Randomizované algoritmy

Karel Richta a kol.

Přednášky byly připraveny s pomocí materiálů, které vyrobili Marko Berezovský, Petr Felkel, Josef Kolář, Michal Píše a Pavel Tvrdík

Katedra počítačů
Fakulta elektrotechnická
České vysoké učení technické v Praze

© Karel Richta a kol., 2020

Datové struktury a algoritmy, B6B36DSA
02/2020, Lekce 5

https://cw.fel.cvut.cz/wiki/courses/b6b36dsa/start

Evropský sociální fond
Praha & EU: Investujeme do vaší budoucnosti

Příklad: Výběr zaměstnance

Chceme najmout zaměstnance. Předchozí pokusy nebyly úspěšné, takže využijeme služeb personální agentury. Ta nám posílá jednoho kandidáta za den. Uděláme s ním interview a rozhodneme, zda ho najmeme, či nikoliv. Předpokládáme, že umíme rozpoznat lepšího kandidáta ihned při interview (lepšího oproti dříve vybraným). Toho pak najmeme, ale pokračujeme s výběrem do celkového počtu kandidátů n.

```
hledání_zaměstnance(n)

best = 0; // nejhorší (neexistující) zaměstnanec
for i = 1 to n
        pohovor se zaměstnancem i
        if zaměstnanec i je lepší než zaměstnanec best
                best = i
                najmutí zaměstnance best
```

Náklady na výběr

- Naším úkolem je spočítat náklady na výběr nejlepšího kandidáta, čas zde není rozhodující.
- Označme c_i cenu za provedení interview a c_h cenu za najmutí jednoho kandidáta. Celkový počet kandidátů je n, z toho m kandidátů najmeme (a později případně nahradíme jiným).
- Obecně jsou náklady: $\mathbf{c_i n + c_h m}$
- Nejhorší případ (najmeme všechny, jsou ve vzrůstajícím pořadí): $\mathbf{(c_i + c_h)n}$
- Nejlepší případ (najmeme jen toho pravého, který je hned první): $\mathbf{c_i n + c_h}$
- Ani jeden z těchto případů není obvyklý, musíme udělat nějaký odhad, co se stane v průměrném případě.

Pravděpodobnostní analýza

- **Pravděpodobnostní analýza** – využití znalostí z teorie pravděpodobnosti při analýze složitosti algoritmu.
- Abychom mohli algoritmus analyzovat, musíme udělat nějaký odhad pravděpodobné **distribuce vstupních dat** (pokud to neumíme udělat, nelze pravděpodobnostní analýzu použít).
- Pak uděláme průměr za všechny možné vstupy dle distribuce - výsledku říkáme **průměrný odhad ceny** (nebo času).
- Pro problém najmutí zaměstnance předpokládejme, že umíme porovnat libovolné dva kandidáty (existuje tedy úplné uspořádání na schopnostech kandidátů).
- Označme kandidáty i = 1..n a jejich schopnosti $rank(i)$ – vyšší „*rank*“ znamená lepší schopnosti.

Průměrné náklady

- Přicházející kandidáty můžeme modelovat posloupností:

  $\langle rank(1), \ldots, rank(n) \rangle$,

  kterou můžeme považovat za permutaci posloupnosti $\langle 1, \ldots, n \rangle$.
- Protože je pořadí náhodné, máme **n!** možností, každá z nich stejně pravděpodobná – tzv. **uniformní rozdělení** – každá možnost má stejnou pravděpodobnost.
- Pro řešení této náhodnosti můžeme využít náhodný výběr kandidátů. Od agentury dostaneme seznam kandidátů a my si každý den jednoho vybereme náhodně.
- Tomu říkáme **náhodný algoritmus** (randomized) – výběr kandidáta záleží na konkrétní distribuci vstupu, ale také na generátoru náhodných čísel, kterým kandidáta vybíráme.
- Předpokládejme, že máme k dispozici funkci *random*(*a*,*b*), která vygeneruje náhodné <u>celé</u> číslo v intervalu *a*..*b*.
- Např. *random*(*0,1*) vrátí 0 s pravděpodobností ½ a 1 s pravděpodobností ½.

