Merge Sort

Solving Recurrences

The Master Theorem

Review: Asymptotic Notation

- Upper Bound Notation:
  - f(n) is O(g(n)) if there exist positive constants c and n_0 such that $f(n) \le c \cdot g(n)$ for all $n \ge n_0$
  - Formally, $O(g(n)) = \{ f(n)\text{: } \exists$ positive constants c and n_0 such that $f(n) \le c \cdot g(n)\ \forall\ n \ge n_0$
- Big O fact:
  - A polynomial of degree k is $O(n^k)$

Review: Asymptotic Notation

- Asymptotic lower bound:
  - f(n) is *$\Omega(g(n))$* if $\exists$ positive constants *c* and n_0 such that $0 \le c \cdot g(n) \le f(n) \quad \forall\ n \ge n_0$
- Asymptotic tight bound:
  - f(n) is *$\Theta(g(n))$* if $\exists$ positive constants c_1, c_2, and n_0 such that $c_1\, g(n) \le f(n) \le c_2\, g(n)\ \forall\ n \ge n_0$
  - $f(n) = \Theta(g(n))$ if and only if $f(n) = O(g(n))$ AND $f(n) = \Omega(g(n))$

Other Asymptotic Notations

- A function f(n) is o(g(n)) if ∃ positive constants c and n_0 such that

  $$f(n) < c\, g(n)\ \forall\ n \geq n_0$$

- A function f(n) is ω(g(n)) if ∃ positive constants c and n_0 such that

  $$c\, g(n) < f(n)\ \forall\ n \geq n_0$$

- Intuitively,
  - o() is like <
  - ω() is like >
  - Θ() is like =
  - O() is like ≤
  - Ω() is like ≥

Merge Sort

```
MergeSort(A, left, right) {
   if (left < right) {
       mid = floor((left + right) / 2);
       MergeSort(A, left, mid);
       MergeSort(A, mid+1, right);
       Merge(A, left, mid, right);
   }
}

// Merge() takes two sorted subarrays of A and
// merges them into a single sorted subarray of A
//     (how long should this take?)
```

Merge Sort: Example

- Show MergeSort() running on the array

```
A = {10, 5, 7, 6, 1, 4, 8, 3, 2, 9};
```

Analysis of Merge Sort

Statement	Effort
`MergeSort(A, left, right) {`	`T(n)`
`if (left < right) {`	`Θ(1)`
`mid = floor((left + right) / 2);`	`Θ(1)`
`MergeSort(A, left, mid);`	`T(n/2)`
`MergeSort(A, mid+1, right);`	`T(n/2)`
`Merge(A, left, mid, right);`	`Θ(n)`
`}`	
`}`	

- So $T(n) =$ $\Theta(1)$ when $n = 1$, and
  $2T(n/2) + \Theta(n)$ when $n > 1$
- So what (more succinctly) is $T(n)$?

Recurrences

- The expression:

$$T(n) = \begin{cases} c & n = 1 \\ 2T\left(\frac{n}{2}\right) + cn & n > 1 \end{cases}$$

is a *recurrence*.

  - Recurrence: an equation that describes a function in terms of its value on smaller functions

Recurrence Examples

$$s(n) = \begin{cases} 0 & n = 0 \\ c + s(n-1) & n > 0 \end{cases}$$

$$s(n) = \begin{cases} 0 & n = 0 \\ n + s(n-1) & n > 0 \end{cases}$$

$$T(n) = \begin{cases} c & n = 1 \\ 2T\left(\frac{n}{2}\right) + c & n > 1 \end{cases}$$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

Solving Recurrences

- Substitution method
- Iteration method
- Master method

Solving Recurrences

- The substitution method
  - A.k.a. the “making a good guess method”
  - Guess the form of the answer, then use induction to find the constants and show that solution works
  - Examples:
    - $T(n) = 2T(n/2) + \Theta(n)$ □ $T(n) = \Theta(n \lg n)$
    - $T(n) = 2T(\lfloor n/2 \rfloor) + n$ □ ???

