NÁHODNOSTNÉ TECHNIKY

- vlastnosti náhodnostných algoritmov
- typy náhodnostných algoritmov
- náhodnostné algoritmy pre optimalizačné problémy

VLASTNOSTI NÁHODNOSTNÝCH ALGORITMOV

VLASTNOSTI NÁHODNOSTNÝCH ALGORITMOV

ZÁKLADNÉ POJMY

ZÁKLADNÉ POJMY

Definícia 1

Nech S je (konečný) priestor (elementárnych) udalostí.

Pravdepodobnostné rozdelenie je každá funkcia

$Pr : \mathcal{P}(S) \to [0,1]$ s vlastnosťami

(i) $Pr(\{x\}) \geq 0$ *pre každú elementárnu udalosť* $x \in S$

(ii) $Pr(S) = 1$

(iii) $Pr(A \cup B) = Pr(A) + Pr(B)$ *pre každé udalosti* $A, B \subseteq S$ *také, že* $A \cap B = \emptyset$.

- hodnota $Pr(A)$ sa nazýva *pravdepodobnosť udalosti* A
- dvojica (S, Pr) sa nazýva *pravdepodobnostný priestor*
- ak $Pr(\{x\}) = \frac{1}{|S|}$ pre všetky elementárne udalosti $x \in S$, tak Pr sa nazýva *uniformné pravdepodobnostné rozdelenie*

Definícia 2

Nech S je konečný priestor udalostí. Každá funkcia $X : S \to \mathbb{R}$ sa nazýva (diskrétna) náhodná premenná na S.

Definícia 3

Nech (S, Pr) je pravdepodobnostný priestor a X náhodná premenná na S. Očakávaná hodnota náhodnej premennej X je

$$E[X] \stackrel{def}{=} \sum_{s \in S} X(s) \cdot Pr(\{s\})$$

ekvivalentné označenie *střední hodnota náhodní veličiny*

- očakávaná hodnota konštantny c je

$$E[c] = c$$

- očakávaná hodnota súčinu náhodnej premennej X a konštantny c je

$$E[cX] = cE[X]$$

- očakávaná hodnota súčtu náhodných premenných X a Y je

$$E[X + Y] = E[X] + E[Y]$$

- pre nezávislé náhodné premenné X a Y je očakávaná hodnota ich súčinu

$$E[XY] = E[X]E[Y]$$

VLASTNOSTI NÁHODNOSTNÝCH ALGORITMOV

PROBLÉM ROVNOSTI

PROBLÉM ROVNOSTI

- na počítači X je uložená databáza x, na počítači Y je uložená databáza y
- x a y sú binárne reťazce dĺžky n
- úlohou je rozhodnúť, či x a y sú zhodné
- zaujíma nás, koľko bitov si musia počítače vymeniť, aby dokázali vyriešiť problém rovnosti
- neexistuje deterministický komunikačný protokol, ktorý by riešil problém rovnosti a pritom si počítače vymenili menej než n bitov
- navrhneme náhodnostný protokol, ktorý je výrazne efektívnejší

Náhodnostný protokol pre problém rovnosti

vstup počítač X má binárny reťazec $x = x_1 x_2 \dots x_n$

počítač Y má binárny reťazec $y = y_1 y_2 \dots y_n$

krok 1 X vyberie náhodne prvočíslo p z intervalu $[2, n^2]$

(*uvažujeme uniformné rozdelenie pravdepodobnosti*)

krok 2 X vypočíta číslo $s \equiv \mathit{Number}(x) \pmod{p}$

X pošle binárnu reprezentáciu čísel s a p počítaču Y

krok 3 Y vypočíta číslo $q \equiv \mathit{Number}(y) \pmod{p}$

ak $q \neq s$, tak Y vráti odpoveď $x \neq y$

ak $q = s$, tak Y vráti odpoveď $x = y$

platí $s < p < n^2$

zložitosť protokolu je rovná dĺžke binárneho zápisu čísel s, p, tj. nanajvýš

$$2 \cdot \lceil \log_2 n^2 \rceil \leq 4 \cdot \lceil \log_2 n \rceil$$

red kvalita protokolu

- protokol môže poskytnúť chybnú odpoveď
- nech $P(n^2) = \{p \mid p \text{ je prvočíslo}, p < n^2\}$
- prvočíslo p z $P(n^2)$ nazveme špatné pre (x, y), ak jeho výber spôsobí, že náhodnostný protokol vráti nesprávnu odpoveď
- pravdepodobnosť chybnej odpovede pre vstup (x, y) je potom

$$\frac{\text{počet špatných prvočísel pre } (x, y)}{|P(n^2)|}$$

- pre $n \geq 9$ platí

$$|P(n^2)| \geq \frac{n^2}{2 \ln n}$$

pokračovanie

- ukážeme, že špatných prvočísel pre (x, y) je nanajvýš $n - 1$
- ak $x = y$, tak $Number(x) \pmod p = Number(y) \pmod p$ pre každé prvočíslo p a odpoveď $x = y$ je vždy správna.
- ak $x \neq y$, tak počítač Y vráti nesprávnu odpoveď práve ak X vybral prvočíslo p také, že

$$Number(x) \pmod p = Number(y) \pmod p$$

  tj. ak p delí číslo $w = |Number(x) - Number(y)|$
- pretože x a y sú binárne reťazce dĺžky n, tak $w < 2^n$
- faktorizáciou čísla w dostávame $w = p_1^{i_1} \cdot p_2^{i_2} \cdot \ldots \cdot p_k^{i_k}$, pričom $k \leq n - 1$
- w má nanajvýš $n - 1$ prvočíselných deliteľov

ZÁVER

pravdepodobnosť, že protokol vráti nesprávnu odpoveď je pre $n \geq 9$

$$\frac{n-1}{|\mathrm{P}(n^2)|} \leq \frac{n-1}{n^2/\ln n^2} \leq \frac{\ln n^2}{n}$$

ZÁVER

pravdepodobnosť, že protokol vráti nesprávnu odpoveď je pre $n \geq 9$

$$\frac{n-1}{|\mathrm{P}(n^2)|} \leq \frac{n-1}{n^2/\ln n^2} \leq \frac{\ln n^2}{n}$$

deterministický vs. náhodnostný protokol

- $n = 10^{16}$
- zložitosť deterministického protokolu je 10^{16}
- zložitosť náhodnostného protokolu je 256
- pravdepodobnosť, že náhodnostný protokol vráti nesprávnu odpoveď je nanajvýš $0.36892 \cdot 10^{-14}$

VLASTNOSTI NÁHODNOSTNÝCH ALGORITMOV

MODELY NÁHODNOSTNÝCH ALGORITMOV

MODEL 1

náhodnostný algoritmus A chápeme ako pravdepodobnostné rozdelenie nad kolekciou $A_1, A_2, \ldots, A_n$ deterministických stratégií

pre vstupnú inštanciu w sa náhodne vyberie stratégia A_i

kvalitu náhodnostného algoritmu určuje jeho

zložitosť očakávaná časová zložitosť

presnosť správnosť výsledku pre rozhodnovacie problémy resp.
cena riešenia pre optimalizačné problémy

očakávaná časová zložitosť

- výpočet A_i na w označme C_i, jeho časovú zložitosť $Time(C_i)$
- definujeme pravdepodobnostný priestor $S_{A,w} = (\{C_1, \ldots, C_n\}, Pr)$, kde Pr je pravdepodobnostné rozdelenie nad stratégiami $A_1, \ldots, A_n$ *(typicky sa uvažuje uniformné rozdelenie)*
- nad $S_{A,w}$ definujeme náhodnú premennú Z, $Z(C_i) \stackrel{\text{def}}{=} Time(C_i)$
- očakávaná zložitosť A na w je

$$ExpTime_A(w) \stackrel{\text{def}}{=} E[Z] = \sum_{i=1}^{n} Pr(\{C_i\}) \cdot Z(C_i)$$