Indikátory náhodných proměnných

- Pro pravděpodobnostní analýzu využíváme tzv. indikátorů náhodných proměnných.
- Předpokládejme, že máme dán prostor S možných hodnot a nějakou událost A. Pak **indikátor náhodné proměnné** $I\{A\}$ je určen:

$$I\{A\} = \begin{cases} 1 & -\,pokud\ A\ nastane \\ 0 & -\,pokud\ A\ nenastane \end{cases}$$

- Uvažme příklad házení mincí – stavový prostor je S = {orel,panna} s pravděpodobnostmi: Pr\{orel\} = Pr\{panna\} = ½ . Můžeme definovat indikátor náhodné proměnné X_{orel}, přiřazený k události, kdy padne orel – pak pro daný hod nabyde hodnoty 1, jinak 0.

$$X_{\text{orel}} = I\{\text{orel}\} = \begin{cases} 1 & -\,pokud\ padne\ orel \\ 0 & -\,pokud\ padne\ panna \end{cases}$$

- Očekávaný počet, kolikrát padne orel pro danou hru, je prostě hodnota indikátorové proměnné X_{orel}.

Očekávaná hodnota náhodné proměnné

- Nejjednodušší a nejlépe použitelný odhad distribuce hodnot náhodné proměnné je “průměr” hodnot, kterých nabývá.
- **Očekávaná hodnota** (očekávání, průměr) E[*X*] diskrétní náhodné proměnné *X* je:

$$E[X] = \sum_{x=1}^{n} x \cdot Pr\{X = x\}$$

- Tato suma je dobře definovaná, jestliže je konečná, nebo konverguje.
- Někdy se očekávaná hodnota *X* zapisuje μ_X (pokud je náhodná proměnná *X* zřejmá z kontextu, stačí μ).
- Očekávání pro součet náhodných proměnných je suma jejich očekávání (linearita očekávání), tj.:

  E[*X* + *Y*] = E[*X*] + E[*Y*]
- Linearita očekávání platí, i když *X* a *Y* nejsou nezávislé, platí i pro konečný počet proměnných, i pro konvergující řady.

Příklad

- Uvažme hru, kdy házíme dvěma mincemi. Výhra je \$3 za každého orla, ale -\$2 za každou pannu.
- Očekávaná hodnota náhodné proměnné *X* reprezentující Vaši výhru je:

$$
\begin{aligned}
E[X] &= 6 \cdot \Pr\{2 \text{ x orel}\} + \\
&\quad + 1 \cdot \Pr\{1 \text{ x orel}, 1 \text{ x panna}\} - \\
&\quad - 4 \cdot \Pr\{2 \text{ x panna}\} = \\
&= 6 \cdot \tfrac{1}{4} + 1 \cdot \tfrac{1}{2} - 4 \cdot \tfrac{1}{4} = 1
\end{aligned}
$$

Indikátory a pravděpodobnost

Očekávaný počet, kolikrát padne orel:

$$E[X_{orel}] = E[I\{X_{orel}\}] = 1 \times Pr\{orel\} + 0 \times Pr\{panna\} =$$
$$= 1 \times ½ + 0 \times ½ = ½$$

Je to tedy ½ . Lze ukázat, že je to vlastně pravděpodobnost události.

Lemma: Buď prostor S možných hodnot a mějme událost A z prostoru S. Nechť $X_A = I\{A\}$. Pak $E[A] = Pr\{A\}$.

Důkaz: $E[A] = E[I\{A\}] = 1 \times Pr\{A\} + 0 \times Pr\{\neg A\} = Pr\{A\}$

kde $\neg A$ je komplement A, tj. $\neg A = S - A$. q.e.d.

Buď $X_i = I\{$výsledek i-tého hodu$\}$. Pak:

$$X = \sum_{i=1}^{n} X_i$$

A očekávaný počet, kdy padne orel je (linearita očekávání):

$$E[X] = E[\sum_{i=1}^{n} X_i] = \sum_{i=1}^{n} E[X_i] = \sum_{i=1}^{n} \frac{1}{2} = n/2$$

Analýza průměrných nákladů

- Zajímá nás počet najmutí X:

$$E[X] = \sum_{x=1}^{n} x \cdot Pr\{X = x\}$$

Použijeme indikátory: $X_i = 1$ pokud je kandidát i najat, 0 jinak. Potom $X = X_1 + \ldots + X_n$.
Pokud jsou všechna pořadí stejné pravděpodobná (linearita očekávání), pak $E[X_i] = 1/i$.

