CIS507: Design & Analysis of Algorithms
Tutorial: Asymptotics and Recurrences
(VERSION WITH ANSWERS)

Cheat Sheet: Master Method

For $T(n) = aT(n/b) + f(n)$, with $a \geq 1$, $b \geq 1$, compare $f(n)$ with $n^{\log_b a}$.

Case	Condition, for $\epsilon > 0$	Solution
1	$f(n) = \mathcal{O}(n^{\log_b a - \epsilon})$	$T(n) = \Theta(n^{\log_b a})$ Number of leafs dominates
2	$f(n) = \Theta(n^{\log_b a})$	$T(n) = \Theta(n^{\log_b a} \lg n)$ All rows have same asymptotic sum
3	$f(n) = \Omega(n^{\log_b a + \epsilon})$	$T(n) = \Theta(f(n))$ provided that $af(n/b) \leq cf(n)$ for some $c < 1$
2 (general)	$f(n) = \Theta(n^{\log_b a} \lg^k n)$ or some constant $k \geq 0$	$T(n) = \Theta(n^{\log_b a} \lg^{k+1} n)$ They grow at 'similar' rate

Note that the last row includes the more general version of case 2, which isn't in the textbook.

1 Asymptotics (Textbook Exercise 3.1-1)

Let $f(n)$ and $g(n)$ be asymptotically nonnegative functions. Using the basic definition of Θ-notation, prove that $\max(f(n), g(n)) = \Theta(f(n) + g(n))$.
ANSWER:
To prove this, we need to find constants $c_1, c_2 > 0$, such that:

$$c_1(f(n) + g(n)) \leq \max(f(n), g(n)) \leq c_2(f(n) + g(n))$$

This holds if we simply choose $c_1 = 1/2$ and $c_2 = 1$.

2 Asymptotics (Textbook Exercise 3.1-2)

Show that for any real constants a and b, where $b > 0$:
$(n + a)^b = \Theta(n^b)$.
ANSWER:

We need to find constants $c_1, c_2 > 0$, such that:

$$c_1 n^b \leq (n+a)^b \leq c_2 n^b$$

Taking the bth root of all sides preserves the inequality, which becomes:

$$\sqrt[b]{c_1} n - a \leq n \leq \sqrt[b]{c_2} n - a$$

And this would be true given $c_1 = (1/2)^b$ and $c_2 = 2^b$

3 Math (Textbook Exercise 3.2-2)

Prove that $a^{\log_b c} = c^{\log_b a}$.
ANSWER:
We know that $a = b^{\log_b a}$. So:

$$\begin{aligned} a^{\log_b c} &= (b^{\log_b a})^{\log_b c} \\ &= (b^{\log_b c})^{\log_b a} \\ &= c^{\log_b a} \end{aligned}$$

4 Asymptotics (Textbook Exercise 3.1-4)

Is $2^{n+1} = \mathcal{O}(2^n)$? Is $2^{2n} = \mathcal{O}(2^n)$? Explain.
ANSWER:
$2^{n+1} = \mathcal{O}(2^n)$, but $2^{2n} \neq \mathcal{O}(2^n)$.

- To show that $2^{n+1} = \mathcal{O}(2^n)$, we must find constants $c, n_0 > 0$ such that

  $$0 \leq 2^{n+1} \leq c.2^n \text{ for all } n \geq n_0$$

  Since $2^{n+1} = 2.2^n$ for all n, we can satisfy the definition with $c = 2$ and $n_0 = 1$.

- To show that $2^{2n} \neq \mathcal{O}(2^n)$, we can use proof by contradiction. Assume there exists constants $c, n_0 \geq 0$ such that

  $$0 \leq 2^{2n} \leq c.2^n \text{ for all } n \geq n_0$$

  This implies that $2^{2n} = 2^n.2^n \leq c.2^n$. Which implies that $2^n \leq c$. But we cannot choose any constant that is greater than all 2^n for any n. So the assumption leads to contradiction.

5 Asymptotics

Rank the following functions by increasing order of growth. That is, find any arrangement g_1; g_2; g_3; g_4; g_5; g_6; g_7 of the functions satisfying $g_1 = \mathcal{O}(g_2)$, $g_2 = \mathcal{O}(g_3)$, $g_3 = \mathcal{O}(g_4)$, $g_4 = \mathcal{O}(g_5)$, $g_5 = \mathcal{O}(g_6)$, $g_6 = \mathcal{O}(g_7)$.

