

## Auxiliar 2 - Cotas Inferiores

CC4102/CC53A - Diseño y Análisis de Algoritmos  
Profesor: Gonzalo Navarro    Auxiliar: Miguel Romero

11 de Agosto del 2014

1. Se desea ordenar un arreglo  $A[1, \dots, n]$ , formado por la concatenación de  $r$  arreglos ordenados de largo  $n_1, \dots, n_r$ . Queremos probar que cualquier algoritmo basado en comparaciones necesita  $\Theta(n + \sum_{i=1}^r n_i \log n/n_i)$  comparaciones en el peor caso.
  - (a) Usando árboles de decisión pruebe una cota inferior de  $\Omega(n + \sum_{i=1}^r n_i \log n/n_i)$ .  
*Hint:* Encuentre una cota inferior de  $\Omega(n)$  y usando árboles de decisión una cota inferior de  $\sum_{i=1}^r n_i \log n/n_i$ . Puede utilizar que  $\log n! - \sum_{i=1}^r \log n_i! = \Omega(n \log n - \sum_{i=1}^r n_i \log n_i)$ .
  - (b) Utilizando la idea de MergeSort, encuentre un algoritmo que hace  $O(n \log r)$  comparaciones.
  - (c) Utilizando el árbol generado en la codificación de Huffman, mejore su algoritmo para que haga  $O(n + \sum_{i=1}^r n_i \log n/n_i)$  comparaciones.  
*Hint:* Puede usar  $r \log r = O(\sum_{i=1}^r n_i \log n/n_i)$  y que  $\sum_{i=1}^r p_i l_i \leq \sum_{i=1}^r p_i \lceil \log 1/p_i \rceil$ , donde los  $l_i$ 's son los largos de los códigos de Huffman.
2. Considere el siguiente problema: Dado un arreglo de bits  $A[1, \dots, n]$ , decidir acaso hay tres 1's consecutivos. Usando la técnica del adversario, demuestre que cualquier algoritmo debe examinar al menos  $4\lfloor n/5 \rfloor$  bits de la entrada.  
*Hint:* Considere un adversario que divide  $A[1, \dots, n]$  en bloques de 5-bit. El último bit de cada bloque contiene 0. El adversario define los primeros 4 bits de cada bloque de manera adecuada, según las preguntas del algoritmo.
3. Queremos verificar si un grafo es acíclico utilizando sólo preguntas de la forma: ¿Existe un arco entre  $u$  y  $v$ ? Demuestre que cualquier algoritmo de esta forma necesita  $\binom{n}{2}$  consultas en el peor caso.  
*Hint:* Considere un adversario que mantiene las componentes conexas conocidas hasta el momento, y responde sí a la pregunta ¿hay un arco entre  $u$  y  $v$ ? siempre y cuando,  $u$  y  $v$  están en componentes distintas. Demuestre que con esta estrategia el algoritmo debe examinar todos los pares de nodos.