Two Methods for Solving Recurrences

Yufei Tao

Department of Computer Science and Engineering
Chinese University of Hong Kong

We have seen how to analyze the running time of recursive algorithms by recurrence. It is important to sharpen our skills in solving recurrences.

Today, we will learn two techniques for this purpose: the master theorem and the substitution method.

Master Theorem

The Master Theorem

Let $f(n)$ be a function that returns a positive value for every integer $n > 0$. We know:

$$\begin{aligned} f(1) &= O(1) \\ f(n) &\leq \alpha \cdot f(\lceil n/\beta \rceil) + O(n^\gamma) \qquad \text{(for } n \geq 2) \end{aligned}$$

where $\alpha \geq 1$, $\beta > 1$, and $\gamma \geq 0$ are constants. Then:

- If $\log_\beta \alpha < \gamma$, then $f(n) = O(n^\gamma)$.
- If $\log_\beta \alpha = \gamma$, then $f(n) = O(n^\gamma \log n)$.
- If $\log_\beta \alpha > \gamma$, then $f(n) = O(n^{\log_\beta \alpha})$.

The theorem can be proved by carefully applying the "expansion method" we saw earlier. The details are tedious and omitted.

Example 1

Consider the recurrence of binary search:

$$
\begin{aligned}
f(1) &\leq c_1 \\
f(n) &\leq f(\lceil n/2 \rceil) + c_2 \qquad \text{(for } n \geq 2)
\end{aligned}
$$

Hence, $\alpha = 1$, $\beta = 2$, and $\gamma = 0$. Since $\log_\beta \alpha = \gamma$, we know that $f(n) = O(n^0 \cdot \log n) = O(\log n)$.

Example 2

Consider the recurrence of merge sort:

$$\begin{aligned} f(1) &\le c_1 \\ f(n) &\le 2 \cdot f(\lceil n/2 \rceil) + c_2 n \qquad \text{(for } n \ge 2\text{)} \end{aligned}$$

Hence, $\alpha = 2$, $\beta = 2$, and $\gamma = 1$. Since $\log_\beta \alpha = \gamma$, we know that $f(n) = O(n^\gamma \cdot \log n) = O(n \log n)$.

Example 3

Consider the recurrence:

$$\begin{aligned} f(1) &\leq c_1 \\ f(n) &\leq 2 \cdot f(\lceil n/4 \rceil) + c_2 \sqrt{n} \qquad \text{(for } n \geq 2) \end{aligned}$$

Hence, $\alpha = 2$, $\beta = 4$, and $\gamma = 1/2$. Since $\log_\beta \alpha = \gamma$, we know that $f(n) = O(n^\gamma \cdot \log n) = O(\sqrt{n} \log n)$.

Example 4

Consider the recurrence:

$$\begin{aligned} f(1) &\leq c_1 \\ f(n) &\leq 2 \cdot f(\lceil n/2 \rceil) + c_2\sqrt{n} \qquad \text{(for } n \geq 2\text{)} \end{aligned}$$

Hence, $\alpha = 2$, $\beta = 2$, and $\gamma = 1/2$. Since $\log_\beta \alpha > \gamma$, we know that $f(n) = O(n^{\log_\beta \alpha}) = O(n)$.

Example 5

Consider the recurrence:

$$\begin{aligned} f(1) &\le c_1 \\ f(n) &\le 13 \cdot f(\lceil n/7 \rceil) + c_2 n^2 \qquad \text{(for } n \ge 2\text{)} \end{aligned}$$

Hence, $\alpha = 13$, $\beta = 7$, and $\gamma = 2$. Since $\log_\beta \alpha < \gamma$, we know that $f(n) = O(n^\gamma) = O(n^2)$.

The Substitution Method

Solving a Recurrence by Mathematical Induction

Example 6

Consider the recurrence:

$$\begin{aligned} f(1) &= 1 \\ f(n) &\leq f(n-1) + 11n \qquad \text{(for } n \geq 2) \end{aligned}$$

We will prove $f(n) = O(n^2)$ by induction.

