2015年12月7日(月)

Aux # 13

P1 Árboles α-balanceados

- $1/2 < \alpha < 1$

$$\begin{array}{ccc} & T & \\ \swarrow & & \searrow \\ T_l & & T_r \end{array} \Rightarrow \begin{array}{l} |T_l| \leq \alpha |T| \\ |T_r| \leq \alpha |T| \end{array}$$

- Al insertar un nodo, debo volver por el camino que bajé y verificar que cada subárbol sea α-balanceado. Si ese subárbol no está α-balanceado $\Rightarrow$ reconstruyo subárbol para que sea perfectamente balanceado.
- Costo de rebalancear T es $\Theta(|T|)$

1) Al bajar por el árbol, en cada descenso descarto al menos $(1-\alpha)$ de los elementos. (Suponemos que en cada descenso, en el peor caso bajo por el subárbol más grande, el que tiene α ellos) Esto sucede hasta quedarnos con 1 elemento.

$\Rightarrow$ cuando $\alpha^t n \leq 1$ (después de t pasos)

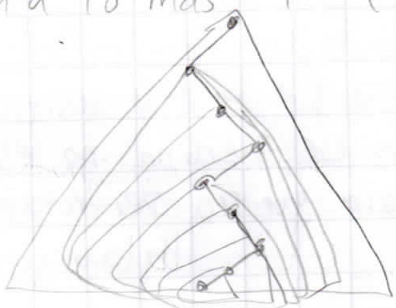
$$\Rightarrow \frac{1}{\alpha^t} \sim n$$

$$\Rightarrow \log n \sim t \log\left(\frac{1}{\alpha}\right)$$

$$\Rightarrow t \sim \frac{\log n}{\log(1/\alpha)} = \log_{(1/\alpha)} n \ldots$$

2) $\phi(T) = \frac{1}{2\alpha - 1} \sum_{T' \in T} \max(||T'_l| - |T'_r|| - 1, 0)$ representa cuán desvalanceada está la estructura, en cada punto

- Al insertar un elemento en un T' uno de los sumandos de ϕ se va a ver afectado. La diferencia entre $|T'_l|$ y $|T'_r|$ va a cambiar en a lo más 1 $(\max(||T'_l| - |T_r|| - 1, 0))$



Como inserto en $\theta(\log n)$ subárboles (largo del camino) el $\Delta\phi$ total es a lo más $\theta(\log n)$
$\Rightarrow$ puede cargarse este costo al descenso en el árbol.

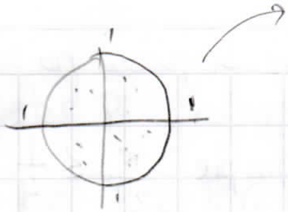
Para la reconstrucción, si rebalanceo T^*,
$\rightarrow \phi_f(T^*) = 0$ (para un árbol perfectamente balanceado, $\phi = 0$)
Por otro lado, antes de un rebalanceo,
$\phi_i = \theta(|T^*|)$ Pq?

$\rightarrow$ Sabemos que si T^* fue reconstruido, (s.p.g)
$|T_l| > \alpha |T| \Rightarrow |T_r| \leq (1-\alpha)|T|$

$\Rightarrow |T_l| - |T_r| \geq \alpha|T| - (1-\alpha)|T| = (2\alpha - 1)|T|$
$\Rightarrow \frac{1}{2\alpha - 1} \max(||T_l| - |T_r|| - 1, 0) \geq |T| - 1$
$\Rightarrow \phi_i(T) \geq |T| - 1 \Rightarrow \Delta\phi = -\theta(|T|)$

Luego esto puede pagar la reconstrucción $\Rightarrow$ reconstrucción es "gratis"

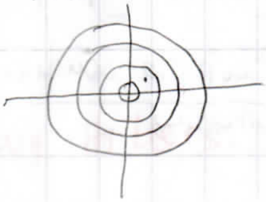
P2



r	x	y

A pesar de ser un dominio continuo, lo dividimos en "cajones" y trabajamos de manera discreta.

Idea: hacer buckets en forma de anillos concéntricos.

$\Rightarrow$  Luego, BucketSort.

Idealmente, en cada bucket queda 1 elemento (construyo tantos buckets como sea necesario para ello)

$\Rightarrow$ Usaremos n buckets con la <u>misma área</u> $= \frac{\pi}{n}$ (para aprovechar distr. uniforme)

$\Rightarrow$ La primera zona, la central, es un círculo de radio r_1

$\Rightarrow A_1 = \pi r_1^2 = \frac{\pi}{n} \Rightarrow r_1 = \frac{1}{\sqrt{n}}$

Las demás áreas son anillos  $\Rightarrow A_j = \pi r_j^2 - \pi r_{j-1}^2$

para $A_2 = \pi r_2^2 - \frac{\pi}{n} = \frac{\pi}{n} \Rightarrow r_2 = \sqrt{\frac{2}{n}}$

De hecho, $r_j = \sqrt{\frac{j}{n}}$ ¿

Luego el algoritmo es simplemente usar bucket sort con estos buckets $\Rightarrow \Theta(n)$ en <u>promedio</u> (en promedio habrá un punto por bucket).