- $f_1(n) = n^4 + \log n$
- $f_2(n) = n + \log^4 n$ (note that $\log^4 n$ is shorthand for $(\log n)^4$)
- $f_3(n) = n \log n$
- $f_4(n) = \binom{n}{3}$
- $f_5(n) = \binom{n}{n/2}$
- $f_6(n) = 2^n$
- $f_7(n) = n^{\log n}$

ANSWER:

First, let us compute the bounds on each:

- $f_1(n) = n^4 + \log n = \mathcal{O}(n^4)$
- $f_2(n) = n + \log^4 n = \mathcal{O}(n)$
- $f_3(n) = n \log n = \mathcal{O}(n \log n)$
- $f_4(n) = \binom{n}{3} = \frac{n(n-1)(n-2)}{6} = \mathcal{O}(n^3)$
- $f_5(n) = \binom{n}{n/2} = \frac{n!}{((n/2)!)^2}$

  Now we use Stirling approximation $n! \approx \sqrt{2\pi n}(n/e)^n$. Therefore:

  $f_5(n) = \frac{\sqrt{2\pi(n)}(n/e)^n}{\left(\sqrt{2\pi(n/2)}\left(\frac{n/2}{e}\right)^{n/2}\right)^2} = \frac{2^{n+1}}{\sqrt{2\pi n}} = \mathcal{O}(\frac{2^n}{\sqrt{n}})$
- $f_6(n) = 2^n = \mathcal{O}(2^n)$
- $f_7(n) = n^{\log n} = (2^{\log n})^{\log n} = 2^{\log n \times \log n} = \mathcal{O}\big(2^{(\log n)^2}\big)$

From the above, we can arrange the functions as follows:

$$f_2; f_3; f_4; f_1; f_7; f_5; f_6$$

6 Substitution (Textbook Exercise 4.3-1)

Show that the solution of $T(n) = T(n-1) + n$ is $\mathcal{O}(n^2)$. Use the substitution method.

ANSWER:

We need to prove that $T(n) \leq cn^2$ for some constant c. Proof by induction:

- *Inductive hypothesis:* We make a (strong) inductive hypothesis that for $k < n$, we already have $T(k) \leq ck^2$.
- *Inductive step:* If for $k < n$, we have $T(k) \leq ck^2$, then $T(n) \leq cn^2$.

To prove the inductive step $T(n) \leq cn^2$:

$$\begin{aligned}
T(n) &= T(n-1) + n \\
&\leq c(n-1)^2 + n \text{ Since, by the induction assumption} \\
&\quad \text{we know that } T(n-1) \leq c(n-1)^2 \\
&= cn^2 - 2cn + c + n \\
&= cn^2 - (2cn - c - n) \text{ Written as desired minus residual} \\
&= cn^2 - (c(2n-1) - n)
\end{aligned}$$

The last quantity is less than cn^2 if the residual is positive. That is, if $c(2n-1) - n > 0$, which yields $c > \frac{n}{(2n-1)}$. This last condition holds for all $n \geq 1$ and $c \geq 1$. The inductive hypothesis is proven.

For the boundary condition, we set $T(1) = 1$, and so $T(1) = 1 \leq c.1^2$. Thus, we can safely choose $n_0 = 1$ and $c = 1$.

General Comment: In a typical proof-by-induction, you start by stating the basis (base case) first, showing that the statement holds when n is equal to the lowest value that n is given in the question (usually, $n = 0$ or $n = 1$). But when solving recurrences, this lowest value n_0 of n may be different (for example the recurrence may have the solution only for $n_0 \geq 12$). This is why, unlike normal induction proofs, we actually compute the base case later, once we know the value of n_0.

7 Substitution (Textbook Exercise 4.3-7)

Using the master method, you can show that the solution to the recurrence $T(n) = 4T(n/3) + n$ is $T(n) = \Theta(n^{\log_3 4})$. Show that a substitution proof with the assumption $T(n) \leq cn^{log_3 4}$ fails. Then show how to subtract off a lower-order term to make a substitution proof work.

ANSWER:

If we try a straight substitution proof, assuming that $T(n) \leq cn^{log_3 4}$, we would get stuck:

$$\begin{aligned}
T(n) &\leq 4(c(n/3)^{log_3 4}) + n \\
&= 4c\left(\frac{n^{log_3 4}}{4}\right) + n \\
&= cn^{log_3 4} + n
\end{aligned}$$

which is greater than $cn^{log_3 4}$ for any positive n. Instead, we subtract off a lower-order term and assume that $T(n) \leq cn^{log_3 4} - dn$ for some constant d. Now we have:

$$\begin{aligned}
T(n) &\leq 4\big(c(n/3)^{log_3 4} - d(n/3)\big) + n \\
&= 4\left(c\frac{n^{log_3 4}}{4} - \frac{dn}{3}\right) + n \\
&= cn^{log_3 4} - \frac{4}{3}dn + n \\
&= cn^{log_3 4} - dn - (\frac{dn}{3} - n) \text{ Separate desired from residual}
\end{aligned}$$

Note that now the desired is $cn^{log_3 4} - dn$ and the residual is $(\frac{dn}{3} - n)$. The above expression is less than or equal to the desired $cn^{log_3 4} - dn$ if we have a positive residual $(\frac{dn}{3} - n) \geq 0$, which is true if $d \geq 3$.

