Faculty of Science
Final Examination

Computer Science 308-251B
Data Structures and Algorithms

Examiner: Prof. Claude Crépeau  
Associate Examiner: Lecturer Martin Courchesne

Date: April 20, 2005  
Time: 14:00 – 17:00

INSTRUCTIONS:
• This examination is worth 50% of your final grade.
• The total of all questions is 105 points.
• Each question is assigned a value found in brackets next to it.
• OPEN BOOKS • OPEN NOTES
• Faculty standard calculator permitted only.
• This examination consists of 4 pages including title page.
• This examination consists of 7 questions.

SUGGESTION : read all the questions and their values before you start.
1) As promised!

Let
\[
T(n) = \begin{cases} 
1 & \text{if } n=1,2 \\
T(\left\lfloor \frac{n}{4} \right\rfloor) + T(\left\lfloor \frac{2n}{3} \right\rfloor) + \Theta(n) & \text{if } n>2
\end{cases}
\]

Prove using constructive induction that \( T(n) \) is \( \Theta(n) \).

2) Consider the following graph.

Find the shortest path from source \( s \) to every vertex \( v \in \{s,a,b,c,d,e\} \) in the graph.
Choose the algorithm you wish to use, and show the step by step progression of the algorithm on this instance.

3) Consider an infinite family of graphs \( \{G_i=(V_i,E_i,W_i)\}_{i>0} \) such that \( |V_i|=i \), and \( W_i \) is the weight function of the edges in \( E_i \). Assume the weight of any edge \( (u,v) \) in \( E_i \) is such that \( 0 < w_i(u,v) < |E_i| \). Explain how to modify Kruskal’s algorithm so that it is asymptotically a better choice than (unmodified) Prim’s algorithm. Compare the running times.

**Hint:** first understand why \( w_i(u,v) < |E_i| \) makes a difference; what is more efficient then?

4) Consider ways of computing the values of the binomial coefficients \( \binom{n}{k} \). First remember that \( \binom{n}{1} = n \div k! \left( n-k \right)! \), and a basic property of binomial coefficients \( \binom{n}{k} = \binom{n-1}{k-1} + \binom{n-1}{k} \).

A) Show that if we blindly use this recursive formula and the fact that \( \binom{n}{1}=n \), it will take exponential time in \( n \) to compute \( \binom{n}{k} \) for certain values of \( k \).

B) Give a dynamic programming algorithm to compute \( \binom{n}{k} \) and analyze its running time.
5) Consider the following table of frequencies of Café’s in Montréal:

<table>
<thead>
<tr>
<th>Café</th>
<th>Acronym</th>
<th>Frequency</th>
</tr>
</thead>
<tbody>
<tr>
<td>Brûlerie St-Denis</td>
<td>BD</td>
<td>14%</td>
</tr>
<tr>
<td>Café Dépôt</td>
<td>CD</td>
<td>4%</td>
</tr>
<tr>
<td>Presse-café</td>
<td>PC</td>
<td>9%</td>
</tr>
<tr>
<td>Second Cup</td>
<td>SC</td>
<td>22%</td>
</tr>
<tr>
<td>Starbuck Cofee</td>
<td>SC</td>
<td>2%</td>
</tr>
<tr>
<td>Tim Hortons</td>
<td>TH</td>
<td>20%</td>
</tr>
<tr>
<td>Van Houtte</td>
<td>VH</td>
<td>13%</td>
</tr>
<tr>
<td>All Others</td>
<td>AO</td>
<td>16%</td>
</tr>
</tbody>
</table>

Build a Huffman tree from these frequencies and use these acronyms as labels of vertices. Show the step by step progression of the HUFFMAN algorithm on this instance.

6) For each statement, say if it is true or false.

<table>
<thead>
<tr>
<th>Correct</th>
<th>Incorrect</th>
<th>No answer</th>
<th>Minimum Total</th>
</tr>
</thead>
<tbody>
<tr>
<td>+1 pt</td>
<td>-0.5 pt</td>
<td>0 pt</td>
<td>0 pt</td>
</tr>
</tbody>
</table>

(a) Red-Black trees are important to get better sorting algorithms.
(b) Finding large prime numbers is mathematically beautiful but practically useless.
(c) The Longest Common Subsequence of X and Y is unique.
(d) The “disjoint sets” data structures of chapter 21 are fundamental to Prim’s algorithm for Minimum Spanning Trees.
(e) We can easily adapt Strassen’s sub-cubic algorithm for matrix multiplication to improve All-pairs Shortest Paths algorithms.
(f) In an array of unsorted integers we can find the element of rank $\lfloor \sqrt{n} \rfloor$ in worst case time $O(n)$.
(g) The worst case to the basic “HIRE-ASSISTANT” algorithm of Chapter 5 is when the candidates are presented in increasing order of quality.
(h) In Radix sort it is mandatory that the underlying sorting algorithm be stable.
(i) The family of hash functions defined as $Ax \oplus b$ on n-bit vectors x, where A is an nxn invertible binary matrix, and b an arbitrary n-bit vector, is universal.
(j) A tree is a forest.
Consider a sorting algorithm A such that the expected running time of A is $< a \cdot n \log n$ to sort any array of $n$ elements. Suppose another sorting algorithm C is such that the worst case running time of C is $< c \cdot n \log n$ to sort any array of $n$ elements, but $c$ is much larger than $a$.

A) Show that the probability that algorithm A will take more than $c \cdot n \log n$ time is at most $a/c$.

Consider the algorithm B that runs A upto $5c \cdot n \log n$ steps and then runs C instead.

B) Show that the worst case running time of B is no more than $6c \cdot n \log n$.

C) Show that the expected running time of B is no more than $(1.2) \cdot a \cdot n \log n$.

D) Conclude from this example that for any $\varepsilon > 0$, we can design from A and C another algorithm D such that the worst case running time of D is $O(n \log n)$, while its expected running time is no more than $(1+\varepsilon) \cdot a \cdot n \log n$. 

- 4 -