- očakávaná zložitosť náhodnostného algoritmu A je

$$ExpTime_A(n) \stackrel{\text{def}}{=} \max\{ExpTime_A(w) \mid |w| = n\}$$

presnosť pre rozhodovací problém

- nad $S_{A,w}$ definujeme náhodnú premennú X predpisom

$$X(C_i) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{ak } C_i \text{ vypočíta pre } w \text{ správnu odpoveď} \\ 0 & \text{inak} \end{cases}$$

- pravdepodobnosť že A vypočíta pre w *správnu* odpoveď je

$$Error_A(w) = E[X] = \sum_{i=1}^{n} Pr(\{C_i\}) \cdot X(C_i)$$

- pravdepodobnosť chyby pre algoritmus A je

$$Error_A = \max_w \{1 - Error_A(w)\}$$

pre optimalizačný problém sa mení definícia náhodnej premennej X tak, aby vypovedala o kvalite vypočítaného riešenia

VLASTNOSTI NÁHODNOSTNÝCH ALGORITMOV

MAXIMÁLNA SPLNITELNOSŤ

MAX-SAT

daná je booleovská formula v konjunktívnom normálnom tvare, hľadáme priradenie, ktoré spĺňa maximálny možný počet klauzulí

náhodnostný algoritmus pre MAX-SAT

vstup formula $\Phi = F_1 \wedge F_2 \wedge \ldots \wedge F_m$ v CNF nad množinou premenných $\{x_1, x_2, \ldots, x_n\}$

výpočet pre $i = 1, \ldots, n$, náhodne uniformne vyber hodnotu $\alpha_i \in \{0, 1\}$ (tj. $Pr(\alpha_i = 1) = Pr(\alpha_i = 0) = \frac{1}{2}$)

výstup $(\alpha_1, \alpha_2, \ldots, \alpha_n)$

očakávaná zložitosť algoritmu je rovná dĺžke výstupu (a je optimálna)

presnosť

- o kvalite algoritmu vypovedá pomer počtu splnených klauzulí k počtu všetkých klauzulí
- pravdepodobnostný priestor je $(\{0,1\}^n, Pr)$, kde Pr je uniformné rozdelenie nad $\{0,1\}^n$
- pre vstup $\Phi = F_1 \wedge F_2 \wedge \ldots \wedge F_m$ definujeme náhodné premenné $Z_1, Z_2, \ldots, Z_m$ predpisom

$$Z_i(\alpha) \stackrel{\text{def}}{=} \begin{cases} 1 & \text{ak klauzula } F_i \text{ je splnená priradením } \alpha \\ 0 & \text{ak klauzula } F_i \text{ nie je splnená priradením } \alpha \end{cases}$$

- náhodná premenná $Z = \sum_{i=1}^{m} Z_i$ vyjadruje počet splnených klauzulí; kvalita algoritmu je tak vyjadrená hodnotou $E[Z]$

$$E[Z] = E\big[\sum_{i=1}^{m} Z_i\big] = \sum_{i=1}^{m} E[Z_i]$$

pokračovanie

- predpokladáme, že žiaden literál sa v klauzule neopakuje, že žiadna klauzula neobsahuje súčasne literály x a $\neg x$ a že všetky klauzule sú navzájom rôzne
- klauzula F_i je nesplnená práve ak všetky jej literály majú hodnotu 0
-

$$\begin{aligned} E[Z_i] &= 1 \cdot Pr(F_i \text{ je splnená}) + 0 \cdot Pr(F_i \text{ nie je splnená}) \\ &= 1 - \frac{1}{2^k} \geq \frac{1}{2} \end{aligned}$$

$$E[Z] = \sum_{i=1}^{m} E[Z_i] \geq \sum_{i=1}^{m} \frac{1}{2} = \frac{m}{2}$$

- očakávaný počet splnených klauzulí pre formulu s m klauzulami je aspoň $m/2$

pokračovanie

- ak každá klauzula obsahuje aspoň k literálov, tak očakávaný počet splnených klauzulí pre formulu s m klauzulami je

$$(1 - (\frac{1}{2})^k)m$$

- algoritmus je výhodný pre formule s dlhými klauzulami

- ak každá klauzula má práve 3 literály (MAX-E3-SAT), tak očakávaný počet splnených klauzulí je $\frac{7}{8}m$
  tj. aproximatívny pomer algoritmu (v očakávanom prípade) je $\frac{7}{8}$

VLASTNOSTI NÁHODNOSTNÝCH ALGORITMOV

MAXIMÁLNY REZ

MAXIMÁLNY REZ GRAFU (MAX-CUT)

daný je neorientovaný hranovo ohodnotený graf

hľadáme taký rozklad (rez) množiny vrcholov grafu, ktorý maximalizuje súčet cien hrán rezu

náhodnostný algoritmus pre MAX-CUT

vstup graf $G = (V, H)$, ohodnotenie hrán $w : H \longrightarrow \mathbb{Z}$

výpočet pre každý vrcholv $v \in V$: s pravdepodobnosťou $\frac{1}{2}$ pridaj v do množiny U

výstup U, $V \setminus U$

očakávaná zložitosť algoritmu je rovná počtu vrcholov (a je optimálna)

presnosť

- definujeme náhodné premenné Z_{ij} pre $i, j \in \{1, \ldots, |V|\}$ predpisom

$$Z_{ij} \stackrel{\text{def}}{=} \begin{cases} 1 & \text{ak } \{i,j\} \in H,\ i \in U,\ j \in V \setminus U \\ 0 & \text{inak} \end{cases}$$

- náhodná premenná $Z = \sum_{\{i,j\} \in H} w_{ij} Z_{ij}$ vyjadruje váhu hrán rezu

$$\begin{aligned} E[Z] &= \sum_{\{i,j\} \in H} w_{ij} E[Z_{ij}] = \sum_{\{i,j\} \in H} w_{ij} Pr[\{i,j\} \text{je hrana rezu}] \\ &= \frac{1}{2} \sum_{\{i,j\} \in H} w_{ij} \\ &\geq \frac{1}{2} OPT \end{aligned}$$

VLASTNOSTI NÁHODNOSTNÝCH ALGORITMOV

MODEL II

MODEL 2

- v každom kroku výpočtu existuje niekoľko možností, ako vo výpočte pokračovať
- každá možnosť má určenú pravdepodobnosť
- pravdepodobnostný priestor sa definuje rovnako ako pre model 1
- hodnota $Pr(C_i)$ je definovaná ako súčin pravdepodobností výberov uskutočnených vo výpočte C_i
- všetky ostatné pojmy sú definované analogicky ako pre model 1

PRÍKLAD - QUICKSORT

Náhodnostný Quicksort

vstup množina čísel A

krok 1 ak $A = \{b\}$ tak výstup je b

ak $|A| \geq 2$, tak náhodne vyber prvok $a \in A$

krok 2 $A_< := \{b \in A \mid b < a\}$

$A_> := \{b \in A \mid b > a\}$

výstup (Quicksort($A_<$), a, Quicksort($A_>$))