Součet harmonické řady

$$E[X] = E\left[\sum_{i=1}^{n} X_i\right] = \sum_{i=1}^{n} E[X_i] = \sum_{i=1}^{n} 1/i = \ln n + O(1)$$

- Přestože tedy máme n kandidátů, v průměru najmeme v očekávaném případě jen: $\ln n$ kandidátů. Průměrné náklady tedy činí: $O(\mathbf{c_h} \times \mathbf{ln}\ \boldsymbol{n})$ oproti nejhoršímu případu: $O(\mathbf{c_h} \times \boldsymbol{n})$, pokud použijeme náhodný výběr.

Pravděpodobnostní analýza a randomizované algoritmy

- Proberme rozdíl mezi pravděpodobnostní analýzou a randomizovanými algoritmy.
- Ukázali jsme, že pokud předpokládáme, že kandidáti přicházejí v náhodném pořadí, je očekávaný počet najmutí ln n.
- Algoritmus je deterministický – pro jakýkoliv konkrétní vstup je počet najmutí stejný. Počet najmutí se pro různé vstupy liší a závisí na schopnostech kandidátů.
- Protože počet najmutí závisí pouze na schopnostech kandidátů, můžeme vstup reprezentovat posloupností jejich schopností:

  <rank(1), rank(2), … , rank(n)>.
- Uvážíme-li posloupnost schopností: A1 = <1, 2, 3, 4, 5, 6, 7, 8, 9, 10>, budeme najímat 10x (každý nový kandidát je lepší, než všichni předchozí).
- Uvážíme-li posloupnost schopností: A2 = <10, 9, 8, 7, 6, 5, 4, 3, 2, 1>, budeme najímat jen jednou (první je nejlepší).

Pravděpodobnostní analýza a randomizované algoritmy (pokr.)

- Uvážíme-li posloupnost schopností: A3 = <5, 2, 1, 8, 4, 7, 10, 9, 3, 6>, budeme najímat 3x (po interview s kandidáty se schopnostmi 5, 8 a 10).
- Cena za najímání závisí na počtu najmutí, vidíme, že existují drahé vstupy (A1), ale i levné (A2) a středně obtížné (A3).
- Uvážíme-li na druhé straně randomizovaný algoritmus, ten nejprve permutuje vstup a pak vybírá kandidáty – není závislý na vstupní distribuci. Např. pro vstup A3 nemůžeme počet najmutí stanovit – to je dáno randomizovaným algoritmem – může to být 1 až 10.
- Pro randomizovaný algoritmus nejde vybrat „nejhorší“ vstup, závisí pouze na generátoru náhodných čísel. Špatné je, pokud vždy generuje „nešťastnou“ permutaci (např. A1).
- Při randomizaci řešení najímání zaměstnance, stačí přidat náhodnou permutaci vstupu.

Randomizovaný výběr

- Co když je personální agentura nespolehlivá?
- Pořadí kandidátů mohu náhodně popřeházet, tím si zajistím linearitu očekávání.
- Generování náhodné permutace:

permutuj_tříděním(A)

vytvoření nového pole P (se stejnou délkou, jako pole A)
vygenerování náhodných hodnot z rozmezí 1. . . A.length^3 do pole P
setřídění pole A s použitím pole P jako klíčů

- Volíme interval 1…n^3 aby bylo pravděpodobné, že všechny priority v P jsou unikátní.
- Lze ukázat, že permutace tříděním generuje uniformní rozdělení.
- Nejtěžší krok v permutaci je setřídění pole A. Dolní mez pro třídění porovnáním je $O(n \ln n)$ a stejná je i složitost permutace.

Randomizované hledání zaměstnance

```
randomizované_hledání_zaměstnance(n)

ulož posloupnost kandidátů délky n do pole A
permutuj_tříděním(A) // náhodně permutuj seznam kandidátů
hledání_zaměstnance(n) // stejné jako dříve

Tj.:
best = 0; // nejhorší (neexistující) zaměstnanec
for i = 1 to n
        pohovor se zaměstnancem i
        if zaměstnanec i je lepší než zaměstnanec best
                best = i
                najmutí zaměstnance best
```

- Očekávaná cena randomizovaného hledání je $\boldsymbol{O}(c_h \ln n)$ – viz složitost hledání zaměstnance.

