nedeljainformatike v7.0

Probabilistički (ne baš uvek ispravni) algoritmi

Bogdana Kolić

Matematička gimnazija

13. 04. 2022.

Agenda

1. Sve što treba znati iz verovatnoće
2. Randomized algoritmi
3. Bonus

Osnovni pojmovi

- **Elementarni događaji** – pojedinačni, međusobno isključivi ishodi eksperimenta
- **Prostor događaja** – skup svih elementarnih događaja (Ω)
  - Prostor događaja je **siguran događaj** – uvek se realizuje
- **Slučajni događaj** – podskup prostora događaja
  - Slučajni događaj S se realizuje samo ako se desi neki ishod iz skupa S
- Ako je A slučajni događaj, definišemo njegov **komplementarni događaj** $\bar{A}$ tako da je $A \cup \bar{A} = \Omega$ i $A \cap \bar{A} = \emptyset$.

Primer 1: Bacanje novčića

- Bacamo novčić 2 puta: Ω = {(P, P), (P, G), (G, P), (G, G)}
- Slučajni događaji:
  - Prvi put se pala glava: S = {(G, P), (G, G)}
  - Pismo se palo bar jedanput: R = {(P, P), (P, G), (G, P)}
- Neka je ishod (P, P) – realizovao se samo događaj R
- Neka je ishod (G, P) – realizovali su se i događaj S i događaj R

Verovatnoća

- Verovatnoću definišemo kao preslikavanje partitivnog skupa prostora događaja u interval [0, 1] – svakom slučajnom događaju dodeljujemo „verovatnoću“ iz intervala [0, 1]
- Označava se slovom p
- Važi:
  - $p(\Omega) = 1$
  - $p(\emptyset) = 0$
  - Neka je $S = E_1 \cup E_2 \cup \ldots \cup E_n$, gde su događaji E_1, E_2, ..., E_n međusobno disjunktni u parovima. Onda je $p(S) = p(E_1) + p(E_2) + \ldots + p(E_n)$
  - $p(A) = 1 - p(\bar{A})$

Uniformna raspodela

- Neka je $\Omega = \{w_1, w_2, \ldots, w_n\}$, gde su $w_1, w_2, \ldots, w_n$ elementarni događaji. Ako važi $p(\{w_1\}) = p(\{w_2\}) = \ldots = p(\{w_n\}) = p$, kažemo da imamo uniformnu raspodelu verovatnoće.
- Elementarni događaji su svi različiti, a njihova unija čini prostor događaja. To znači da je $p(\Omega) = p(\{w_1\}) + p(\{w_2\}) + \ldots + p(\{w_n\})$, odakle dobijamo

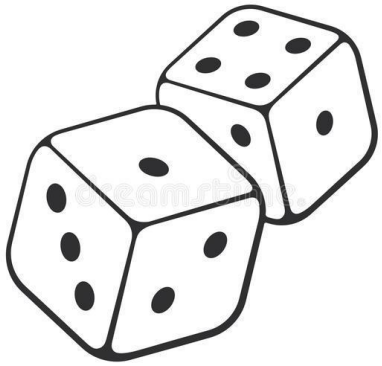
$$1 = n \times p, \text{ tj. } p = \frac{1}{n}$$

Verovatnoća događaja (uniformna raspodela)

- *P(slučajni događaj)* = $\frac{broj\ povoljnih\ ishoda}{broj\ svih\ ishoda}$
- P(S) = $\frac{\#S}{\#\Omega}$
- U primeru 1: $p(S) = \frac{2}{4} = \frac{1}{2}$ $p(R) = \frac{3}{4}$
- Primer 2: Bacamo istovremeno dva novčića, kolika je verovatnoća da će se pasti i glava i pismo?
  - Pala su se dva pisma, dve glave ili pismo i glava – dakle, rešenje je $\frac{1}{3}$
  - Ne!!!
  - Ω = {(P, P), (P, G), (G, P), (G, G)}
  - Naš slučajni događaj se realizuje u dva slučaja – kada se pismo palo na prvom, a glava na drugom novčiću, ili glava na prvom, a pismo na drugom novčiću
  - Rešenje je $p = \frac{1}{2}$