TYPY NÁHODNOSTNÝCH ALGORITMOV

KLASIFIKÁCIA NÁHODNOSTNÝCH ALGORITMOV

Las Vegas

- bez chybnej odpovede
- s prípustnou odpoveďou *neviem*

Monte Carlo

- s jednostrannou chybou
- s dvojstrannou chybou
- s neohraničenou chybou

TYPY NÁHODNOSTNÝCH ALGORITMOV

LAS VEGAS

LAS VEGAS ALGORITMY

Náhodnostný algoritmus A sa nazýva **Las Vegas algoritmus** pre funkciu F ak pre každý vstup x

$$Pr(A(x) = F(x)) = 1.$$

- algoritmus pre každý vstup vypočíta správny výstup
- kľúčovou charakteristikou Las Vegas algoritmu je jeho očakávaná časová zložitosť
- *príklad: Quicksort s očakávanou časovou zložitosťou* $\theta(n \log n)$

LAS VEGAS ALGORITMY S ODPOVEĎOU NEVIEM

predpokladáme, že algoritmus ako výstupnú hodnotu môže poskytnúť tzv. neutrálnu odpoveď, označujeme ju *neviem*

Nech A je náhodnostný algoritmus, ktorý môže ako výstup poskytnúť hodnotu *neviem*. A sa nazýva **Las Vegas algoritmus** pre funkciu F ak pre každý vstup x

(i) $Pr(A(x) = F(x)) \geq 1/2$

(ii) $Pr(A(x) = \mathit{neviem}) = 1 - Pr(A(x) = F(x))$

- algoritmus pre každý vstup buď vypočíta správny výstup, alebo ako výstup poskytne hodnotu *neviem*
- správny výstup vypočíta s pravdepodobnosťou aspoň $1/2$

PRÍKLAD - ROVNOSŤ PODREŤAZCOV

- počítač X má desať reťazcov $x_1, x_2, \ldots, x_{10} \in \{0,1\}^n$
  počítač Y má desať reťazcov $y_1, y_2, \ldots, y_{10} \in \{0,1\}^n$
- počítač X má rozhodnúť, či existuje $j \in \{1, \ldots, 10\}$ také, že $x_j = y_j$
- zaujíma nás, koľko bitov si musia X a Y vymeniť, aby dokázali problém vyriešiť
- deterministický protokol potrebuje v najhoršom prípade $10n$ bitov
- ukážeme, že existuje randomizovaný protokol, ktorému stačí $n + \mathcal{O}(\log n)$ bitov

náhodnostný protokol RP pre rovnosť podreťazcov

vstup X má reťazce $x_1, x_2, \ldots, x_{10} \in \{0,1\}^n$

Y má reťazce $y_1, y_2, \ldots, y_{10} \in \{0,1\}^n$

krok 1 X vyberie náhodne (uniformne) 10 prvočísel $p_1, \ldots, p_{10}$ z intervalu $[2, n^2]$

krok 2 X vypočíta $s_i = Number(x_i) \pmod{p_i}$ pre $i = 1, \ldots, 10$ a pošle čísla $p_1, \ldots, p_{10}, s_1, \ldots, s_{10}$ počítaču Y

krok 3 Y vypočíta $q_i = Number(y_i) \pmod{p_i}$ pre $i = 1, \ldots, 10$

krok 4a ak $s_i \neq q_i$ pre všetky $i = 1, \ldots, 10$, tak Y pošle X hodnotu 0 a X vráti odpoveď *zamietam*

krok 4b v opačnom prípade nech j je najmenšie číslo pre ktoré $s_j = q_j$

Y pošle reťazce j a y_j počítaču X

krok 5 X porovná x_j a y_j

ak sú zhodné, vráti odpoveď *akceptujem*

ak sú rôzne, vráti odpoveď *neviem*

zložitosť protokolu RP

- v najhoršom prípade si počítače vymenia čísla $p_1, \ldots, p_{10}$, $s_1, \ldots, s_{10}$, j, y_j *(presnejšie: ich binárne zápisy)*
- čísla $p_1, \ldots, p_{10}$, $s_1, \ldots, s_{10}$ sú menšie než n^2 a každé sa dá reprezentovať binárnym reťazcom dĺžky $2\lceil \log_2 n \rceil$
- číslo j sa dá reprezentovať 4 bitmi, y_j má n bitov
- spolu $n + 40\lceil \log_2 n \rceil + 4 = n + \mathcal{O}(\log n)$ bitov

presnosť protokolu RP

nech $r = x_1, \ldots, x_{10}, y_1, \ldots, y_{10}$ je vstup taký, že $x_i \neq y_i$ pre $i \in \{1, \ldots, 10\}$

- ak pre všetky vybrané prvočísla platí $Number(x_i) \pmod{p_i} \neq Number(y_i) \pmod{p_i}$, tak protokol vráti správnu (zamietajúcu) odpoveď
- pre náhodne vybrané $p_i \in P(n^2)$ je pravdepodobnosť toho, že $Number(x_i) \pmod{p_i} \neq Number(y_i) \pmod{p_i}$ aspoň $1 - \frac{2 \ln n}{n}$
- $$Pr(RP(r) = zamietam)) \geq (1 - \frac{2 \ln n}{n})^{10} \geq 1 - \frac{20 \ln n}{n} \geq \frac{1}{2}$$
- naopak, ak pre nejaké vybrané prvočíslo platí $Number(x_i) \pmod{p_i} = Number(y_i) \pmod{p_i}$, tak protokol vráti odpoveď *neviem*

nech $r = x_1, \ldots, x_{10}, y_1, \ldots, y_{10}$ je vstup a nech j je najmenšie číslo také, že $x_j = y_j$

- zrejme $Number(x_j) \pmod{p_j} = Number(y_j) \pmod{p_j}$ a protokol akceptuje práve ak $Number(x_i) \pmod{p_i} \neq Number(y_i) \pmod{p_i}$ pre všetky $i \in \{1, \ldots, j-1\}$
- analogicky ako v predchádzajúcom prípade sa dá ukázať, že pravdepodobnosť takejto udalosti je

$$\left(1 - \frac{2 \ln n}{n}\right)^{j-1} \geq 1 - \frac{2(j-1) \cdot \ln n}{n}$$

- výraz nadobúda minimálnu hodnotu pre $j = 10$ a preto

$$Pr(RP(r) = akceptujem) \geq 1 - \frac{18 \ln n}{n} > \frac{1}{2}$$

- naopak, ak existuje $i \in \{1, \ldots, j-1\}$ také, že $Number(x_i) \mod p_i = Number(y_i) \mod p_i$, tak protokol vráti odpoveď *neviem*.

