UNIVERSIDAD DE CHILE
ESCUELA DE INGENIERIA
DEPARTAMENTO DE CIENCIAS DE LA COMPUTACION

Diseño y Análisis de Algoritmos – CC4102
Control 1 - Semestre Otoño 2015

1. Diseñe un algoritmo eficiente para memoria secundaria que dada una relación binaria R sobre el dominio $\{1, \ldots, n\}$ determine si R es simétrica. Asuma que la relación R se presenta en disco como un archivo secuencial de todos sus pares $(i, j) \in \{1, \ldots, n\} \times \{1, \ldots, n\}$.

   Solución: Tomamos una copia de R y la ordenamos con respecto a su primera componente, y luego con respecto a la segunda en caso de empates. El resultado lo dejamos en cinta C. Luego hacemos una copia de R^{-1} y la ordenamos igual que la anterior. El resultado lo dejamos en cinta C'. Luego verificamos que ambas cintas contengan lo mismo. El costo es $O(sort(R))$ en término de llamadas a disco.

2. Sea $k \geq 1$ un entero fijo y considere el siguiente algoritmo para seleccionar un buen candidato para el máximo de un arreglo $x_1, x_2, \ldots, x_n$ de n enteros positivos distintos (donde $n > k$): (1) Compute el máximo m entre $x_1, \ldots, x_k$. (2) Elija como buen candidato para el máximo al menor j con $k + 1 \leq j \leq n$ tal que $m < x_j$. Si no existe tal x_j entonces simplemente elija a x_n como su buen candidato.

   Sea S el evento que expresa que el algoritmo anterior computa efectivamente el máximo. Queremos computar la probabilidad $Pr(S)$ del evento S. Para eso, note que $Pr(S)$ es igual $\sum_{i=k+1}^{n} Pr(S_i)$, donde S_i es el evento de que nuestro algoritmo compute el máximo y que tal máximo esté en la i-ésima posición del arreglo (¿por qué?). Además, para cada $k+1 \leq i \leq n$ se tiene que $Pr(S_i)$ es igual a $Pr(B_i \cap O_i)$, donde B_i es el evento de que el máximo esté en la posición i y O_i es el evento de que ningún elemento entre $k + 1$ y $i - 1$ sea elegido como buen candidato por el algoritmo (¿por qué?).

   a) (2 pts) Demuestre que para cada $k + 1 \leq i \leq n$ se tiene que $Pr(B_i \cap O_i) = \frac{k}{n(i-1)}$.

   Solución: Los eventos B_i y O_i son independientes. Claramente, $Pr(B_i) = 1/n$. Además, para que el evento O_i ocurra se debe cumplir que el máximo entre $x_1, \ldots, x_{i-1}$ está en (cualquiera) de las primeras k posiciones. La probabilidad de que esto ocurra es $k/(i - 1)$.

   b) (2 pts) Demuestre que $Pr(S) = \frac{k}{n} \sum_{i=k}^{n-1} \frac{1}{i}$, y que por tanto $Pr(S) \geq \frac{k}{n}(\ln n - \ln k)$.

   Solución: Sabemos que $Pr(S)$ es igual $\sum_{i=k+1}^{n} Pr(S_i)$, y por tanto $Pr(S) = \sum_{i=k+1}^{n} \frac{k}{n(i-1)}$. Esto último es igual a $\frac{k}{n} \sum_{i=k}^{n-1} \frac{1}{i}$. Claramente, $\int_k^n \frac{1}{x} dx \leq \sum_{i=k}^{n-1} \frac{1}{i}$. Por tanto, $Pr(S) \geq \frac{k}{n}(\ln n - \ln k)$.

   c) (2 pts) Enucentre un valor de k que maximice la probabilidad de $Pr(S)$. ¿Cuál es una cota inferior para la probabilidad de S en tal caso?

   Solución: Derivaremos la expresión $\frac{k}{n}(\ln n - \ln k)$ y la igualaremos a 0. La derivada de esta expresión es $\frac{1}{n}(\ln n - \ln k - 1)$. Igualando a 0 obtenemos un valor óptimo de $k = n/e$. En tal caso la probabilidad de S es al menos $1/e$.

