Reading #5

Analyzing the Running Time of a Simple Recursive Algorithm

Recurrences

Example: The Algorithm We Started With

Recall the recursive algorithm `maxInRange` considered in the notes for Reading #2: If the precondition for the problem it solves is satisfied when an execution of this algorithm begins then it receives an integer array `A` with some positive length n. and integers `low` and `high` such that

$$0 \leq \texttt{low} \leq \texttt{high} \leq n - 1$$

as its inputs.

It will be useful to analyze the running time of this algorithm as a function of the value

$$k = \texttt{high} - \texttt{low} + 1$$

because this is the number of entries of the array `A` that this algorithm examines when executed with these inputs. Since `low` $\leq$ `high`, $k \geq 1$ if the precondition for the computational problem being solved is satisfied.
The algorithm is as follows.

```
integer maxInRange (integer[] A, integer low, integer high) {
1.  if (low == high) {
2.    return A[low]
    } else {
3.    integer mid := floor((low + high)/2)
4.    return max(maxInRange(A, low, mid), maxInRange(A, mid+1, high))
    }
}
```

Let $T(k)$ be the number of steps executed by this algorithm for a given value of k — assuming, as usual, that the uniform cost criterion is used to define this.

- If $k = 1$ then $\texttt{low} = \texttt{high}$, so that the test at line 1 is passed and the execution of the algorithm ends after the execution of step 2: $T(1) = 2$.

Consider, instead, an execution of the algorithm such that $k \geq 2$ — so that $\texttt{high} = \texttt{low} + k - 1 \geq \texttt{low} + 1$.

- This time, the test at line fails, so the execution of the algorithm continues with an execution of step 3.
- It will be useful to notice that, when this step is executed, `mid` receives the value

$$\left\lfloor \frac{(\texttt{low} + \texttt{high})}{2} \right\rfloor = \left\lfloor \frac{2 \times \texttt{low} + k - 1}{2} \right\rfloor = \texttt{low} + \left\lfloor \frac{k-1}{2} \right\rfloor .$$

- The execution of the algorithm ends with an execution of step 4 — but it *also* calls itself recursively, twice — with the same input array `A`, but
  - with the same input `low`, but with `high` replaced by `mid`, and
  - with `low` replaced by $\texttt{mid} + 1$ and with the same input `high`.

  When defining running times for recursive algorithms we *must* also include the times for these recursive calls, and we should do so as accurately as we can.
- For the first recursive execution, $k = \texttt{high} - \texttt{low} + 1$ is replaced by

$$\texttt{mid} - \texttt{low} + 1 = \left\lfloor \frac{k-1}{2} \right\rfloor + 1.$$

  The cost of this recursive application is, therefore,

$$T\left(\left\lfloor \frac{k-1}{2} \right\rfloor + 1 \right).$$

- For the second recursive execution, $k = \texttt{high} - \texttt{low} + 1$ is replaced by

$$\begin{aligned} \texttt{high} - (\texttt{mid} + 1) + 1 = \texttt{high} - \texttt{low} - \left\lfloor \frac{k-1}{2} \right\rfloor \\ = k - \left\lfloor \frac{k-1}{2} \right\rfloor \\ = \left\lceil \frac{k-1}{2} \right\rceil . \end{aligned}$$

  The cost of this recursive application is, therefore,

$$T\left(\left\lceil \frac{k-1}{2} \right\rceil \right).$$

Conclusion: When executed with the problem's precondition satisfied, and such that $k = \texttt{high} - \texttt{low} + 1 \geq 1$, the algorithm carries out $T(k)$ steps, where

$$T(k) = \begin{cases} 2 & \text{if } k = 1, \\ T\left(\lfloor \frac{k-1}{2} \rfloor + 1\right) + T\left(\lceil \frac{k-1}{2} \rceil\right) + 3 & \text{if } k \geq 2. \end{cases}$$

Note: When algorithms are more complicated we will often get *inequalities* that look like this, instead of *equations*.

That's OK... They will still be useful!

Recurrences: Definition

Definition 1. In mathematics, a **recurrence** (or "recurrence relation") is generally defined to be a relation that recursively defines the elements of a sequence of values.

If one considers the sequence of values

$$T(1), T(2), T(3), T(4), \ldots$$

then one can view the above expression

$$T(k) = \begin{cases} 2 & \text{if } k = 1, \\ T\left(\lfloor \frac{k-1}{2} \rfloor + 1\right) + T\left(\lceil \frac{k-1}{2} \rceil\right) + 3 & \text{if } k \geq 2. \end{cases}$$

to be a "recurrence" (that defines this sequence).

