Estadístico de orden k

Estadístico de orden k

Entrada: Un arreglo *A* con *N* valores numéricos y un entero *k* ($1 \leq k \leq N$)

Salida: Estadístico de orden *k* de *A* (*k*-ésimo menor valor de *A*)

¿Qué tal ordenar y luego acceder a
la posición *k* del arreglo ordenado?

O(N*log(N)) ¿Se puede hacer mejor?

Solución mediante “divide & conquer”

Truco: Utilizar la idea del *quickSort* pero considerando este problema como de menor complejidad que el de ordenar

```
function findStatistic(A, i, j, k):
   if i=j:
      return A[i]
   else:
      p = choosePivot(i,j)
      h = partition(A, i, j, p)
      if k = h:
         return A[h]
      else if k > h:
         return findStatistic(A, h+1, j, k)
      else:
         return findStatistic(A, i, h-1, k)
```

Solución mediante "divide & conquer"

En el peor de los escenarios (por ejemplo si el pivote siempre es el menor del sub-arreglo y se busca el *n*-ésimo estadístico) ¿Cuál es la eficiencia del algoritmo? $O(N^2)$

Y en el mejor de los escenarios (si el pivote siempre es la media del sub-arreglo y se busca el *k*-ésimo estadístico con *k* diferente a dicha media)?

Según, el método maestro: *a=1, b=2, d=1*, es decir $a < b^d$ (caso 2) lo que implica n^d → O(N)

¿Será que de forma análoga al *quickSort*, en promedio dominan los buenos resultados sobre los malos?

Análisis de la eficiencia promedio vía "fases"

El algoritmo (*findStatistic*) realiza *c*m* operaciones por fuera del llamado recursivo (la función *partition*), con *c>0 y m* el tamaño del subarreglo.

<u>Notación</u>: decimos que *findStatistic* se encuentra en la fase *f* si el tamaño del subarreglo se encuentra entre: $N * \left(\frac{3}{4}\right)^{f+1}$ y $N * \left(\frac{3}{4}\right)^{f}$ es decir, si se ha reducido en un 75% *f* veces.

Ejemplos: si *f* = 0 (75%-100%), si *f* = 1 (56.25% - 75%), etc.

Sea entonces X_f el número de llamados recursivos durante *f*

De esta forma eficiencia = *r*c*N* ≤ $\sum_{fases\ f} X_f * c * [N * \left(\frac{3}{4}\right)^{f}]$

Siendo *r* la cantidad de llamados recursivos (*log(N)* ≤ *r* ≤ *N*)

Siendo así: $\mathrm{E}[\text{complejidad}] \leq \mathrm{E}[\sum_{fases\ f} X_f * c * N * \left(\frac{3}{4}\right)^f\ ]$

$$\leq c * N * \mathrm{E}[\sum_{fases\ f} X_f * \left(\frac{3}{4}\right)^f\ ]$$

$$\leq c * N * \sum_{fases\ f} \left(\frac{3}{4}\right)^f\ \mathrm{E}[X_f]$$

Como X_f es una variable aleatoria, ¿cuál es su valor esperado? Notemos que si durante la fase *f* la función *partition* tiene un buen pivote (una división 25%-75% como en el caso del *quickSort*) la fase termina. Si esto ocurre el nuevo sub-arreglo tendría un tamaño como mucho de 75% del anterior.

Como la probabilidad de obtener dicha división es 0.5 (igual a la de obtener “cara” al lanzar una moneda), el valor esperado de X_f es análogo al valor esperado del número de veces que se debe lanzar una moneda para obtener “cara”, es decir, 2.

Análisis de la eficiencia promedio vía “fases”

Por tanto: $\mathrm{E}[\text{complejidad}] \leq \mathrm{E}[\sum_{fases\ f} X_f * c * N * \left(\frac{3}{4}\right)^f\ ]$

$$\leq c * N * \mathrm{E}[\sum_{fases\ f} X_f * \left(\frac{3}{4}\right)^f\ ]$$

$$\leq c * N * \sum_{fases\ f} \left(\frac{3}{4}\right)^f\ \mathbf{E}[\boldsymbol{X_f}]$$

$$\leq c * N * \mathbf{2} * \boxed{\sum_{fases\ f} \left(\frac{3}{4}\right)^f}$$

$$\leq c * N * 2 * 4$$ → O(N)

La **serie geométrica** real de término inicial $a \in \mathbb{R}$ no nulo y de razón $r \in \mathbb{R}$ es convergente si y solamente si $|r| < 1$. En tal caso, su suma vale:

$$\sum_{n=0}^{\infty} ar^n = \frac{a}{1-r}$$