Verovatnoća preseka događaja

- Primer 3: Bacamo klasičnu kockicu za igru 3 puta. Koliko je verovatnoća će se svaki put pasti broj manji od 5?
- Rešenje: Naš slučajni događaj događaj se realizuje ako se u svakom bacanju padne broj iz skupa {1, 2, 3, 4}, a to se dešava u 4^3 slučajeva. Verovatnoća je $p = \frac{4^3}{6^3}$. Primetimo još nešto – verovatnoća da se u jednom bacanju padne broj manji od 5 je $\frac{4}{6}$, a $p = \frac{4}{6} \times \frac{4}{6} \times \frac{4}{6}$. Pritom, nijedno bacanje ne utiče na ishod ostalih.

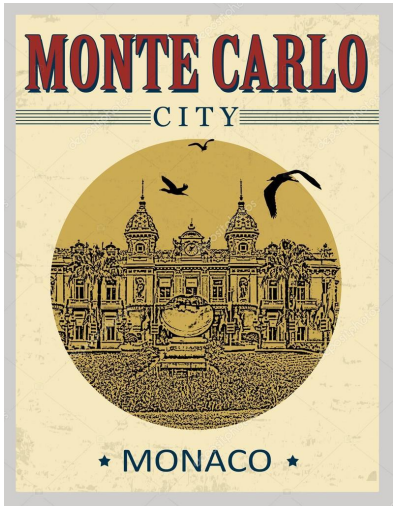


Čemu nam ovo služi u računarstvu?

- Cilj : optimizacija kompleksnih algoritama
- Deo algoritma zasniva se na **nasumičnim** izborima – **Randomized algoritam**
  - Interesuje nas verovatnoća da program daje tačno rešenje, da ima optimalno vreme izvršavanja, ...
- Razlikujemo:

Vreme izvršavanja nije uvek dobro

Rešenje je uvek ispravno



Vreme izvršavanja je uvek optimalno

Rešenje je tačno u velikom broju slučajeva

Agenda

1. Sve što treba znati iz verovatnoće
2. Randomized algoritmi
3. Bonus

Las Vegas optimizacija Quick sort-a

- Kako radi Quick sort algoritam?
  1. Niz je sortiran kada je prazan ili sadrži samo jedan element – stajemo
  2. Izaberemo jedan element iz niza – delitelj (pivot)
  3. Sve elemente manje od pivota prebacimo na jedan, a sve veće na drugi kraj niza
  4. Primenimo isti postupak na dva nova niza dobijena deljenjem početnog niza uspomoć pivota

Koja je složenost?

- Ako smo imali sreće i delili svaki niz približno na pola – O(nlogn)
- Ako smo uvek birali element koji je na početku/kraju sortiranog niza – $O(n^2)$
- U proseku, složenost će biti O(nlogn), ali u najgorem slučaju (na primer kada je lista već sortirana), složenost je uvek $O(n^2)$

Randomized Quick sort

- Postoji način da poboljšamo kompleksnost Quick sort-a, a da sačuvamo korektnost algoritma. Umesto da uvek biramo pivot sa kraja niza, biraćemo ga nasumično.
- I dalje može da se desi da je pivot izabran baš tako da treba da stoji na početku/ kraju sortiranog niza, ali verovatnoća da će se to desiti je mnogo manja
- Ako je niz jednom sortiran u složenosti $O(n^2)$ to ne znači da će svaki put biti sortiran u toj složenosti
  - Više ne postoje konkretni primeri niza koji se uvek sortiraju u „najgoroj složenosti“

Monte Karlo algoritmi

- Želimo da ubrzamo program i spremni smo da žrtvujemo 100% korektnost zarad te optimizacije
- Pri jednoj iteraciji algoritma, verovatnoća da je rešenje tačno je $p > 0$
- Kako možemo da umanjimo šansu da pravimo grešku:
  - Prisetimo se da važi i $p < 1$. Verovatnoća da pravimo grešku je $q = 1 - p$. Ako rezultati pojedinačnih izvršenja algoritma ne zavise jedna od drugih i ako izvršimo algoritam k puta, verovatnoća da pravimo grešku postaje $q^k << 1$.

