1 Oznaczenia

Notacja $\mathbf{1}_{\mathcal{E}}$, gdzie $\mathcal{E}$ jest dowolnym zdarzeniem będzie oznaczać indykator zdarzenia $\mathcal{E}$, czyli zmienną losową która przyjmuje wartość 1 gdy $\mathcal{E}$ zaszło i 0 w przeciwnym przypadku (formalnie, $\mathbf{1}_{\mathcal{E}}(\omega) = [\omega \in \mathcal{E}]$).

Z reguły w sytuacjach omawianych w tym tekście mamy do czynienia z jedną funkcją haszującą h (być może wybraną w jakiś sposób z jakiejś rodziny funkcji haszujących). Jeśli $h(x) = i$ (lub $h(x) \in A$) to powiemy, że x *haszuje* do i (lub do A).

2 Haszowanie Liniowe

W tym rozdziale przedstawimy kolejne bardzo naturalne rozwiązanie problemu dynamicznego słownika oparte na haszowaniu. Liczby całkowite ze zbioru S reprezentowanego przez nasz słownik będą przechowywane bezpośrednio w tablicy $T[0..m-1]$. Rozmiar T będzie większy niż $n = |S|$, choć liniowy względem n. Zwykle w zastosowaniach przyjmuje się $m = 2n$, do udowodnienia asymptotycznych oszacowań na czasy działania operacji wystarczy $m = (1+\varepsilon)\cdot n$, natomiast my dla ustalenia uwagi przyjmiemy $m = 3n$. Algorytmy operacji słownikowych będą korzystały z pojedynczej funkcji haszującej $h : U \to \{0..m-1\}$.

Algorytmy poszczególnych operacji są następujące:

insert(x): Wstawia x w pierwsze wolne miejsce w ciągu kolejnych liczb $h(x), h(x)+1, h(x)+2, h(x)+3, \ldots$ (oczywiście liczymy modulo m).

lookup(x): Sprawdza komórki T o indeksach $h(x), h(x)+1, h(x)+2, \ldots$, aż do znalezienia x (wtedy zwraca True) lub komórki pustej (wtedy zwraca False).

delete(x): wyszukuje x tak jak opisano powyżej — powiedzmy, że $T[p] = x$. Zamiast x wstawiamy znak pusty $\perp$. Nie możemy jednak tak zostawić tablicy T, ponieważ w spójnym bloku niepustych komórek zaczynającym się od pozycji $p+1$ może znajdować się element (lub elementy) y taki, że $h(y) \leq p$; wówczas wynik operacji lookup(y) byłby niepoprawny. W tym celu znajdujemy pierwszy taki element $y = T[r]$, wstawiamy go w $T[p]$, i czyścimy jego komórkę, tzn. $T[r] \leftarrow \perp$. Oczywiście tę operację powtarzamy tak długo, jak występuje opisana powyżej nieporządana sytuacja. Dla jasności poniżej podajemy pseudokod.

procedure delete(x):
  $p \leftarrow h(x)$
  while $(T[p] \neq x)$ **and** $T[p] \neq \perp$ **do**
    $p \leftarrow p + 1$
  $T[p] \leftarrow \perp$
  fill(p)

procedure fill(p):
  $r \leftarrow p + 1$

while $(T[r] \neq \perp)$ **and** $h(T[r]) > p$ **do**
    $r \leftarrow r + 1$
if $(T[r] \neq \perp)$ **then**
    $T[p] \leftarrow T[r]$
    $T[r] \leftarrow \perp$
    fill(r)

Poprawność opisanych algorytmów powinna być dość jasna (w zasadzie tylko w przypadku delete jest się nad czym zastanawiać). *Spójną składową* tablicy T będziemy nazywać dowolny maksymalny ciąg kolejnych niepustych komórek tablicy T.

Pamiętamy, że przy haszowaniu przez łańcuchowanie, jeśli h jest wybrana losowo z rodziny uniwersalnej, to oczekiwane czasy operacji słownikowych są stałe. Od czasu uzyskania tego wyniku przez Cartera i Wegmana w 1977r. pozostawało otwartym pytanie, czy można uzyskać analogiczny wynik dla haszowania liniowego, używając rodziny uniwersalnej lub jakiejś innej rodziny funkcji haszujących. Dopiero w 2007 odpowiedzi udzielili Rasmus Pagh, Anna Pagh i Milan Ruzic.

