The University of Melbourne
Department of Computing and Information Systems

COMP20007
Design of Algorithms
July (Supplementary), 2013

Student Number:

Identical Examination papers: None.

Common Content: None.

Exam Duration: Three hours.

Reading Time: Fifteen minutes.

Length: This paper has 14 pages including this cover page.

Total Marks: 76 (just over 2 minutes per mark)

Authorized Materials: None.

Instructions to Invigilators: Students will write all of their answers on this examination paper. Students may not remove any part of the examination paper from the examination room.

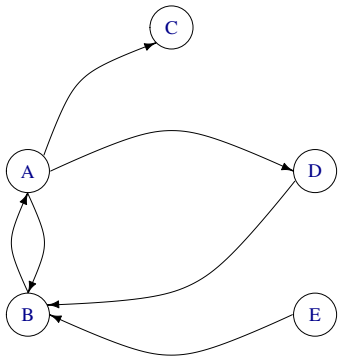
Instructions to Students: All questions should be answered in the spaces provided on the examination paper. You may make rough notes, and prepare draft answers, on the reverse of any page, and then copy them neatly into the boxes provided. You are not required to write comments in any of your code fragments or functions.

Throughout you should assume a RAM model of computation where input items fit in a word of memory, and basic operations such as $+ - \times /$ and memory access are all constant time.

Calculators: Calculators are not permitted.

Library: This paper may not be held by the Baillieu Library.

Question 1 (9 marks).



(a) (5 marks) For the above graph, complete the table of pre and post numbers for each vertex when a Depth First Search is performed beginning at A. If there is a choice of edge, choose the lower letter vertex in lexicographic order. Begin your numbering at 1.

Vertex	Pre number	Post number
A		
B		
C		
D		
E		

(b) (1 mark) List any back edges discovered in the DFS from (a).

(c) (1 mark) List any cross edges discovered in the DFS from (a).

(d) (1 mark) Name one source vertex in the graph at the top of this page.

(e) (1 mark) Name one sink vertex in the graph at the top of this page.

Question 2 (5 marks).

(a) (2 marks) What is a Euler path in a directed graph $G(V, E)$?

(b) (3 marks) Complete the following C code function.

```
#define BAD_VERTEX -1

/*
** INPUTS: G is an n by n adjacency matrix of a graph
**         where G[u][v]=1 indicates an edge from u to v.
**         v is a vertex number (hence index into G).
** OUTPUTS: the in-degree of the supplied vertex v in G,
**          or BAD_VERTEX if v is not a valid index into G.
** SIDE EFFECTS: None.
*/
int
get_in_degree(int **G, unsigned int n, unsigned int v) {
```

}

Question 3 (18 marks).

(a) (5 marks) What is the Burrows Wheeler Transform of the string $abcabc\$$? Assume that \$ is less than all other characters.

(b) (5 marks) Draw the Wavelet Tree for your BWT string in (a) assuming the shape of the tree is based on the prefix code: \$=00, a=01, b=10, c=11
You should show both the bitvector and the character strings at each node, even though only the bitvectors are actually stored.

(c) (3 marks) Build a Huffman code on the frequencies of each character in the string from (a). Show the Huffman tree, and the final codewords for each character.

(d) (2 marks) Write down an expression for Shannon's entropy of probability distribution $p = \{p_1, p_2, \ldots, p_n\}$.

(e) (3 marks) Explain why it is impossible for a compression program to compress every possible input of length n bits to a size that is smaller than n bits so that the original n bits are recovered perfectly upon decompression.

Question 4 (9 marks).

During Kruskal's greedy algorithm for building a Minimum Spanning Tree, each edge is added to the solution if it does not form a cycle in the tree that is constructed so far. A data structure is required to keep track of connected components in the partial solution during Kruskal's algorithm.

(a) (2 marks) What two operations must be supported by this data structure if it is to be used in Kruskal's algorithm? Give both the arguments and a short description of each operation.

(b) (5 marks) Describe an efficient data structure for the task, and include the big-O cost of each operation you mentioned in part (a) on this data structure for Kruskal's algorithm to find a MST on a graph of m edges and n vertices. If you have incorrect operations in (a), the maximum marks possible for this part is 3 out of 5.