TRANSFORMÁCIE LAS VEGAS ALGORITMOV I

transformácia algoritmu A s odpoveďou *neviem* na algoritmus B, ktorý vypočíta vždy správnu odpoveď

- B simuluje A
- ak A vypočíta odpoveď *neviem*, tak B začne nový beh algoritmu A
- v opačnom prípade vráti B rovnaký výsledok ako A

vlastnosti algoritmu B

- ak výpočet B skončí, tak vždy vráti správnu odpoveď
- očakávána časová zložitosť algoritmu B je $ExpTime_B(n) \in \mathcal{O}(Time_A(n))$

očakáváná časová zložitosť algoritmu B

- pravdepodobnosť, že B skončí po jednom behu A je aspoň $1/2$,
  po dvoch behoch $3/4$, atď.
- pravdepodobnosť, že B skončí v čase nanajvýš $k \cdot Time_A(w)$ je aspoň $1 - \frac{1}{2^k}$
- búno všetky výpočty A na w s výsledkom *neviem* majú dĺžku $Time_A(w)$
- pre $i \in \mathbb{N}$ nech Set_i označuje množinu tých výpočtov C algoritmu B na w, pre ktoré
  $(i-1)\,Time_A(w) < Time(C) \leq i \cdot Time_A(w)$
- $$\sum_{C \in Set_i} Pr(\{C\}) \leq \frac{1}{2^{i-1}}$$

$$
\begin{aligned}
ExpTime_B(n) &= \sum_{i=1}^{\infty} \sum_{C \in Set_i} Time_B(C) \cdot Pr(\{C\}) \\
&\leq \sum_{i=1}^{\infty} i \cdot Time_A(w) \cdot \frac{1}{2^{i-1}} \\
&= Time_A(w) \cdot \sum_{i=1}^{\infty} \frac{1}{2^{i-1}} \\
&< 6 \cdot Time_A(w)
\end{aligned}
$$

TRANSFORMÁCIE LAS VEGAS ALGORITMOV II

transformácia algoritmu A, ktorý poskytne vždy správnu odpoveď, na algoritmus B s prípustnou odpoveďou *neviem*

- transformácia má opodstatnenie v prípadoch, keď niektoré výpočty algoritmu A sú neúmerne dlhé
- ako zvoliť hramicu, po ktorej B zastaví beh A a vráti odpoveď *neviem* tak, aby B bol stále Las Vegas algoritmus?
- postačujúcou hranicou je

$$2 \cdot ExpTime_A(w)$$

pretože viac než polovica výpočtov A na w nemôže mať časovú zložitosť väčšiu než je dvojnásobok priemeru (tj. $ExpTime_A(w)$)

TYPY NÁHODNOSTNÝCH ALGORITMOV

MONTE CARLO

MONTE CARLO S JEDNOSTRANNOU CHYBOU

pre rozhodovacie problémy

Nech A je randomizovaný algoritmus pre jazyk L. A je Monte Carlo algoritmus s jednostrannou chybou (1MC) pre L ak

(i) pre každé $x \in L$ platí $Pr(A(x) = 1) \geq 1/2$

(ii) pre každé $x \notin L$ platí $Pr(A(x) = 0) = 1$

špeciálny prípad 1MC algoritmu: hodnota $Pr(A(x) = 1)$ *sa s rastúcou dĺžkou* x *limitne blíži k 1*

- príkladom Monte Carlo algoritmu s jednostrannou chybou je randomizovaný protokol pre problém rovnosti
- protokol akceptuje jazyk
  $L = \{(x, y) \mid x, y \in \{0,1\}^n, x \neq y, n \in \mathbb{N}\}$
- protokol akceptuje slovo z jazyka L s pravdepodobnosťou aspoň $1 - \frac{2 \cdot \ln n}{n}$
- pre slovo nepatriace do L vypočíta protokol vždy správnu (zamietajúcu) odpoveď
- pre uvedený protokol platí, že pravdepodobnosť akceptovania sa pre slová z L s rastúcou dĺžkou slov limitne blíži k 1

AMPLIFIKÁCIA

úprava algoritmu s cieľom znížiť pravdepodobnosť chybnej odpovede pri zachovaní jeho (asymptotickej) zložitosti

motivácia

- presnosť vs dôveryhodnosť
- rozdiel medzi *hodom mincou* a náhodnostným výpočtom

amplifikácia algoritmu A typu Monte Carlo s jednostrannou chybou

konštruujeme algoritmus A_k

- algoritmus A_k zopakuje výpočet algoritmu A na vstupe x nezávisle(!) k krát za sebou
- nech $\alpha_1, \alpha_2, \ldots, \alpha_k$ sú výstupy jednotlivých výpočtov
- ak existuje index j taký, že $\alpha_j = 1$, tak algoritmus A_k akceptuje vstup; v tomto prípade x zaručene patrí do L
- naopak, ak $\alpha_1 = \alpha_2 = \ldots = \alpha_k = 0$, tak algoritmus A_k zamietne; zamietajúca odpoveď algoritmu A_k môže byť nesprávna
- pravdepodobnosť nesprávnej odpovede algoritmu A_k je $(Pr(A(x) = 0))^k$; vlastnosť (i) 1MC algoritmu A garantuje, že $(Pr(A(x) = 0))^k \leq (1/2)^k$

vlastnosti amplifikovaného algoritmu A_k

- voľba parametru k je daná požadovanou presnosťou amplifikovaného algoritmu: ak požadujeme presnosť ε, tak volíme k tak, aby $1 - \left(1/2\right)^k \leq \varepsilon$
- časová zložitosť algoritmu A_k je k násobkom zložitosti algoritmu A (lineárny nárast)
- pravdebobnosť chybnej odpovede klesá exponenciálne s rastúcim parametrom k

ALGORITMUS MONTE CARLO S OHRANIČENOU CHYBOU

Randomizovaný algoritmus A je Monte Carlo algoritmus s ohraničenou chybou pre funkciu F práve ak existuje $\varepsilon \in \mathbb{R}$, $0 < \varepsilon < 1/2$, také, že pre každý vstup x

$$Pr(A(x) = F(x)) \geq \frac{1}{2} + \varepsilon.$$

Las Vegas a Monte Carlo algoritmy s jednostrannou chybou sú špeciálnym prípadom Monte Carlo algoritmov s ohraničenou chybou

amplifikácia algoritmu A typu Monte Carlo s ohraničenou chybou

konštruujeme algoritmus A_t, ktorý pre vstup x

- zopakuje výpočet A na x (nezávisle!) t krát za sebou výstupy jednotlivých výpočtov označí $\alpha_1, \ldots, \alpha_t$
- ak existuje hodnota α, ktoré sa v postupnosti $\alpha_1, \ldots, \alpha_t$ opakuje aspoň $\lceil \frac{t}{2} \rceil$ krát, tak výstupom výpočtu je α
- v opačnom prípade je výstupom *neviem*

algoritmus A_t vypočíta správny výsledok s pravdepodobnosťou aspoň

$$1 - (1 - 4 \cdot \varepsilon^2)^{\frac{t}{2}}$$

dôkaz vlastností amplifikovaného algoritmu A_t

- pre vstup x označme p a ε_x čísla také, že

$$p = Pr(A(x) = F(x)) = \frac{1}{2} + \varepsilon_x$$

- určíme, aká je pravdepodobnosť $pr_i(x)$ toho, že algoritmus A_t vypočíta správny výsledok v *práve* i výpočtoch ($i < \lceil t/2 \rceil$)

$$
\begin{aligned}
pr_i(x) &= \binom{t}{i} \cdot p^i \cdot (1-p)^{t-i} && t \geq 2i \\
&= \binom{t}{i} \cdot \big(p \cdot (1-p)\big)^i \cdot (1-p)^{t-2i} && p = \frac{1}{2} + \varepsilon_x \\
&= \binom{t}{i} \cdot \big((\frac{1}{2} + \varepsilon_x) \cdot (\frac{1}{2} - \varepsilon_x)\big)^i \cdot (\frac{1}{2} - \varepsilon_x)^{2\cdot(\frac{t}{2}-i)} \\
&= \binom{t}{i} \cdot (\frac{1}{4} - \varepsilon_x^2)^i \cdot ((\frac{1}{2} - \varepsilon_x^2)^2)^{\frac{t}{2}-i} && \frac{1}{2} - \varepsilon_x < \frac{1}{2} + \varepsilon_x \\
&< \binom{t}{i} \cdot (\frac{1}{4} - \varepsilon_x^2)^i \cdot \big((\frac{1}{2} - \varepsilon_x) \cdot (\frac{1}{2} + \varepsilon_x)\big)^{\frac{t}{2}-i} \\
&= \binom{t}{i} \cdot (\frac{1}{4} - \varepsilon_x^2)^i \cdot (\frac{1}{4} - \varepsilon_x^2)^{\frac{t}{2}-i} \\
&= \binom{t}{i} \cdot (\frac{1}{4} - \varepsilon_x^2)^{\frac{t}{2}} \\
&\leq \binom{t}{i} \cdot (\frac{1}{4} - \varepsilon^2)^{\frac{t}{2}}
\end{aligned}
$$

