CS 456: Advanced Algorithms

Fall 2014
Midterm Exam

Instructions

• Write your name and the last 3-digits of your SIUE ID in the space provided above
• This is a closed-book, closed-note examination
• There are 7 total questions printed on 12 pages (including the title page). Please confirm that all pages are present
  - All students must answer Q1-Q5
  - Master’s students must answer Q6
  - Q7 is an optional bonus question that all students can attempt
• Total points: 120* points for BS Students; 140* points for MS Students
• Total time: 1 hour and 15 mins

*Including the bonus points
Short Answers

Q1. [20 points] State whether the following statements are true or false and provide a 1 sentence justification if false

a. [2 points] A function with a faster order of growth is more efficient than a one with a slower order of growth

\[ \text{false. faster order of growth means less efficient} \]

b. [2 points] If \( f(n) \in O(g(n)) \) and \( h(n) \in \Omega(g(n)) \) then \( f(n) \in \Omega(h(n)) \)

\[ \text{false. } h(n) \in \Omega(g(n)) \Rightarrow g(n) \in O(h(n)) \text{ thus, } f(n) \in O(h(n)) \]

c. [2 points] If \( f(n) \in \Theta(g(n)) \) then \( g(n) \in \Theta(f(n)) \)

\[ \text{true. they are both in the same complexity class.} \]

d. [2 points] Partial correctness = total correctness + termination

False \[ T \cdot C = P \cdot C + \text{termination} \]

e. [2 points] By definition, a spanning tree is cyclic

False \[ \text{acyclic} \]

f. [2 points] Divide-and-conquer strategy always chooses the locally optimal solution

False \[ \text{Greedy} \]

g. [2 points] Both dynamic programming and greedy strategies solve problems by dividing them into several overlapping subproblems

False \[ \text{Dynamic Programming and D-n-P.} \]

h. [2 points] Kruskal's algorithm greedily grows a minimum spanning tree while Prim's algorithm greedily combines a forest to arrive at a minimum spanning tree

False \[ \text{Differing around l.} \]

i. [2 points] Codewords that are prefixes of other codewords are called prefix codes

False \[ \text{Prefix codes are actually prefix-free.} \]
j. **[2 points]** Dijkstra's algorithm applies to both directed and undirected graphs

> True although undirected graphs may not yield efficient use of Dijkstra.

**Long Answers**

Q2. **[20 points]** For each of the following cases, calculate whether \( f(n) \in O(g(n)) \) or \( f(n) \in \Omega(g(n)) \) or \( f(n) \in \Theta(g(n)) \).

a. \( f(n) = \sqrt{n}, g(n) = \log(n) \)

\[
\lim_{n \to \infty} \frac{\sqrt{n}}{\log n} = \lim_{n \to \infty} \frac{1}{2} \frac{n^{1/2}}{\log n} = \lim_{n \to \infty} \left( \frac{n^{1/2}}{2} \right) \sqrt{n} = \infty
\]

thus, \( f(n) \in \Omega(g(n)) \)

b. \( f(n) = \log(n^2), g(n) = \log(n)^2 \)

\[
\lim_{n \to \infty} \frac{\log(n^2)}{\log(n)^2} = \lim_{n \to \infty} \frac{2 \log n}{\log(n)^2} = \lim_{n \to \infty} \frac{2}{\log n} = 0
\]

thus \( f(n) \in O(g(n)) \)
c. \( f(n) = n^32^n, g(n) = 3^n \)

\[
\lim_{{n \to \infty}} \frac{n^32^n}{3^n} = \lim_{{n \to \infty}} \left( \frac{2}{3} \right)^n \xrightarrow{n \to \infty} 0
\]

\( f(n) \in O(g(n)) \)

\[f(n) \in O(g(n))\]

d. \( f(n) = (n + 1)!, g(n) = n! \)

\[
\lim_{{n \to \infty}} \frac{(n+1)!}{n!} = \lim_{{n \to \infty}} (n+1) \xrightarrow{n \to \infty} \infty
\]

\[f(n) \in \Omega(g(n))\]
Q3. [20 points] Greedy Strategy

<table>
<thead>
<tr>
<th></th>
<th>a</th>
<th>b</th>
<th>c</th>
<th>d</th>
<th>e</th>
<th>f</th>
</tr>
</thead>
<tbody>
<tr>
<td>a</td>
<td>0</td>
<td>3</td>
<td>0</td>
<td>1</td>
<td>6</td>
<td>5</td>
</tr>
<tr>
<td>b</td>
<td>3</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>c</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>4</td>
</tr>
<tr>
<td>d</td>
<td>0</td>
<td>6</td>
<td>0</td>
<td>8</td>
<td>5</td>
<td></td>
</tr>
<tr>
<td>e</td>
<td>6</td>
<td>0</td>
<td>0</td>
<td>8</td>
<td>0</td>
<td>2</td>
</tr>
<tr>
<td>f</td>
<td>5</td>
<td>4</td>
<td>4</td>
<td>5</td>
<td>2</td>
<td>0</td>
</tr>
</tbody>
</table>

a. [10 points] Use Kruskal’s algorithm to find the MST for graph represented by the above adjacency list. Show each intermediate step.
b. [10 points] Construct a Huffman code for the following data. Show your work.

