**CMSC 451 Homework 2**

1. Given the following two functions:

* + *f(n) = 3n2 + 5*
  + *g(n) = 53n + 9*

Use L’Hopital’s rule and limits to prove or disprove each of the following:

* + *f* ∈ Ω(*g*)
  + *g* ∈ ϴ(*f*)

**SOLUTION**

*f* ∈ Ω(*g*)

Definition of Big-Omega:

Finding value of constant where n ≥ 1 is true:

If

**Base Case Test**

🡪

Adding the inequalities gives us the inequality that satisfies the definition of Big-Omega:

L’Hopital’s rule (Alternative Proof):

This shows that f(n) is growing faster than g(n) and thus the statement *f* ∈ Ω(*g*) is **true**.

*g* ∈ ϴ(*f*)

Definition of Big-Theta:

Finding value of constant where n ≥ 1 is true:

There is no value for that will make this statement true for

L’Hopital’s rule (Alternative Proof):

🡨 Limit cannot be 0

This shows that g(n) is growing slower than f(n) and thus the statement *g* ∈ ϴ(*f*)is **false**, because for the statement to be true both g(n) and f(n) need to grow at the same rate.

1. Rank the following functions from lowest asymptotic order to highest. List any two or more that are of the same order on the same line.

**SOLUTION**

**Asymptotic Order** **Big-O Notation**

* 🡨 Bases of log functions are constants



* , 🡨 n3 is the fastest growing for both

Here is a useful link for asymptotic ordering:

<https://en.wikipedia.org/wiki/Big_O_notation#Orders_of_common_functions>

1. Draw the recursion tree when *n* = 8, where *n* represents the length of the array, for the following recursive method:

int sum(int[] array, int first, int last)

{

if (first == last)

return array[first];

int mid = (first + last) / 2;

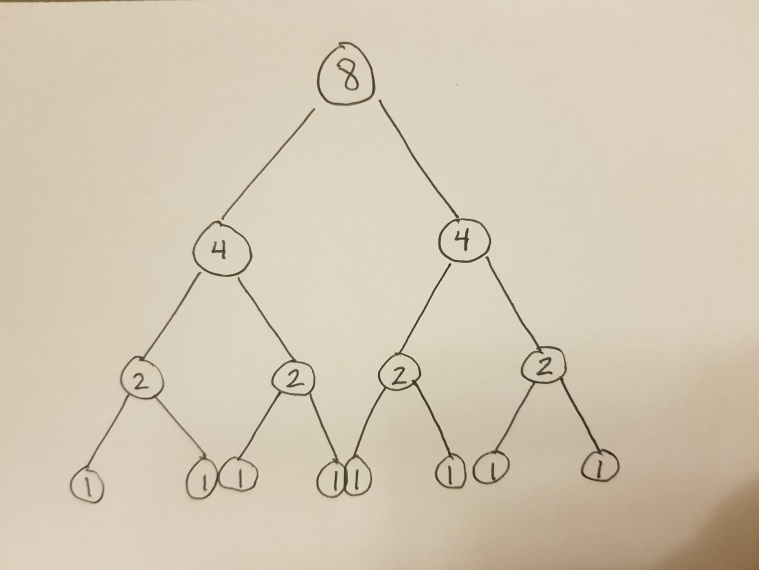
return sum(array, first, mid) + sum(array, mid + 1, last);

}

* Determine a formula that counts the numbers of nodes in the recursion tree.
* What is the Big-ϴ for execution time?
* Determine a formula that expresses the height of the tree.
* What is the Big-ϴ for memory?
* Write an iterative solution for this same problem and compare its efficiency with this recursive solution.