Rovnoměrnost očekávání

$$Pr\{E_1 \cap E_2 \cap \cdots \cap E_n\} = Pr\{E_1\} \cdot Pr\{E_2|E_1\} \cdot Pr\{E_3|E_1 \cap E_2\} \ldots$$

- Rychlejší algoritmus:

```
permutuj_prohazováním(A)

n = A.length
for i = 1 to n
        prohoď A[i] s A[random(i,n)]
```

- Rovnoměrnost očekávání:

Pomocí invariantu: pro každou permutaci prvků vstupního pole platí, že na začátku $i$té iterace cyklu obsahuje pole A na indexech 1 až $i-1$ prvních $i-1$ prvků této permutace s pravděpodobností $(n-i+1)!/n!$.

Příklad: Maximální prvek

Uvažme problém hledání maxima v poli čísel a[1..n]:

m:=a[1]; for i:=2 to n do if a[i] > m then m:=a[i]

Kolikrát se provede přiřazení m:=a[i]?

- V nejhorším případě se vykoná v každé iteraci, tj. n-1-krát.
- V nejlepším případě není vykonáno ani jednou (maximum je první).
- Co v průměrném případě? Začneme definicí pravděpodobnostního prostoru. Předpokládáme, že pole obsahuje n různých elementů a že jejich libovolné pořadí je stejně pravděpodobné. Jinými slovy – náš prostor obsahuje n! permutací. Každá permutace je stejně pravděpodobná a její pravděpodobnost je 1/n!.
- Protože konkrétní hodnoty v poli nejsou důležité, můžeme předpokládat, že pole obsahuje čísla od 1 do n v nějakém pořadí.
- Co nás nyní bude zajímat, je průměrný počet tzv. maxim **zleva-do-prava** (left-to-right).

Příklad: Maximální prvek (pokr.)

- **Maximum zleva-do-prava** v posloupnosti je prvek, který je větší než všechny předchozí prvky. Např.: posloupnost (1,2,4,3) má 3 taková maxima, (3,1,2,4) má 2 maxima zleva-do-prava.
- Pro permutaci π celých čísel od 1 do n, označme $M_n(\pi)$ počet maxim zleva-do-prava.
- Jaké bude očekávání $E[M_n]$? Lze to popsat dvěma způsoby:
  - Pro malá n, můžeme $E[M_n]$ přímo spočítat. Pro n = 1 existuje jen jedna permutace (1) a má jedno maximum, tj. $E[M_1] = 1$. Pro n = 2, existují dvě permutace (1,2) a (2,1). První má 2 maxima a druhá má 1 maximum, $E[M_2] = 1.5$.
  - Pro větší n postupujeme následovně – označme M_n sumu indikátorových proměnných I_1 až I_n, tj. $M_n = I_1 + \ldots + I_n$, kde I_k je rovno 1 pro permutaci π, kde k-tý element π je maximum zleva-do-prava. Např.: $I_3((3,1,2,4)) = 0$ and $I_4((3,1,2,4)) = 1$.

Příklad: Maximální prvek (pokr.)

- Vzhledem k linearitě E a protože I_k jsou indikátorové proměnné.

$$E[M_n] = E[I_1 + I_2 + \ldots + I_n] = E[I_1] + E[I_2] + \ldots + E[I_n] =$$
$$= \Pr\{I_1 = 1\} + \Pr\{I_2 = 1\} + \ldots + \Pr\{I_n = 1\}$$

- Zbývá určit pravděpodobnost, že $I_k = 1$. Platí, že k-tý element náhodné permutace je maximum zleva-do-prava tehdy a právě tehdy, když je k-tý element největší ze všech předcházejících prvních k elementů.
- V náhodné permutaci je pro každou pozici stejně pravděpodobné, že obsahuje maximum, takže hledaná pravděpodobnost je $\Pr\{I_k = 1\} = 1/k$.

$$E[M_n] = \sum_{1 \le k \le n} \Pr\{I_k = 1\} = \sum_{1 \le k \le n} 1/k$$

- Pak např.: $E[M_4]=1+1/2+1/3+1/4=(12+6+4+3)/12=25/12$.
- Součet $\sum_{1 \le k \le n} 1/k$ se počítá jako „n-tá harmonická“ a značí se H_n. Je známo, že platí: $\ln n \le H_n \le 1 + \ln n$, tj.: $H_n \approx \ln n$.
- Proto můžeme usoudit, že průměrný počet maxim zleva-do-prava je mnohem menší (ln n), než v nejhorším případě (n).