Traženje elementa u nizu

- Zadatak: Dat je (nesortiran) niz od n elemenata. Znamo da se element x pojavljuje u njemu bar m puta. Na kojoj se poziciji nalazi jedan od elemenata x?
- Ako bismo redom pretraživali niz, složenost bi bila O(n)
  - Za veliko n, vreme izvršavanja nam ne odgovara
  - ... ali ako je i m dovoljno veliko, možemo vrlo lako da ga smanjimo uspomoć probabilističkog algoritma!

Traženje elementa u nizu

- Ideja: k puta nasumično generišemo indeks i, i proverimo da li se element x nalazi na poziciji i.
- Složenost će biti O(k)
- Verovatnoća da ne pronađemo odgovarajući indeks je $(\frac{n-m}{n})^k$

Traženje elementa u nizu

- Primer : Zamislimo da je dat promešani špil karata i da želimo da pronadjemo poziciju na kojoj se nalazi neko srce.
- n = 52 – dužina niza, sve karte u špilu
- m = 13 – imamo 13 srca
- Neka je k = 15 – nasumično biramo kartu 15 puta i proveravamo znak
- Pronalazimo srce u 98.9977% slučajeva
- Čak i da smo pomešali 104987078185 špilova i proverili samo 15 nasumično izabranih karata iz tog niza, verovatnoća da pronađemo srce bi ostala 0.9899 – srce čini četvrtinu karata u nizu!

Proizvod dve matrice

- Zadatak: Date su tri matrice A, B i C, dimenzija n×n. Proveriti da li je A×B = C
- Naivno rešenje bi bilo da pomnožimo matrice i uradimo proveru – složenost bi bila $O(n^3)$
  - Čak i kada bismo koristili Štrasenov algoritam za množenje matrica, složenost bi i dalje bila loša – $O(n^{2.8074})$

- $$\begin{bmatrix} 1 & 7 & 2 \\ 8 & 0 & 3 \\ 0 & 5 & 0 \end{bmatrix} \times \begin{bmatrix} 3 & 0 \\ 1 & 4 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 1\times 3 + 7\times 1 + 2\times 1 & 1\times 0 + 7\times 4 + 2\times 0 \\ 8\times 3 + 0\times 1 + 3\times 1 & 8\times 0 + 0\times 4 + 3\times 0 \\ 0\times 3 + 5\times 1 + 0\times 1 & 0\times 0 + 5\times 4 + 0\times 0 \end{bmatrix}$$

$$= \begin{bmatrix} 12 & 28 \\ 27 & 0 \\ 5 & 20 \end{bmatrix}$$

- Probabilističko rešenje: Frejvaldsov algoritam.

Frejvaldsov algoritam

- Ideja je da nasumično kreiramo pomoćni vektor $\underset{r}{\rightarrow}$ (matricu dimenzija n×1). Ako je A×B = C, onda je i $A \times \left(B \times \underset{r}{\rightarrow}\right) - C \times \underset{r}{\rightarrow} = \underset{0}{\rightarrow}$.