**SOLUTION**

**Recursion Tree**



**Formula for Number of Nodes**

**Big-ϴ for Execution Time**

Two recursive calls to sum function and one calculation of mid in constant time gives:

Recurrence Relation:

According to Part 1 of Master Theorem: a = 2, b = 2, and f(n) = ϴ(1)

**Formula for Height of Tree**

**Big-ϴ for Memory**

**Iterative Solution**

int sum(int[] array, int first, int last) Time Complexity

{

int sum = 0; ϴ(1)

for (int i = first; i <= last; i++) ϴ(1)

sum += array[i]; ϴ(n)

return sum; ϴ(1)

}

Execution Time: ϴ(n)

Memory Space: ϴ(1) 🡨 Constant for iterative functions

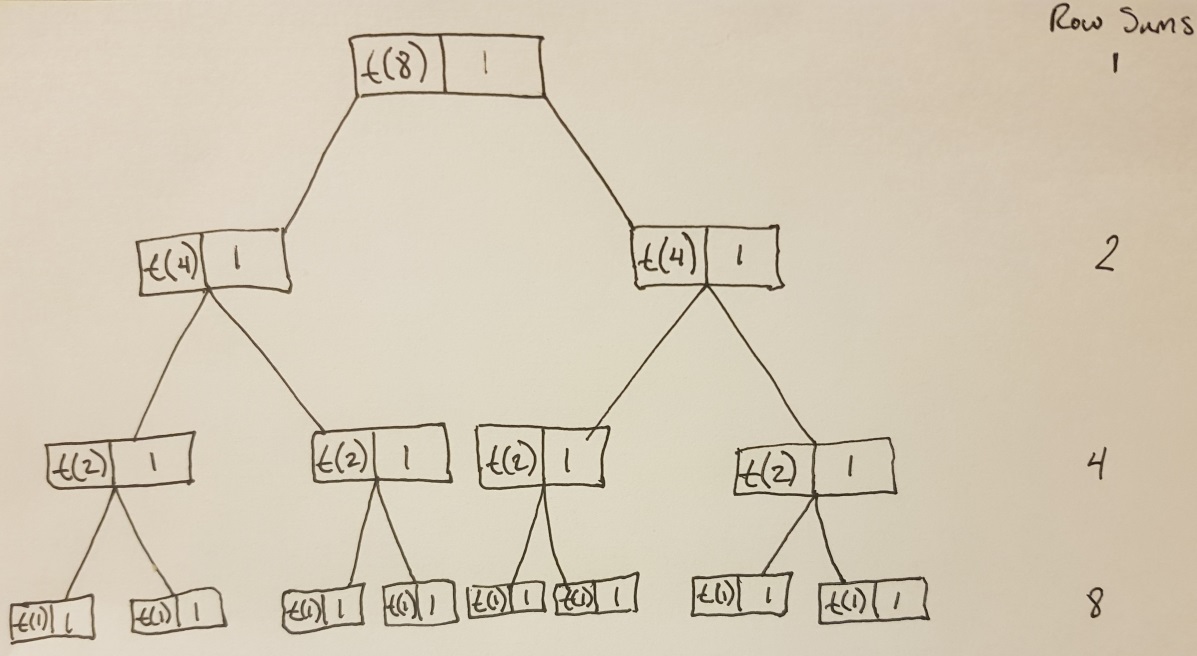
Thus, the recursive and iterative functions have the same execution time, but the iterative uses memory more efficiently.

4. Using the recursive method in problem 3 and assuming *n* is the length of the array.

* Modify the recursion tree from the previous problem to show the amount of work on each activation and the row sums.
* Determine the initial conditions and recurrence equation.
* Determine the critical exponent.
* Apply the Little Master Theorem to solve that equation.
* Explain whether this algorithm optimal.

**SOLUTION**

**Modified Recursion Tree**



**Initial Condition**

if (first == last)

return array[first];

, when only one element in array exists.

**Recurrence Equation**

🡨 Explained in Question 3

**Critical Exponent**

branching factor (b) = 2

cutting factor (c) = 2

**Solving Equation**

🡨 More detail in Question 3

The lower bound for this algorithm is and as such, this algorithm is optimal.