Twierdzenie 1 ([1]). *Jeśli w haszowaniu liniowym h jest wybrana losowo z uniwersalnej rodziny funkcji haszujących Cartera i Wegmana, to istnieje $S \subset U$ taki, że całkowity oczekiwany czas wstawienia wszystkich elementów z S wynosi $\Omega(n \log n)$.*

Równocześnie pokazali oni także następujący wynik pozytywny:

Twierdzenie 2 ([1]). *Jeśli w haszowaniu liniowym h jest wybrana losowo z dowolnej 5-niezależnej rodziny funkcji haszujących, to oczekiwany czas operacji słownikowych jest stały, a dokładniej wynosi $O((1-\alpha)^{-7/6})$, gdzie $\alpha = n/m$.*

Zacznijmy od tego, że wynik $O((1-\alpha)^{-7/6})$ jest naprawdę dobry, gdyż $O((1-\alpha)^{-1})$ to oczekiwany czas jaki dostajemy, gdy zamiast szukać wolnego miejsca na x używając kolejnych pozycji w tablicy T, korzystamy z w pełni losowej permutacji pozycji (patrz np. podręcznik Cormena i innych), co jest bardzo wyidealizowanym założeniem. Twierdzenie 2 jest dość zaskakujące, gdyż w świetle twierdzenia 1, które „z grubsza" mówi, że 2-niezależność nie wystarcza, mogłoby się wydawać, że również $O(1)$-niezależność nie jest wystarczająca. Dodatkowo dziwi tu liczba 5, na pierwszy rzut oka wygląda to na niedokładne szacowanie, nieoptymalny wynik. Nic bardziej błędnego, gdyż w 2010r. Patrascu i Thorup wykazali, że

Twierdzenie 3 ([2]). *Istnieje 4-niezależna rodzina funkcji haszujących $\mathcal{H}$ taka, że jeśli w haszowaniu liniowym h jest wybrana losowo z $\mathcal{H}$, to istnieje $S \subset U$ taki, że całkowity oczekiwany czas wstawienia wszystkich elementów z S wynosi $\Omega(n \log n)$.*

W dalszej części udowodnimy uproszczoną wersję twierdzenia 2. (Dowód jest zainspirowany blogiem Mihai Patrascu.)

Twierdzenie 4. *Jeśli w haszowaniu liniowym h jest wybrana losowo z dowolnej 5-niezależnej rodziny funkcji haszujących $\mathcal{H}$, oraz $m \geq 3n$, to oczekiwany czas operacji słownikowych jest stały.*

Zauważmy, że przyjęliśmy $\alpha \leq \frac{1}{3}$. Niech $cc(p)$ oznacza spójną składową zawierającą komórkę p, natomiast $|cc(p)|$ jej długość. Łatwo widać, że prawdziwy jest następujący lemat.

Lemat 1. *Czas działania każdej z operacji insert(x), lookup(x) oraz delete(x) można oszacować przez $O(|cc(h(x))|)$.*

Weźmy dowolne $x \in U$. Zgodnie z lematem 1 wystarczy, że oszacujemy przez pewną stałą oczekiwaną długość spójnej składowej zawierającej pozycję $h(x)$. Oczywiście zawsze istnieje takie k, że $|cc(h(x))| \in \{2^k, \ldots, 2^{k+1}-1\}$. Podzielmy komórki T na równe bloki długości 2^{k-2}.

Lemat 2. $\mathbb{E}[|h(S) \cap B|] = \alpha|B| = \frac{1}{3}|B|$

Dowód. Korzystamy z liniowości wartości oczekiwanej i jednostajności $\mathcal{H}$, czyli faktu, że $\mathbb{P}[h(e) = b] = 1/m$ dla dowolnych $e \in U$, $b \in [m]$. Zauważmy, że $\mathbb{E}[|h(S) \cap B|] = \sum_{e \in S} \mathbb{E}[\mathbf{1}_{h(e) \in B}] = \sum_{e \in S} \mathbb{P}[h(e) \in B] = \sum_{e \in S} \sum_{b \in B} \mathbb{P}[h(e) = b] = n \cdot |B| \cdot \frac{1}{m} = \alpha|B| = \frac{1}{3}|B|$. □

Powiemy, że blok $B = \{i, i+1, \ldots, i+2^{k-2}-1\}$ jest *niebezpieczny*, gdy $|h(S) \cap B| \geq \frac{2}{3}|B|$. Uwaga! Zauważmy, że nie liczymy tu ile komórek B jest zajętych, a jedynie do ilu komórek B haszują się elementy S.

