Randomized Algorithms
ITU Copenhagen, Fall 2004
Exercises and hand-in

G. S. Frandsen, A. Pagh, and R. Pagh

September 8, 2004

Exercises on September 17, 13-15

1. Analyze the expected number of swaps made by bubblesort if the input is a random permutation of n distinct numbers.
2. Exercises from Motwani-Raghavan: 1.1, 1.2, 1.4, 1.8.

Hand-in (deadline: Lecture on September 21)

C. A. R. Hoare published the original (randomized, recursive) version of QUICKSORT in 1961 along with a recursive selection algorithm FIND sharing code with quicksort. Below is a self-contained version of FIND. It returns the k'th smallest element in a list.

In this problem you are required to upper bound the expected number of comparisons made by FIND. You must use the "sum of indicator variables" technique (see Motwani and Raghavan, Example 1.1 and the prose leading up to it). You must prove a bound that is no larger than cn for some explicit constant c.

For the ambitious student: prove a bound $\leq (2 + 2\ln 2)\, n \approx 3.3863\, n$.

Below is defined some notation and set up some subproblems that may be of help.

Algorithm: FIND

Input: $L = [a_1, \ldots, a_n]$, a nonempty list of distinct numbers
k, an integer such that $1 \leq k \leq n$.

Output: b, where $b \in L$ and $|\{a \in L : a \leq b\}| = k$

Method:

1. Select e randomly from L using the uniform distribution
2. Split $L' = L - \{e\}$ into the two sublists
   $L_1 = [a_i \in L | a_i < e]$
   and
   $L_2 = [a_i \in L | a_i > e]$
   by comparing e to each element of L'.
3. If $|L_1| = k - 1$ then return e.
   If $|L_1| > k - 1$ then make a recursive call on L_1 and k.
   If $|L_1| < k-1$ then make a recursive call on L_2 and $k-1-|L_1|$.

Notation: Let π be the unique permutation that sorts the elements of L, i.e. $a_{\pi(1)} < \cdots < a_{\pi(n)}$

Subproblem: Using the above notation, which element does FIND return?

Subproblem: Given i, j with $1 \leq i < j \leq n$. What is the probability that $a_{\pi(i)}$ is compared to $a_{\pi(j)}$ during the execution of FIND on input $[a_1, \ldots, a_n], k$? - it may be convenient to divide into cases depending on whether $i < j < k$ or $i < k < j$ or $k < i < j$. `Added in version 3:` As an example, consider the case of $i < k < j$. Argue that during the execution of FIND there must be a first recursive call, where e is chosen from the subset of elements $\{a_{\pi(i)}, a_{\pi(i+1)}, \ldots, a_{\pi(k)} \ldots a_{\pi(j)}\}$. Argue that if $a_{\pi(i)}$ is compared to $a_{\pi(j)}$ it must necessarily happen in precisely this call. Argue that the wanted probability in this case is $\frac{2}{j-i+1}$.

Subproblem: Define suitable indicator variables and use the result from the previous subproblem to write down an expression for the expected number of comparisons that FIND makes on input $[a_1, \ldots, a_n], k$.

Subproblem: Use some math to evaluate or upper bound the value of the above expression. `Added in version 2:` When considering the case $i < k < j$, you may find it helpful to determine how many pairs i, j that satisfy $j - i = l$ for any fixed number l.