Příklad: Výběr ceny

- Předpokládejte, že máte šanci účastnit se TV show.
- Je tam 100 skříněk, které budete otevírat v pořadí, které si stanovíte Vy, moderátor Vám může napovídat.
- Skříňka s číslem *i* obsahuje *mi* peněz. Nevíte, kolik to je, ale po otevření to zjistíte.
- Žádné dvě skříňky neobsahují stejné množství peněz.
- Pravidla hry jsou jednoduchá:
  - Na začátku hry dostanete 10 žetonů.
  - Když otevřete skříňku, která obsahuje více peněz, než je obsah dosud otevřených skříněk, musíte vrátit 1 žeton.
  - Když musíte vrátit žeton a nemáte ho – prohrál jste.
  - Když se Vám podaří otevřít všechny skříňky, vyhrál jste a můžete si ponechat všechny peníze.

Příklad: Výběr ceny (pokr.)

- Analyzujme nejprve případ, kdy vždy dáte na radu moderátora – jeho nápověda bude řídit hru.
- Nejhorší případ bude, když Vás nechá otvírat schránky v obráceném pořadí dle obsahu. Kdykoliv otevřete skřínku, musíte vrátit žeton a po otevření 11-té skříňky prohráváte. To by se ale nelíbilo divákům a proto to asi takto nebude.
- Nejlepší případ by byl, kdyby Vám hned prozradil nejdražší skříňku. To by se líbilo hráči, ale nebyl by prostor pro reklamy během hry, a proto to asi takto také nebude.
- Problém lze přirovnat k hledáním maxima zleva-do-prava. Kdykoliv narazíme na maximum zleva-do-prava, musíme vrátit žeton. Jak jsme ukázali dříve, je očekávaný počet maxim zleva-do-prava v náhodné posloupnosti „n-tá harmonická“ H_n. Pro n=100, $H_{100} \leq 1+\ln 100 = 1+4.61$ a $H_{100} < 6$. Pokud Vám moderátor ukazuje skříňky v náhodném pořadí, měli byste v průměru vrátit 6 žetonů a vyhrát.

Příklad: Výběr ceny (pokr.)

- Proč by Vám ale moderátor ukazoval skříňky v náhodném pořadí, když by to v průměrném případě vedlo k Vaší výhře. On přece po Vaší výhře netouží – přesněji netouží po mnoha výhercích, občas ale musí někoho nechat vyhrát – mohli byste to být právě Vy.
- Řešení je vzít věci do svých rukou, ignorovat nápovědu moderátora a otevírat skříňky v náhodném pořadí. Vyberte si náhodně jednu skříňku a otevřete ji. Pak si opět náhodně vyberte skříňku ze zbývajících a tak dále.
- Jak náhodně vybrat skříňku? Pokud zbývá k skříněk, hoďte si kostkou s k stranami a dostanete číslo v intervalu 1..k. Tímto způsobem generujete náhodnou permutaci a analýza bude stejná, tj. v průměru budete muset vrátit méně než 6 žetonů, ale protože jich máte 10, vyhráváte.
- To je randomizovaný algoritmus. Můžete se obávat, že ačkoliv to matematicky vychází, ve skutečnosti jste vydáni na milost moderátorovi. To ale platí pouze tehdy, když necháte na něm, zda Vám radí skříňky skutečně v náhodném pořadí, nebo ne. V případě randomizovaného algoritmu na tom nezáleží. Jistotu výhry nemáte, ale lépe to nejde.

Příklad: Výběr ceny (pokr.)

- Zhodnotíme-li celkovou situaci, máme 10 žetonů. Očekávaný počet vrácených žetonů je méně než 6. Jak si můžete být jisti, že odejdete s výhrou?
- Můžeme to postavit tak, že pravděpodobnost M_n je větší než 11, pokud otevřete více než 11 skříněk, nebo více než maximum zleva-do-prava.
- Markovova nerovnost nám umožňuje omezit tuto pravděpodobnost. Říká, že pro nezápornou náhodnou proměnnou X a libovolnou konstantu $c \geq 1$, platí, že:

$$\Pr\{X \geq c \cdot E[X]\} \leq 1/c.$$

- Pokud tuto nerovnost aplikujeme pro $X = M_n$ a $c = 11/6$, dostáváme:

$$\Pr\{M_n \geq 11\} \leq \Pr\{M_n \geq \frac{11}{6} E[M_n]\} \leq \frac{6}{11}.$$

- Pravděpodobnost výhry je proto více než 5/11.