A_t vypočíta správny výsledok $F(x)$ práve ak aspoň $\lceil t/2 \rceil$ behov A vypočíta správny výsledok a preto

$$
\begin{aligned}
Pr(\mathbf{A_t}(x) = F(x)) &= 1 - \sum_{i=0}^{\lfloor \frac{t}{2} \rfloor} pr_i(x) \\
&> 1 - \sum_{i=0}^{\lfloor \frac{t}{2} \rfloor} \binom{t}{i} \cdot \left(\frac{1}{4} - \varepsilon_x^2\right)^{\frac{t}{2}} \\
&= 1 - \left(\frac{1}{4} - \varepsilon_x^2\right)^{\frac{t}{2}} \cdot \sum_{i=0}^{\lfloor \frac{t}{2} \rfloor} \binom{t}{i} \\
&> 1 - \left(\frac{1}{4} - \varepsilon_x^2\right)^{\frac{t}{2}} \cdot 2^t \\
&\geq 1 - \left(1 - 4 \cdot \varepsilon_x^2\right)^{\frac{t}{2}} \\
&\geq 1 - \left(1 - 4 \cdot \varepsilon^2\right)^{\frac{t}{2}}
\end{aligned}
$$

- horná hranica pre veľkosť chyby $(1 - 4 \cdot \varepsilon^2)^{\frac{t}{2}}$ sa s rastúcim t blíži k 0
- ak chceme, aby pre danú konštantu δ platilo

$$Pr(A_t(x) = F(x)) \geq 1 - \delta,$$

tak stačí zvoliť t také, že

$$t \geq \frac{2 \cdot \ln \delta}{\ln(1 - 4 \cdot \varepsilon^2)}$$

- ak δ a ε sú dané konštanty, tak t je tiež konštanta a

$$Time_{A_t}(n) \in \mathcal{O}(Time_A(n))$$

MONTE CARLO S NEOHRANIČENOU CHYBOU

Randomizovaný algoritmus A je Monte Carlo algoritmus s neohraničenou chybou pre funkciu F práve ak pre každý vstup x platí

$$Pr(A(x) = F(x)) > \frac{1}{2}.$$

amplifikácia algoritmu A typu Monte Carlo s ohraničenou chybou

- pre algoritmus A s neohraničenou chybou sa rozdiel medzi pravdepodobnosťou chyby a hodnotou $1/2$ môže s rastúcou dĺžkou vstupu blížiť k 0
- ak chceme, aby pre dané fixné δ platilo

$$Pr(A_t(x) = F(x)) \geq 1 - \delta,$$

tak počet nezávislých opakovaní algoritmu A musí byť

$$t(|x|) \geq \frac{2 \cdot \ln \delta}{\ln(1 - 4 \cdot \varepsilon_x^2)}$$

- uvážme situáciu keď $Pr(A(x) = F(x)) = \frac{1}{2} + \frac{1}{2^{|x|}} > \frac{1}{2}$
- aký musí byť počet t nezávislých opakovaní výpočtu A na x, ak požadujeme, aby pre danú konštantu δ platilo $Pr(A_t(x) = F(x)) > 1 - \delta$
-

$$\begin{aligned} t(|x|) &\geq \frac{2 \cdot \ln \delta}{\ln(1 - 4 \cdot \varepsilon_x^2)} \\ &= \frac{2 \cdot \ln \delta}{\ln(1 - 4 \cdot 2^{-|x|})} && \varepsilon_x = 2^{-|x|} \\ &\geq \frac{2 \cdot \ln \delta}{-2^{-2 \cdot |x|}} && \ln(1-y) \leq -y \text{ pre } 0 < y < 1 \\ &= (-2 \cdot \ln \delta) \cdot 2^{2 \cdot |x|}. \end{aligned}$$

- časová zložitosť je exponenciálna voči dĺžke x,

$$Time_{A_t}(x) \geq (-2 \cdot \ln \delta) \cdot 2^{2 \cdot |x|} \cdot Time_A(x)$$

AMPLIFIKÁCIA

AMPLIFIKÁCIA

- úprava algoritmu s cieľom znížiť pravdepodobnosť chybnej odpovede pri zachovaní jeho (asymptotickej) zložitosti
- triviálny postup: opakovanie celého výpočtu
- *ako postupovať v situácii, keď opakovanie celého výpočtu vedie k neakceptovateľnému zvýšeniu zložitosti?*

MINIMÁLNY REZ

vstup neorientovaný graf

prípustné riešenie rez grafu, tj. rozklad množiny vrcholov grafu na množiny (A, B)

cena riešenia veľkosť rezu (A, B). tj. je počet hrán (s, t) grafu takých, že $s \in A$, $t \in B$

cieľ prípustné riešenie s minimálnou cenou

DETERMINISTCKÝ ALGORITMUS

- redukcia na problém nájdenia minimálneho $s - t$ rezu v orientovanom grafe
- v grafe G nahradíme neorientovanú hranu $(u, v) \in E$ dvojicou opačne orientovaných hrán (u, v) a (v, u) s kapacitou 1
- zvolíme pevne vrchol s a pre všetky vrcholy $t \in V \setminus \{s\}$ vypočítame veľkosť minimálneho $s - t$ rezu
- kapacita minimálneho $s - t$ rezu je rovná hodnote maximálneho $s - t$ toku
- celková zložitosť $\mathcal{O}(n^3)$

NÁHODNOSTNÝ ALGORITMUS

- algoritmus založený na kontrakcii vrcholov
- multigraf $G = (V, E)$ - násobné hrany medzi vrcholmi
- kontrakcia: náhodne vyberieme hranu (u, v) grafu G a vytvoríme graf G'
  - u a v sú spojené do jedného nového vrcholu w; všetky ostatné vrcholy zostávaju nezmenené
  - hranu, ktorej jeden z koncových vrcholov tvorí u alebo v, nahradíme hranou s koncovým vrcholom w; ostatné hrany sú nezmenené
- postup rekurzívne aplikujeme na G'
- G' má supervrcholy w; každý supervrchol w zodpovedá množine $S(w) \subseteq V$
- výpočet končí keď G' obsahuje len dva supervrcholy v_1 a v_2
- množiny $S(v_1)$ a $S(v_2)$ určujú rozklad množiny vrcholov

iniciálne volanie: graf $G(V, E)$ *a funkcia* $S(v) = \{v\}$

Funkce KONTRAKCE$(G = (\overline{V}, \overline{E}), S\colon \overline{V} \to 2^V)$

---

1 **if** G má dva vrcholy v_1 a v_2 **then return** (rez $(S(v_1), S(v_2))$

2 náhodne vyber hranu $(u, v) \in \overline{E}$

3 $\overline{V} \leftarrow \overline{V} \cup \{z_{uv}\} \setminus \{u, v\}$

4 $\overline{E} \leftarrow \overline{E} \cup \{\ \{x, z_{uv}\} \mid \{x, y\} \in \overline{E}, y = u \text{ alebo } y = v\}$

5 $\overline{E} \leftarrow \overline{E} \setminus \{\ \{x, y\} \mid \{x, y\} \in \overline{E}, y = u \text{ alebo } y = v\}$

6 $S(z_{uv}) \leftarrow S(u) \cup S(v)$

7 Kontrakce$(G = (\overline{V}, \overline{E}), S\colon \overline{V} \to 2^V)$

Algorimus kontrakcií nájde globálny minimálny rez s pravdepodobnosťou aspoň $1/\binom{n}{2}$.

