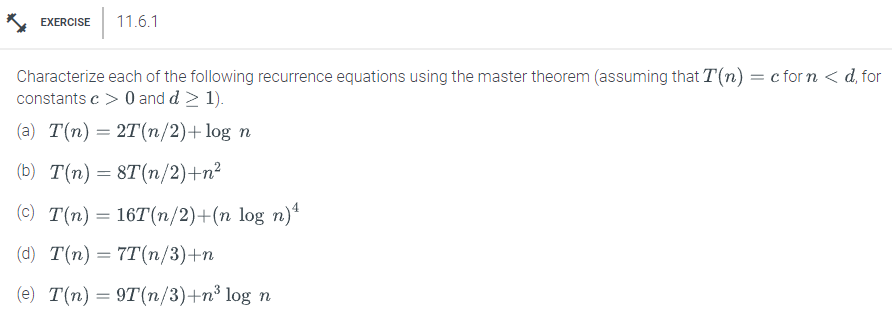
# Michael Chillemi

# 07/20/2023

# CS 590 - Algorithms

# M9.B1: Module 9 Divide-and-Conquer Reinforcement Exercises

Problem 11.6.1



Answer:

1. In this recurrence relation, a = 2, b = 2, and f(n) = log n are all equal to two. For each > 0, the logarithmic function f(n) is smaller than . As a result, we may apply case 2 of the master theorem, which states that if T(n) is O( log n), then f(n) is O() for some constant k. We have f(n) = log n in this situation, which is O() because k = 0. T(n) is therefore O ( log n) = O (log n).
2. We have a = 8, b = 2, and f(n) = in this recurrence relation. Polynomial describes the function f(n), which has the same order as = . As a result, we may apply case 3 of the master theorem, which states that if T(n) is O(), then f(n) is O(nk) for some constant k > logb(a). In this instance, f(n) = , which is equal to (n2), and log2(8) = 3 > 2 are present. T(n) is thus equal to ().
3. We have a = 16 in this recurrence relation, b = 2, and f(n) = (n log n)4. For each > 0, the function f(n) is greater than . Since f(n) is () for some constant k, and af(n/b) cf(n) for some constant c 1 and sufficiently large n, then T(n) is O (f(n)), we can utilize example 1 of the master theorem. In this instance, f(n) = (n log n)4 is equal to (n4). Additionally, the following is evident: for n > 16. The master theorem's instance 1 can be used to select c = 1/16 and get to the conclusion that T(n) equals O (f(n)) = .
4. We have a = 7, b = 3, and f(n) = n in this recurrence relation. For any , the function f(n) is smaller than . As a result, we may apply case 2 of the master theorem, which states that if T(n) is O(nk), then f(n) is O(nk) for some constant k. We have f(n) = n in this situation, which is O(n1). T(n) is hence O (n).
5. We have a = 9 in this recurrence relation, b = 3, and f(n) = . For each > 0, the function f(n) is greater than . Consequently, we may apply case 1 of the master theorem, which states that if T(n) is O (f(n) and f(n) is (nk) for some constant c and sufficiently large n, respectively. In this instance, f(n) = n3 log n, or (n3), is what we have. Additionally, it is evident that: for n > 27. The master theorem's instance 1 can be used to select c = ) and get to the conclusion that T(n) is O (f(n)) = .

In conclusion, we have investigated the time complexity of five recurrence equations using the master theorem. We have determined the values of a, b, and f(n) for each equation and, using the master theorem, calculated the time complexity in terms of f(n).

These are the outcomes:

(a) T(n) = 2T(n/2) + log n has time complexity (n log n).

(b) T(n) = 8T(n/2) + n2 has time complexity .

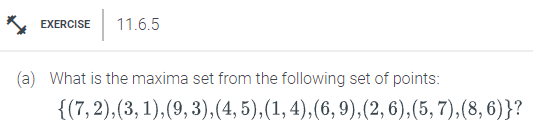
(c) T(n) = 16T(n/2) + (n log n)4 has time complexity.

(d) T(n) = 7T(n/3) + n has time complexity .

(e) T(n) = 9T(n/3) + n3 log n has time complexity .

These findings show the master theorem's effectiveness in examining the temporal complexity of divide-and-conquer algorithms. We may quickly and easily compute the time complexity of the algorithm by identifying the recurrence relation and using the appropriate case of the master theorem.

Problem 11.6.5



Answer:

If there exist np points in a finite set of points S whose coordinates are all greater than or equal to the corresponding coordinate of P, then that point is said to be maximum. All of a point set's maximal points are its maxima. From the list of points previously stated, the following stand out: (3,1); (7,2); (9,3); (1,4); (4,5); (6,9); (2,6); (5,7); (6,9). As a result, the points (9,3), (6,9), and (8,6) are visible here. fulfill the criteria for maximum points. The required maxima set is therefore (9,3), (6,9), and (8,6).