Solving Recurrences

Sometimes a recurrence is simple enough that it suffices to compute initial values, look for a pattern, and ***guess*** a solution.

Example: The recurrence for $T(k)$ can be used to confirm that

- $T(1) = 2$;
- $T(2) = T(1) + T(1) + 3 = 2 + 2 + 3 = 7$;
- $T(3) = T(2) + T(1) + 3 = 7 + 2 + 3 = 12$;
- $T(4) = T(2) + T(2) + 3 = 7 + 7 + 3 = 17$;
- $T(5) = T(3) + T(2) + 3 = 12 + 7 + 3 = 22$; and
- $T(6) = T(3) + T(3) + 3 = 12 + 12 + 3 = 27$.

Notice that $T(k) = T(k-1) + 5$ for every positive integer k such that $2 \leq k \leq 6$. Based on this, one might ***guess*** that there is a constant c such that

$$T(k) = 5k + c$$

for every positive integer k.
Since $T(1) = 2 = 5 \cdot 1 - 3$, it must be the case that $c = -3$, so that

$$T(k) = 5k - 3$$

for every positive integer k, if this ***guess*** is correct.

Verifying Recurrences

If a guessed solution for a recurrence is correct then it is often possible to use **mathematical induction** to prove this — as shown, for the ongoing example, below.

Claim 2. *If T is a function of a positive integer k such that*

$$T(k) = \begin{cases} 2 & \text{if } k = 1, \\ T\left(\left\lfloor \frac{k-1}{2} \right\rfloor + 1\right) + T\left(\left\lceil \frac{k-1}{2} \right\rceil\right) + 3 & \text{if } k \geq 2. \end{cases}$$

then $T(k) = 5k - 3$ for every positive integer k.

The following lemma will be of use when proving this claim.

Lemma 3. *Suppose that h is an integer such that $h \geq 1$.*

(a) $1 \leq \left\lfloor \frac{h}{2} \right\rfloor + 1 \leq h$.

(b) $1 \leq \left\lceil \frac{h}{2} \right\rceil \leq h$.

(c) $\left(\left\lfloor \frac{h}{2} \right\rfloor + 1\right) + \left\lceil \frac{h}{2} \right\rceil = h + 1$.

Proof. It will be helpful to consider the cases that h is even, and that h is odd, separately.

Case: h is even. Since h is a positive integer, $h = 2\ell$ for an integer $\ell \geq 1$.

- In this case, $\left\lfloor \frac{h}{2} \right\rfloor + 1 = \left\lfloor \frac{2\ell}{2} \right\rfloor + 1 = \ell + 1$, so that $1 \leq \left\lfloor \frac{h}{2} \right\rfloor + 1 = \ell + 1 \leq 2\ell = h$, as required to establish part (a) of the claim in this case.
- Furthermore, $\left\lceil \frac{h}{2} \right\rceil = \left\lceil \frac{2\ell}{2} \right\rceil = \ell$, so that $1 \leq \left\lceil \frac{h}{2} \right\rceil = \ell \leq 2\ell = h$, as required to establish part (b) of the claim in this case as well.
- Finally, $\left(\left\lfloor \frac{h}{2} \right\rfloor + 1\right) + \left\lceil \frac{h}{2} \right\rceil = (\ell + 1) + \ell = 2\ell + 1 = h + 1$, as required to establish part (c) of the claim in this case.

Case: h is odd. Since h is a positive integer, $h = 2\ell + 1$ for an integer $\ell \geq 0$.

- In this case, $\left\lfloor \frac{h}{2} \right\rfloor + 1 = \left\lfloor \frac{2\ell+1}{2} \right\rfloor + 1 = \ell + 1$, so that $1 \leq \left\lfloor \frac{h}{2} \right\rfloor + 1 = \ell + 1 \leq 2\ell + 1 = h$, as required to establish part (a) of the claim in this case.
- Furthermore, $\left\lceil \frac{h}{2} \right\rceil = \left\lceil \frac{2\ell+1}{2} \right\rceil = \ell + 1$ as well, so that $1 \leq \left\lceil \frac{h}{2} \right\rceil = \ell + 1 \leq 2\ell + 1 = h$, as required to establish part (b) of the claim in this case as well.
- Finally, $\left(\left\lfloor \frac{h}{2} \right\rfloor + 1\right) + \left\lceil \frac{h}{2} \right\rceil = (\ell + 1) + \ell + 1 = 2\ell + 2 = h + 1$, as required to establish part (c) in this case, and to complete the proof of the lemma. □

Proof of Claim 2. The claim will be proved by induction on k. The strong form of mathematical induction will be used, and the case that $k = 1$ will be considered in the basis.