(c) (2 marks) For some graph $G(V, E)$, what are the first edges added to the partial solution by Prim's greedy algorithm for building a Minimum Spanning Tree? Assume vertex $v \in V$ is processed first.

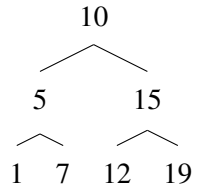
Question 5 (9 marks).

Answer True (T) of False (F) to the following statements. You will **lose** one mark for an incorrect answer in this question. Do not guess. The minimum possible mark for this question is zero.

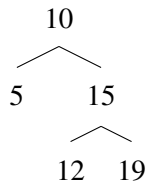
$f(n) = n^3$ is in $\Omega(n)$	
$f(n) = n^3 \log n$ is in $O(n^3)$	
$f(n) = n^2 / \log n$ is in $O(n^2)$	
$f(n) = \log \log n$ is in $O(\log^* n)$	
$f(n) = \sqrt{n}$ is in $\Omega(\log n)$	
It is possible to devise an algorithm that can answer a `rank` query on a bitvector in $O(1)$ time using no more than $2n$ extra bits of space	
It is possible to devise an algorithm that can sort an array of integers in $O(n)$ time if the maximum integer is restricted to be $O(n)$.	
The travelling salesman problem is in class NP, but the minimum spanning tree problem is in P.	
An n symbol string that is the result of a Burrows Wheeler Transform can be reversed in $O(n)$ time, regardless of the number of distinct symbols in the string.	

Question 6 (14 marks).

(a) (2 marks) Draw this tree after rotating the root once to the right so that 5 becomes the new root.



(b) (5 marks) Draw this AVL tree after inserting 13. Show the tree before rebalancing and after each rotation.



After inserting 13

After first rotation

After second rotation

(c) (7 marks) Complete the following table for each of these data structures using big-Oh expressions assuming they contain n items.

What is the space required for a hash table that uses Open Addressing and has m slots?	
What is the space required for an AVL tree?	
What is the worst case time to find an item in an AVL tree?	
What is the space required for a Wavelet Tree based on a prefix code with all codewords of $\lceil \log_2 n \rceil$ bits?	
What is the worst case time required to find the successor of an item in an AVL tree?	
What is the worst case time required to find the successor of an item in a Hash Table that has m slots and uses Open Addressing for collision resolution?	
What is the worst case time required to find an item in a Hash Table that uses separate chaining and has a load factor of n?	

Question 7 (12 marks).

(a) (3 marks) What are the attributes of a problem that would indicate that applying Dynamic Programming might lead to an efficient (polynomial) algorithm to solve the problem?

(b) (6 marks) Given two strings $x[1..n]$ and $y[1..m]$, the Edit Distance between the two strings can be calculated as $E(n, m)$, where
$E(0, j) = j$, $E(i, 0) = i$;
$\delta(i, j) = 1$ when $x[i] \neq y[j]$ and 0 otherwise; and
$E(i, j) = \min\{1 + E(i - 1, j), 1 + E(i, j - 1), \delta(i, j) + E(i - 1, j - 1)\}$ when $i \in [1, n]$ and $j \in [1, m]$.

Using the Dynamic Programming technique, give pseudo code for a polynomial time algorithm for computing $E(n, m)$.

(c) (1 mark) What is a tight upper bound on the worst case running time of your algorithm in (c)?

(d) (1 mark) What is a tight lower bound on the best case running time of your algorithm in (c)?

(e) (1 mark) Given your answers to (d) and (e), is there a precise statement you can make about the running time of your algorithm that holds for best, average and worst case inputs? If so, what is it?

Overflow Answers

The boxes here are for emergency use only. If you do need to use this page, indicate **CLEARLY** in your previous answer that you have continued onto this page. Without such an indication, it is possible that this part of your answer will be overlooked.

The University of Melbourne
Department of Computing and Information Systems

COMP20007
Design of Algorithms
July (Supplementary), 2013

Student Number: ANSWER

Identical Examination papers: None.

Common Content: None.

Exam Duration: Three hours.

Reading Time: Fifteen minutes.

Length: This paper has 14 pages including this cover page.

Total Marks: 76 (just over 2 minutes per mark)

Authorized Materials: None.

Instructions to Invigilators: Students will write all of their answers on this examination paper. Students may not remove any part of the examination paper from the examination room.