- označme k veľkosť minimálneho rezu
  nech F je množina hrán, ktoré tvoria minimálny rez (A, B)
- ak algoritmus v niektorom z krokov kontrahuje hranu z F, tak jeho výstupom nemôže byť rez (A, B)
- pre odhad pravdepodobnosti kontrakcie hrany z F
  potrebujeme odhad počtu hrán: každý vrchol má stupeň aspoň k a preto $|E| \geq \frac{1}{2}kn$
- pravdepodobnosť, že v prvej iterácii je kontrahovaná hrana z F je nanajvýš

$$\frac{k}{\frac{1}{2}kn} = \frac{2}{n}$$

- uvažujme situáciu po j iteráciach, keď graf $\overline{G}$ má $n-j$ supervrcholov a žiadna hrana z F nebola kontrahovaná
- každý rez grafu $\overline{G}$ je rezom grafu G a preto každý vrchol v $\overline{G}$ má stupeň aspoň k
- celkový počet hrán v $\overline{G}$ je aspoň $\frac{1}{2}k(n-j)$
- pravdepodobnosť, že v iterácii $j+1$ je kontrahovaná hrana z F je nanajvýš

$$\frac{k}{\frac{1}{2}k(n-j)} = \frac{2}{n-j}$$

- algoritmus nájde rez (A, B) práve ak v žiadnej iterácii sa nekontrahuje hrana z F

- udalosť ε_j - v iterácii j nie je kontrahovaná hrana z F
- $Pr(\varepsilon_1) \geq 1 - 2/n$
- $Pr(\varepsilon_{j+1} \mid \varepsilon_1 \cap \varepsilon_2 \cap \ldots \cap \varepsilon_j) \geq 1 - 2/(n-j)$

$$
\begin{aligned}
&Pr(\varepsilon_1) \cdot Pr(\varepsilon_2 \mid \varepsilon_1) \cdots Pr(\varepsilon_{n-2} \mid \varepsilon_1 \cap \ldots \cap \varepsilon_{n-3}) \\
&\qquad \geq (1 - \frac{2}{n})(1 - \frac{2}{n-1}) \cdots (1 - \frac{2}{n-j}) \cdots (1 - \frac{2}{3}) \\
&\qquad = (\frac{n-2}{n})(\frac{n-3}{n-1})(\frac{n-4}{n-2}) \cdots (\frac{2}{4})(\frac{1}{3}) \\
&\qquad = \frac{2}{n(n-1)} = \binom{n}{2}^{-1}
\end{aligned}
$$

zložitosť algoritmu kontrakcií

- pravdepodobnosť, že algoritmus nenájde globálny minimálny rez je nanajvýš $(1 - 1/\binom{n}{2})$
- opakovaním výpočtu môžeme znížiť pravdepodobnosť neúspechu (z nájdených rezov vyberáme najmenší)
- ak opakujeme výpočet $\binom{n}{2}$ krát, tak pravdepodobnosť neúspechu je nanajvýš

$$(1 - 1/\binom{n}{2})^{\binom{n}{2}} \leq \frac{1}{e}$$

- *celková zložitosť vyššia ako u deterministického algoritmu*

optimalizácia algoritmu kontrakcií

1. graf kontrahujeme až kým nemá $l(n)$ vrcholov, ďalej počítame deterministicky
   (optimálna voľba funkcie $l(n) = \lfloor n^{2/3} \rfloor$)
2. pretože pravdepodobnosť chybnej voľby hrany rastie s počtom iterácií, pri opakovaní výpočtu opakujeme len kontrakcie v záverečných iteráciach

pri optimálnej voľba parametra h a počtu opakovaní je pravdepodobnosť neúspechu ohraničená konštantou a zložitosť výpočtu je $\mathcal{O}(n^2(\log n)^3)$

DERANDOMIZÁCIA

DERANDOMIZÁCIA

- algoritmická technika transformácie randomizovaného algoritmu na deterministický algoritmus, ktorý garantuje rovnako dobré výsledky ako randomizovaný algoritmus

- prečo navrhujeme randomizovaný algoritmus?
- (veľmi často) je jednoduchšie analyzovať a dokázať vlastnosti randomizovaného algoritmu než jeho deterministickej varianty
- randomizácia prináša jednoduchosť návrhu a analýzy algoritmu, derandomizácia zachováva parametry algoritmu

Náhodnostný algoritmus pre MAX-SAT

vstup formula $\Phi = F_1 \wedge F_2 \wedge \ldots \wedge F_m$ v CNF nad množinou premenných $\{x_1, x_2, \ldots, x_n\}$

výpočet náhodne vyber priradenie $(\alpha_1, \alpha_2, \ldots, \alpha_n) \in \{0,1\}^n$ s pravdepodobnosťou $Pr(\alpha_i = 1) = Pr(\alpha_i = 0) = \frac{1}{2}$,

výstup $(\alpha_1, \alpha_2, \ldots, \alpha_n)$

- náhodná premenná Z vyjadruje počet splnených klauzulí
- očakávaný počet splnených klauzulí je

$$E[Z] \geq m/2 = \frac{1}{2} OPT$$

DERANDOMIZÁCIA ALGORITMU PRE MAX-SAT

- namiesto náhodného rozhodnutia, či premenná x_i má hodnotu *true* alebo *false*, priradíme hodnotu premennej x_i deterministicky
- postupujeme sekvenčne: najprv určíme hodnotu premennej x_1, potom x_2, atď.
- využíva sa vlastnosť *self-redukcie* problému

- ako určíme hodnotu premennej x_1?
- predstavme si, že hodnotu premennej x_1 určíme deterministicky, ale hodnoty všetkých ostatných premenných určíme náhodne
- v takom prípade má zmysel zvoliť hodnotu premennej x_1 tak, aby sa maximalizovala očakávaná hodnota počtu splnených klauzulí výsledného riešenia
- predpokladajme, že vieme vypočítať očakávaný počet splnených klauzulí $E[Z|_{x_1=1}]$ formule $F_{x_1:=1}$, ktorá vznikne z F dosadením hodnoty *true* za x_1
- premennej x_1 priradíme hodnotu *true* práve ak očakávaný počet je vyšší pre formulu $F_{x_1:=1}$ než pre formulu $F_{x_1:=0}$

- predpis pre určenie hodnoty premennej x_1

$$x_1 := \begin{cases} 1 & \text{ak } E[Z|_{x_1=1}] \geq E[Z|_{x_1=0}] \\ 0 & \text{inak} \end{cases}$$