P3 $|S| = n$ $S_i \subseteq S \ \forall i$, los S_i pueden no ser disjuntos

k cjtos $S_1, \dots, S_k$, $|S_i| = r$; se cumple $\frac{k}{2^r} \leq \frac{1}{4}$

1.- Pintaremos los elementos al azar (en forma indep y uniforme)
Esto toma $\Theta(n + \underbrace{kr}_{\text{verificar que se cumpla condición}})$

Para un S_i, sea $X_i := \begin{cases} 1 & \text{si } S_i \text{ es monocromático} \\ 0 & \sim \end{cases}$

Queremos (para una sol correcta), $\sum X_i = 0$.
Para un S_i
$$E(X_i) = \sum_{\text{rojo, azul}} \left(\frac{1}{2}\right)^r = \frac{1}{2^{r-1}}$$

$$\Rightarrow E(\sum X_i) = \sum E(X_i) = \frac{k}{2^{r-1}}$$

Ahora, por la desigualdad de Markov (sirve para pasar de $E(\cdot)$ a Pr con una cota)
$$E(X \geq a) \geq P(X > a)$$

$$\Rightarrow E(\sum X_i \geq 1) = \frac{k}{2^{r-1}}$$
$$\Rightarrow P(\sum X_i \geq 1) \leq \frac{k}{2^{r-1}} \leq \boxed{\frac{1}{2}}$$ enunciado, $\frac{k}{2^r} \leq \frac{1}{4}$

2.- OK, repito t veces verificando. Si alguna vez obtengo un coloreo válido, lo retorno
$$\Rightarrow P[\text{fallar } t \text{ veces}] = \prod_{i=1}^{t} [\text{fallar 1 vez}] = (1/2)^t$$

tiempo $\Theta(t(n + kr))$

3.- Repetir por siempre ($t \to \infty$) ← fallar $i-1$ veces ← achuntarle

$$\mathbb{E}(t) = \sum i \cdot \mathbb{P}(i) = \sum_{i=1}^{\infty} i \cdot p^{i-1}(1-p)$$

$$= \sum_{i=1}^{\infty} \frac{i}{2^i}$$ ← para resolver usar $\sum x \cdot \alpha^x = \alpha \frac{d}{d\alpha}\left(\sum \alpha^x\right)$

$\boxed{P4}$ Heavy Hitters.

Stream $A = a_1 \ldots a_m$, $a_i \in \{1 \ldots n\}$

Encontrar los i tq $f_i > \phi \cdot m$, para ϕ dado.

Me gustaría tener un contador para ca $a_i \in \{1, \ldots n\}$, pero n puede ser gigante! Uso "buckets" contadores donde a_i's #'s caen en el mismo bucket $\longrightarrow$ hash con colisiones. Usaremos varios hash

$\Rightarrow$ Algoritmo Count-Min

- Usamos t funciones de hash $h_i : n \to k$, con $k < n$
- También tenemos una tabla C:

$$10 \to \quad \begin{array}{c|ccccc}
h_1 & C_{11} & C_{12} & \cdots & \underset{++}{\boxed{10}} & \cdots \; C_{1k} \\
h_2 & C_{21} & C_{22} & \cdot & C{++}\;\boxed{10} & C_{2k} \\
\vdots & & \boxed{10}\; C{++} & & & \\
h_t & & & & & C_{tk}
\end{array}$$

Count-Min:

$C_{i,j} \leftarrow 0 \quad \forall i,j$

cuando llega un a_i,

for $j = 1 \ldots t$

$\quad C_{j, h_j(a_i)}$ ++

$$\Rightarrow \hat{f}_q = \min_{j \in 1..t} C_{i, h_j(q)}$$

$$C_{j, h_j(q)} \geq f_q$$

Claramente $\hat{f_q} \geq f_q$ } queremos acotar $\hat{f_q}$ en un rango aceptable
Ahora, $\hat{f_q} \leq f_q + \underset{\sim}{W}$

Definimos $Y_{i,j}$ = overcount en h_i debido a j.

$$Y_{i,j} = \begin{cases} f_j & \text{con prob } 1/k \\ 0 & \sim \end{cases}$$ ← pues h es buena fn. de hash

$\Rightarrow E[Y_{i,j}] = f_j / k$ lo que hace fallar 1 elemento al contador de q

$X_i = \sum_{j \neq q} Y_{i,j} \Rightarrow E[X_i] = \sum_{j \neq q} \frac{f_j}{k} = \frac{F_1}{k}$ ← cuanto falla la función h_i en total.

notación $\Rightarrow \frac{\varepsilon F_1}{2}$

(*) Luego $\mathbb{P}[X_i \geq \varepsilon F_1] \leq \frac{1}{2}$ (Markov)

Para que fallo el conjunto deben fallar todas por separado

$\mathbb{P}[\underset{\sim}{\hat{f_q}} - f_q \geq \varepsilon F_1]$ ← indep.

$= \mathbb{P}[\min X_i \geq \varepsilon F_1] = \mathbb{P}[\forall_{i \in 1..t} (X_i \geq \varepsilon F_1)]$

$= \prod_{i=1}^{t} \mathbb{P}[X_i \geq \varepsilon F_1]$ ← aquí ganamos por usar muchas fn. de hash $\neq$'s

$\overset{(*)}{=} \frac{1}{2^t}$

Definimos δ tq $t = \log \frac{1}{\delta} \Rightarrow \mathbb{P}[\text{fallar}] = \frac{1}{2^{\log \frac{1}{\delta}}} = \frac{1}{\frac{1}{\delta}} = \delta$

$\Rightarrow f_q \leq \hat{f_q} \leq \hat{f_q} + \varepsilon F_1$

con $\mathbb{P} = 1 - \delta$ al menos.

ε relacionado con k
δ relacionado con t.

Falta análisis de espacio