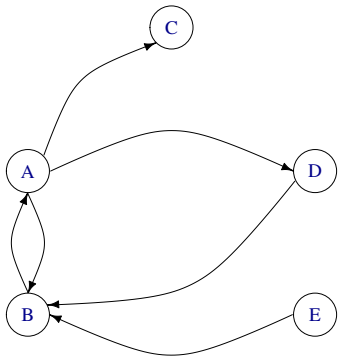
Instructions to Students: All questions should be answered in the spaces provided on the examination paper. You may make rough notes, and prepare draft answers, on the reverse of any page, and then copy them neatly into the boxes provided. You are not required to write comments in any of your code fragments or functions.

Throughout you should assume a RAM model of computation where input items fit in a word of memory, and basic operations such as $+ - \times /$ and memory access are all constant time.

Calculators: Calculators are not permitted.

Library: This paper may not be held by the Baillieu Library.

Question 1 (9 marks).



(a) (5 marks) For the above graph, complete the table of pre and post numbers for each vertex when a Depth First Search is performed beginning at A. If there is a choice of edge, choose the lower letter vertex in lexicographic order. Begin your numbering at 1.

ANSWER

Vertex	Pre number	Post number
A	1	8
B	2	3
C	4	5
D	6	7
E	9	10

Half a mark per cell

(b) (1 mark) List any back edges discovered in the DFS from (a).

ANSWER
1 mark BA

(c) (1 mark) List any cross edges discovered in the DFS from (a).

ANSWER
1 mark DB

(d) (1 mark) Name one source vertex in the graph at the top of this page.

ANSWER
1 mark E

(e) (1 mark) Name one sink vertex in the graph at the top of this page.

ANSWER
1 mark C

Question 2 (5 marks).

(a) (2 marks) What is a Euler path in a directed graph $G(V, E)$?

ANSWER
2 marks A path in G that visits every edge of G precisely once.
1 mark only for visits each edge once (ie not exactly/precisely)

(b) (3 marks) Complete the following C code function.

```
#define BAD_VERTEX -1

/*
** INPUTS: G is an n by n adjacency matrix of a graph
**         where G[u][v]=1 indicates an edge from u to v.
**         v is a vertex number (hence index into G).
** OUTPUTS: the in-degree of the supplied vertex v in G,
**          or BAD_VERTEX if v is not a valid index into G.
** SIDE EFFECTS: None.
*/
int
get_in_degree(int **G, unsigned int n, unsigned int v) {
```

ANSWER

```
    1 mark   if (v < 0 || v >= n) return BAD_VERTEX;
 0.25 mark   int count = 0;
  0.5 mark   for (i = 0 ; i < n ; i++)
  0.5 mark       if (G[i][v] == 1)
  0.5 mark           count++;
 0.25 mark   return count;                                  }
```

So one mark for using a counter (=0, ++, return),
0.5 mark for the loop
and 0.5 mark for getting in-degree, not out-degree.

Question 3 (18 marks).

(a) (5 marks) What is the Burrows Wheeler Transform of the string $abcabc\$$? Assume that \$ is less than all other characters.

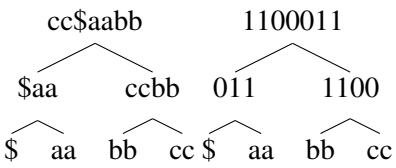
ANSWER

\$abcabc
abc\$abc
abcabc\$
bc\$abca
bcabc\$a
c\$abcab
cabc\$ab

(3 mark) for attempting to sort suffixes/rotations
(2 mark) BWT = cc\$aabb

(b) (5 marks) Draw the Wavelet Tree for your BWT string in (a) assuming the shape of the tree is based on the prefix code: \$=00, a=01, b=10, c=11
You should show both the bitvector and the character strings at each node, even though only the bitvectors are actually stored.

ANSWER



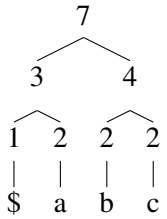
1 mark for the same root as given in their (a)
1 mark for the same shape as tree given in their (b)
3 marks 1 for each internal node correct

(c) (3 marks) Build a Huffman code on the frequencies of each character in the string from (a). Show the Huffman tree, and the final codewords for each character.