- z definície očakávanej hodnoty platí

$$\begin{aligned} E[Z] &= E[Z|_{x_1=1}]Pr[x_1 = 1] + E[Z|_{x_1=0}]Pr[x_1 = 0] \\ &= \frac{1}{2}(E[Z|_{x_1=1}] + E[Z|_{x_1=0}]) \end{aligned}$$

- ak zvolíme $x_1 = 1$, tak $E[Z|_{x_1=1}] \geq E[Z]$
- ak zvolíme $x_1 = 0$, tak $E[Z|_{x_1=0}] \geq E[Z]$

- nech $a_1, \ldots, a_i$ sú hodnoty, ktoré sme priradili premenným $x_1, \ldots, x_i$
- ako určíme hodnotu premennej x_{i+1}?
- analogickým postupom:

$$x_{i+1} := \begin{cases} 1 & \text{ak } E[Z|_{x_1=a_1,\ldots,x_i=a_i,x_{i+1}=1}] \geq E[Z|_{x_1=a_1,\ldots,x_i=a_i,x_{i+1}=0}] \\ 0 & \text{inak} \end{cases}$$

- z definície očakávanej hodnoty platí

$$\begin{aligned} E[Z|_{x_1=a_1,\ldots,x_i=a_i}] &= \frac{1}{2}E[Z|_{x_1=a_1,\ldots,x_i=a_i,x_{i+1}=1}] \\ &\quad + \frac{1}{2}E[Z|_{x_1=a_1,\ldots,x_i=a_i,x_{i+1}=0}] \end{aligned}$$

- pokračujeme až kým neurčíme hodnoty všetkých premenných
- očakávaný počet splnených klauzulí $E[Z|_{x_1=a_1,\ldots,x_n=a_n}]$ vo formule $F_{x_1=a_1,\ldots,x_n=a_n}$ je rovný počtu splnených klauzulí pre zvolené hodnoty premenných
- indukciou overíme, že

$$\text{počet splnených klauzulí} = E[Z|_{x_1=a_1,\ldots,x_n=a_n}] \geq E[Z] \geq \frac{1}{2}OPT$$

ako vypočítame hodnotu $E[Z|_{x_1=a_1,\ldots,x_i=a_i}]$?

$$E[Z|_{x_1=a_1,\ldots,x_i=a_i}] = \sum_{i=1}^{m} Pr[\text{klauzula } C_i \text{ formule } F_{x_1=a_1,\ldots,x_i=a_i} \text{ je splnená}]$$

pre klauzulu C_i je

- pravdepodobnosť jej splnenia rovná 1 ak klauzula je po dosadení hodnôt splnená
- pravdepodobnosť je $(1 - (\frac{1}{2})^k)$ ak po dosadení hodnôt obsahuje klauzula k literálov

príklad

$F = (x_1 \vee x_2 \vee \neg x_3 \vee x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge x_2 \wedge (x_3 \vee \neg x_4)$

$Z = \frac{15}{16} + \frac{7}{8} + \frac{1}{2} + \frac{3}{4} = \frac{49}{16}$

$F_{x_1=1} = \mathit{true} \wedge (\neg x_2 \vee x_4) \wedge x_2 \wedge (x_3 \vee \neg x_4)$

$Z_{x_1=1} = 1 + \frac{3}{4} + \frac{1}{2} + \frac{3}{4} = \frac{48}{16}$

$F_{x_1=0} = (x_2 \vee \neg x_3 \vee x_4) \wedge \mathit{true} \wedge x_2 \wedge (x_3 \vee \neg x_4)$

$Z_{x_1=0} = \frac{7}{8} + 1 + \frac{1}{2} + \frac{3}{4} = \frac{50}{16}$

pre x_1 volíme hodnotu *false*

príklad - pokračovanie

$F = (x_1 \vee x_2 \vee \neg x_3 \vee x_4) \wedge (\neg x_1 \vee \neg x_2 \vee x_4) \wedge x_2 \wedge (x_3 \vee \neg x_4)$

$F_{x_1=0,x_2=0} = (\neg x_3 \vee x_4) \wedge \mathit{true} \wedge \mathit{false} \wedge (x_3 \vee \neg x_4)$

$Z_{x_1=0,x_2=0} = \frac{3}{4} + 1 + 0 + \frac{3}{4} = \frac{40}{16}$

$F_{x_1=0,x_2=1} = \mathit{true} \wedge \mathit{true} \wedge \mathit{true} \wedge (x_3 \vee \neg x_4)$

$Z_{x_1=0,x_2=1} = 1 + 1 + 1 + \frac{3}{4} = \frac{60}{16}$

pre x_2 volíme hodnotu *false*

DERANDOMIZÁCIA - ZHRNUTIE

- *method of conditional expectations*
- „selfredukovateľnosť“ problému
- technika sa dá aplikovať na ďalšie prezentované algoritmy

NÁHODNOSTNÉ ALGORITMY PRE OPTIMALIZAČNÉ PROBLÉMY

aproximatívny pomer R_A algoritmu A chápeme ako náhodnú premennú

cieľom je

- odhadnúť strednú hodnotu náhodnej premennej R_A, alebo
- garantovať, že aproximatívny pomer δ sa dosiahne s pravdepodobnosťou aspoň $1/2$.

RANDOMIZOVANÉ APROXIMATÍVNE ALGORITMY

Nech δ je konštanta, $\delta > 1$. Randomizovaný algoritmus A je randomizovaný $E[\delta]$-aproximatívny algoritmus pre optimalizačný problém U práve ak pre každý vstup x platí, že $A(x)$ je prípustné riešenie x a zároveň $E[(R_A(x)] \leq \delta$

Nech δ je konštanta, $\delta > 1$. Randomizovaný algoritmus A je randomizovaný δ-aproximatívny algoritmus pre optimalizačný problém U práve ak pre každý vstup x platí, že $A(x)$ je prípustné riešenie x a zároveň $Prob(R_A(x) \leq \delta) \geq 1/2$

$R_A(x)$ je aproximatívny pomer algoritmu A na vstupe x

NÁHODNOSTNÉ ALGORITMY PRE OPTIMALIZAČNÉ PROBLÉMY

MAXIMÁLNA SPLNITEĽNOSŤ

Náhodnostný algoritmus pre MAX-SAT

vstup formula $\Phi = C_1 \wedge \ldots \wedge C_m$ v CNF nad množinou premenných $\{x_1, x_2, \ldots, x_n\}$
váhová funkcia $w : \mathcal{C} = \{C_1, \ldots C_m\} \to \mathbb{R}$

výpočet náhodne vyber priradenie $(\alpha_1, \alpha_2, \ldots, \alpha_n) \in \{0,1\}^n$
s pravdepodobnosťou $Pr(\alpha_i = 1) = Pr(\alpha_i = 0) = \frac{1}{2}$,

výstup $(\alpha_1, \alpha_2, \ldots, \alpha_n)$

analýza algoritmu

- náhodná premenná W = váha splnených klauzulí
- pre klauzulu c, náhodná premenná W_c je rovná váhe, ktorou klauzula c prispieva do W, $W = \sum_{c \in \mathcal{C}} W_c$

