Week 3: Lab

Collaboration level 0 (no restrictions). Open notes.

1. Consider the linear-time merge algorithm discussed in the notes and a possible implementation below. How many element comparisons will the standard merge function take to merge the following left and right lists ?

   $left = [1, 3, 4, 5, 6, 7, 8]$, $right = [1, 5, 9, 11, 12, 16]$

   Merge(left, right)

```
result = []
i=0
j=0
while i < len(left) and j < len(right):
     if left[i] < right[j]:
          result.append(left[i])
          i = i+1
   else:
        result.append(right[j])
        j = j+1

  # add any left overs
  while i < len(left):
      result.append(left[i])
      i = i+1
  while j < len(right):
      result.append(right[j])
      j = j+1

  return result
```

   A 8

   B 9

   C 10

   D 13

Find a $\Theta()$ bound for the following recurrences using iteration.

What we expect: show **the process***: the first $O(1)$ steps of your iteration leading to the general formula, the recursion depth, and the final $\Theta()$ bound for $T(n)$. Do not write your answers on this page because there isn't enough space. Use plenty of space for each problem and show your work.*

2. $T(n) = T(n/2) + \Theta(1)$ (assume $T(1) = 1$).

3. $T(n) = T(2n/3) + \Theta(1)$ (assume $T(i) = 1$ for $i = 1, 2$).

4. $T(n) = T(n-1) + \Theta(1)$ (assume $T(1) = 1$).

5. $T(n) = T(n-2) + \Theta(1)$ (assume $T(i) = 1$ for $i = 1, 2$).

6. $T(n) = T(n/2) + \Theta(n)$ (assume $T(1) = 1$).

7. $T(n) = T(n/3) + \Theta(n)$ (assume $T(i) = 1$ for $i < 3$).

8. $T(n) = 5T(n/5) + \Theta(n)$ (assume $T(i) = 1$ for $i < 5$).

9. $T(n) = T(n-1) + 2n - 3$, with $(T(1) = 1$.
   (Hint: you can write this in a simpler form as $T(n) = T(n-1) + \Theta(n)$.)

10. $T(n) = T(\sqrt{n}) + 1$ (What is the base case here?)

11. $T(n) = 7T(n/2) + \Theta(n^3)$ (assume $T(1) = 1$).

12. $T(n) = 7T(n/2) + \Theta(n^2)$ (assume $T(1) = 1$).

13. $T(n) = 4T(n/3) + 2n - 1$, with $T(1) = T(2) = 1$
    (Hint: You can write this in a simpler form as $T(n) = 4T(n/3) + \Theta(n)$).

14. $T(n) = 3T(n/2) + \Theta(n^2)$, assume $T(1) = 1$.

15. $T(n) = 2T(n-1) + \Theta(1)$, assume $T(1) = 1$.

16. Based on all examples seen so far, list recurrences that solve to:

    (a) $\Theta(\lg n)$
    (b) $\Theta(n)$
    (c) $\Theta(n \lg n)$
    (d) $\Theta(n^2)$
    (e) exponenital

    For each category, enumerate all recurrences seen so far that fall into that category, and add at last one *new* one.

17. Consider the following algorithm to compute $n!$: :

    Mystery(n):

    - //Input: a nonnegative integer n
    - //Output: the value of $n!$
    - if n==0: return 1
    - else: return $n\times$ Mystery($n-1$)

    Analyze the running time of Mystery(n) (i.e. write a recurrence for its running time and find its $\Theta()$) .

18. For the algorithm below, give its runtime recurrence and its order of growth.

    AlgorithmC(n):

    - //We don't know what this algorithm does.
    - Do something that takes $O(1)$
    - AlgorithmC($n/3$)
    - Do something that takes $O(n)$
    - AlgorithmC($n/3$)

Optional

Based on your intuition from working through the previous examples, what is a $\Theta()$ for $T(n)$ in the following recurrences?

