QuickSort, parte 2

Complejidad del QuickSort

- ¿Se puede utilizar el método maestro? ¿Si, No, Por qué?

- ¿Qué pasa en el mejor de los casos?

  *O(N*log(N))*

- ¿Qué pasa en el peor de los casos?

  $O(N^2)$

Ventajas del QuickSort

- Es muy utilizado en la práctica incluyendo en las librerías de muchos lenguajes de programación.

- Es elegante y relativamente simple a la hora de implementarlo. Es de tipo *“divide & conquer”* pero no requiere etapa de combinación.

- Con la alternativa de no usar arreglos adicionales trabaja *in situ*, es decir que necesita un mínimo de memoria extra.

- Al agregarle un componente aleatorio tiene una eficiencia <u>promedio</u> de *O(N*log(N))*. Esta característica es conocida como el “Teorema del *quickSort”*

Selección de un buen pivote

Un pivote es bueno si particiona el arreglo en dos sub-arreglos de aproximadamente igual tamaño. En otras palabras si balancea los sub-problemas.

Expectativa: Un pivote seleccionado aleatoriamente con una distribución uniforme es más o menos bueno, casi siempre.

Conjetura: Si siempre se logra un particionamiento 25-75, este es lo suficientemente bueno para obtener una eficiencia $O(N*\log(N))$. Consideremos que la mitad de los elementos nos darían ese particionamiento o mejor.

Pero vamos mejor a la demostración estadística …

Análisis estadístico

Dada la elección aleatoria de pivotes, *T(N)* está "dominado" por el número de comparaciones puesto que el quickSort no tiene etapa de combinación y que el particionamiento básicamente se limita a comparar (el intercambio no siempre se da). Sean entonces:

C: número de comparaciones entre los elementos del arreglo

$T(n) = E[C]$

Como este valor esperado es difícil de calcular es mejor utilizar una aproximación por descomposición (descomponer *C* en otras variables aleatorias "más sencillas")

Sea Z_i el *i*-ésimo elemento más pequeño del arreglo

Sea X_{ij} el número de comparaciones entre Z_i y Z_j

¿Qué valores puede tomar esta variable?

1 ó 0 (una si una de ellas es elegida como pivote y hacen parte del mismo llamado recursivo, o cero si durante la partición caen en lados opuestos del pivote), por tanto es una variable binaria

Podemos entonces definir a *E[C]* como $\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} E[X_{ij}]$

Y recordando que: $E[Y] = \mu_y = \begin{cases} \sum_y y * \Pr(y) \quad para\, V.A.D. \\ \int_{-\infty}^{\infty} yFr(y)dy \quad para\, V.A.C. \end{cases}$

Podemos decir que *E[C]* = $\sum_{i=1}^{N-1} \sum_{j=i+1}^{N} Pr(X_{ij} = 1)$

Análisis estadístico

¿Cuál es $Pr(X_{ij} = 1)$?

Dados Z_i y Z_j con $i<j$ tendríamos el siguiente sub-arreglo:

$Z_i, Z_{i+1}, \ldots, Z_{j-1}, Z_j$ y podría suceder que:

a) El pivote sea elegido por fuera de este sub-arreglo, caso en el cual todos pasarían al mismo llamado recursivo; ó

b) Z_i o Z_j sea elegido, caso en el cual $X_{ij} = 1$; ó

c) Ni Z_i ni Z_j sea elegido, caso en el cual $X_{ij} = 0$ pues pasarían a diferentes llamados recursivos

Concentrándonos en el caso b, $Pr(X_{ij} = 1) = \frac{2}{j-i+1}$

Reemplazando tendríamos que $\mathrm{E}[\mathrm{C}] = \sum_{i=1}^{N-1} \sum_{j=i+1}^{N} \frac{2}{j-i+1}$

Con lo cual, la sumatoria interna se convierte en:

$\sum_{j=i+1}^{N} \frac{1}{j-i+1} = \frac{1}{2} + \frac{1}{3} + \frac{1}{4} + \cdots (\text{a lo su}mo\ si\ i = 1) \frac{1}{N}$

Es decir, $\mathrm{E}[\mathrm{C}] \leq \sum_{i=1}^{N-1} \sum_{x=1}^{N} \frac{2}{x} = 2 * (N-1) * \sum_{x=1}^{N} \frac{1}{x}$

Y dado que $\sum_{x=1}^{N} \frac{1}{x} \leq \int_{1}^{N} \frac{1}{x} \partial x = \ln(x) \Big|_{1}^{N} = \ln(N) - \ln(1)$

Tenemos finalmente $\mathrm{E}[\mathrm{C}] \leq 2(N-1)\ln(N)$

Pero una propiedad de la función *log* es que $log_b(x) = \frac{\ln(x)}{\ln(b)}$

Por tanto $\mathrm{E}[\mathrm{C}] \leq 2\ln(2)(N-1)log_2(N)$ es decir ***O(N*log(N))***