<table>
<thead>
<tr>
<th>symbol</th>
<th>δ</th>
<th>θ</th>
<th>γ</th>
<th>a</th>
<th>else</th>
</tr>
</thead>
<tbody>
<tr>
<td>frequency</td>
<td>20</td>
<td>10</td>
<td>25</td>
<td>40</td>
<td>5</td>
</tr>
</tbody>
</table>

```
5 20 25 40 40
```

```
15
  __________
  | else |
  | θ    |
```

```
35
  __________
  | else |
  | γ    |
```

```
60
  __________
  | else |
  | δ    |
```

```
100
  __________
  | else |
  | τ    |
```

```
35
  __________
  | else |
  | α    |
```
Q4. **[20 points]** Divide-and-Conquer

a. **[10 points]** Find the order of growth for the following recurrences. Assume that $T(n)$ is constant for $n \leq 2$. Justify your answers.

i. $T(n) = 2T(n/4) + \sqrt{n}$

\[ a = 2 \]
\[ b = 4 \]
\[ n^{\log_b a} = n^{\log_4 2} = \sqrt{n} \]
\[ f(n) = \sqrt{n} \Rightarrow f(n) = \Theta(n^{\log_b a}) \]

**Case 2**: $T(n) \in \Theta(\sqrt{n \log n})$

ii. $T(n) = 2T(n/2) + n^2$

\[ a = 2 \]
\[ b = 2 \]
\[ n^{\log_b a} = n \]
\[ f(n) = n^2 \Rightarrow f(n) = \Omega(n^{\log_b a + \varepsilon}) \text{ where } \varepsilon = 1. \]

For sufficiently large $n$, $af(n/b) = 2 \cdot (\frac{n}{2})^2 = \frac{1}{2} n^2 < \alpha f(n)$

with $\varepsilon = 1$.

$T(n) = \Theta(n^2)$
b. [10 points] Develop a recurrence as a function of $n$ to determine the number of times
the following code prints 'Jumbo!'. Solve your recurrence using a recurrence tree. (You
may assume $n = 2^k$)

```cpp
void jumbo(int n)
{
    if (n > 1)
    {
        jumbo(n/2);
        jumbo(n/2);
        for (i = O; i < n; i++)
        {
            cout << "Jumbo!" << endl;
        }
    }
}
```

$$T(n) = 2T \left( \frac{n}{2} \right) + cn.$$
Q5. [20 points] Present the optimal order of parenthesizing $M_1 \times M_2 \times M_3 \times M_4 \times M_5 \times M_6$ using the following dynamic programming tabulation. Show each step clearly.

<table>
<thead>
<tr>
<th></th>
<th>$m_{1,6}$</th>
<th>$m_{1,5}$</th>
<th>$m_{1,4}$</th>
<th>$m_{1,3}$</th>
<th>$m_{1,2}$</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>best $k$</td>
<td>best $k$</td>
<td>best $k$</td>
<td>best $k$</td>
<td>best $k$</td>
</tr>
<tr>
<td>$m_{1,6}$ =</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$m_{1,5}$ =</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$m_{1,4}$ =</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$m_{1,3}$ = 330</td>
<td>$m_{2,4}$ = 330</td>
<td>$m_{3,5}$ = 240</td>
<td>$m_{4,6}$ = 480</td>
<td></td>
<td></td>
</tr>
<tr>
<td>best $k$ = 3</td>
<td>best $k$ = 3</td>
<td>best $k$ = 5</td>
<td>best $k$ = 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$m_{1,2}$ = 150</td>
<td>$m_{2,3}$ = 360</td>
<td>$m_{3,4}$ = 180</td>
<td>$m_{4,5}$ = 240</td>
<td>$m_{5,6}$ = 120</td>
<td></td>
</tr>
<tr>
<td>best $k$ = 1</td>
<td>best $k$ = 2</td>
<td>best $k$ = 3</td>
<td>best $k$ = 4</td>
<td>best $k$ = 5</td>
<td></td>
</tr>
<tr>
<td>$m_{1,1}$ = 0</td>
<td>$m_{2,2}$ = 0</td>
<td>$m_{3,3}$ = 0</td>
<td>$m_{4,4}$ = 0</td>
<td>$m_{5,5}$ = 0</td>
<td>$m_{6,6}$ = 0</td>
</tr>
<tr>
<td>[5x10]</td>
<td>[10x3]</td>
<td>[3x12]</td>
<td>[12x5]</td>
<td>[5x4]</td>
<td>[4x6]</td>
</tr>
</tbody>
</table>

$$m_{1,3} = \min \left\{ M_1 \times (M_2 \times M_3) , (M_1 \times M_2) \times M_3 \right\} = \min \left\{ \begin{array}{l} 150 + 5 \times 10 \times 12 \ 360 \end{array} \right\} = 150 + 180 = 330$$

