–150

[BV] Buttyán L., Vajda I.:*Kriptográfia és alkalmazásai*, Typotex, 2004

[Cs] L. Csánky: Fast parallel matrix inversion algorithms, *SIAM Journal on Computing*, 5(1976), 618–622

[D] P. Diaconis: The Markov chain Monte Carlo revolution, *Bulletin of the American Math. Society,* 46(2009), 179–205

[DH] C. J. H. McDiarmid, R. Hayward: Strong concentration for quicksort, *Proc. of the 3rd ACM-SIAM SODA*, 1992, 414–421

[EK] D. Easley, J. Kleinberg: *Networks, Crowds, and Markets: Reasoning About a Highly Connected World,* Cambridge University Press, 2010

[F] W. Feller: *Bevezetés a valószínűségszámításba és alkalmazásaiba*, Műszaki Könyvkiadó, 1978

[FGy] Freud R., Gyarmati E.: *Számelmélet*, Nemzeti Tankönyvkiadó, 2006

[FGMSZ] M. Furer, O. Goldreich, Y. Mansour, M. Sipser, S. Zachos: On completeness and soundness of interactive proof systems, *Advances in Computing Research*, 5(1989), 429–442

[GMR] S. Goldwasser, S. Micali, C. Rackoff: The Knowledge Complexity of Interactive Proofs, *Proceedings of 17th ACM Symposium on the Theory of Computation*, Providence, RI, 1985

[GS] C. M. Grinstead, J. L. Snell: *Introduction to probability,* American Mathematical Society, 1988

[Gy] Györfi L.: Mindentudás egyeteme előadás http://mindentudas.hu

[GyGyP] Györfi L., Győri S., Pintér M.: *Tömegkiszolgálás informatikai rendszerekben*, Műegyetemi Kiadó, 2005

[GyGyV] Györfi L., Győri S., Vajda I.: *Információ és kódelmélet*, Typotex, 2002

[GyKKW] L. Györfi, M. Kohler, A. Krzyżak, H. Walk: *A Distribution-Free Theory of Nonparametric Regression*, Springer-Verlag, 2002

[I] *Informatikai algoritmusok*, szerk.: Iványi A., ELTE Eötvös Kiadó, 2005

[J] J. JaJa: A perspective on quicksort, *Computing in Science and Engineering*, Jan/Feb (2000), 43–49

[KKT] D. R. Karger, P. N. Klein, R. E. Tarjan: A Randomized Linear-Time Algorithm to Find Minimum Spanning Trees, *Journal of the ACM,* 42(1995), 321–328

[KRSz] Katona Gy., Recski A., Szabó Cs.: *A számítástudomány alapjai*, Typotex, 2002

[Ke] Ketskeméty L.: *Valószínűségszámítás*, Műegyetem Kiadó, 2005

[Ki] V. King: A simpler minimum spanning tree verification algorithm, *Algorithmica,* 18(1997), 263–270

[Ko] J. Komlós: Linear verification for spanning trees, *Combinatorica,* 5(1985), 57–65

[L] Lovász L.: *Algoritmusok bonyolultsága*, Tankönyvkiadó, 1989

[MRRTT] N. Metropolis, A. W. Rosenbluth, M. N. Rosenbluth, A. H. Teller, and E. Teller: Equation of state calculation by fast computing machines, *Journal of Chemical Physics,* 21(6)(1953), 1087–1092

[P] Prékopa A.: *Valószínűségelmélet műszaki alkalmazásokkal*, Műszaki Könyvkiadó, 1962

[R] Rényi A.: *Valószínűségszámítás*, Tankönyvkiadó, 1968

[RISz] Rónyai L., Ivanyos G., Szabó R.: *Algoritmusok*, Typotex Kiadó, 2005

[SA] R. Seidel, C. R. Aragon: Randomized Search Trees, *Algorithmica*, 16(1996), 464–497

[Z] D. Zagier: Newman's short proof of the prime number theorem, *American Mathematical Monthly*, 104(1997), 705–708