Universidad de Chile
Departamento de Ciencias de la Computación
CC4102 - Diseño y Análisis de Algoritmos

Soluciones Control 1

Profesor: Gonzalo Navarro
Auxiliar: ~~Manuel~~ Ariel Cáceres Reyes

P1. Sean $T_1, \ldots, T_n$ las tuercas en el orden que vienen en el input y $t_1, \ldots, t_n$ los tornillos según vienen en el input. Anotemos $T_i \leq t_j$ si la tuerca i es más chica o calza con el tornillo j. Veamos por otro lado, que una asignación de tuercas-tornillos corresponde exactamente a una permutación de los tornillos que los hace calzar con las tuercas según el orden en el que vienen en el input, por lo que hay $n!$ asignaciones posibles. Veamos ahora que el resultado de una comparación $T_i \leq t_j$ reduce los posibles input a aquellas permutaciones de los tornillos en donde $T_i \leq t_j$ y a otras en donde $T_i > t_j$, y por lo tanto particiona el espacio de posibles asignaciones en 2. Finalmente, si suponemos que cada una de las asignaciones aparecen en el input con igual probabilidad, gracias al teorema de Shannon tenemos la cota inferior $\log n! \sim n \log n$.

P2.

1. Tendremos $N/M \leq M/B$ archivos y mandamos aleatoriamente elementos del input a algunos de estos archivos (tenemos buffers de B elementos que nos caben en memoria principal), si alguno de los archivos alcanza los M elementos no se le considera más en la repartición aleatoria del input. Luego uno por uno nos traemos estos archivos a memoria principal, les hacemos shuffle y los mandamos de vuelta a memoria externa. Finalmente vamos eligiendo al azar elementos de alguno de los archivos y los vamos mandando al output (todo esto con buffers). Notar que en términos de I/Os el algoritmo solo unas cuantas pasadas lineales por los datos, es decir $\mathcal{O}(N/B)$.
2. El problema con que $N > M^2/B$ es que ya no podemos dividir el input en N/M archivos, pues solo tenemos la capacidad de hacerlo en M/B archivos con lo que los archivos nos quedan de tamaño $N/(M/B) > M$ y no podemos hacer la segunda fase de traer los archivos a memoria y hacerles shuffle ahí. Para resolver lo anterior le haremos shuffle recursivamente a los archivos de tamaño $N/(M/B)$ lo que nos dará archivos de tamaño $N/(M/B)^2$ y así hasta que $N/(M/B)^i = M$ y nos quepan en memoria para hacerles el shuffle. Ahora la complejidad es un proceso lineal, pero que se hace $i = \log_m n$ veces por lo que la complejidad final es la de ordenar.

P3. Veamos que en una operación de I/O nos podemos traer B elementos a memoria principal. Esto particiona el conjunto de inputs en a los más $B+1$ conjuntos, lo que genera un árbol de decisión de aridad $B+1$ cuyas hojas son todas las posibles ubicaciones del elemento buscado, es decir, el árbol tiene N hojas. Así, la altura de este árbol será al menos $\log_{B+1} N$ y por lo tanto cualquier algoritmo debe hacer al menos $\log_{B+1} N \mathcal{O}(\log_B N)$ comparaciones en el peor caso. Para mostrar que esto se cumple también en el caso promedio podemos transformar este árbol de decisión $B+1$-ario a un árbol de decisión binario equivalente con N hojas equiprobables sobre el cual podemos aplicar Shannon y obtener la cota inferior en caso promedio.