1 Introduction

In this lecture we look at Divide-and-Conquer Recurrence Relations and the Akra-Bazzi or Master Theorem.

2 Divide-and-Conquer Recurrence Relations

In divide-and-conquer algorithms (like Mergesort), the problem is divided into smaller subproblems, each subproblem is solved recursively and a combined algorithm is used to obtain the final solution from the solutions of the subproblems. Assume there are \( a \) subproblems, each of size \( \frac{1}{b} \) of the original problem and the algorithm to combine the subproblems takes polynomial time given by \( cn^k \) for some constants \( a, b, c \) & \( k \). Then we have

\[
T(n) = aT\left(\frac{n}{b}\right) + cn^k
\]

By an expanding method very similar to the one in the last lecture for \( a = 2a, \frac{n}{2} + n \), we can prove the following theorem: (it is a good exercise to prove it for yourself or see the proof in the book [1]).

**Theorem 1** If \( T(n) = aT\left(\frac{n}{b}\right) + cn^k \) for integers \( a \geq 1, b \geq 2 \) and positive constants \( c \) and \( k \), then

\[
T(n) = \begin{cases} 
O(n^{\log_b a}) & \text{if } a > b^k \\
O(n^k \log n) & \text{if } a = b^k \\
O(n^k) & \text{if } a < b^k 
\end{cases}
\]

This formula is very useful in many divide-and-conquer algorithms and you should memorize it in this course.

A more general theorem which consider a general function \( f(n) \) instead of \( cn^k \) in Equation [1] and obtains \( \Theta \) instead of \( O \), is called the Master Theorem.
See the Wikipedia article on the Master Theorem. Again there are three cases to consider conditioned on the function $f(n)$.

An even more general theorem which gives one formula instead of the three cases above is called Akra-Bazzi theorem. In our class we always approximate $f(n)$ with appropriate $cn^k$ and use Theorem 1 instead and we obtain $\theta$ usually (and not just $O$).

3 Recurrence Relations With Full History

A full history recurrence relation is one that depends on all the previous functions. We use the method of elimination of history, in which we will try to write the recurrence in such a way that most of the terms will be cancelled (we used such an approach before while computing sums).

3.1 A recurrence in simplest form

A simplest form for a recurrence relation is

$$T(n) = c + \sum_{i=1}^{n-1} T(i)$$

Then $T(n+1) - T(n) = T(n+1) = 2T(n) \Rightarrow T(n+1) = 2^n T(1)$. But say if $T(1) = 1$ and $c = 5$ then $T(2) = 6 \neq 2T(1)$. The base case is $T(2) - T(1) = c \neq T(1)$. Thus $T(2) = T(1) + c$ (by definition) and $T(n+1) = 2T(n)$ for $n > 2$. Hence $T(n+1) = (T(1) + c)2^{n-1}$. Note that $c$ did not appear in the formula which was strange. Always try for more base cases to avoid such situations.

3.2 A More Involved Example

Now let us see a more complicated example which we will use later in the analysis of Quicksort. Base case is $T(1) = 0$ and the general term for $n \geq 2$ is given by:

$$T(n) = (n - 1) + \frac{2}{n} \sum_{i=1}^{n-1} T(i)$$

(2)

Multiplying both sides by $n$ gives

$$nT(n) = n(n - 1) + 2 \sum_{i=1}^{n-1} T(i)$$

(3)

Shifting the index ahead by 1 gives

$$(n + 1)T(n + 1) = n(n + 1) + 2 \sum_{i=1}^{n} T(i)$$

(4)
Subtracting Equation 3 from Equation 4 gives 

\[(n+1)T(n+1) - nT(n) = 2n + 2T(n)\]

which implies 

\[T(n+1) = \frac{2n}{n+1} + \frac{n+2}{n+1}T(n) \leq 2 + \frac{n+2}{n+1}T(n).\]

Expanding we have

\[T(n) \leq 2 + \frac{n+1}{n}T(n-1)\]

\[= 2 + \frac{n+1}{n} \left(2 + \frac{n}{n-1} \left(2 + \frac{n-1}{n-2} \left(\ldots \frac{4}{3}\right)\right)\right)\]

\[= 2 \left(1 + \frac{n+1}{n} + \frac{n+1}{n-1} + \frac{n+1}{n-2} + \ldots + \frac{n+1}{n-1} n - 2 \ldots \frac{4}{3}\right)\]

\[= 2(n+1) \left(\frac{1}{n+1} + \frac{1}{n} + \frac{1}{n-1} + \ldots + \frac{1}{3}\right)\]

\[= 2(n+1) \left(H(n+1) - \frac{3}{2}\right)\]

where \(H(n) = 1 + \frac{1}{2} + \frac{1}{3} + \ldots + \frac{1}{n}\) is called as the Harmonic Series. It is easy to see that (say by integration) that \(H(n) = \ln n + 0.577 + O\left(\frac{1}{n}\right)\). Thus \(T(n)\) is \(O(n \ln n)\).

References