Formální model randomizace

- Formálně lze situaci řešit tak, že náš RAM model obohatíme o novou instrukci: x:=randInt(C), která celočíselné proměnné x přiřadí náhodné celé číslo z intervalu mezi 0 a C−1.
- Cena za náhodný výběr je jedna časová jednotka, jako u všech ostatních instrukcí.
- Algoritmy (programy), které ji nepoužijí nazýváme deterministické.
- Čas běhu randomizovaného algoritmu obecně závisí na generovaných hodnotách. Není to tedy číslo, ale náhodná proměnná závislá na náhodných výběrech.
- Závislost můžeme eliminovat tím, že náš stroj vybavíme časovačem. Na začátku výpočtu nastavíme časovač na hodnotu T(n), která může záviset na velikosti problému (n).
- Stroj se po plynutí času T(n) zastaví – můžeme garantovat, že čas je omezen na T(n). Nemusí ale v této době spočítat výsledek.

Formální model randomizace (pokr.)

- Také výstup randomizovaných algoritmů může záviset na generovaných náhodných hodnotách.
- Může se zdát, že to způsobí nepoužitelnost algoritmu – odpověď na stejnou instanci problému může být jednou „ANO“ a podruhé „NE“. Pokud mají oba případy stejnou pravděpodobnost, odpověď není k ničemu. Pokud má ale správná odpověď mnohem větší pravděpodobnost, má výstup určitou hodnotu.

Příklad:

- Alice a Bob jsou ve spojení pomocí pomalé telefonní linky. Alice má celé číslo x_A a Bob má celé číslo x_B, obě čísla mají n cifer. Chtějí si ověřit, že mají stejné číslo. Protože je komunikace pomalá, snaží se minimalizovat množství předávané informace. Lokální výpočty de-facto nic nestojí.
- Jedno možné řešení je, že Alice pošle své číslo Bobovi, ten ho porovná se svým a vrátí výsledek (0/1). Toto řešení vyžaduje přenos n+1 cifer.

Formální model randomizace (pokr.)

- Alternativně: Alice může posílat cifru za cifrou, Bob přenos zastaví v okamžiku, kdy už zná výsledek – buď shodu, nebo neshodu. V nejhorším případě se ale musí opět přenést n+1 cifer (v nejlepším 2).
- Pokusíme se to vylepšit randomizací. Po přenosu O(ln n) cifer, může být rovnost či nerovnost rozhodnuta s vysokou pravděpodobností.
- Alice a Bob se budou řídit následujícím protokolem. Každý si připraví uspořádaný seznam prvočísel p. Seznam obsahuje nejmenších L prvočísel, které mají k a více cifer a začínají cifrou 1. Každé takové prvočíslo má hodnotu nejméně 2^k. Důležité je, že je garantováno, že Alice i Bob generují stejný seznam.
- Pak Alice vybere náhodně index i, $1 \leq i \leq L$, a pošle i a (x_A mod p_i) Bobovi. Bob spočítá x_B mod pi. Pokud x_A mod $p_i \neq x_B$ mod p_i, deklaruje, že čísla jsou odlišná. Jinak jsou shodná.
- Pokud $x_A \neq x_B$, ale x_A mod $p_i = x_B$ mod p_i, Bob chybně označí čísla jako stejná. Jaká je pravděpodobnost chyby?

Formální model randomizace (pokr.)

- Chyba nastane, pokud $x_A \neq x_B$, ale $x_A \bmod p_i = x_B \bmod p_i$. Tj. $x_A \equiv x_B (\bmod p_i)$.
- Poslední podmínka je ekvivalentní podmínce, že p_i dělí rozdíl $D = x_A - x_B$. Rozdíl je nejvýše 2^n v absolutní hodnotě. Protože každé prvočíslo p_i má podle zadání hodnotu nejméně 2^k, seznam obsahuje nejvýše n/k prvočísel, která dělí rozdíl a proto pravděpodobnost chyby je nejvýše (n/k)/L. Tuto pravděpodobnost můžeme libovolně snížit volbou dostatečně velkého L.
- Pokud např. chceme, aby pravděpodobnost byla menší než 0.000001 = 10^{-6}, stačí zvolit $L = 10^6(n/k)$.
- Jak zvolíme vhodné k? Mezi čísly s k bity je přibližně $2^k/k$ prvočísel. Proto, pokud $2^k/k \geq 10^6(n/k)$, bude seznam obsahovat jen celá čísla délky k.
- Podmínka aby $2^k \geq 10^6 n$ je stejná jako $k \geq \ln n + 6 \ln 10$.
- Pro tuto volbu k, protokol přenáší: $\ln L + k = \ln n + 12 \ln 10$ bitů. To je exponenciálně lepší, než naivní řešení.

