COMP20003 Algorithms and Data Structures

Second Semester 2016
Final Examination
15/11/2016

Student Number:

Identical Examination papers: None.
Exam Duration: Three hours.
Reading Time: 15 minutes.
Length: This paper has 6 pages including this cover page.
Authorised Materials: English language dictionaries and foreign language dictionaries are the only authorised materials. Other paper and printed materials are not permitted. University regulations prohibit the use of electronic dictionaries. Calculators and other electronic devices: Calculators are not permitted.
Instructions to Invigilators: Students should be supplied with the examination paper and a script book. They may have unlimited script books on request. Students should write their ID number on the examination paper and on all script books. They may not remove any part of the examination paper from the room and must return any used and unused script books with their examination paper.
Instructions to Students: There are 5 questions. You should attempt all questions. Answers to all questions are to be written in your script book. You may use the reverse side of the page to make notes or prepare draft answers. The reverse sides will not be looked at or marked unless you clearly indicate that you wish this to be your final answer.
You may use pencil. This paper counts for 60% of your final grade. Marks shown are out of 60.

Grade Table (for teacher use only)

Question	Points	Score
1	18	
2	6	
3	10	
4	16	
5	10	
Total:	60	

Programming

1. (18 points) This question is about the C programming language.

   Given an array of integers, the goal is to efficiently find the (contiguous) subarray whose prime elements have the greatest sum: when considering negative numbers, they are considered prime if their absolute value is prime. Note that because some elements of the array may be negative, the problem is not solved by simply picking the start and end elements of the array to be the subarrray, and summing the entire subarray.

   For example, given the following array

```
1 int foo[5] = {1, 1, -5, 3, -3, 2};
```

   the maximum sum of its subarrays is 3. It is possible for the subarray to be zero elements in length (if every element of the array is negative or not prime).

   Before you write the code, take some time to think of the most efficient solution possible.

   (a) (6 points) Write the function int `is_prime`(int `num`), that given a number returns 1 if it is prime or 0 otherwise.

   (b) (6 points) Write the function int `max_sum`(int `*vector`, int `len`) that given an array and its length returns the maximum sum as defined above.

   (c) (6 points) Write the main function, use a dynamic array called foo, allocate the memory, initialize it with the example foo array shown above and print the solution using your implemented functions. Last but not least, free your memory. You do not need to write the `include` statements.

   You should write the function in your script book. Include brief comments in the code to explain your reasoning and help with readability. Marking of your code will be based on accuracy (1 mark out of 18), completeness (1 mark), efficiency (2 marks), best practice use of C (1 mark), and readability (1 mark). Efficiency will be mostly checked in question (a) and (b). For partial marks in question (b), write a function that sums every element in the subarray, not just prime numbers.

Computational Complexity

2. (6 points) This question is about computational complexity

   For the remaining parts of this question you are required only to give an answer. You are not required to explain or justify your answers in this section. Each correct answer gives 1 point.

   (a) (1 point) What is the time complexity of inserting one item into a sorted linked list of n items? Give the closest upper bound in big-O notation.

   (b) (1 point) What is the time complexity of sorting an array of n items using quicksort? Give the closest upper bound in big-O notation.

   (c) (1 point) What is the time complexity for searching an item in a perfectly balanced ternary tree of n items? Each node in a ternary tree has at most 3 children. Give the closest upper bound in big-O notation.

   (d) (1 point) What is the extra space complexity of sorting an array of n items using quicksort? Give the closest upper bound in big-O notation.

   (e) (1 point) What is the extra space complexity of sorting an array of n items using bottom-up mergesort, where the range of values of the items is r? Give the closest upper bound in big-O notation.

   (f) (1 point) Given a hash table of size m, containing $m-1$ items, give the worst case complexity of inserting one more item when collision resolution is achieved with chaining. Give the closest upper bound in big-O notation.

Advanced Data Structures

3. (10 points) This question is about disjoint sets, balanced trees, heaps and priority queues.

   (a) (4 points) You are given the following sets of items:

   {10},{20},{30},{40},{50},{60}

   i. (3 points) Using a *forest of trees* implementation of disjoint sets, show the resulting forest after each operation in order: Merge(10,20), Merge(50,60), Merge(30,40), Merge(30,60) and Merge(30,10). In this implementation we always select the minimum key as a representative. Show your workings and justify any operation if needed. Show your final answer for the forest in *both* the array *and* the tree representations.

   ii. (1 point) Given the naive *array* of representatives implementation of disjoint sets, what is the least upper bound big-O complexity of *finding* to which set an element belongs to? Be specific in your answer.

(b) (4 points) Prim's algorithm for Minimum Spanning Trees on an undirected, weighted graph uses a priority queue to get the next vertex closest to the fringe. If the priority queue is implemented using a balanced tree, give the least upper bound big-O complexity of Prim's algorithm, in terms of V , the number of vertexes in the graph, and E, the number of edges in the graph. Assume in-order traversal is used to find out the predecessor and successor candidates for deletion. Explain your answers *briefly*.