- $$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \times \begin{bmatrix} b_{11} & \cdots & b_{1n} \\ \vdots & \ddots & \vdots \\ b_{n1} & \cdots & b_{nn} \end{bmatrix} = \begin{bmatrix} c_{11} & \cdots & c_{1n} \\ \vdots & \ddots & \vdots \\ c_{n1} & \cdots & c_{nn} \end{bmatrix} \qquad O(n^3)$$

- $$\begin{bmatrix} a_{11} & \cdots & a_{1n} \\ \vdots & \ddots & \vdots \\ a_{n1} & \cdots & a_{nn} \end{bmatrix} \times \begin{bmatrix} r_1 \\ \vdots \\ r_n \end{bmatrix} = \begin{bmatrix} c_1 \\ \vdots \\ c_n \end{bmatrix} \qquad O(\mathrm{n}^2)$$

Proizvod dve matrice

- Pravimo grešku ako je rezultat koji dobijemo 0, a $A \times B \neq C$.
- Dokazano je da je verovatnoća da se to desi najviše ½ .
  - Dakle, ako ponovimo ovaj postupak k puta, verovatnoća da dobijamo loše rešenje će biti $(½)^k$, a složenost $O(kn^2)$.
- Vrednost k zavisi od toga koliku tačnost želimo da dostignemo:

k	tačnost
1	50%
2	75%
3	87,5%
4	93,75%
5	96,875%
6	98,4375%
7	99,21875%
8	99,609375%
9	99,8046875%

Da li je broj prost?

- Klasičan algoritam – proverimo da li je naš broj n jednak 2, paran, ili deljiv sa bilo kojim od neparnih brojeva između 3 i $\sqrt{n}$ - složenost $O(\sqrt{n})$?
  - Kod jako velikih vrednosti n, složenost je nešto gora – više ne možemo da posmatramo proveru deljivosti kao osnovnu operaciju, već kao funkciju broja cifara naših brojeva u binarnom sistemu.

10110 : 110 = 011
- 110
•1010
- 110
100

2^k	2^{k-1}	2^{k-2}	2^{k-3}	2^{k-4}		2^3	2^2	2^1	2^0
1	0	1	1	0	...	1	0	0	1

$2^{k+1} = n \Rightarrow k + 1 = \log(n)$

- Prava složenost će biti $O(log^2(n)\sqrt{n})$
- Ako zadržimo notaciju k = log(n) – broj cifara broja n u binarnom sistemu, složenost je **eksponencijalna**: $O\left(k^2 \times 2^{\frac{k}{2}}\right)$

Da li je broj prost?

- Postoji li bolje rešenje?
  - Da - AKS (Agraval, Kajal, Saksena) algoritam radi u složenosti O($log^6(n)$), pritom, ovaj algoritam je deterministički – uvek daje tačno rešenje!
- Možemo li još bolje?
  - I da i ne. Ako zanemarimo mali broj slučajeva kada algoritam karakteriše složene brojeve kao proste, složenost koju postižemo je čak O($log^3(n)$)
  - Fermaov test
  - Miler – Rabinov test

Da li je broj prost?

- Zašto nam je uopšte bitna još bolja složenost?
  - Prosti brojevi igraju veliku ulogu u kriptografiji – RSA algoritam za enkripciju/dekripciju je zasnovan na tome da rastavljanje velikih brojeva na proste činioce zahteva nepojmljivo mnogo vremena (čak i kvantnim kompjuterima!)
  - Potrebno nam je da generišemo dva prosta broja (privatni ključevi) i izračunamo njihov proizvod (javni ključ)

Fermaov test

- Mala Fermaova teorema: Broj p je prost AKKO je $a^{p-1} \equiv 1 \ (mod \ n)$ za svako $1 \leq a \leq p-1$.
- Neka su n složen i x ceo broj tako da n∤ x. Ako je $x^{n-1} \equiv 1 \ (mod \ n)$, kažemo
- da je x **Fermaov lažov**. Ako je $x^{n-1} \not\equiv 1 \ (mod \ n)$, kažemo da je x **Fermaov svedok**.
- Ideja: biramo broj $1 \leq x \leq n-1$ i proveravamo da li je Fermaov svedok ili Fermaov lažov.