Formální model randomizace (pokr.)

- Co uděláme, pokud budeme chtít, aby pravděpodobnost chyby byla menší než 10^{-12}?
- Můžeme přepočítat předchozí kalkulaci pro $L=10^{12}n$.
- Alternativně, můžeme použít protokol dvakrát a deklarovat jiná čísla (pokud se dají najít). Tento dvou-stavový protokol bude chybový, pouze pokud budou chybové obě fáze – pravděpodobnost chyby je nejvýše součin pravděpodobností: $10^{-6} \cdot 10^{-6} = 10^{-12}$.

Las Vegas versus Monte Carlo

- Existují dvě varianty randomizovaných algoritmů: Las Vegas a Monte Carlo.
- Algoritmy typu Las Vegas vždy spočtou správný výsledek, ale čas běhu je náhodná proměnná. Řešení pro TV show bylo typu Las Vegas – v průměrném očekávaném případě vždy najdete skříňku obsahující maximum, počet maxim zleva-do-prava je náhodná proměnná.
- Algoritmus typu Monte Carlo pracuje vždy ve stejném čase, ale může dávat nekorektní výsledky. Pravděpodobnost nekorektní odpovědi je nejvýše 1/4. Algoritmus pro porovnání dvou čísel po telefonní lince je typu Monte Carlo. Ukázali jsme ale, že pravděpodobnost chyby lze libovolně snížit.

Las Vegas versus Monte Carlo (pokr.)

- Předpokládejme, že máme algoritmus typu Las Vegas, kde očekávaný čas je $t(n)$. Spusťme jej pro $4 \cdot t(n)$ kroků. Jestliže vrátí odpověď v očekávaném čase, bude to výsledná odpověď. Jinak vrátí libovolnou odpověď. Výsledkem je algoritmus typu Monte Carlo.
- Předpokládejme, že máme algoritmus typu Monte Carlo, kde očekávaný čas je $m(n)$, který dává korektní odpověď s pravděpodobností p. Dále máme deterministický algoritmus, který verifikuje v čase $v(n)$, zda algoritmus Monte Carlo dává korektní odpověď. Kombinací těchto dvou algoritmů lze získat algoritmus typu Las Vegas s očekávaným časem exekuce: $(m(n)+v(n))/(1-p)$.

Příklad na pravděpodobnost - řešení

```
boolean f(int[] a, int[] b) {
  int r1 = random(0, a.length - 1); // funkce random vrati nahodne cislo
                                    // mezi 0 a a.length - 1 vcetne
  int r2 = random(0, b.length - 1);
  return a[r1] == b[r2];
}
```

Řešení:

- Označme si a.length = n. Pravděpodobnost, že na indexu r1 je hodnota k, je 1/n. Pravděpodobnost, že na indexu r2 je hodnota k, je 1/n. Oba jevy jsou na sobě nezávislé, takže pravděpodobnost, že a[r1] == b[r2], je:

$$\sum_{k=1}^{n} \frac{1}{n} \cdot \frac{1}{n} = \frac{1}{n}$$

Příklad: Narozeninový paradox

- Kolik lidí musí být v místnosti, aby byla šance 1:1 (50%), že tam budou nejméně dvě osoby, které se narodily ve stejný den v roce.
- Odpověď je překvapivě nízká - paradox spočívá v tom, že je to mnohem méně, než počet dnů v roce, nebo dokonce než polovina roku.
- Přidělme osobám v místnosti čísla od 1 do k, kde k je počet osob v místnosti.
- Zanedbáme přestupné roky a budeme předpokládat, že rok má n=365 dní.
- Nechť b_i je den narození osoby $1 \leq i \leq k$ v roce. Předpokládáme uniformní rozdělení dat narození v roce, tj. $Pr\{b_i = r\} = 1/n$, kde i=1,..,k, r=1,..n.
- Pravděpodobnost, že se dvě osoby narodily ve stejný den (předpokládáme nezávislost dnů narození) je:

$$Pr\{b_i = r \text{ a } b_j = r\} = Pr\{b_i = r\} \cdot Pr\{b_j = r\} = 1/n^2.$$

- Pravděpodobnost, že se narodily ve stejný den:

$$Pr\{b_i = b_j\} = \sum_{r=1}^{n} Pr\{bi = r \text{ and } bj = r\} = \sum_{r=1}^{n} 1/n^2 = 1/n.$$

Příklad: Narozeninový paradox (pokr.)

