Substitution Method

Adapted from the CLRS book slides

SUBSTITUTION METHOD

Guess the solution.

Use induction to find the constants and show that the solution works.

Example:

Determine an asymptotic upper bound on $T(n) = 2T(\lfloor n/2 \rfloor) + \Theta(n)$.
Floor function ensures that $T(n)$ is defined over integers.

Guess: $T(n) = O(n \lg n)$.

SUBSTITUTION METHOD (continued)

Inductive step: Assume that $T(n) \leq cn \lg n$ for all numbers $\geq n_0$ and $< n$. If $n \geq 2n_0$, holds for $\lfloor n/2 \rfloor \Rightarrow T(\lfloor n/2 \rfloor) \leq c \lfloor n/2 \rfloor \lg \lfloor n/2 \rfloor$. Substitute into the recurrence:

$$
\begin{aligned}
T(n) &\leq 2(c \lfloor n/2 \rfloor \lg(\lfloor n/2 \rfloor)) + \Theta(n) \\
&\leq 2(c(n/2) \lg(n/2)) + \Theta(n) \\
&= cn \lg(n/2) + \Theta(n) \\
&= cn \lg n - cn \lg 2 + \Theta(n) \\
&= cn \lg n - cn + \Theta(n) \\
&\leq cn \lg n \; .
\end{aligned}
$$

SUBSTITUTION METHOD (continued)

Base cases: Need to show that $T(n) \le cn \lg n$ when $n_0 \le n < 2n_0$. Add new constraint: $n_0 > 1 \Rightarrow \lg n > 0 \Rightarrow n \lg n > 0$. Pick $n_0 = 2$. Because no base case is given in the recurrence, it's algorithmic $\Rightarrow$ $T(2), T(3)$ are constant. Choose $c = \max\{T(2), T(3)\} \Rightarrow T(2) \le c < (2 \lg 2)c$ and $T(3) \le c < (3 \lg 3)c \Rightarrow$ inductive hypothesis established for the base cases.

Wrap up: Have $T(n) \le cn \lg n$ for all $n \ge 2 \Rightarrow T(n) = O(n \lg n)$.

In practice: Don't usually write out substitution proofs this detailed, especially regarding base cases. For most algorithmic recurrences, the base cases are handled the same way.

SUBSTITUTION METHOD (CONTINUED)

When the additive term uses asymptotic notation

Name the constant in the additive term.

Show the upper (O) and lower (Ω) bounds separately. Might need to use different constants for each.

Example:

$T(n) = 2T(n/2) + \Theta(n)$. If we want to show an upper bound of $T(n) = 2T(n/2) + O(n)$, we write $T(n) \le 2T(n/2) + cn$ for some positive constant c.

Important: We get to name the constant hidden in the asymptotic notation (*c* in this case), but we do ***not*** get to choose it, other than assume that it's enough to handle the base case of the recursion.

SUBSTITUTION METHOD (CONTINUED)

Upper bound:

Guess: $T(n) \leq d\, n \lg n$ for some positive constant d. This is the inductive hypothesis.

Important: We get to both name and choose the constant in the inductive hypothesis (d in this case). It OK for the constant in the inductive hypothesis (d) to depend on the constant hidden in the asymptotic notation (c).

Substitution:

$$\begin{aligned}
T(n) &\leq 2T(n/2) + cn \\
&= 2\left(d\frac{n}{2}\lg\frac{n}{2}\right) + cn \\
&= dn\lg\frac{n}{2} + cn \\
&= dn\lg n - dn + cn \\
&\leq dn\lg n \qquad \text{if } -dn + cn \leq 0\,, \\
&\qquad\qquad\qquad\qquad d \geq c
\end{aligned}$$

Therefore, $T(n) = O(n\lg n)$.

SUBSTITUTION METHOD (CONTINUED)

Lower bound:

Write $T(n) \geq 2T(n/2) + cn$ for some positive constant c.

Guess: $T(n) \geq d\, n \lg n$ for some positive constant d.

Substitution:

$$\begin{aligned}
T(n) &\geq 2T(n/2) + cn \\
&= 2\left(d\frac{n}{2}\lg\frac{n}{2}\right) + cn \\
&= dn\lg\frac{n}{2} + cn \\
&= dn\lg n - dn + cn \\
&\geq dn\lg n \qquad \text{if } -dn + cn \geq 0\,, \\
&\qquad\qquad\qquad\qquad d \leq c
\end{aligned}$$

Therefore, $T(n) = \Omega(n \lg n)$.

Therefore, $T(n) = \Theta(n \lg n)$. [For this particular recurrence, we can use $d = c$ for both the upper-bound and lower-bound proofs. That won't always be the case.]

SUBTRACTING A LOW-ORDER TERM

Might guess the right asymptotic bound, but the math doesn't go through in the proof. Resolve by subtracting a lower-order term.

Example:

$T(n) = 2T(n/2) + \Theta(1)$. Guess that $T(n) = O(n)$, and try to show $T(n) \le cn$ for $n \ge n_0$, where we choose c, n_0:

$$\begin{aligned} T(n) &\le 2(c(n/2)) + \Theta(1) \\ &= cn + \Theta(1) . \end{aligned}$$

But this doesn't say that $T(n) \le cn$ for *any* choice of c.

SUBTRACTING A LOW-ORDER TERM
(continued)

Could try a larger guess, such as $T(n) = O(n^2)$, but not necessary. We're off only by $\Theta(1)$, a lower-order term. Try subtracting a lower-order term in the guess: $T(n) \le cn - d$, where $d \ge 0$ is a constant:

$$\begin{aligned} T(n) &\le 2(c(n/2) - d) + \Theta(1) \\ &= cn - 2d + \Theta(1) \\ &\le cn - d - (d - \Theta(1)) \\ &\le cn - d \end{aligned}$$

as long as d is larger than the constant in $\Theta(1)$.

SUBTRACTING A LOW-ORDER TERM
(continued)

Why subtract off a lower-order term, rather than add it? Notice that it's subtracted twice. Adding a lower-order term twice would take us further away from the inductive hypothesis. Subtracting it twice gives us $T(n) \leq cn - d - (d - \Theta(1))$, and it's easy to choose d to make that inequality hold.

Important: Once again, we get to name and choose the constant c in the inductive hypothesis. And we also get to name and choose the constant d that we subtract off.

SUBTRACTING A LOW-ORDER TERM

(continued)

Be careful when using asymptotic notation

A false proof for the recurrence $T(n) = 2T(\lfloor n/2 \rfloor) + \Theta(n)$, that $T(n) = O(n)$:

$$\begin{aligned} T(n) &\leq 2 \cdot O(\lfloor n/2 \rfloor) + \Theta(n) \\ &= 2 \cdot O(n) + \Theta(n) \\ &= O(n) . \qquad \Longleftarrow \textit{wrong!} \end{aligned}$$

This "proof" changes the constant in the Θ-notation. Can see this by using an explicit constant. Assume $T(n) \leq cn$ for all $n \geq n_0$:

$$\begin{aligned} T(n) &\leq 2(c \lfloor n/2 \rfloor) + \Theta(n) \\ &\leq cn + \Theta(n) , \end{aligned}$$

but $cn + \Theta(n) > cn$.

MAKING A GOOD GUESS?

No general way to make a good guess. Experience helps. Can also draw out a recursion tree.