We aim to find a constant c such that $f(n) \leq c \cdot n^2$ for $n \geq 1$. To that end, we want to gather all the conditions that c should satisfy.

For the base case of $n = 1$, for $f(1) \leq c$ to hold, we require $c \geq 1$.

Suppose that $(n) \leq cn^2$ for all $n \leq k - 1$ where $k \geq 2$. Then, we have:

$$\begin{aligned} f(k) &\leq f(k-1) + 11k \leq c \cdot (k-1)^2 + 11k \\ &= ck^2 - 2ck + c + 11k \end{aligned}$$

To make the above at most ck^2, we need

$$c \geq 11k/(2k-1)$$

For $k \geq 2$, the fraction $\frac{11k}{2k-1} \leq 22/3$ (maximum taken at $k = 2$). The requirement becomes $c \geq 22/3$.

Any $c \geq 22/3$ gives a working argument. We will set $c = 8$ to simplify the calculation in the argument, given in the next slide.

Proof (for the claim $f(n) = O(n^2)$): We will prove $f(n) \le 8n^2$ for all $n \ge 1$.

For the base case of $n = 1$, we have $f(1) = 1 \le 8$.

Suppose that $(n) \le 8n^2$ for all $n \le k - 1$ where $k \ge 2$. Then, we have:

$$\begin{aligned} f(k) &\le f(k-1) + 11k \le 8 \cdot (k-1)^2 + 11k \\ &= 8k^2 - 5k + 8 \end{aligned}$$

which is at most $8k^2$ because $k \ge 2$.

This completes the proof. □

Try to use the method to "prove" $f(n) \leq cn$. You will never succeed because $f(n) = \Omega(n^2)$, but it is worth trying to see how the argument will fail.

Example 7

Consider the recurrence:

$$f(1) = f(2) = f(3) = 1$$
$$f(n) \le f(\lceil n/5 \rceil) + f\left(\left\lceil \frac{7n}{10} \right\rceil\right) + n \qquad \text{(for } n \ge 4)$$

This is really a non-trivial recurrence (the master theorem is not applicable here). We will prove that $f(n) = O(n)$ using the substitution method.

Goal: To prove the existence of a constant α such that $f(n) \leq \alpha n$ for all $n \geq 1$.

Base case ($n \leq \beta$): We need

$$\alpha \cdot n \geq f(n) \text{ for all } n \in [1, \beta]. \tag{1}$$

Induction: Assuming $f(n) \leq \alpha n$ under $n \leq k - 1$, we aim to show $f(k) \leq \alpha k$, where $k \geq \beta + 1$.

We have:

$$\begin{aligned} f(k) &\leq \alpha(\lceil k/5 \rceil) + \alpha(\lceil (7/10)k \rceil) + k \\ &\leq \alpha(k/5 + 1) + \alpha((7/10)k + 1) + k \\ &= \alpha(9/10)k + 2\alpha + k \end{aligned}$$

We need:

$$\begin{aligned} \alpha(9/10)k + 2\alpha + k &\leq \alpha k \\ \Leftrightarrow \alpha(k/10 - 2) &\geq k \end{aligned} \quad (2)$$

We will make sure $k \geq \beta + 1 > 20$ so that $k/10 - 2 > 0$. With this, we derive:

$$(2) \Leftrightarrow \alpha \geq \frac{k}{k/10 - 2} \quad (3)$$

For $k \geq \beta + 1$, the value $\frac{k}{k/10-2} \leq \frac{(\beta+1)}{(\beta+1)/10-2}$ (maximum taken at $k = \beta + 1$). Requirement (3) becomes

$$\alpha \geq \frac{(\beta + 1)}{(\beta + 1)/10 - 2}. \quad (4)$$

All we need to do now is to find α and β to satisfy the red constraints, namely, (1), $\beta > 19$, and (4). There are infinitely many such values, e.g.:

$$\begin{aligned}\beta &= 39 \\ \alpha &= \max\left\{\frac{f(39)}{39}, \frac{f(38)}{38}..., \frac{f(1)}{1}, 20\right\}.\end{aligned}$$

You can now use this pair of values to construct a working inductive argument.