CIS 575. Introduction to Algorithm Analysis
Material for February 5, 2024

Solving Recurrences, Using Mathematical Induction

This note covers material also covered in *Cormen*'s Section 4.3.

1 Solving Recurrences

For the (worst case) running time $T(n)$ of a given recursive algorithm, there will often exist constants a and b, and a function f, such that for "big enough" n it holds that

$$T(n) = aT(\frac{n}{b}) + f(n) \tag{1}$$

A recurrence of the form (1) shows up when we have a recursive function that on input of size n (that is not "small")

- calls itself recursively a times, where each recursive call is on an argument b times smaller than the input parameter, and
- has an *overhead*, that is running time apart from the recursive calls, described by $f(n)$.

In the previous note, we analyzed **Merge Sort** where $a = 2$, $b = 2$, and $f(n) \in \Theta(n)$.

We must require that $b > 1$ (as otherwise the recursion will go on forever), perhaps even $b \geq 2$ (though some applications have $b = 1.5$). Since $T(n)$ is defined only with n a natural number, we will need to wrap either "floor" ($\lfloor\ \rfloor$) or "ceiling" ($\lceil\ \rceil$) around $\frac{n}{b}$.

We shall now address how to **solve** such recurrences so as to find the **asymptotic** behavior of $T(n)$ (which turns out *not* to depend on how $T(n)$ behaves for "small" n, or whether we wrap floor or ceiling around $\frac{n}{b}$). That is, we want to find $g(n)$ such that $T(n) \in \Theta(g(n))$, though we may be less ambitious and only go for $T(n) \in O(g(n))$ or $T(n) \in \Omega(g(n))$. Our agenda is:

1. first we shall present a general method, the **substitution method**, for verifying that a **given** g solves a given recurrence;
2. next we shall present a (quite general but occasionally too narrow) recipe for **finding** the solution to a given recurrence.

To prepare for the substitution method, it is important to be familiar with the principle of

Mathematical Induction which we shall illustrate by a simple example: looking at the first n natural numbers, we may *conjecture* that they add up to $n(n+1)/2$. With $\sum_{i=1}^{n} i$ the sum of the first n natural numbers, we may thus believe that for all $n \geq 0$ (in many situations we would consider only $n \geq 1$) it holds that

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

We may check that claim for various values of n:

$n = 0$**:** The claim is $0 = \frac{0 \cdot 1}{2}$ which is obviously true

$n = 1$**:** The claim is $1 = \frac{1 \cdot 2}{2}$ which is obviously true

$n = 2$**:** The claim is $1 + 2 = \frac{2 \cdot 3}{2}$ which is obviously true

$n = 3$**:** The claim is $1 + 2 + 3 = \frac{3 \cdot 4}{2}$ which is obviously true

We may go on like this, and become increasingly confident that our conjecture is true, but we can never be certain, no many how many numbers we consider. To verify our conjecture beyond any doubt, we need a **proof by induction**. The key part of such a proof is the **inductive step** which aims to show that, for a given $m > 0$, that **if** (2) holds for all $n \geq 0$ with $\mathbf{n < m}$ **then** (2) also holds for $\mathbf{n = m}$. Let us embark on the inductive step: for $n = m > 0$, we have the calculation (**IH** marks the use of the **Induction Hypothesis**)

$$\begin{aligned}
& \sum_{i=1}^{m} i \\
= \; & \sum_{i=1}^{m-1} i + m \\
\mathbf{IH} \quad = \; & \frac{(m-1)((m-1)+1)}{2} + m \\
= \; & \frac{m^2 - m + 2m}{2} \\
= \; & \frac{m(m+1)}{2}
\end{aligned}$$

which shows that (2) *does* hold for $n = m$ if we **inductively** assume that (2) holds for all $n \geq 0$ with $n < m$, in particular for $n = m - 1$.

We have already seen the **base case** of the inductive proof: that (2) holds for $n = 0$. The inductive step now makes it possible to infer that (2) holds for **all** $n \geq 0$: first for $n = 1$, next for $n = 2$, then for $n = 3$, etc, etc.