CS524

MIDTERM EXAM

Name_____

Date: October 19, 2004

All documentation permitted

1. (25 points) Using Θ-notation, find a simplest solution to the recurrences

a) $T(n) = 9T(n/3) + n^2$

b) $T(n) = 9T(n/3) + n^2\sqrt{n}$

2. (20 points) The efficiency of QUICKSORT depends upon the distance between $n/2$ and a random number, *pivot*, $pivot \in \{1, 2, ..., n\}$, where we assume that n is evenly divisible by 4. Define *pivot* to be *good* if $.25n \leq pivot < .75n$, and suppose that *pivot* will be randomly chosen (from a uniform distribution) over [1,n]. Assuming that the successive values of *pivot* are independent, what is the expected number of values which must be sampled until a *good* value is drawn?

3. (30 points) Consider the problem of testing for MEMBERSHIP of an element x in an array $A[1..n]$, where the benchmark operation is testing for equality, $x \stackrel{?}{=} A[i]$, where $1 \le i \le n$. For example, when n=5 and $A = (22, -3, 15, 22, 85)$, then the algorithm should return false when x=6 and true when x=15.

a) Prove that an upper bound on the worst-case complexity of testing for MEMBERSHIP is $O(n)$.

b) Prove that a lower bound on the worst-case complexity of testing for MEMBERSHIP is $\Omega(n)$.

4. (25 points) An algorithm for HASH CODING samples bins until an empty bin is found. In order to analyze the algorithm, assume that there are m>1 bins, and n of the bins are OCCUPIED, where $m > n \geq 0$. Assume that

- the algorithm probes bins from a uniform distribution over $\{1, ..., m\}$,
- the probes are determined independently,
- the algorithm can detect if a bin is OCCUPIED,
- the algorithm never probes a bin that a previous probe determined was OCCUPIED,
- the algorithm continues probing until it finds a bin which is not OCCUPIED.

For example, if m=10, and bins 2, 3, 4, 8 and 9 are OCCUPIED, then probe sequence 2, 9, 3, 1 is a sequence of length 4 until it finds an empty bin. Find an expression (it need not be closed form), as a function of m and n, of the expected length of a probe sequence until an empty bin is found.

CS524

Solutions to Midterm Exam

1. ***a***) Using the Master Theorem, a=9, b=3 and $f(n) = n^2$. So $\log_b a = \log_3 9 = 2$, and $f(n) = n^2 = \Theta(n^2)$, yielding (by the second case, with k=0) that $T(n) = \Theta(n^2 \lg n)$.

b) Using the Master Theorem, a=9, b=3 and $f(n) = n^2\sqrt{n} = n^{2.5}$. So $\log_b a = \log_3 9 = 2$, and $f(n) = n^{2.5} = \Omega(n^{2+\varepsilon})$ for any $\varepsilon < 0.5$, yielding (by the third case) that $T(n) = \Theta(n^{2.5})$.

2. The probability that *pivot* satisfies $.25n \le pivot < .75n$ is ½. Defining this event to be success, we sample values of *pivot* independently until success. The number of trials until success is geometrically distributed, and the expected value of this random variable is $\sum_{k\ge1} k * \Pr\{\text{failure}\}^{k-1}\Pr\{\text{success}\} = \sum_{k\ge1} k * \left(\frac{1}{2}\right)^k = 2$.

3. ***a*** To get the $O(n)$ upper bound, we demonstrate a linear time algorithm to solve the problem.

```
MEMBERSHIP(x,A)
    for i ← 1 to n do
        if x=A[i] then return true
    return false
```

b To get the $\Omega(n)$ lower bound, we assume that there is an algorithm to solve the problem using fewer than a linear number of tests for equality. For some element x and list A which doesn't include x, the purported algorithm uses fewer than n tests for equality in returning false. During this execution, the algorithm did not examine at least one element of A, say $A[i^*]$. We now construct a new instance of the problem, with input x and $B[1..n]$ where

$$B[i] = \begin{cases} A[i], \text{ if } i \ne i^* \\ x, \text{ if } i = i^* \end{cases}$$

We note that the algorithm must return false on input x and B since it receives the same answers to its queries, though x is a member of B. By this contradiction, the assumption of the existence of a sublinear algorithm to test for MEMBERSHIP must be wrong.

4. If k-1 ≥ 0 probes have been made, $k \ge 1$, then we note that for probe k, all previous k-1 probes were of OCCUPIED bins. So the probability that the $k^{\underline{th}}$ bin probed will be OCCUPIED is $\frac{n-k+1}{m-k+1}$. The probability that the algorithm will quit on probe k is the probabilty that the first k-1 probes all hit OCCUPIED bins times the probability that the $k^{\underline{th}}$ bin probed is

not OCCUPIED, or $\frac{n}{m}\frac{n-1}{m-1}*...*\frac{n-k+2}{m-k+2}\left(1-\frac{n-k+1}{m-k+1}\right)$. So the expected number of probes is $\sum_{k\geq 1}\frac{n}{m}\frac{n-1}{m-1}*...*\frac{n-k+2}{m-k+2}\left(1-\frac{n-k+1}{m-k+1}\right)k$.