Solving Recurrences

- The substitution method
- A.k.a. the "making a good guess method"
  - Guess the form of the answer, then use induction to find the constants and show that solution works
  - Examples:
    - $T(n) = 2T(n/2) + \Theta(n) \rightarrow T(n) = \Theta(n \lg n)$
    - $T(n) = 2T(\lfloor n/2 \rfloor) + n \rightarrow T(n) = \Theta(n \lg n)$
    - $T(n) = 2T(\lfloor n/2 \rfloor)+ 17) + n \rightarrow$???

Solving Recurrences

- The substitution method
- A.k.a. the “making a good guess method”
  - Guess the form of the answer, then use induction to find the constants and show that solution works
  - Examples:
    - $T(n) = 2T(n/2) + \Theta(n) \rightarrow T(n) = \Theta(n \lg n)$
    - $T(n) = 2T(\lfloor n/2 \rfloor) + n \rightarrow T(n) = \Theta(n \lg n)$
    - $T(n) = 2T(\lfloor n/2 \rfloor + 17) + n \rightarrow \Theta(n \lg n)$

Solving Recurrences

- Another option is what the book calls the "iteration method"
  - Expand the recurrence
  - Work some algebra to express as a summation
  - Evaluate the summation
- We will show several examples

$$s(n) = \begin{cases} 0 & n = 0 \\ c + s(n-1) & n > 0 \end{cases}$$

- s(n) = c + s(n-1)

  c + c + s(n-2)

  2c + s(n-2)

  2c + c + s(n-3)

  3c + s(n-3)

  …

  kc + s(n-k) = ck + s(n-k)

$$s(n) = \begin{cases} 0 & n = 0 \\ c + s(n-1) & n > 0 \end{cases}$$

- So far for n >= k we have
  - s(n) = ck + s(n-k)
- What if k = n?
  - s(n) = cn + s(0) = cn

$$s(n) = \begin{cases} 0 & n = 0 \\ c + s(n-1) & n > 0 \end{cases}$$

- So far for n >= k we have
  - s(n) = ck + s(n-k)
- What if k = n?
  - s(n) = cn + s(0) = cn
- So

$$s(n) = \begin{cases} 0 & n = 0 \\ c + s(n-1) & n > 0 \end{cases}$$

- Thus in general
  - s(n) = cn

$$s(n) = \begin{cases} 0 & n = 0 \\ n + s(n-1) & n > 0 \end{cases}$$

- s(n)

= n + s(n-1)

= n + n-1 + s(n-2)

= n + n-1 + n-2 + s(n-3)

= n + n-1 + n-2 + n-3 + s(n-4)

= …

= n + n-1 + n-2 + n-3 + … + n-(k-1) + s(n-k)

$$s(n)=\begin{cases}0 & n=0\\ n+s(n-1) & n>0\end{cases}$$

- s(n)

= n + s(n-1)

= n + n-1 + s(n-2)

= n + n-1 + n-2 + s(n-3)

= n + n-1 + n-2 + n-3 + s(n-4)

= …

= n + n-1 + n-2 + n-3 + … + n-(k-1) + s(n-k)

$$= \sum_{i=n-k+1}^{n} i \quad + \quad s(n-k)$$

$$s(n) = \begin{cases} 0 & n = 0 \\ n + s(n-1) & n > 0 \end{cases}$$

- So far for n >= k we have

$$\sum_{i=n-k+1}^{n} i \quad + \quad s(n-k)$$

$$s(n) = \begin{cases} 0 & n = 0 \\ n + s(n-1) & n > 0 \end{cases}$$

- So far for n >= k we have

$$\sum_{i=n-k+1}^{n} i \quad + \quad s(n-k)$$

- What if k = n?

