CC4102 - Diseño y Análisis de Algoritmos
Auxiliar 1

Prof. Gonzalo Navarro; Aux. Mauricio Quezada

2 de Noviembre, 2011

1 Notación Asintótica

1. Defina $O(\cdot)$, $\Omega(\cdot)$, $\Theta(\cdot)$, $o(\cdot)$ y $\omega(\cdot)$. ¿Es cierto que si todo algoritmo resuelve un cierto problema en tiempo $\Omega(f(n))$ en el peor caso, entonces *ningún* algoritmo que resuelve ese problema corre en tiempo $o(f(n))$?.

1.1 Solucion

1. $O(g(n)) = \{f(n) \ : \ \exists c > 0, n_0 > 0, \forall n \geq n_0, \ 0 \leq f(n) \leq cg(n)\}$
2. $\Omega(g(n)) = \{f(n) \ : \ \exists c > 0, n_0 > 0, \forall n \geq n_0, \ 0 \leq cg(n) \leq g(n)\}$
3. $o(g(n)) = \{f(n) \ : \ \forall c > 0, \exists n_0 > 0, \forall n \geq n_0, \ 0 \leq f(n) < cg(n)\}$
4. $\omega(g(n)) = \{f(n) \ : \ \forall c > 0, \exists n_0 > 0, \forall n \geq n_0, \ 0 \leq cg(n) < f(n)\}$

2 Cotas inferiores

1. ¿Cuál es la cota inferior del problema de encontrar el máximo de un arreglo desordenado usando: 1) Estrategia del adversario, 2) *Teoría de la Información*?

2. Muestre usando árboles de decisión la cota inferior del problema de ordenamiento de n elementos distintos de un arreglo desordenado en el *modelo de comparaciones*.

3. Muestre mediante la estrategia del adversario cómo probar una cota inferior de $\lceil 3n/2 \rceil - 2$ comparaciones para el problema de encontrar el mínimo y el máximo de un arreglo desordenado.

4. De un algoritmo eficiente para ordenar n elementos distintos en un arreglo desordenado, donde el arreglo está formado de r secuencias contiguas ya ordenadas, de largos $n_1, n_2, \ldots, n_r$, respectivamente.

2.1 Solucion

1. Bajo el modelo de comparaciones:

   (a) Como el arreglo está desordenado, al no comparar todos los elementos, un adversario siempre podrá decir que uno de los elementos restantes es el máximo. Por lo que necesitamos por lo menos $n - 1$ comparaciones para determinar el máximo.

(b) (Propuesto)

2. Un árbol de decisión tendrá $n!$ hojas, correspondiente a la cantidad de permutaciones posibles del arreglo de n elementos. Ahora el problema es determinar la permutación correcta (ordenada), lo que corresponde a la altura del árbol de decisión, que es $\lg n!$, o $\Omega(n \log n)$.

3. Visto en clases

4. La cota inferior de ordenamiento es $\Omega(n \log n)$ en el caso general, pero si el arreglo está formado de secuencias ya ordenadas, podemos tener una mejor cota:

   - Si el arreglo consiste en r secuencias contiguas ordenadas de largos $n_1, n_2, \ldots, n_r$, la cantidad de arreglos de ese tipo es $n!/(n_1!n_2! \cdots n_r!)$
   - La cota inferior nos da entonces $\lg(n!/(n_1!n_2! \cdots n_r!)) = n \lg n - \sum_{i=1}^{r} n_i \lg n_i = \sum_{i=1}^{r} n_i \lg(n/n_i)$

Un algoritmo posible para ordenar este tipo de arreglo es mergesort r-ario que tomaría $O(n \log r)$, pero que aun no es suficiente.

Si usamos un árbol de Huffman para hacer el merge de las r secuencias considerándolas como símbolos de frecuencia n_i/n, vemos que los n_i elementos de la i-ésima hoja hacen merge l_i veces, donde l_i es su profundidad en el árbol. Por lo que $\sum_{i=1}^{r} n_i l_i$ es la cantidad total de merges a realizar.

Como el largo promedio de cada símbolo en este árbol es menor que $H + 1$, donde H es la entropía, $H = \sum_{i=1}^{r} \frac{n_i}{n} \log \frac{n}{n_i}$, y el largo promedio de cada símbolo es $\frac{1}{n} \sum_{i=1}^{r} n_i l_i$, se garantiza que el costo de los merges es

$$\begin{aligned} \frac{1}{n} \sum_{i=1}^{r} n_i l_i &< 1 + \sum_{i=1}^{r} \frac{n_i}{n} \log \frac{n}{n_i} \\ \sum_{i=1}^{r} n_i l_i &< n + \sum_{i=1}^{r} n_i \log \frac{n}{n_i} \\ &= \Omega\Big(\sum_{i=1}^{r} n_i \log \frac{n}{n_i} \Big) \end{aligned}$$