OHJ-2016 Utilization of Data Structures

Class Exam

Tuesday 25 May 2010

Imed Hammouda

Make sure you read the questions carefully before giving your answer. Put your name and student number on each answer sheet.
This exam consists of **2** pages / **5** exercises. The maximum amount of points is 30. Each exercise is worth 6 points.
Written material, mobile phones and calculators are NOT allowed in the exam.
Good luck!

Exercise 1

(a) Define the *load factor* of a hash table. (1p)
(b) Briefly explain the connection between *memoization*, *dynamic programming* and *greedy algorithms*. (2p)
(c) When you insert an item into a *skip list*, and build a tower, you toss a fair coin that comes up tails with probability *1/2* and heads with probability *1/2*. Suppose that you use an unfair coin instead. This coin comes up tails with probability p and heads with probability $1-p$. Suppose that p is very close to *1*. How is search time affected? Suppose that p is very close to *0*. How is search time affected? Do not derive formulas, just describe your intuition. Why is the choice of $p=1/2$ important? (3p)

Exercise 2

Decide whether these statements are *True* or *False*. (0.5p/point)

1. If $f(n) = \Theta(g(n))$ and $g(n) = \Theta(h(n))$, then $h(n) = \Theta(f(n))$
2. If $f(n) = O(g(n))$ and $g(n) = O(h(n))$, then $h(n) = \Omega(f(n))$
3. If $f(n) = O(g(n))$ and $g(n) = O(f(n))$ then $f(n) = g(n)$
4. $n/100 = \Omega(n)$
5. There exists a comparison sort of 5 numbers that uses at most 6 comparisons in the worst case.
6. *Heapsort* can be used as the auxiliary sorting routine in radix sort, because it operates in place.
7. Let P be a shortest path from some vertex s to some other vertex t in a graph. If the weight of each edge in the graph is increased by one, P will still be a shortest path from s to t.
8. If an in-place sorting algorithm is given a sorted array, it will always output an unchanged array.
9. Dijkstra's algorithm works on any graph without negative weight cycles.
10. The Relax function never increases any shortest path estimate $d[v]$.
11. Any Dynamic Programming algorithm with n subproblems will run in $O(n)$ time.
12. Every binary search tree on n nodes has height $O(\log n)$.

Exercise 3

You are given an array of n keys, each with one of the values *red*, *white*, and *blue*.
Give an $O(n)$ algorithm for rearranging the keys so that all the *reds* come before all the *whites*, and all the *whites* come before all the *blues*. The only operations permitted are examination of a key to find out what color it is, and swap of two keys (specified by their indices). Explain your algorithm and derive its running time.

Exercise 4

1. The sequence $\langle 20, 15, 18, 7, 9, 5, 12, 3, 6, 2 \rangle$ is a max-heap. True or False? Explain! (1.5p)

2. Where in a max-heap can the smallest element reside, assuming all elements are distinct? Include both the location in the array and the location in the implicit tree structure. (1.5p)

3. Suppose that instead of using *Build-Heap* to build a max-heap in place, the *Insert* operation is used n times. Starting with an empty heap, for each element, use *Insert* to insert it into the heap. After each

insertion, the heap still has the max-heap property, so after n *Insert* operations, it is a max-heap on the n elements.
(i) Argue that this heap construction runs in $O(n \log n)$ time. (1.5p)
(ii) Argue that in the worst case, this heap construction runs in $\Omega(n \log n)$ time. (1.5p)

Exercise 5

You are given an n-by-n grid, where each *square* (i, j) contains $c(i, j)$ gold coins. Assume that $c(i, j) \geq 0$ for all squares. You must start in the upper-left corner and end in the lower-right corner, and at each step you can only travel one square down or right. When you visit any square, including your starting or ending square, you may collect all of the coins on that square. Give an algorithm to find the maximum number of coins you can collect if you follow the optimal path.

mikael.niemela@tut.fi