Univerzálne triedy hashovacích funkcií

Veronika Krajčová, Martin Kvočka, Marek Trgiňa

20.5.2010

1 Úvod

Rôzne vstupy pre program môžeme vnímať ako prvky z triedy problémov, pričom výstupom programu je správne riešenie daného problému. Ak hovoríme o priemernom výkone programu, spravidla priemerujeme jeho výsledky cez triedu problémov, ktoré program rieši. Páni Gill, Rabin, Strassen a Solovay na niektorých triedach problémov použili iný prístup. Navrhli, aby si program náhodne vybral algoritmus z triedy algoritmov, ktorým bude problém riešiť. Týmto spôsobom boli schopný ohraničiť priemerný výkon triedy algoritmov pre najhorší možný vstup. Tento priemer na najhoršom vstupe môže byť lepší než výkon ľubovoľného konkrétneho algoritmu na tomto vstupe. Tento prístup prekonáva niektoré problémy, ako:

1. Klasická analýza (priemerovanie cez triedu vstupov) musí počítať s distribúciou vstupov. Tieto predpoklady však v určitých prípadoch nemusia platiť.

2. Dôsledkom (1) je že nemôžme klasicky analyzovať priemerný výkon podprocedúry nezávisle na hlavnej časti programu, pretože hlavná časť môže rozhodiť distribúciu dát.

3. Ak programu predložíme najhorší možný vstup, tak neexistuje spôsob ako sa vyhnúť jeho slabej výkonnosti. Avšak, pri použití algoritmu z triedy algoritmov by sme mohli detekovať, že je pomalý na danom vstupe, a zvoliť iný algoritmus.

V nasledujúcom texte ukážeme použitie tohto prístupu pri hashovaní. Ukážeme že ak je trieda funkcií zvolená správne, potom priemerný výkon programu na ľubovoľnom vstupe bude aspon taký dobrý ako jednotlivá funkcia, zvolená s vedomím konkrétneho vstupu. Taktiež predvedieme niektoré triedy hashovacích funkcií ktoré zaručia, že každá vzorka z priestoru vstupov bude rovnomerne distribuovaná dostatočným počtom funkcií tak aby kompenzovala slabý výkon algoritmu v prípade neštastnej volby funkcie.

2 Univerzálne triedy hash funkcií

Značenie:
Všetky hashovacie funkcie budú mapovat' množinu A do B, pričom vždy bude $|A| > |B|$. A budeme nazývat' množinou kľúčov, B zas množinou indexov. Nech f je hash funkcia a $\delta_f(x, y)$ je 1 ak $x \neq y$ a $f(x) = f(y)$, inak 0. Teda $\delta_f(x, y)$ je 1 ak x a y sú rôzne prvky A ktoré sa mapujú na rovnakú hodnotu pomocou funkcie f. Ak f, x, y nahradíme množinou, potom sčítavame cez všetky prvky v množine. Teda, ak H je kolekcia hash funkcií, $x \in A$ a $S \subset A$, potom $\delta_H(x, S)$ znamená:

$$\Sigma_{f \in H} \Sigma_{y \in S} \delta_f(x, y)$$

Vlastnosti univerzálnych tried:
Nech H je trieda funkcií z A do B. O H vravíme že je $universal_2$ ak pre všetky x, y z A, $\delta_H(x, y) \leq \frac{|H|}{|B|}$. To znamená, že H je $universal_2$ ak žiaden pár rôznych kľúčov nie je mapovaný na tie isté indexy viac než jedno $|B|$-tinou funkcií.

Tvrdenie 1:
Pre každú kolekciu hash funkcií (nie nutne $universal_2$) existuje $x, y \in A$ také že

$$\delta_H(x, y) > \frac{|H|}{|B|} - \frac{|H|}{|A|}$$

Tvrdenie 2:
Nech x je ľubovoľný prvok A a S podmnožina A. Nech f je funkcia zvolená náhodne z $universal_2$ triedy funkcií (s rovnakou pravdepodobnost'ou). Potom očakávaný počet prvkov z S s ktorými x koliduje, teda $\delta_f(x, S)$, je $\leq \frac{|S|}{|B|}$.

Zaujíma nás použitie týchto funkcii pri operáciach ukladania a získavania dát. Ak máme sekvenciu R požiadaviek (vloženie alebo výber) do nejakej databáze, a hash funkciu f, potom definujeme cenu R vzhľadom na f ako $C(f, R)$, teda ako sumu cien jednotlivých požiadaviek. Cena jednotlivej požiadavky zodpovedajúcej prvku x je jedna plus počet rôznych predtým vložených y pre ktoré $f(x) = f(y)$.

