Data Structures and Algorithms

COSC 336 Assignment 3

Exercise 1. Analyze the following recurrences using the method that is indicated. In case you use the Master Theorem, state what the corresponding values of a, b, and $f(n)$ are and how you determined which case of the theorem applies.

- $T(n) = 3T(\frac{n}{4}) + 3$. Use the Master Theorem to find a $\Theta()$ evaluation, or say "Master Theorem cannot be used", if this is the case.
- $T(n) = 2T(\frac{n}{2}) + 3n$. Use the Master Theorem to find a $\Theta()$ evaluation, or say "Master Theorem cannot be used", if this is the case.
- $T(n) = 9T(\frac{n}{3}) + n^2 \log n$. Use the Master Theorem to find a $\Theta()$ evaluation, or say "Master Theorem cannot be used", if this is the case.

Solution: (a) $a = 3, b = 4, f(n) = 3$. We compare $n^{\log_b a} = n^{\log_4 3}$ vs $f(n) = 3$. The Winner is $n^{\log_4 3}$ and the Loser is 3, and the ratio Winner/Loser is $n^{\log_4 3}$ is a polynomial (so it is n^{constant}). So we can use the Master Theorem, and $T(n) = \Theta(\text{Winner}) = \Theta(n^{\log_4 3})$.

(b) $a = 2, b = 2, f(n) = 3n$. We compare $n^{\log_b a} = n^{\log_2 2}$ vs $f(n) = 3n$. Since $n^{\log_2 2} = n$ we have a TIE. So we can use the Master Theorem, $T(n) = \Theta(n \log n)$.

(c) $a = 9, b = 3, f(n) = n^2 \log n$. We compare $n^{\log_b a} = n^{\log_3 9}$ vs $f(n) = n^2 \log n$. Since $n^{\log_3 9} = n^2$, the Winner is $n^2 \log n$, but the ratio Winner/Loser is $\frac{n^2 \log n}{n^2} = \log n$, which is smaller than any polynomial. Therefore, the Master Theorem cannot be used. $\square$

Exercise 2.

- $T(n) = 2T(n-1) + 1$, $T(0) = 1$. Use the iteration method to find a $\Theta()$ evaluation for $T(n)$.
- $T(n) = T(n-1) + 1$, $T(0) = 1$. Use the iteration method to find a $\Theta()$ evaluation for $T(n)$.
- Give a $\Theta(\cdot)$ evaluation for the runtime of the following code:

```
 i= n
 while(i >=1) {
    for (j=1;  j <=n;  j++)
        x=x+1
    i = i/2
}
```

- Give a $\Theta(\cdot)$ evaluation for the runtime of the following code:

```
i= n
while(i >=1) {
   for (j=1;  j <=i;  j++)
       x=x+1
   i = i/2
}
```

Solution: (a) The recurrence does not have the format in the Master Theorem. So, we use the substitution method.

$$\begin{aligned} T(n) &= 2T(n-1)+1 \\ &= 2(2T(n-2)+1)+1 = 2^2T(n-2)+(2+1) \\ &= 2^2(2T(n-3)+1)+(2+1) = 2^3T(n-3)+(2^2+2+1) \vdots \end{aligned}$$

We see the pattern, and we obtain that

$$T(n) = 2^nT(0)+(2^{n-1}+2^{n-2}+\ldots+1) = 2^n+(2^n-1) = 2\cdot 2^n - 1.$$

(The sum is a geometric series and I used the formula for it.) We conclude that $T(n) = \Theta(2^n)$.

(b) As above, the recurrence does not have the format in the Master Theorem. We use the substitution method.

$$\begin{aligned} T(n) &= T(n-1)+1 \\ &= (T(n-2)+1)+1 = T(n-2)+2 \\ &= (T(n-3)+1)+2 = T(n-3)+3 \vdots \end{aligned}$$

We see the pattern, and we obtain that

$$T(n) = T(0)+n = 1+n = \Theta(n).$$

(c) The runtime is $\Theta(n \log n)$, because the inner loop has runtime $\Theta(n)$, and the outer loop has $\log n$ iterations (i goes from n down to 1, by halving at each iteration).

(d) It is convenient to assume $n = 2^k$ (n is a power of 2). The inner loop has runtime i, and from the outer loop we see that i takes the values (in order): $2^k, 2^{k-1}, 2^{k-2} \ldots 2^0$.

So the total runtime is Θ of the sum $2^k+2^{k-1}+2^{k-2}+\ldots 2^0 = 2^{k+1}-1 = 2\cdot 2^k - 1 = 2n-1$. We conclude that the runtime is $\Theta(n)$. $\square$