Homework 2

CSE321 - Introduction to Algorithm Design

Deadline: 21/11/2021 - 23:55

1) Solve the following recurrence relations by using **Master theorem**. If any of the problems cannot be solved by using Master theorem, state that it cannot be solved and explain the reason.

- **a**) $T(n) = 16T(\frac{n}{4}) + n!$
- **b**) $T(n) = \sqrt{2}T(\frac{n}{4}) + \log n$
- **c**) $T(n) = 8T(\frac{n}{2}) + 4n^3$
- **d**) $T(n) = 64T(\frac{n}{8}) - n^2 \log n$
- **e**) $T(n) = 3T(\frac{n}{3}) + \sqrt{n}$
- **f**) $T(n) = 2^n T(\frac{n}{2}) - n^n$
- **g**) $T(n) = 3T(\frac{n}{3}) + \frac{n}{\log n}$

2) Consider the following three algorithms:

- **a**) Algorithm X solves problems by dividing them into nine subproblems that have one-third of the size, recursively solving each subproblem, and then combining the solutions in quadratic time.
- **b**) Algorithm Y solves problems of size n by recursively solving 8 subproblems that have half the size and then combining the solutions in $O(n^3)$.
- **c**) Algorithm Z solves problems of size n by dividing them into two subproblems that have a quarter of the size, recursively solving each subproblem, and then combining the solutions in $O(\sqrt{n})$ time.

What are the running times of each of these algorithms (in big-O notation), and which would you choose? Explain your reasoning in detail.

3) **a**) Consider the Mergesort algorithm.

i. Give an example of an 8-element array which requires the maximum number of comparisons, and explain your answer.

ii. Give an example of an 8-element array which requires the minimum number of comparisons, and explain your answer.

b) Consider the QuickSort algorithm.

i. Give an example of an 8-element array which requires the maximum number of swap operations (Assume the pivot is the first element), and explain your answer.

ii. Give an example of an 8-element array which requires the minimum number of swap operations (Assume the pivot is the last element), and explain your answer.

4) Analyze the following algorithm. Write the recurrence relation of the algorithm, and find the running time complexity by deriving from the recurrence relation.

```
algorithm(left, right)
    mid = (left+right)/2
    if A[mid]==0
        return mid
    else
        if A[mid] > 0
            right = mid
            algorithm(left, right)
        else
            left = mid
            algorithm(left, right)
```

5) Ali is planning to visit his family. Before going, he bought n gifts of different sizes and n corresponding boxes. **Write an efficient algorithm that helps Ali to match each gift to its corresponding box by getting help from the QuickSort algorithm**. You are allowed to try a gift and box together, from which you can determine whether the gift is larger than the box, smaller than the box, or fits in the box exactly. However, there is no way to compare two gifts together or two boxes together.

a) Write your algorithm in pseudocode.

b) Analyze your algorithm, and write the corresponding recurrence relation. Find the running time complexity of the algorithm by deriving from the recurrence relation.

c) Implement your algorithm in Phyton.

Notes

1. Late submissions will not be accepted.
2. No collaboration is allowed, the homework must be done individually.

3. The homework will be submitted in ZIP format. The file hierarchy will be this:
   CSE321_HW2_[StudentID].zip
   | → CSE321_HW2_ANS_[StudentID].pdf
   | → matchGiftBox_[StudentID].py