Táto funkcia ceny odráža najhoršiu cenu vloženia alebo nájdenia prvkov v úložisti a návratovú schému v ktorej každý prvok B je asociovaný so spojovaným zoznamom, a prvok x je uložený v tomto zozname asociovaný s $f(x)$.

Nasledujúci teorém dáva pekný odhad na očakávanú cenu pomocou $universal_2$ tried hashovacích funkcií používajúcich spojovaný zoznam na riešenie kolízií.

Tvrdenie 3:
Nech R je sekvencia r požiadaviek ktoré zahŕňajú k vložení. Ďalej nech H je $universal_2$ trieda hashovacích funkcií. Potom ak zvolíme f náhodne z H, očakávaná cena je:

$$C(f, R) \leq r(1 + \frac{k}{|B|})$$

Špeciálny prípad tohto tvrdenia je ak veľkost'k je zhruba rovnaká B, potom je očakávaná cena $2r$. Všimnite si, že toto lineárne ohraničenie platí pre ľubovoľnú sekvenciu požiadaviek, nie len pre "priemernú" požiadavku. Vo väčšine aplikácií však existuje horná hranica na počet prvkov ktoré je možno uložiť, teda B može byt' vhodne zvolené. Ak žiadna horná hranica neexistuje, môžme jej veľkost' volit' dynamicky a znovu prehashovat' ak sa voľba ukáže ako príliš malá. Rehashovanie totiž môžme urobit' v lineárnom čase, niekedy aj real-time.

Môžme ukázat', že očakávaná cena (spriemerovaná cez hashovacie funkcie) ľubovoľnej požiadavky je prakticky rovnaká ako očakaváná cena (spriemerovaná cez možné požiadavky) ľubovoľnej jednotlivej hash funkcie aplikovanej na nahodnú požiadavku po náhodných vloženiach. Dôvod je nasledovný: Nech $a = |A|$ a $b = |B|$. Argument z tvrdenia 1 implikuje že ak je f lubovoľná hash funkcia a x a y sú zvolené náhodne z A, tak pravdepodobnost' $\delta_f(x, y)$ je $\geq (1/b - 1/a)$. Potom teda pravdepodobnost' $\delta_f(x, S)$ je $\geq |S|(1/b - 1/a)$, kde S je náhodná podmnožina A ktorá bola predtým uložená. Teda cena požiadavky je aspoň $1 + |S|(1/b - 1/a)$.

Taktiež je možné ohranicit' pravdepodobnost' že pre danú sekvenciu

požiadaviek R je výkon náhodne zvolenej funkcie horší než tolerovateľný na R. Kedže vieme že $C(f, R)$ musí byt'aspoň r, môžme predpokladat'že ak k je zhruba rovnaké ako $|B|$, pravdepodobnost'že $C(f, R) > t - r$ je menej ako $1/(t-1)$. Pre niektoré triedy hashovacích funkcii je možné vyvodit' odhad na štandardnú odchýlku alebo vyššie momenty ceny náhodne zvolenej funkcie na konkrétnom R. Toto nám umožní získat'oveľa lepší odhad toho že pravdepodobnost' $C(f, R)$ bude veľká.

3 Niektoré $universal_2$ triedy

Prvá trieda $universal_2$ hashovacích funkcií ktorú uvedieme je vhodná na použitie v situáciách, keď bitové reťazce reprezentujúce klúče je možné jednoducho násobit'počítačom. Zoberme $A = 0, 1, ..., a-1$ a $B = 0, 1, \cdots, b-1$. Nech p je prvočíslo také že $p \geq a$. Nech g je nejaká funkcia zo Z_p do B, ktorá čo najpresnejšie mapuje rovnaký počet prvkov Z_p na každý prvok z B. Formálne požadujeme $|\{y \in Z_p | g(y) = z\}| \leq \lceil \frac{p}{q} \rceil$ pre všetky $z \in B$. Prirodzená voľba pre g je zvyšok modulo b. Ak $b = 2^k$ pre nejaké k, toto predstavuje posledných k bitov v binárnej reprezentácii y.

Nech m a n sú prvky Z_p kde $m \neq 0$. Definujeme $h_{m,n} : A \to Z_p$ ako $h_{m,n}(x) = (mx + n) mod p$. Potom definujeme $f_{m,n}(x) = g(h_{m,n}(x))$. Trieda H je množina $\{f_{m,n} | m, n \; in Z_p m \neq 0\}$. Takáto trieda H je $universal_2$.