**Grading Rubric**

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
|  | **Problem** |  | **Meets** |  | **Does Not Meet** |  |
|  |  |  | **10 points** |  | **0 points** |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  | Used L'Hopital's rule to correctly |  | Did not use L'Hopital's rule to correctly |  |
|  | **Problem 1** |  | determine limits (2) |  | determine limits (0) |  |
|  |  |  |  |  |  |
|  |  | Provided correct proof or disproof of |  | Did not provide correct proof or |  |
|  |  |  |  |  |
|  |  |  | Big-Theta relationship (4) |  | disproof of Big-Theta relationship (0) |  |
|  |  |  |  |  |  |  |
|  |  |  | Provided correct proof or disproof of |  | Did not provide correct proof or |  |
|  |  |  | Big-Omega relationship (4) |  | disproof of Big-Omega relationship (0) |  |
|  |  |  |  |  |  |  |
|  |  |  | **10 points** |  | **0 points** |  |
|  |  |  |  |  |  |  |
|  |  |  |  |  |  |  |
|  |  |  | Positioned exponential functions |  | Did not position exponential functions |  |
|  |  |  | correctly (2) |  | correctly (0) |  |
|  |  |  |  |  |  |  |
|  |  |  | Positioned polynomial functions |  | Did not position polynomial functions |  |
|  | **Problem 2** |  | correctly (2) |  | correctly (0) |  |
|  |  |  |  |  |  |
|  |  | Positioned constant functions correctly |  | Did not position constant functions |  |
|  |  |  |  |  |
|  |  | (2) | |  | correctly (0) |  |
|  |  |  |  |  |  |  |
|  |  |  | Positioned logarithmic functions |  | Did not position logarithmic functions |  |
|  |  |  | correctly (2) |  | correctly (0) |  |
|  |  |  |  |  |  |  |
|  |  |  | Positioned root functions correctly (2) |  | Did not position root functions |  |
|  |  |  |  |  | correctly (0) |  |
|  |  |  |  |  |  |  |

|  |  |  |
| --- | --- | --- |
|  | **10 points** | **0 points** |
|  |  |  |
|  |  |  |
|  | Correctly drew recursion tree (2) | Did not correctly draw recursion tree |
|  |  | (0) |
|  |  |  |
|  | Provided correct formula for number of | Did not provide correct formula for |
|  | nodes (2) | number of nodes (0) |
|  |  |  |
|  | Provided correct Big-Theta for | Did not provide correct Big-Theta for |
|  | execution time (1) | execution time (0) |
|  |  |  |
| **Problem 3** | Provided correct formula for tree | Did not provide correct formula for |
|  | heights (2) | tree heights (0) |
|  |  |  |
|  | Provided correct Big-Theta for memory | Did not provide correct Big-Theta for |
|  | (1) | memory (0) |
|  |  |  |
|  | Wrote correct iterative solution (1) | Did not write correct iterative solution |
|  |  | (0) |
|  |  |  |
|  | Provided correct comparison of | Did not provide correct comparison of |
|  | efficiency of recursive and iterative | efficiency of recursive and iterative |
|  | solutions (1) | solutions (0) |
|  |  |  |
|  | **10 points** | **0 points** |
|  |  |  |
|  |  |  |
|  | Correctly drew modified recursion tree | Did not correctly draw modified |
|  | (2) | recursion tree (0) |
|  |  |  |
|  | Provided correct initial condition (1) | Did not provide correct initial condition |
|  |  | (0) |
|  |  |  |
|  | Provided correct recurrence equation | Did not provide correct recurrence |
| **Problem 4** | (2) | equation (0) |
|  |  |  |
|  | Provided correct critical exponent (1) | Did not provide correct critical |
|  |  | exponent (0) |
|  |  |  |
|  | Correctly applied Little Master | Did not correctly apply Little Master |
|  | Theorem to correctly solve recurrence | Theorem to correctly solve recurrence |
|  | equation (3) | equation (0) |
|  | Provided correct explanation of | Did not provide correct explanation of |
|  | whether algorithm is optimal (1) | whether algorithm is optimal (0) |
|  |  |  |