$$s(n) = \begin{cases} 0 & n = 0 \\ n + s(n-1) & n > 0 \end{cases}$$

- So far for n >= k we have

$$\sum_{i=n-k+1}^{n} i \quad + \quad s(n-k)$$

- What if k = n?

$$\sum_{i=1}^{n} i \quad + \quad s(0) \quad = \quad \sum_{i=1}^{n} i \quad + \quad 0 \quad = \quad n\frac{n+1}{2}$$

$$s(n) = \begin{cases} 0 & n = 0 \\ n + s(n-1) & n > 0 \end{cases}$$

- So far for n >= k we have

$$\sum_{i=n-k+1}^{n} i \quad + \quad s(n-k)$$

- What if k = n?

$$\sum_{i=1}^{n} i \quad + \quad s(0) \quad = \quad \sum_{i=1}^{n} i \quad + \quad 0 \quad = \quad n\frac{n+1}{2}$$

- Thus in general

$$s(n) \quad = \quad n\frac{n+1}{2}$$

$$T(n) = \begin{cases} c & n = 1 \\ 2T\left(\dfrac{n}{2}\right) + c & n > 1 \end{cases}$$

- $T(n) =$

  $2T(n/2) + c$

  $2(2T(n/2/2) + c) + c$

  $2^2T(n/2^2) + 2c + c$

  $2^2(2T(n/2^2/2) + c) + 3c$

  $2^3T(n/2^3) + 4c + 3c$

  $2^3T(n/2^3) + 7c$

  $2^3(2T(n/2^3/2) + c) + 7c$

  $2^4T(n/2^4) + 15c$

  …

  $2^kT(n/2^k) + (2^k - 1)c$

$$T(n) = \begin{cases} c & n = 1 \\ 2T\left(\dfrac{n}{2}\right) + c & n > 1 \end{cases}$$

- So far for n > 2k we have
  - $T(n) = 2^k T(n/2^k) + (2^k - 1)c$
- What if k = lg n?
  - $T(n) = 2^{\lg n} T(n/2^{\lg n}) + (2^{\lg n} - 1)c$

    $= n\,T(n/n) + (n - 1)c$

    $= n\,T(1) + (n-1)c$

    $= nc + (n-1)c = (2n - 1)c$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\dfrac{n}{b}\right) + cn & n > 1 \end{cases}$$

- T(n) =

  $aT(n/b) + cn$

  $a(aT(n/b/b) + cn/b) + cn$

  $a^2T(n/b^2) + cna/b + cn$

  $a^2T(n/b^2) + cn(a/b + 1)$

  $a^2(aT(n/b^2/b) + cn/b^2) + cn(a/b + 1)$

  $a^3T(n/b^3) + cn(a^2/b^2) + cn(a/b + 1)$

  $a^3T(n/b^3) + cn(a^2/b^2 + a/b + 1)$

  …

  $a^kT(n/b^k) + cn(a^{k-1}/b^{k-1} + a^{k-2}/b^{k-2} + \ldots + a^2/b^2 + a/b + 1)$

$$T(n)=\begin{cases} c & n=1 \\ aT\left(\dfrac{n}{b}\right)+cn & n>1 \end{cases}$$

- So we have
  - $T(n) = a^k T(n/b^k) + cn(a^{k-1}/b^{k-1} + \ldots + a^2/b^2 + a/b + 1)$
- For $k = \log_b n$
  - $n = b^k$
  - $T(n) = a^k T(1) + cn(a^{k-1}/b^{k-1} + \ldots + a^2/b^2 + a/b + 1)$

    $= a^k c + cn(a^{k-1}/b^{k-1} + \ldots + a^2/b^2 + a/b + 1)$

    $= ca^k + cn(a^{k-1}/b^{k-1} + \ldots + a^2/b^2 + a/b + 1)$

    $= cna^k/b^k + cn(a^{k-1}/b^{k-1} + \ldots + a^2/b^2 + a/b + 1)$

    $= cn(a^k/b^k + \ldots + a^2/b^2 + a/b + 1)$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + ... + a^2/b^2 + a/b + 1)$
- What if $a = b$?
  - $T(n) = cn(k + 1)$

    $= cn(\log_b n + 1)$

    $= \Theta(n \log n)$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + \ldots + a^2/b^2 + a/b + 1)$
- What if $a < b$?