- Intuitivně: pokud zvolíme b_i , pravděpodobnost, že vybereme b_j se stejným dnem narození je 1/n.
- Proto pravděpodobnost, že se osoby i a j narodily ve stejný den je stejná, jako pravděpodobnost, že narození jedné z nich padne na stejný den. Povšimněme si, že předpoklad závisí na nezávislosti data narození.
- Můžeme analyzovat pravděpodobnost, že nejméně 2 osoby v naší místnosti mají stejný den narození, pomocí komplementárního tvrzení, že žádné dvě stejný den nemají.
- Pravděpodobnost, že nejméně dvě osoby mají stejný den narození je doplněk pro případ, kdy jsou všechna data odlišná, tj.

  1 – pravděpodobnost, že jsou stejná.
- Stav, kdy má k osob různé dny narození, kde A_i je případ, že osoba i má den narození odlišný od osoby j pro všechna j < i. Můžeme psát $B_k = A_k \cap B_{k-1}$, a podle Bayesova teorému:

Příklad: Narozeninový paradox (pokr.)

- $Pr\{B_k\} = Pr\{B_{k-1}\}\, Pr\{A_k \mid B_{k-1}\}$,
  kde uvažujeme: $Pr\{B_1\} = Pr\{A_1\} = 1$ jako počáteční podmínku.
- Jinými slovy, pokud uvažujeme, že $b_1, b_2, \dots, b_k$ jsou různá data narození, je to stejné, jako když $b_1, b_2, \dots, b_{k-1}$ jsou různá data narození krát pravděpodobnost, že $b_k \neq b_i$, pro $i = 1, \dots, k-1$.
- Pokud jsou $b_1, b_2, \dots, b_{k-1}$ různá data, pak podmíněná pravděpodobnost, že $b_k \neq b_i$, pro $i = 1, \dots, k-1$ je $Pr\{A_k \mid B_{k-1}\} = (n-k+1)/n$, protože z n dnů se $n-(k-1)/n$ nebere v úvahu.
- Pokud iterativně aplikujeme výše uvedenou úvahu:
- $Pr\{B_k\} = Pr\{B_{k-1}\}\, Pr\{A_k \mid B_{k-1}\} =$

Příklad: Narozeninový paradox (pokr.)

$= Pr\{B_{k-2}\}\, Pr\{A_{k-1} \mid B_{k-2}\}\, Pr\{A_k \mid B_{k-1}\} =$

. . .

$= Pr\{B_1\}\, Pr\{A_2 \mid B_1\}\, Pr\{A_3 \mid B_2\} \ldots\ Pr\{A_k \mid B_{k-1}\} =$

$= 1 \cdot (\frac{n-1}{n}) \cdot (\frac{n-2}{n}) \ldots (\frac{n-k+1}{n}) =$

$= 1 \cdot (1 - \frac{1}{n}) \cdot (1 - \frac{2}{n}) \ldots (1 - \frac{k-1}{n}) \leq$ (protože $e^x \geq 1+x$)

$\leq e^{-1/n}\, e^{-2/n} \ldots e^{-(k-1)/n} =$

$= e^{-\sum_{i=1}^{k-1} k/n} =$

$= e^{-k(k-1)/2n} \leq$

$\leq 1/2$

kde: $-k(k-1)/2n \leq \ln(1/2)$

Příklad: Narozeninový paradox (pokr.)

- Pravděpodobnost, že budou všechna data narození různá je nejvýše ½, pokud k(k-1) ≥ 2n·ln(2), pokud:
  k ≥ (1+$\sqrt{1+(8ln2)n}$)/n
- Pro n = 365 vychází k ≥ 23.
- Pokud je tedy v místnosti 23 a více lidí, je pravděpodobnost ½ (50%), že alespoň dva lidé budou mít stejné datum narození.

The End