1. $T(n) = T(n/3) + \Theta(1)$ (assume $T(i) = 1$ for $i = 1, 2$).

2. $T(n) = T(n/10) + \Theta(1)$ (assume $T(i) = 1$ for $i < 10$).

3. $T(n) = T(99n/100) + \Theta(1)$ (assume $T(i) = 1$ for $i < 100$).

4. $T(n) = T(n-3) + \Theta(1)$ (assume $T(i) = 1$ for $i = 1, 2, 3$).

5. $T(n) = T(n-2) + \Theta(n)$ (assume $T(i) = 1$ for $i = 1, 2$).

6. (challenge) $T(n) = T(n/3) + T(2n/3) + \Theta(n)$ (cannot iterate; only guess the solution)

7. (challenge) $T(n) = T(n/3) + T(n/4) + \Theta(n)$ (cannot iterate; only guess the solution)

8. (challenge) $T(n) = T(n/2) + T(n/4) + T(n/10) + \Theta(n)$ (only guess the solution)

Partial Answers

- B (9 comparisons)
- $T(n) = T(n/2) + \Theta(1)$: $\Theta(\lg n)$.
- $T(n) = T(n/3) + \Theta(1)$ (assume $T(i) = 1$ for $i = 1, 2$): $\Theta(\lg n)$
- $T(n) = T(n/10) + \Theta(1)$ (assume $T(i) = 1$ for $i < 10$): $\Theta(\lg n)$
- $T(n) = T(2n/3) + \Theta(1)$ (assume $T(i) = 1$ for $i = 1, 2$): $\Theta(\lg n)$
- $T(n) = T(n-1) + \Theta(1)$: $\Theta(n)$
- $T(n) = T(n-2) + \Theta(1)$ (assume $T(i) = 1$ for $i = 1, 2$): $\Theta(n)$
- $T(n) = T(n-3) + \Theta(1)$ (assume $T(i) = 1$ for $i = 1, 2, 3$): $\Theta(n)$
- $T(n) = T(n/2) + \Theta(n)$: $\Theta(n)$
- $T(n) = T(n/3) + \Theta(n)$ (assume $T(i) = 1$ for $i < 3$): $\Theta(n)$
- $T(n) = 3T(n/3) + \Theta(n)$ (assume $T(i) = 1$ for $i < 3$): $\Theta(n \lg n)$
- $T(n) = 5T(n/5) + \Theta(n)$ (assume $T(i) = 1$ for $i < 5$): $\Theta(n \lg n)$
- $T(n) = T(n-1) + \Theta(n)$: $\Theta(n^2)$
- $T(n) = T(n-2) + \Theta(n)$ (assume $T(i) = 1$ for $i = 1, 2$): $\Theta(n^2)$
- $T(n) = T(\sqrt{n}) + \Theta(1)$: $\Theta(\lg \lg n)$ (what base case do we need here?)
- $T(n) = 7T(n/2) + \Theta(n^3)$: $\Theta(n^3)$
- $T(n) = 7T(n/2) + \Theta(n^2)$: $T(n) = \Theta(n^{\lg 7})$
- $T(n) = 4T(n/3) + 2n - 1$, with $(T(1) = T(2) = 1$: $T(n) = \Theta(n^{\log_3 4})$
- $T(n) = 3T(n/2) + n^2$, with $(T(1) = 1$: $T(n) = \Theta(n^2)$
- $T(n) = 2T(n-1) + \Theta(1)$: $T(n) = \Theta(2^n)$ Note: For exponential recurrences we are usually happy with just a lower bound.
- (challenge) $T(n) = T(n/3) + T(2n/3) + \Theta(n)$: talk to us!
- (challenge) $T(n) = T(n/3) + T(n/4) + \Theta(n)$: talk to us!
- (challenge) $T(n) = T(n/2) + T(n/4) + T(n/10) + \Theta(n)$: talk to us!