Lemat 3. *Jeśli $h(x)$ jest w spójnej składowej długości $\in \{2^k, \ldots, 2^{k+1}-1\}$ to jeden z $O(1)$ bloków przecinających $cc(h(x))$ jest niebezpieczny.*

Dowód. Oznaczmy kolejne bloki przecinające $cc(h(x))$ przez $B_1, \ldots, B_k$. Zauważmy, że $4 \leq k \leq 9$. Załóżmy przeciwnie, że wszystkie te bloki są bezpieczne. W szczególności oznacza to, że mniej niż $\frac{2}{3}2^{k-2}$ elementów haszujących do B_1 znajduje się w kolejnych blokach. W blokach B_2 i B_3 jest sumarycznie więcej niż $2 \cdot \frac{1}{3}|B|$ komórek, które nie zawierają elementów haszujących do B_2 i B_3. To oznacza, że nie wszystkie z tych komórek zostaną zajęte przez elementy haszujące do B_1, a więc co najmniej jedna komórka pozostanie pusta (gdyż inne elementy nie mogą tam się pojawić). W takim razie $k \leq 3$, sprzeczność. □

Załóżmy teraz, że znamy wartość $\rho = h(x)$ i chcemy oszacować prawdopodobieństwo, że $|cc(\rho)| \in \{2^k, \ldots, 2^{k+1}-1\}$. Ponieważ $cc(\rho)$ przecina co najwyżej 9 bloków, to jest co najwyżej 17 bloków $B_1, \ldots, B_k$, które potencjalnie mogą przecinać się z $cc(\rho)$. Z lematu 3 wiemy, że jeśli $|cc(\rho)| \in \{2^k, \ldots, 2^{k+1}-1\}$ to jeden z tych bloków jest niebezpieczny. Stąd,

$$\mathbb{P}[|cc(\rho)| \in \{2^k, \ldots, 2^{k+1}-1\} \mid h(x) = \rho] \leq \sum_{i=1}^{k} \mathbb{P}[|h(S) \cap B_i| \geq \tfrac{2}{3}|B_i| \mid h(x) = \rho].$$

Z symetrii i lematu 2 mamy dalej

$$\begin{aligned} &\mathbb{P}[|cc(\rho)| \in \{2^k, \ldots, 2^{k+1}-1\} \mid h(x) = \rho] = \\ &O(1) \cdot \mathbb{P}[|h(S) \cap B| \geq \mathbb{E}(|h(S) \cap B|) + \tfrac{1}{3}|B| \mid h(x) = \rho], \end{aligned} \tag{1}$$

gdzie B jest pewnym konkretnym blokiem długości 2^{k-2}.

Od tej pory, dla uproszczenia zakładamy, że wszsytko jest warunkowane przez zdarzenie $h(x) = \rho$ i będziemy pomijać w prawdopodobieństwach (i wartościach oczekiwanych) napis „$|h(x) = \rho$". Zauważmy, że przy tym warunkowaniu, rodzina $\mathcal{H}$ jest 4-niezależna.

Niech $X_e = \mathbf{1}_{h(e)\in B}$ oraz $X = \sum_{e\in S} X_e$. Przy takiej notacji, chodzi nam o to, żeby oszacować $\mathbb{P}[X > 2\mathbb{E}(X)]$. Narzuca się, żeby w celu oszacowania powyższego prawdopodobieństwa użyć nierówności Chernoffa, problem polega jednak na tym, że zmienne X_e nie są niezależne (a jedynie 4-niezależne). Z nierówności Markowa dostajemy szacowanie przez $\frac{1}{2}$, lecz jak się później okaże jest ono o wiele za słabe. Kolejny pomysł to użycie nierówności Czebyszewa – dałaby ona lepsze oszacowanie, lecz w dalszym ciągu niewystarczające. Okazuje się, że wystarczy zrobić jeden krok dalej – mianowicie użyć następującego faktu:

Lemat 4 (Nierówność czwartego momentu)**.** $\mathbb{P}[|X - \mathbb{E}X| > d] \leq \frac{\mathbb{E}[(X-\mathbb{E}X)^4]}{d^4}$.

Dowód. Dowodzimy tak samo jak nierówność Czebyszewa, tylko podnosimy do 4-tej potęgi:

$$\mathbb{P}[|X - \mathbb{E}X| > d] = \mathbb{P}[(X - \mathbb{E}X)^4 > d^4] \leq \frac{\mathbb{E}[(X - \mathbb{E}X)^4]}{d^4},$$

gdzie ostatnia nierówność wynika z nierówności Markowa. □

Pozostaje jedynie oszacować czwarty moment, czyli $\mathbb{E}[(X - \mathbb{E}X)^4]$. Oznaczmy

$$Y_e = X_e - \mathbb{E}X_e \quad \text{oraz} \quad Y = \sum_{e\in S} Y_e.$$

Wówczas $X - \mathbb{E}X = Y$ i interesuje nas $\mathbb{E}(Y^4)$. Mamy:

$$\mathbb{E}[Y^4] = \mathbb{E}\left[\left(\sum_{e\in S} Y_e\right)^4\right] = \sum_{e_1,e_2,e_3,e_4\in S} \mathbb{E}(Y_{e_1}Y_{e_2}Y_{e_3}Y_{e_4}).$$