ANSWER

1 mark $f = \{1, 2, 2, 2\}$

1 mark



1 mark $c = \{\$ = 00, a = 01, b = 10, c = 11\}$

(d) (2 marks) Write down an expression for Shannon's entropy of probability distribution $p = \{p_1, p_2, \ldots, p_n\}$.

ANSWER

$-\sum_{i=1}^{n} p_i \log_2 p_i$.

0.5 mark	negative out the front
0.5 mark	some sum over all symbols
0.5 mark	some log base 2
0.5 mark	p_i's in the right spot

(e) (3 marks) Explain why it is impossible for a compression program to compress every possible input of length n bits to a size that is smaller than n bits so that the original n bits are recovered perfectly upon decompression.

ANSWER

1 mark	there are 2^n possible files
1 mark	there are 2^{n-1} possible compressed files
1 mark	pigeon hole principal (or some equivalent)

Question 4 (9 marks).

During Kruskal's greedy algorithm for building a Minimum Spanning Tree, each edge is added to the solution if it does not form a cycle in the tree that is constructed so far. A data structure is required to keep track of connected components in the partial solution during Kruskal's algorithm.

(a) (2 marks) What two operations must be supported by this data structure if it is to be used in Kruskal's algorithm? Give both the arguments and a short description of each operation.

ANSWER
1 mark Find(Vertex u) - find a label of the component containing u
1 mark Union(Vertex u, Vertex v) - the ability to join two connected components

(b) (5 marks) Describe an efficient data structure for the task, and include the big-O cost of each operation you mentioned in part (a) on this data structure for Kruskal's algorithm to find a MST on a graph of m edges and n vertices. If you have incorrect operations in (a), the maximum marks possible for this part is 3 out of 5.

ANSWER
Either LL with head pointers or Forest with/without path compression. 1 mark per big-O for union and find. 2 marks for description of data structure, (what is it, what are the labels returned?) and 1 mark for including some mention of a map indexing into the data structure.

(c) (2 marks) For some graph $G(V, E)$, what are the first edges added to the partial solution by Prim's greedy algorithm for building a Minimum Spanning Tree? Assume vertex $v \in V$ is processed first.

ANSWER
2 marks all edges leading out of v

Question 5 (9 marks).

Answer True (T) of False (F) to the following statements. You will **lose** one mark for an incorrect answer in this question. Do not guess. The minimum possible mark for this question is zero.

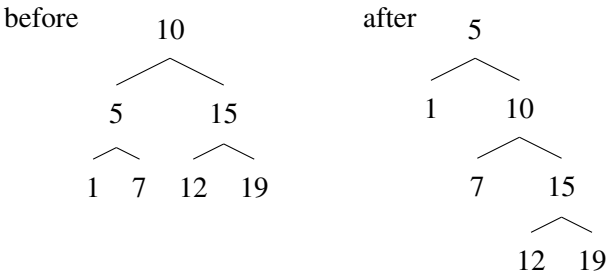
ANSWER

$f(n) = n^3$ is in $\Omega(n)$	T
$f(n) = n^3 \log n$ is in $O(n^3)$	F
$f(n) = n^2 / \log n$ is in $O(n^2)$	T
$f(n) = \log \log n$ is in $O(\log^* n)$	F
$f(n) = \sqrt{n}$ is in $\Omega(\log n)$	T
It is possible to devise an algorithm that can answer a `rank` query on a bitvector in $O(1)$ time using no more than $2n$ extra bits of space	T
It is possible to devise an algorithm that can sort an array of integers in $O(n)$ time if the maximum integer is restricted to be $O(n)$.	T
The travelling salesman problem is in class NP, but the minimum spanning tree problem is in P.	T
An n symbol string that is the result of a Burrows Wheeler Transform can be reversed in $O(n)$ time, regardless of the number of distinct symbols in the string.	T

Question 6 (14 marks).

(a) (2 marks) Draw this tree after rotating the root once to the right so that 5 becomes the new root.