The main operations of Prim algorithm are:

```
1  pq = makePQ(G);
2  while(!emptyPQ(pq))
3  {
4    u = deletemin(pq);
5    for(/*each vertex v reached from u */)
6    {
7      if(edgeweight(u,v) < dist[v] and inmst[v] == FALSE)
8        update(v,pred,dist,pq);
9    }
10   inmst[u]=True;
11  }
```

(c) (2 points) A ternary heap is a heap in which each non-leaf node in the tree representation has *3* children, except possibly the rightmost non-leaf node. In an array representation of a ternary heap, what are the array indexes of the children of the node at position i in the array? Be explicit about any assumptions or conditions you have made.

Graphs

4. (16 points) This question is about graphs and graph algorithms.

(a) (3 points) Which is the best algorithm to compute the Minimum Spanning Tree over a graph $G(V, E)$:

i. (1.5 points) When the number of edges is $E = V$

ii. (1.5 points) When the number of edges is $E = V^2$.

Justify your answer and give the worse case behavior in big-O notation.

(b) (4 points) Which of the following are true and which are false? *Explain your answers for full marks*

i. (1 point) One Depth-first traversal is sufficient to visit all the vertexes in an undirected graph.

ii. (1 point) Breadth First Searh and Dijkstra will always find the shortest path over weighted DAGs.

iii. (1 point) A^* always finds the shortest path connecting two vertexes in a graph.

iv. (1 point) You can prove that *P=NP* if you find an exponential time reduction from an *NP-Complete* problem into a *P* problem.

(c) (4 points) In a directed graph, every vertex has an *in-degree*, the number of edges coming into the vertex, and an *out-degree*, the number of edges leaving that vertex.

For any given directed graph G(V, E), where v is an individual vertex, state the relationship between the sum of the *out-degrees* for each vertex, $\sum_{v=1}^{V} out\_degree(v)$, and the sum of the *in-degrees* for each vertex, $\sum_{v=1}^{V} in\_degree(v)$.

(d) (5 points) Give an example of a graph such that running Dijkstra on it would give incorrect distances. Explain why.

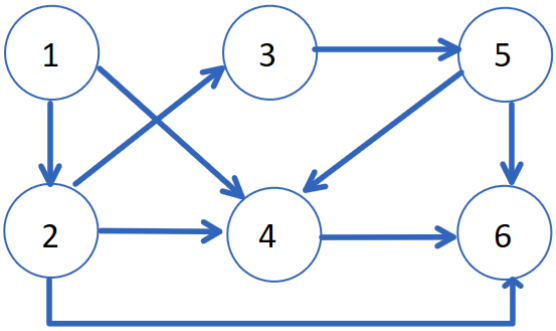
Searching and Sorting

5. (10 points) This question is about searching, sorting and balanced trees.

(a) (1.5 points) Given a list of 7 integers, how many mergesort calls are going to be triggered in the bottom-up implementation? Give the exact number and justify your answer.

(b) (1.5 points) Given the following list of integers: {13, 4, 5, 43, 22, 2, 3, 1}, which pivots will be chosen by quicksort if we always select the left most element? Keep the relative ordering between integers while partitioning, you do not need to run the algorithm using the indexes. Justify your answer.

(c) (2 points) Compute the topological sort of the following graph:



Show your workings to get full marks.

(d) (3 points) You are given records with the following keys, to be stored in a hash table

1 23 43 99 44 79 89

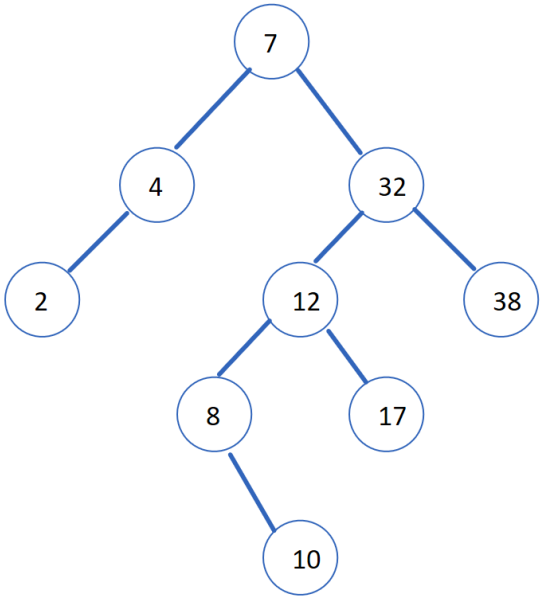
The hash table contains a maximum of 10 items, and the hash function `hash(key)` returns *key*%10. Linear probing is used for collision resolution.

i. (1.5 points) Show the hash table after insertion of these records

ii. (1.5 points) Can chaining resolution improve the lookup operation in terms of key comparisons when searching for number 89? Explain your answer.

(e) (2 points) The rotation operation is used in AVL trees, red-black trees, and splay trees to adjust the balance of a binary tree.

In the binary search tree shown below, you are to perform a rotation to balance the tree.



Explain which rotation is needed and justify your choice. Draw the resulting balanced tree.