3. Sean $x_1, x_2, \ldots, x_n$ números naturales distintos con pesos positivos $w_1, w_2, \ldots, w_n$ tal que se cumple que $\sum_{i=1}^{n} w_i = 1$. Definimos su *mediana de pesos (inferior)* (mp) como el elemento x_k que satisface lo siguiente:

$$\sum_{x_i < x_k} w_i < 1/2 \quad \text{y} \quad \sum_{x_i > x_k} w_i \leq 1/2.$$

a) (2 pts) Demuestre que la mediana (inferior) de $x_1, x_2, \ldots, x_n$ es la mp de los x_i con pesos $w_i = 1/n$, para $i = 1, 2, \ldots, n$.

Solución: La mediana inferior x_k de $x_1, x_2, \ldots, x_n$ es estrictamente mayor que $\lfloor n+1/2 \rfloor - 1 = \lfloor n - 1/2 \rfloor$ elementos. Por tanto, se cumple que $\sum_{x_i < x_k} w_i = 1/n(\lfloor n - 1/2 \rfloor) \leq (n-1)/2n < 1/2$. Además, x_k es menor que $n - \lfloor n+1/2 \rfloor$ elementos. Por tanto, se cumple que $\sum_{x_i > x_k} w_i = 1/n(n - \lfloor n+1/2 \rfloor) = 1 - 1/n(\lfloor n+1/2 \rfloor) \leq 1 - 1/2 \leq 1/2$.

b) Diseñe un algoritmo que compute la mp en tiempo $O(n)$ (4 pts). Si no puede, diseñe un algoritmo que compute la mp en tiempo $O(n \log n)$ (2 pts).

Hint: Para el algoritmo en $O(n)$ puede utilizar el hecho que la mediana inferior de los x_i se puede computar en $O(n)$ (como vimos en clases).

Solución: Primero presentamos una solución sencilla en $O(n \log n)$. Ordene los x_i's utilizando mergesort. Esto toma $O(n \log n)$. Luego compute el mayor elemento x_k tal que $\sum_{x_i < x_k} w_i < 1/2$ en tiempo $O(n)$. Para demostrar que este x_k es la mp solo falta demostrar que $\sum_{x_i > x_k} w_i \leq 1/2$. Note que $\sum_{x_i > x_k} w_i = 1 - (\sum_{x_i < x_k} w_i) - w_k$. Sabemos que $\sum_{x_i < x_k} w_i + w_k \geq 1/2$ (porque de otra forma el algoritmo no hubiera elegido a x_k). Por tanto, $\sum_{x_i > x_k} w_i \leq 1/2$.

El algoritmo que trabaja en tiempo lineal es un poco más complejo:

mp(A,l)
$n \leftarrow$ length(A)
$m \leftarrow$ median(A)
$B \leftarrow \emptyset$ % B contiene todos los elementos de A menores a m
$C \leftarrow \emptyset$ % C contiene todos los elementos de A mayores o iguales a m
$w_B \leftarrow 0$ % w_B es suma de pesos de elementos en B
if $n = 1$
  then return $A[1]$
for $i \leftarrow 1$ **to** n
  do if $A[i] < m$
    then $w_B \leftarrow w_B + w_i$
      Append $A[i]$ to B
    else Append $A[i]$ to C
if $l + w_B < m$ % mediana de pesos pertenece a B
  then mp(B,l)
  else mp(C,w_B)

El algoritmo se aplica en entrada $(X, 0)$, asumiendo X es el arreglo de los x_i. En cada paso aplicamos el algoritmo en entrada (A, l), donde l es la suma de los pesos de los elementos en X que son menores que todos los elementos de A. El costo del algoritmo es $T(n) := T(n/2) +$

$O(n)$ (ya que la mediana se puede computar en tiempo lineal, como vimos en clases). Por tanto, $T(n)$ es $O(n)$ según Teorema Maestro.