Zauważmy, że zmienne Y_e są rownież 4-niezależne (gdyż X_e są takie). Stąd, jeśli w ostatniej sumie $e_1, \ldots, e_4$ są parami różne, to zmienne $Y_{e_1}, Y_{e_2}, Y_{e_3}, Y_{e_4}$ są niezależne, czyli $\mathbb{E}(Y_{e_1}Y_{e_2}Y_{e_3}Y_{e_4}) = \mathbb{E}Y_{e_1}\mathbb{E}Y_{e_2}\mathbb{E}Y_{e_3}\mathbb{E}Y_{e_4} = 0$, gdzie ostatnia równość bierze się stąd, że $\mathbb{E}Y_e = 0$. Ogólniej, jeśli $e_1 \notin \{e_2, e_3, e_4\}$ to dwie zmienne Y_{e_1} i $Y_{e_2}Y_{e_3}Y_{e_4}$ są niezależne i dostajemy $\mathbb{E}(Y_{e_1}Y_{e_2}Y_{e_3}Y_{e_4}) = \mathbb{E}Y_{e_1}\mathbb{E}[Y_{e_2}Y_{e_3}Y_{e_4}] = 0$. Stąd,

$$\mathbb{E}[Y^4] = \sum_{e\in S} \mathbb{E}[Y_e^4] + \sum_{e,f\in S; e<f} \binom{4}{2} \mathbb{E}[Y_e^2]\mathbb{E}[Y_f^2].$$

Dla dowolnego e i parzystego $j > 0$ mamy

$$\begin{aligned}\mathbb{E}[Y_e^j] &= \left(1 - \tfrac{|B|}{m}\right)^j \tfrac{|B|}{m} + \left(-\tfrac{|B|}{m}\right)^j \left(1 - \tfrac{|B|}{m}\right) = \left(1 - \tfrac{|B|}{m}\right)^j \tfrac{|B|}{m} + \left(\tfrac{|B|}{m}\right)^j \left(1 - \tfrac{|B|}{m}\right) = \\ &= \tfrac{|B|}{m}\left(1 - \tfrac{|B|}{m}\right)\left(\left(1 - \tfrac{|B|}{m}\right)^{j-1} + \left(\tfrac{|B|}{m}\right)^{j-1}\right) < \tfrac{|B|}{m}.\end{aligned}$$

Stąd, pamiętając o oznaczeniu $\alpha = n/m$,

$$\mathbb{E}[Y^4] < n\frac{|B|}{m} + 6\frac{n^2}{2}\left(\frac{|B|}{m}\right) = \alpha|B| + 3(\alpha|B|)^2 < 4(\alpha|B|)^2.$$

Stąd i z lematu 4 mamy, że

$$\mathbb{P}[X > 2\mathbb{E}X] = \mathbb{P}[X - \mathbb{E}X > \mathbb{E}X] < \frac{4(\alpha|B|)^2}{(\alpha|B|)^4} = \frac{4}{\alpha^4}|B|^{-2} = O(|B|^{-2}). \tag{2}$$

Stąd i z (1) mamy

$$\begin{aligned}
\mathbb{E}[|cc(\rho)| \mid h(x) = \rho] &= \sum_l l \cdot \mathbb{P}[|cc(\rho)| = l \mid h(x) = \rho] \\
&< \sum_k 2^{k+1} \cdot \mathbb{P}[|cc(\rho)| \in \{2^k, \ldots, 2^{k+1} - 1\} \mid h(x) = \rho] \\
&<^{(1),\,(2)} \sum_k 2^{k+1} \cdot O(1) \cdot (2^{k-2})^{-2} \\
&= O(1) \sum_k 2^{-k} \\
&= O(1).
\end{aligned}$$

Ponieważ jest to prawdziwe dla dowolnego $\rho \in [m]$, więc $\mathbb{E}[|cc(h(x))|] = O(1)$, a tym samym dowód twierdzenia 4 jest zakończony.

Literatura

[1] A. Pagh, R. Pagh, and M. Ruzic. Linear probing with constant independence. *SIAM J. Comput.*, 39(3):1107–1120, 2009.

[2] M. Patrascu and M. Thorup. On the -independence required by linear probing and minwise independence. In S. Abramsky, C. Gavoille, C. Kirchner, F. M. auf der Heide, and P. G. Spirakis, editors, *ICALP (1)*, volume 6198 of *Lecture Notes in Computer Science*, pages 715–726. Springer, 2010.