Wollongong Joint Institute of Central China Normal University

JICSCI803 Algorithms and Data Structures

2020

Assignment 1 (Due 11:59 PM <10 days> 30th October)

Objectives

Utilise mathematical concepts related to the study of algorithms Analyse
the correctness and complexity of algorithms.

Questions

1. (1 mark) Using only the exponent and log laws given in lectures, prove that

$$\log_b(x^n) = n \log_b x$$

for real numbers x, $b \neq 0$ and integer n> 0.

2. (4 marks)

(a) Using $k > 2+k^{-1}$, prove that $2k^2 > (k + 1)^2$. For what integer values of k is this proof valid?

(b) Prove by induction that $2^n/2n > n$ for all integers n> 6

(c) Use the above result to prove from first principles that, for integer n,

$$\lim_{m\to\infty} \frac{2^n}{2n} = \infty_+$$

(d) What result do you expect for

$$\lim_{m\to\infty} \frac{\log_2 m}{\sqrt{m}}$$

explain your reasoning.

3. (5 marks) A community of N pirates has recently conducted an election to choose their new leader. All pirates vote, and any pirate may run as a candidate. There is no preferential system, each pirate simply writes the number of their preffered leader on the ballot paper. If a single pirate gets more than 50% of the votes, then that pirate is declared the new leader. If no pirate gets more than 50% of the votes, then the old leader is retained.

N pirates have voted, and their choices have been collected in an array A. Your task is to determine whether there will be a new leader and, if there will, who the new leader will be. For example, given the array

4, 4, 7, 1, 7, 7, 2, 7, 7

pirate 7 has 5 votes out of 9, and becomes the new leader, whereas given the array

4, 4, 7, 1, 4, 7, 2, 4

pirate 4 has 4 votes out of 8 but doesn't manage to get over 50% so the old leader is retained.

Your team has proposed the following algorithm to determine which, if any, candidate wins:

First, a possible winner must be found. This possible winner the only candidate that could possibly have more than 50% of the votes. To find a possible winner in the array, A, create a second array, B. Compare A[0] and A[1]. If they are equal, put one of them in B; otherwise do nothing. Then compare A[2] and A[3]. Again if they are equal, add one of these to B; otherwise do nothing. Continue like this until the entire array A has been used. Recursively find a possible winner on the array B, stopping when the array has less than 2 elements. If a possible winner has been found, scan through the array and count the number of votes received by the possible winner. If the possible winner has more than N/2 votes, print the possible winner out as the winner. If no possible winner is found, or the possible winner does not have enough votes, print that the old leader is retained.

(a) What is the worst case space complexity for this algorithm (consider the array(s) B only)? Explain your reasoning.

(b) Give the O, Ω and, if possible, Θ time complexities for this algorithm. Explain your reasoning.

(c) Why does this algorithm work? You may assume the array size is even for all function calls.

(d) What problem occurs when the array size is odd? Propose a fix for this problem, or describe an alternative algorithm.

(e) What are the (Big-Oh) time and space[1] complexities of your new algorithm? Compare this to the previous algorithm and explain under what circumstances would you use one algorithm or the other?

[1] not including the initial array

Submission instructions

Your completed assignment should be a readable pdf file.

1 Late submissions will be marked with a 25% deduction for each day.

2 Submissions more than three days late will not be marked, unless an extension has been granted.

3 If you need an extension apply through SOLS, if possible before the assignment deadline.

4 Plagiarism is treated seriously. If we suspect any work is copied, all students involved are likely to receive zero for the entire assignment.