Fermaov test - korektnost

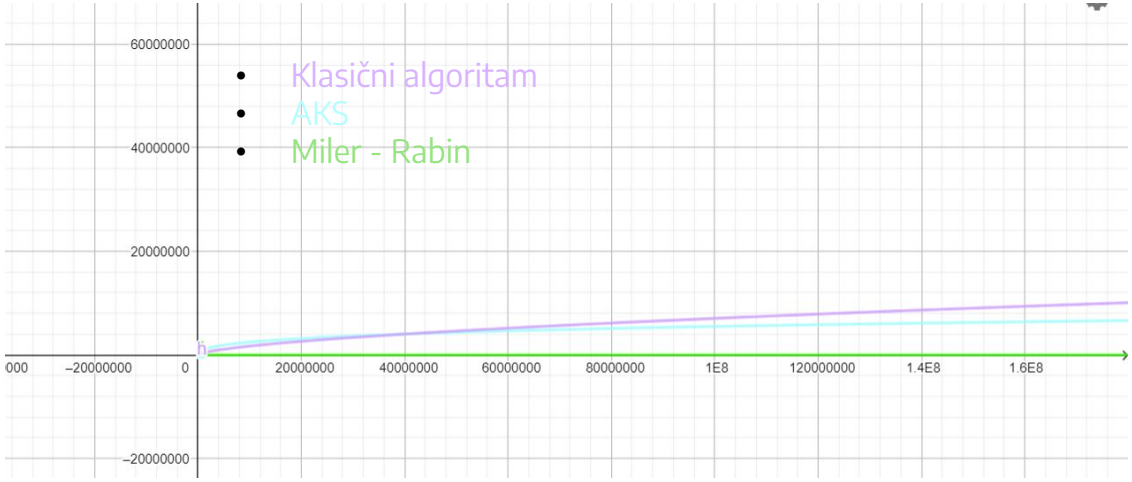
- Ako je broj prost, program će svakako vratiti tačno rešenje.
- Ako je broj složen i deljiv sa 2 program vraća tačno rešenje.
- Interesuju nas neparni složeni brojevi:
  - Teorema: Ako je n složen broj koji ima bar jednog Fermaovog svedoka sa kojim je uzajamno prost, onda n ima bar $\left\lfloor \frac{n-1}{2} \right\rfloor$ Fermaovih svedoka iz skupa {1, 2, ..., n-1}.
  - Karmajklov broj – neparan složen broj n takav da za svako x uzajamno prosto sa n važi da je x Fermaov lažov broja n.
  - Dakle, ako je n složen i nije Karmajklov broj, program vraća tačno rešenje u bar 50% slučajeva.

Miler – Rabinov test

- Teorema: Ako su $x, n \in Z^+$ takvi da je $x^2 \equiv 1 \ (mod\ n)$, ali n ne deli ni x-1 ni x+1, onda je n složen broj.
- Nova ideja: Zapisaćemo broj n-1 u obliku $2^p q$ i izabraćemo nasumično broj x iz skupa {1, 2, ..., n-1}. Zatim proveravamo koji ostatak pri deljenju sa n daje svaki od brojeva x^{2iq}, gde je $1 \leq i \leq p$.
- Dokazano je da ovaj algoritam prepoznaje Karmajklove brojeve sa verovatnoćom bar ¾ , a da je i u ostalim slučajevima ¼ gornje ograničenje verovatnoće da složene brojeve prijavljujemo kao proste.

Miler – Rabinov test

- Složenost ovog algoritma je $O(log^3(n))$.
- Ispravnost je $1 - (\frac{1}{4})^k$



k	tačnost
1	75 %
2	93.75 %
3	98.4375 %
4	99.609375 %
5	99.90234375 %
6	99.9755859 %
7	99.99389648 %
8	99.99847412 %

Agenda

1. Sve što treba znati iz verovatnoće
2. Randomized algoritmi
3. Bonus

Bonus

- Još neki primeri randomized algoritama:
  - Kargerov algortiam
  - Računanje površine preseka nekih figura
  - ...

Hvala na pažnji!

Pitanja?