Basis: If $k = 1$ then $T(k) = 2$, by definition, while $5 \cdot k - 3 = 5 \cdot 1 - 3 = 2$ as well, as required to establish the claim in this case.

Inductive Step: Let h be an integer such that $h \geq 1$. It is necessary and sufficient to use the following

> Inductive Hypothesis: $T(\,j\,) = 5j - 3$ for every integer j such that $1 \leq j \leq h$.

to prove the following

> Inductive Claim: $T(h + 1) = 5(h + 1) - 3$.

With that note, let h be an integer such that $h \geq 1$, that is, a positive integer.
In this case

$$
\begin{aligned}
T(h+1) &= T\left(\left\lfloor \frac{h}{2} \right\rfloor + 1\right) + T\left(\left\lceil \frac{h}{2} \right\rceil\right) + 3 && \text{(by definition, since } h+1 \geq 2\text{)} \\
&= 5 \cdot \left(\left\lfloor \frac{h}{2} \right\rfloor + 1\right) - 3 + T\left(\left\lceil \frac{h}{2} \right\rceil\right) + 3 \\
& && \text{(by the Inductive Hypothesis and part (a) of the above lemma)} \\
&= 5 \cdot \left(\left\lfloor \frac{h}{2} \right\rfloor + 1\right) - 3 + 5 \cdot \left\lceil \frac{h}{2} \right\rceil - 3 + 3 \\
& && \text{(by the Inductive Hypothesis and part (b) of the above lemma)} \\
&= 5 \cdot \left(\left(\left\lfloor \frac{h}{2} \right\rfloor + 1\right) + \left\lceil \frac{h}{2} \right\rceil\right) - 3 && \text{(gathering terms)} \\
&= 5 \cdot (h+1) - 3 && \text{(by part (c) of the above lemma)}
\end{aligned}
$$

as to required to establish the Inductive Claim, complete the Inductive Step, and establish the claim. □

An Occasional Mistake

On tests and assignments, students are sometimes asked to give ***recurrences*** bounding the running times of algorithms like `maxInRange`.
If the algorithm given in the question *is* `maxInRange` then the expected answer would be

$$T(k) = \begin{cases} 2 & \text{if } k = 1, \\ T\left(\left\lfloor \frac{k-1}{2} \right\rfloor + 1\right) + T\left(\left\lceil \frac{k-1}{2} \right\rceil\right) + 3 & \text{if } k \geq 2. \end{cases}$$

where $k = \texttt{low} - \texttt{high} + 1$, as above.
Instead of this, they sometimes give either the answer

$$T(k) = 5k - 3$$

or, possibly, the answer

$$T(k) = \begin{cases} 2 & \text{if } k = 1, \\ 5k - 3 & \text{if } k \geq 2. \end{cases}$$

There are several reasons why this is problematic:

1. The other answers are — at least arguably — not ***recurrences*** at all! They do not given an expression that is equal to (or bounds) the value of a function at one value, using values of the function (or bounds for it) at smaller values.

   So, a marker would have reason to wonder whether the student even knows what the word "recurrence" means!

2. While the "expected answer" is something that can be derived directly from the source code (or pseudocode) the other answers require some extra calculations — or, perhaps, guesswork. Even if a marker is willing to accept a "solution for a recurrence" instead of the recurrence itself, some sort of *explanation* of how the student actually *obtained* the solution — or a *proof* — would be also be expected.

3. The situation is even more complicated if the student has made an *arithmetic* or *logical error* somewhere along the way — so that answer returned was $2k$, k^2, or 2^k instead of $5k - 3$... It is hard to give part marks under these circumstances.

4. Finally if the expected recurrence was reasonably easy to derive, the *next question* might be something like

   > "Now use your recurrence to prove that $T(k) = 5k - 3$ for every integer $n \geq 0$."

   That is hard to do if the student never gave a recurrence at all — especially if arithmetic or logical errors also got made.