Paper Revision

Xin Liu

September 12, 2023

Discuss with Charles Troutman

Question 1: Given the recurrence relation:

$$T(n) = T\left(\frac{3}{10}n + 18\right) + T\left(\frac{2}{5}n - 15\right) + an$$

Guess:

$$T(n) = \Theta(n)$$

Assumption:

$$T(n) \leq an$$

Let's substitute the assumed value of $T(n)$ into the recurrence relation:

$$\begin{aligned}
T(n) &= a\left(\frac{3}{10}n + 18\right) + a\left(\frac{2}{5}n - 15\right) + an \\
&\leq \frac{3}{10}an + 18a + \frac{2}{5}an - 15a + an \\
&\leq \frac{7}{10}an + 3a \\
&\leq an \text{ (When n large than 10a)}
\end{aligned}$$

Runtime

So we proved, the runtime is $\Theta(n)$

Question 2: The largest number of peek is $\lceil \frac{n}{2} \rceil$. We can set a matrix, in which odd row is $1, -1, 1, -1, 1, -1, ..., 1, -1$, the even row is $-1, 1, -1, 1, -1, 1, ..., -1, 1$. In this matrix, every 1 is a peek.

If the number of row is odd, we have one more $1, -1, 1, -1, 1, -1, ..., 1, -1$, so the number of peek is one more than non-peek number. It is $\lceil \frac{n}{2} \rceil$

If the number of row is even, we have same number of peek and non-peek, so the number of peek is $n/2$.

Question 3: $\Theta(n\log(n))$

Question 4: I come up with a $\Theta(n^2)$ divide-conquer algorithm.

Given an $n \times n$ matrix of distinct integers, we aim to identify a peak, defined as a number that is strictly larger than each of its neighbors (top, bottom, left, right). A proposed divide and conquer algorithm to achieve this is as follows:

1. **Divide:** Recursively divide the $n \times n$ matrix into four smaller $n/2 \times n/2$ matrices until reaching the base case of a 1×1 matrix. In this base case, the single element is treated as a peak.
2. **Conquer:** For each sub-matrix, identify peaks using the same recursive strategy.
3. **Combine:** When combining the results from the four smaller matrices, only check the boundary elements to see if they qualify as peaks. Given an $n \times n$ matrix, at most $4n$ boundary elements need checking(Row $\frac{n-1}{2}$ and $\frac{n+1}{2}$, Column $\frac{n-1}{2}$ and $\frac{n+1}{2}$). Most of these elements have only one neighbor, while corner elements have two neighbors. So the combine time is $\Theta(n)$.

Question 5: The recurrence relation for the algorithm is given by:

$$T(n) = 4T\left(\frac{n}{2}\right) + \Theta(n)$$

Easily, use Master theory,

Watershed function is:$n^log_2^4$.

Driving function is n.

So the watershed function is dominant.

The run time is $\Theta(n^2)$