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + \ldots + a^2/b^2 + a/b + 1)$
- What if $a < b$?
  - Recall that $\Sigma(x^k + x^{k-1} + \ldots + x + 1) = (x^{k+1} - 1)/(x-1)$

$$T(n)=\begin{cases} c & n=1 \\ aT\left(\dfrac{n}{b}\right)+cn & n>1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + \ldots + a^2/b^2 + a/b + 1)$
- What if $a < b$?
  - Recall that $(x^k + x^{k-1} + \ldots + x + 1) = (x^{k+1} - 1)/(x-1)$
  - So:

$$\frac{a^k}{b^k}+\frac{a^{k-1}}{b^{k-1}}+\cdots+\frac{a}{b}+1 \quad = \quad \frac{(a/b)^{k+1}-1}{(a/b)-1} \quad = \quad \frac{1-(a/b)^{k+1}}{1-(a/b)} \quad < \quad \frac{1}{1-a/b}$$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + ... + a^2/b^2 + a/b + 1)$
- What if $a < b$?
  - Recall that $\Sigma(x^k + x^{k-1} + \ldots + x + 1) = (x^{k+1} - 1)/(x-1)$
  - So:

$$\frac{a^k}{b^k} + \frac{a^{k-1}}{b^{k-1}} + \cdots + \frac{a}{b} + 1 = \frac{(a/b)^{k+1} - 1}{(a/b) - 1} = \frac{1 - (a/b)^{k+1}}{1 - (a/b)} < \frac{1}{1 - a/b}$$

  - $T(n) = cn \cdot \Theta(1) = \Theta(n)$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + ... + a^2/b^2 + a/b + 1)$
- What if $a > b$?

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + \ldots + a^2/b^2 + a/b + 1)$
- What if a > b?

$$\frac{a^k}{b^k} + \frac{a^{k-1}}{b^{k-1}} + \cdots + \frac{a}{b} + 1 \quad = \quad \frac{(a/b)^{k+1} - 1}{(a/b) - 1} \quad = \quad \Theta\left((a/b)^k\right)$$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + \ldots + a^2/b^2 + a/b + 1)$
- What if $a > b$?

$$\frac{a^k}{b^k} + \frac{a^{k-1}}{b^{k-1}} + \cdots + \frac{a}{b} + 1 = \frac{(a/b)^{k+1} - 1}{(a/b) - 1} = \Theta\left((a/b)^k\right)$$

  - $T(n) = cn \cdot \Theta(a^k / b^k)$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\dfrac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + \ldots + a^2/b^2 + a/b + 1)$
- What if $a > b$?

$$\frac{a^k}{b^k} + \frac{a^{k-1}}{b^{k-1}} + \cdots + \frac{a}{b} + 1 \quad = \quad \frac{(a/b)^{k+1} - 1}{(a/b) - 1} \quad = \quad \Theta\left((a/b)^k\right)$$

  - $T(n) = cn \cdot \Theta(a^k / b^k)$

$$= cn \cdot \Theta(a^{\log n} / b^{\log n}) = cn \cdot \Theta(a^{\log n} / n)$$

$$T(n)=\begin{cases} c & n=1 \\ aT\left(\frac{n}{b}\right)+cn & n>1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + \ldots + a^2/b^2 + a/b + 1)$
- What if $a > b$?

$$\frac{a^k}{b^k}+\frac{a^{k-1}}{b^{k-1}}+\cdots+\frac{a}{b}+1 \quad = \quad \frac{(a/b)^{k+1}-1}{(a/b)-1} \quad = \quad \Theta\left((a/b)^k\right)$$