- označme k počet literálov v klauzule c
- klauzula je nesplnená práve ak všetky jej literály sú *false*, tj. s pravdeposobnosťou 2^{-k}
- očakávaná hodnota náhodnej premennej W_c je $E[W_c] = w_c Pr[c \text{ je splnená}] + 0 Pr[c \text{ nie je splnená}] = w_c(1 - 2^{-k})$
- pre $k \geq 1$ platí $1 - 2^{-k} \geq 1/2$ a preto

$$E[W] = \sum_{c \in \mathcal{C}} E[W_c] \geq \frac{1}{2} \sum_{c \in \mathcal{C}} w_c \geq \frac{1}{2} OPT$$

kde OPT je hodnota optimálneho riešenia

MAX-SAT A ÚLOHA LINEÁRNEHO PROGRAMOVANIA

- pre konštruckiu náhodnostného algoritmu pre MAX-SAT využijeme redukciu na úlohu lineárneho programovania
- pre každú klauzulu $c \in \mathcal{C}$ označme S_c^+ (S_c^-) množinu tých literálov, ktoré sa v c vyskytujú v pozitívnom tvare (s negáciou)
- každej premennej x_i booleovskej formule F zodpovedá premenná y_i úlohy lineárneho programovania
- každej klauzule c zodpovedá premenná z_c; hodnota $z_c = 1$ indikuje, že klauzula je splnená
- lineárne ohraničenia garantujú, že $z_c = 1$ len ak aspoň jeden literál klauzuly je splnený

redukcia MAX-SAT na úlohu LP

maximalizovať $$\sum_{c \in \mathcal{C}} w_c z_c$$

pri splnení $$\sum_{i \in S_c^+} y_i + \sum_{i \in S_c^-} (1 - y_i) \geq z_c \qquad c \in \mathcal{C}$$

$$z_c \in \{0, 1\} \qquad c \in \mathcal{C}$$

$$y_i \in \{0, 1\} \qquad i = 1, \ldots, n$$

relaxovaná úloha LP

maximalizovať $$\sum_{c \in \mathcal{C}} w_c z_c$$

pri splnení $$\sum_{i \in S_c^+} y_i + \sum_{i \in S_c^-} (1 - y_i) \geq z_c \qquad c \in \mathcal{C}$$

$$0 \leq z_c \leq 1 \qquad c \in \mathcal{C}$$

$$0 \leq y_i \leq 1 \qquad i = 1, \ldots, n$$

- vyriešime relaxovanú úlohu LP
- nech $(y_1, \ldots, y_n, z_1, \ldots, z_m)$ je optimálne riešenie
- booleovskej premennej x_i priradíme hodnotu *true* s pravdepodobnosťou y_i
- kvalitu nájdeného riešenia vyjadríme pomocou náhodných premenných W a W_c

Ak klauzula c má k literálov, tak

$$E[W_c] = (1 - (1 - \frac{1}{k})^k) w_c z_c \ .$$

- búno $c = (x_1 \vee \ldots \vee x_k)$
- c je splnená práve ak nie všetky jej literály majú hodnotu *false*; pravdepodobnosť takejto udalosti je

$$\begin{aligned} 1 - \prod_{i=1}^{k}(1 - y_i) &\geq 1 - \left(\frac{\sum_{i=1}^{k}(1-y_i)}{k}\right)^k = 1 - \left(1 - \frac{\sum_{i=1}^{k} y_i}{k}\right)^k \\ &\geq 1 - \left(1 - \frac{z_c}{k}\right)^k \qquad z_c \in [0,1] \\ &\geq 1 - (1 - \tfrac{1}{k})^k z_c \end{aligned}$$

Ak všetky klauzule majú veľkosť nanajvýš k, tak $E[W] \geq (1 - (1 - \frac{1}{k})^k)\mathrm{OPT}$

- označme k_c veľkosť klauzule c a OPT_f hodnotu optimálneho riešenia relaxovanej úlohy LP
- využijeme fakt, že funkcia $(1 - (1 - \frac{1}{k})^k)$ je klesajúca
- $$\begin{aligned} E[W] &= \sum_{c \in \mathcal{C}} E[W_c] \\ &\geq \sum_{c \in \mathcal{C}} (1 - (1 - \tfrac{1}{k_c})^{k_c}) w_c z_c \\ &\geq (1 - (1 - \tfrac{1}{k})^k) \sum_{c \in \mathcal{C}} w_c z_c \\ &\geq (1 - (1 - \tfrac{1}{k})^k)\mathrm{OPT}_f \geq (1 - (1 - \tfrac{1}{k})^k)\mathrm{OPT} \end{aligned}$$

Algoritmus založený na redukcii na úlohu LP je randomizovaným $(1 - 1/e)$ aproximatívnym algoritmom pre MAX-SAT.

KOMBINOVANÝ NÁHODNOSTNÝ ALGORITMUS

- pozorovanie
  - algoritmus, ktorý náhodne priraďuje hodnoty premenným, sa chová dobre pre formule s veľkými klauzulami
  - naopak, algoritmus založený na redukcii na úlohu LP, sa chová dobre pre formule s malými klauzulami
- otázka
  - ako postupovať pre *obecnú* formulu?

kombinovaný algoritmus

náhodne, s pravdepodobnosťou 1/2, vyber jeden z uvažovaných algoritmov

Pre kombinovaný algoritmus platí $E[W] \geq \frac{3}{4}\mathbf{OPT}$.

- $\mathcal{A}_1$ - algoritmus náhodného výberu
  $\mathcal{A}_2$ - algoritmus redukcie na úlohu LP
- dokázali sme, že ak klauzula c má veľkosť k, tak

$$\begin{aligned} E[W_c|\mathcal{A}_1] &= (1-2^{-k})w_c \\ E[W_c|\mathcal{A}_2] &\geq (1-\frac{1}{k})^k)w_c z_c \end{aligned}$$

- pre kombinovaný algoritmus platí

$$\begin{aligned} E[W_c] &= \frac{1}{2}(E[W_c|\mathcal{A}_1]) + \frac{1}{2}E[W_c|\mathcal{A}_2]) \\ &\geq \frac{(1-2^{-k}+(1-\frac{1}{k})^k))}{2} w_c z_c \geq \frac{3}{4} w_c z_c \end{aligned}$$

$$E[W] = \sum_{c\in\mathcal{C}} E[W_c] \geq \frac{3}{4}\sum_{c\in\mathcal{C}} w_c z_c = \frac{3}{4}\mathrm{OPT}_f \geq \frac{3}{4}\mathrm{OPT}$$

DERANDOMIZÁCIA KOMBINOVANÉHO ALGORITMU

deterministický algoritmus

1. použi derandomizovaný 1/2 aproximatívny algoritmus pre MAX-SAT
2. použi derandomizovaný 1 - 1/e aproximatívny algoritmus pre MAX-SAT
3. výstupom je lepšie z vypočítaných priradení

jedno z vypočítaných riešení musí mať cenu aspoň $E[W] \geq \frac{3}{4}\text{OPT}$
a preto deterministický algoritmus 3/4-aproximatívnym algoritmom pre MAX-SAT

tesné príklady pre 3/4-aproximatívny algoritmus pre MAX-SAT

- $f = (x_1 \vee x_2) \wedge (\neg x_1 \vee x_2) \wedge (x_1 \vee \neg x_2) \wedge (\neg x_1 \vee \neg x_2)$
  s jednotkovou váhou klauzulí (OPT=3 a $\mathrm{OPT}_f = 4$)

- $f = (x \vee y) \wedge (x \vee \neg y) \wedge (\neg a \vee z)$ s váhami klauzulí 1, 1 a $2 + \varepsilon$