8 Master Method

Find an asymptotic solution of the following functional recurrences.

1. $T(n) = 9T(n/3) + n^3$
2. $T(n) = 2T(n/4) + \sqrt{n}$
3. $T(n) = 3T(n/4) + n \lg n$
4. $T(n) = 4T(n/2) + n^2 \log n$

ANSWER:

1. Using the master theorem, $a = 9$, $b = 3$, $\log_b a = \log_3 9 = 2$, and $f(n) = n^3$. Thus, we compare $n^{\log_b a} = n^2$ with $f(n) = n^3$. Since $n^3 = \Omega(n^{2+\epsilon})$, we have that $f(n)$ dominates the recurrence, and we are in Case 3 of the master theorem.

   But Case 3 of the master theorem also requires the regularity condition that $af(n/b) \leq cf(n)$ for some constant $c < 1$ and all sufficiently large n. We have:

   $9f(n/3) = 9(\frac{n}{3})^3 = \frac{n^3}{3}$. This is less than cn^3 for $c = \frac{1}{2}$. Therefore, the regularity condition holds. By the master theorem, $T(n) = \Theta(n^3)$.

2. Using the master theorem, $a = 2$, $b = 4$, $\log_b a = \log_4 2 = \frac{1}{2}$, and $f(n) = \sqrt{n}$. Thus, we compare $n^{\log_b a} = n^{1/2}$ with $f(n) = \sqrt{n}$. Since these are asymptotically equivalent, we are in Case 2 of the master theorem. This means that we gain an additional $\log n$ factor and $T(n) = \Theta(\sqrt{n} \log n)$.

3. Using the master theorem, $a = 3$, $b = 4$, $n^{\log_b a} = n^{\log_4 3} \approx n^{0.793}$, and $f(n) = n \lg n$. It is true that $f(n) = \Omega(n^{0.793+0.2})$, where $\epsilon = 0.2$, since $n \lg n \geq n \geq n^{0.993}$.

But Case 3 of the master theorem also requires the regularity condition that $3f(n/4) \le cf(n)$ for some constant $c < 1$ and all sufficiently large n. We have:

$3f(n/4) = 3\left(\frac{n}{4}\lg\frac{n}{4}\right) \le \frac{3}{4}n\lg n = cf(n)$ for $c = \frac{3}{4}$. Therefore, the regularity condition holds for $f(n)$. Case 3 holds, so $T(n) = \Theta(n\lg n)$

4. Using the master theorem, $a = 4$, $b = 2$, $\log_b a = \log_2 4 = 2$, and $f(n) = n^2 \log n$. Thus, we compare $n^{\log_b a} = n^2$ with $f(n) = n^2 \log n$. It is clear that $f(n)$ grows faster. However, it does *not* grow *polynomially* faster. However, we notice that we can apply the more general version of Case 2 in the master method (see cheat sheet at the beginning of this tutorial). We have $f(n) = n^2 \lg^k n = n^{\log_b a} \lg^k n$ for $k = 1$. Therefore, we obtain the solution $T(n) = \Theta(n^{\log_b a} \lg^{k+1} n) = \Theta(n^2 \lg^2 n)$.

9 Trick of Changing the Variable

Solve the recurrence $T(n) = 2T(\sqrt{n}) + 1$ using the master method. *Hint:* To make this possible, you can define your own variables and a new recurrence function to make the recurrence look like the form in the master method.

ANSWER:

Let $m = \lg n$ (this is an often useful trick). Then:

$$T(n) = T(2^m) = 2T(\sqrt{2^m}) + 1 = 2T(2^{m/2}) + 1$$

Let $S(m) = T(2^m)$. Therefore:

$$S(m) = 2T(2^{m/2}) + 1 = 2S(m/2) + 1$$

Now we solve the new recurrence

$$S(m) = 2S(m/2) + 1$$

which can be done using the master method (Case 1), giving $S(m) = \Theta(m)$. Therefore, $T(n) = T(2^m) = S(m) = \Theta(m) = \Theta(\lg n)$.