ANSWER

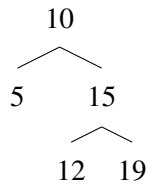
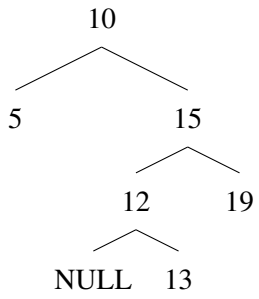
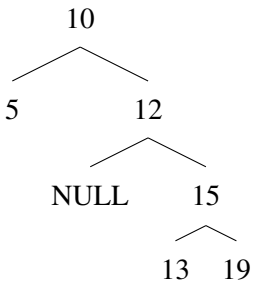
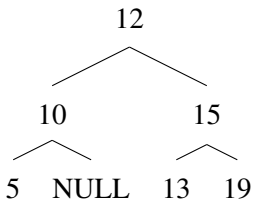


1 mark 1 as the left child

1 mark for 7 as the right-left child

(b) (5 marks) Draw this AVL tree after inserting 13. Show the tree before rebalancing and after each rotation.

ANSWER

Orig	After insert (1 mark)	After rot 1 (2 marks)	After rot 2 (2 marks)
			

Not necessary to show NULLs, or heights.

(c) (7 marks) Complete the following table for each of these data structures using big-Oh expressions assuming they contain n items.

ANSWER

What is the space required for a hash table that uses Open Addressing and has m slots?	$O(m)$
What is the space required for an AVL tree?	$O(n)$
What is the worst case time to find an item in an AVL tree?	$O(\log n)$
How many bits are required to store a Wavelet Tree based on a prefix code with all codewords of $\lceil \log_2 n \rceil$ bits?	$O(n \log n)$
What is the worst case time required to find the successor of an item in an AVL tree?	$O(\log n)$
What is the worst case time required to find the successor of an item in a Hash Table that has m slots and uses Open Addressing for collision resolution?	$O(m)$

Question 7 (12 marks).

(a) (3 marks) What are the attributes of a problem that would indicate that applying Dynamic Programming might lead to an efficient (polynomial) algorithm to solve the problem?

ANSWER

0.5 marks	Problem can be decomposed into sub-problems
0.5 marks	The solution can be composed of the solutions to the sub-problems
0.5 marks	The sub-problems overlap (repetition)
0.5 marks	The sub-problems can be dependency ordered
1 marks	There is a polynomial number of sub-problems

(b) (6 marks) Given two strings $x[1..n]$ and $y[1..m]$, the Edit Distance between the two strings can be calculated as $E(n, m)$, where
$E(0, j) = j$, $E(i, 0) = i$;
$\delta(i, j) = 1$ when $x[i] \neq y[j]$ and 0 otherwise; and
$E(i, j) = \min\{1 + E(i - 1, j), 1 + E(i, j - 1), \delta(i, j) + E(i - 1, j - 1)\}$ when $i \in [1, n]$ and $j \in [1, m]$.

Using the Dynamic Programming technique, give pseudo code for a polynomial time algorithm for computing $E(n, m)$.

ANSWER

1 marks	Allocate a two dimensional table[n][m]
1 marks	Set $t[0][*] = 0$ and $t[*][0] = 0$
0.5 marks	for i in 1 to n
0.5 marks	for j in 1 to m
0.5 marks	$a = 1 + t[i - 1][j]$
0.5 marks	$b = 1 + t[i][j - 1]$
1 marks	$c = (x[i] \neq y[j]) + t[i - 1][j - 1]$
0.5 marks	$t[i][j] = \min(a, b, c)$
0.5 marks	return $t[n][m]$

2 marks for mentioning table and initialise

1 mark for double loop over table

1 mark for the two easy cases

1 mark for the harder case

1/2 mark for min of three cases

1/2 mark for saying $t[n][m]$ is answer

(c) (1 mark) What is a tight upper bound on the worst case running time of your algorithm in (c)?

ANSWER

1 marks $O(nm)$

(d) (1 mark) What is a tight lower bound on the best case running time of your algorithm in (c)?

ANSWER

1 marks $O(nm)$

(e) (1 mark) Given your answers to (d) and (e), is there a precise statement you can make about the running time of your algorithm that holds for best, average and worst case inputs? If so, what is it?

ANSWER

1 marks $\Theta(nm)$

Award the mark if wrong but consistent with (d) and (e).

Overflow Answers

The boxes here are for emergency use only. If you do need to use this page, indicate **CLEARLY** in your previous answer that you have continued onto this page. Without such an indication, it is possible that this part of your answer will be overlooked.