$$m_{2,4} = \min \left\{ M_3 \times (M_4 \times M_5) , (M_2 \times M_3) \times M_4 \right\} = \min \left\{ \begin{array}{l} 180 + 180 \ 360 + 10 \times 3 \times 5 \end{array} \right\} = 180 + 180 = 330$$

$$m_{3,5} = \min \left\{ M_3 \times (M_4 \times M_5) , (M_3 \times M_4) \times M_5 \right\} = \min \left\{ \begin{array}{l} 180 + 3 \times 12 \times 4 \ 240 \end{array} \right\} = 180 + 240 = 330$$

$$m_{4,6} = \min \left\{ M_3 \times (M_5 \times M_6) , (M_4 \times M_5) \times M_6 \right\} = \min \left\{ \begin{array}{l} 360 + 120 \ 240 + 12 \times 4 \times 6 \end{array} \right\} = 360 + 120 = 480.$$
Q6. **[20 points]** Asymptotic Order of Growth

a. **[5 points]** Let \( f(n) = 4n^2 + 9n - 81 \) and \( g(n) = n^2 \). Find \( c_1, c_2, \) and \( n_0 \) to formally show that \( f(n) \in \Theta(g(n)) \)

\[
\begin{align*}
c_1 n^2 &\leq 4n^2 + 9n - 81 \leq c_2 n^2 \\
0 &\leq c_1 \leq 4 + \frac{9}{n} - \frac{81}{n^2} \leq c_2 \\
\end{align*}
\]

\[ n = 9, \Rightarrow c_1 \leq 4 + \frac{9}{9} - 1 \leq c_2 \]

\[ c_1 \leq 4 \leq c_2 \]

b. **[5 points]** Let \( h(n) = n^3 \). Formally show that \( f(n) \notin \Omega(h(n)) \)

\[
\begin{align*}
4n^2 + 9n - 81 &> c_1 n^3 \\
0 &\leq c_1 \leq 4 + \frac{9}{n} - \frac{81}{n^2} \\
0 &\leq c_1 \leq 4 + \frac{9}{n} - \frac{81}{n^3} \\
\end{align*}
\]

\[ \text{there is no } c_1 \text{ can be found for all } n > n_0 \]

\[ n_0 = 9 \]

\[
\begin{align*}
c_1 &= 3 \\
c_2 &= 5 \\
\end{align*}
\]

c. **[5 points]** Consider two algorithms with running times \( T_A(n) = \frac{1}{2} n^2 \log_2 n \) and \( T_B(n) = 100n^2 \). At what value of \( n \) does algorithm \( A \) start to take less time than algorithm \( B \).

\[
\begin{align*}
\frac{1}{2} n^2 \log_2 n &\leq 100n^2 \\
\log n &\leq 200 \\
n &\leq 2^{200}. \text{ up to } n = 2^{200} \text{, } T_A \text{ takes less time than } T_B, \text{ after which it starts to take more time.} \\
\end{align*}
\]

d. **[5 points]** If a step in algorithm \( A \) takes 1 ns, how much is the above runtime in years?

\[
\begin{align*}
\frac{1}{2} (2^{200})^2 \cdot \log_2 2^{200} &\text{ seconds per steps} \\
&= 100 \times 2^{400} \times 10^{-9} \text{ seconds} \\
&= \text{seconds per year.}
\end{align*}
\]
Extra Credit

Q7. [20 points] Algorithmic correctness proofs  
   a. Using mathematical induction, prove that $6^n - 1$ is divisible by 5 for any $n \geq 1$.
   
   **Base case** $n=1$  
   $\frac{6-1}{5} = 1 \checkmark$

   **I.H:** $n=k \implies \frac{6^k-1}{5}$ divisible by 5

   **I. Step**  
   $6^{k+1}-1 = 6(6^k-1)$
   
   $6 \left( \frac{6^k-1}{5} \right) + 6 - 1 \implies 6 \left( \frac{6^k-1}{5} \right) + 5$ divisible by 5.
b. Prove the correctness of the following pseudocode by considering the invariant $[I: \text{At the beginning of } j\text{th iteration, } A[1\ldots j-1] \text{ is sorted and is a permutation of the original } A[1\ldots j-1]]$

$$I$$

$$\text{for } j = 2 \rightarrow A\text{.length do}$$
$$\text{(I } \land j \leq A\text{.length)}$$
    $$key = A[j]$$
    $$i = j - 1$$
    $$\text{while } (i > 0) \land A[i] > key \text{ do}$$
    $$A[i + 1] = A[i]$$
    $$i = i - 1$$
$$\text{end}$$
$$A[i + 1] = key$$
$$\text{(I)}$$
$$\text{end}$$
$$\text{(I } \land j = A\text{.length)}$$

Initialization: $j = 1 \quad A[1\ldots i] \text{ trivially sorted}$

Check your notes. This problem was discussed in class.