  - $T(n) = cn \cdot \Theta(a^k / b^k)$

$$= cn \cdot \Theta(a^{\log n} / b^{\log n}) = cn \cdot \Theta(a^{\log n} / n)$$

recall logarithm fact: $a^{\log n} = n^{\log a}$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + \ldots + a^2/b^2 + a/b + 1)$
- What if $a > b$?

$$\frac{a^k}{b^k} + \frac{a^{k-1}}{b^{k-1}} + \cdots + \frac{a}{b} + 1 \quad = \quad \frac{(a/b)^{k+1} - 1}{(a/b) - 1} \quad = \quad \Theta\left((a/b)^k\right)$$

  - $T(n) = cn \cdot \Theta(a^k / b^k)$

$= cn \cdot \Theta(a^{\log n} / b^{\log n}) = cn \cdot \Theta(a^{\log n} / n)$

recall logarithm fact: $a^{\log n} = n^{\log a}$

$= cn \cdot \Theta(n^{\log a} / n) = \Theta(cn \cdot n^{\log a} / n)$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So with $k = \log_b n$
  - $T(n) = cn(a^k/b^k + \ldots + a^2/b^2 + a/b + 1)$
- What if $a > b$?

$$\frac{a^k}{b^k} + \frac{a^{k-1}}{b^{k-1}} + \cdots + \frac{a}{b} + 1 = \frac{(a/b)^{k+1} - 1}{(a/b) - 1} = \Theta\left((a/b)^k\right)$$

  - $T(n) = cn \cdot \Theta(a^k / b^k)$

$= cn \cdot \Theta(a^{\log n} / b^{\log n}) = cn \cdot \Theta(a^{\log n} / n)$

recall logarithm fact: $a^{\log n} = n^{\log a}$

$= cn \cdot \Theta(n^{\log a} / n) = \Theta(cn \cdot n^{\log a} / n)$

$= \Theta(n^{\log a})$

$$T(n) = \begin{cases} c & n = 1 \\ aT\left(\frac{n}{b}\right) + cn & n > 1 \end{cases}$$

- So…

$$T(n) = \begin{cases} \Theta(n) & a < b \\ \Theta(n \log_b n) & a = b \\ \Theta\left(n^{\log_b a}\right) & a > b \end{cases}$$

The Master Theorem

- Given: a *divide and conquer* algorithm
  - An algorithm that divides the problem of size n into a subproblems, each of size n/b
  - Let the cost of each stage (i.e., the work to divide the problem + combine solved subproblems) be described by the function f(n)
- Then, the Master Theorem gives us a cookbook for the algorithm's running time:

The Master Theorem

- if T(n) = aT(n/b) + f(n) then

$$
T(n) = \begin{cases}
\Theta\left(n^{\log_b a}\right) & f(n) = O\left(n^{\log_b a - \varepsilon}\right) \\
\Theta\left(n^{\log_b a} \log n\right) & f(n) = \Theta\left(n^{\log_b a}\right) \\
\Theta\left(f(n)\right) & f(n) = \Omega\left(n^{\log_b a + \varepsilon}\right) \text{AND} \\
 & af(n/b) < cf(n) \text{ for large } n
\end{cases}
\left.\vphantom{\begin{matrix}1\\1\\1\\1\end{matrix}}\right\} \begin{matrix}\varepsilon > 0 \\ c < 1\end{matrix}
$$

Using The Master Method

- $T(n) = 9T(n/3) + n$
  - $a=9$, $b=3$, $f(n) = n$
  - $n^{\log_b a} = n^{\log_3 9} = \Theta(n^2)$
  - Since $f(n) = O(n^{\log_3 9 - \varepsilon})$, where $\varepsilon=1$, case 1 applies:

$$\boldsymbol{T}(\boldsymbol{n}) = \Theta\left(\boldsymbol{n}^{\log_{\boldsymbol{b}} \boldsymbol{a}}\right) \text{when } \boldsymbol{f}(\boldsymbol{n}) = \boldsymbol{O}\left(\boldsymbol{n}^{\log_{\boldsymbol{b}} \boldsymbol{a}-\varepsilon}\right)$$