Algoritmy sú často analyzované za predpokladu že násobenie zaberie jednotku času a počet násobení býva považovaný za cenu algoritmu. Tento model je vhodný v prípadoch, keď neexistujú operácie ktoré by bolo možné vykonat'neohraničený počet krát za každé násobenie. Keď je použítá trieda $universal_2$ hashovacích funkcií, počet odkazov do pamäte za každú požiadavku je možné ohraničit' spriemerovaním nad všetkými funkciami v triede ako v tvrdení (3). Takže tento model je vhodný a hashovacie funkcie v triede uvedenej vyššie je v ňom možné vykonat' za koštantný čas. Takže v tomto modeli, pre ľubovoľnú sekvenciu požiadaviek má algoritmus lineárnu časovú zložitost' vzľadom k počtu požiadaviek.

Trieda $universal_2$ popísaná vyššie nie je vhodná v prípadoch keď sú klúče príliš dlhé na to, aby ich bolo možné násobit' použítím jednej inštrukcie. Nasledujúce tvrdenie dáva spôsob akým rozšírit' triedu funkcií pre dlhé klúče.

Tvrdenie 4:

Predpokladajme že $|B|$ je mocninou 2 a H je trieda funkcií z A do B taká, že pre každé $i \in B$: $|\{f \in H | f(x) \oplus f(y) = i\}| = \frac{|H|}{|A|}$, kde $\oplus$ značí operáciu XOR. Potom môžme definovať *universal*$_2$ triedu hashovacích funkcií z AxA do B takto. Pre $f, g \in H$ definujeme $H_{f,g}((x,y)) = f(x) \oplus g(y)$. Potom táto nová trieda hashovacích funkcií $J = \{h_{f,g} | f, g \in H\}$ je *universal*$_2$ a zároveň spĺňa podmienky v tomto tvrdení. Toto tvrdenie sa úplne nevzťahuje na *universal*$_2$ triedu funkcií, ktorá bola uvedená vyššie, a to preto že H nie je mocninou 2 a preto že počet funkcií pre ktoré platí $f(x) \oplus f(y) = 0, i.e. \delta_H(x,y)$, je v skutočnosti menší ako $|H|/|B|$.

Nasleduje trieda funkcií, ktoré nevyžadujú násobenie, čo môže byť výhoda v rôznych aplikáciách. Zoberme si A ako množinu i-ciferných čísel o základe α a B ako množinu binárnych čísel dĺžky j. Potom $|A| = \alpha^i a |B| = 2^j$. Nech M je trieda polí dĺžky $i\alpha$, ktorých prvky sú binárne reťazce dĺžky j. Pre $m \in M$, nech $m(k)$ je binárny reťazec ktorý je k-ty prvok m, a pre $x \in A$, nech x_k je k-ta číslica x v bázi α. Definujeme

$$f_m(x) = m(x_1 + 1) \oplus m(x_1 + x_2 + 2) + \cdots + (m(sum_{k=1}^{i} x_k + k)$$

Trieda H je množina $\{f_m | m \in M\}$. Táto trieda spĺňa vlastnosti *universal*$_2$. Pre dané B, každá funckia v H má lineárnu časovú zložitosť vzhľadom k dĺžke klúča.

4 Záver

Programátori často strávia dlhú dobu pri tom, aby odladili hashovacie funkcie pre aplikácie kde je kritické dosiahnutie uniformného rozloženia. Dosiahnutie tohto ciela môže byť zložité, pretože je potrebné aby očakávaná vstupná množina bola usporiadaná takým spôosobom, že to nespôsobí nedostatočný výkon hashovacej funkcie. Jednou z praktických vlastností triedy *universal*$_2$ funkcií je, že sa v triede nachádza mnoho použiteľných funkcií. Jednoduchým náhodným výberom hashovacej funkcie z tejto triedy je možné dosiahnuť uniformné rozloženie. A tým, že sa použitá funkcia mení pri každom vykonaní programu, je možné dosiahnuť dobrý výkon v priemernom prípade. Použitie *universal*$_2$ tried dáva možnosť dobrého odhadu medzí, v ktorých sa bude pohybovať priemerný výkon algoritmu používajúceho hashovanie.

Literatúra

[1] DILIP V. SARWATE. *A Note on Universal Classes of Hash Functions.*. [cited 2010-5-14]. Available at: journal Information Processing Letters, vol. 10

[2] GEORGE MARKOWSKY, J.LAWRENCE CARTER, MARK N. WEGMAN. *Analysis of universal class of hash functions* [online]. [cited 2010-5-14]. Available at: http://laptops.maine.edu/HashFunctions.pdf.

[3] MARTIN DIETZFELBINGER AND FRIEDHELM MEYER. *A new universal class of hash functions and dynamic hashing in real time* . [cited 2010-5-14]. Available at: Lecture Notes in Computer Science, vol. 443/1990.