  - Thus the solution is $T(n) = \Theta(n^2)$

Master Theorem

- Let T(n) be <u>a monotonically increasing</u> function that satisfies

  $$T(n) = a\,T(n/b) + f(n)$$

  $$T(1) = c$$

  where $a \geq 1$, $b \geq 2$, $c>0$. If $f(n)$ is $\Theta(n^d)$ where $d \geq 0$ then

$$T(n) = \begin{cases} \Theta(n^d) & \text{if } a < b^d \\ \Theta(n^d \log n) & \text{If } a = b^d \\ \Theta(n^{\log_b a}) & \text{if } a > b^d \end{cases}$$

Master Theorem: Pitfalls

- You cannot use the Master Theorem if
  - T(n) is not monotone, e.g. T(n) = sin(x)
  - f(n) is not a polynomial, e.g., $T(n)=2T(n/2)+2^n$
  - b cannot be expressed as a constant, e.g.

$$T(n) = T(\sqrt{n})$$

- Note that the Master Theorem does not solve the recurrence equation
- Does the base case remain a concern?

Master Theorem: Example 1

- Let $T(n) = T(n/2) + ½ n^2 + n$. What are the parameters?

  a = *1*

  b = *2*

  d = *2*

  Therefore, which condition applies?

  $1 < 2^2$, case 1 applies

- *We conclude that*

$$T(n) \in \Theta(n^d) = \Theta(n^2)$$

Master Theorem: Example 2

- Let T(n)= 2 T(n/4) + √n + 42. What are the parameters?

  a = *2*

  b = *4*

  d = *1/2*

  Therefore, which condition applies?

  $2 = 4^{1/2}$, case 2 applies

- *We conclude that*

$$T(n) \in \Theta(n^d \log n) = \Theta(\log n \sqrt{n})$$

Master Theorem: Example 3

- Let T(n)= 3 T(n/2) + 3/4n + 1. What are the parameters?

  a = *3*

  b = *2*

  d = *1*

  Therefore, which condition applies?

  $3 > 2^1$, case 3 applies

- *We conclude that*

$$T(n) \in \Theta(n^{\log_b a}) = \Theta(n^{\log_2 3})$$

- *Note that $\log_2 3 \approx 1.584\ldots$, can we say that $T(n) \in \Theta(n^{1.584})$*

  No, because $\log_2 3 \approx 1.5849\ldots$ and $n^{1.584} \notin \Theta(n^{1.5849})$

Practice Problems

For each of the following recurrences, give an expression for the runtime $T(n)$ if the recurrence can be solved with the Master Theorem. Otherwise, indicate that the Master Theorem does not apply.

1. $T(n) = 3T(n/2) + n^2$

2. $T(n) = 4T(n/2) + n^2$

3. $T(n) = T(n/2) + 2^n$

4. $T(n) = 2^n T(n/2) + n^n$

5. $T(n) = 16T(n/4) + n$

6. $T(n) = 2T(n/2) + n \log n$

Solutions

1. $T(n) = 3T(n/2) + n^2 \implies T(n) = \Theta(n^2)$ (Case 3)

2. $T(n) = 4T(n/2) + n^2 \implies T(n) = \Theta(n^2 \log n)$ (Case 2)

3. $T(n) = T(n/2) + 2^n \implies \Theta(2^n)$ (Case 3)

4. $T(n) = 2^n T(n/2) + n^n \implies$ Does not apply (a is not constant)

5. $T(n) = 16T(n/4) + n \implies T(n) = \Theta(n^2)$ (Case 1)

6. $T(n) = 2T(n/2) + n \log n \implies T(n) = n \log^2 n$ (Case 2)