PS1

June 18, 2025

Asymptotic complexity, recurrence relations, peak finding

problem 1-1

The order of the functions in increase assymptotic order
(a)

$$O\left(n^{0.9999}\log n\right), O\left(10^8 n\right), O\left(n^2\right), O\left(1.00001^n\right)$$

(b)

$$O\left(2^{2^{10^9}}\right), n^{3/2}, n\left(n-1\right), 2^{n10^6}$$

(c)

$$\sum_{i=1}^{n}\left(i+1\right) = n + \frac{n\left(n+1\right)}{2} = O\left(n^2\right)$$

$$log\left(n^{\sqrt{n}}\right) = O\left(\sqrt{n}logn\right)$$

$$\log\left(n^{10}2^{n/2}\right) = O\left(\frac{n}{2}\right)$$

$$O\left(\sum_{i=1}^{n}\left(i+1\right)\right), O\left(n^{\sqrt{n}}\right), O\left(n^{10}2^{n/2}\right), O\left(2^n\right)$$

Problem 1-2

(a)

$$T\left(x,c\right) = \Theta\left(x\right)$$

for $c \leq 2$

$$T\left(c,y\right) = \Theta\left(y\right)$$

for $c \leq 2$

$$T\left(x,y\right) = \Theta\left(x+y\right) + T\left(x/2, y/2\right)$$

$$T(n,n) = \Theta(2n) + T(n/2, n/2) = \Theta(2n) + \Theta(n) + \Theta\left(\frac{n}{2}\right) + \cdots$$
$$= \Theta\left(n\left(2 + 1 + \frac{1}{2} + \cdots\right)\right) = \Theta(4n) = \Theta(n)$$

(b)

$$T(x,c) = \Theta(x)$$

for $c \leq 2$

$$T(c,y) = \Theta(y)$$

for $c \leq 2$

$$T(x,y) = \Theta(x) + T(x, y/2)$$
$$T(n,n) = \Theta(n) + T(n, n/2) = \Theta(n) + \Theta(n) + T(n, n/4) = O(n\log(n))$$

(c)

$$T(x,c) = \Theta(x)$$

for $c \leq 2$

$$T(x,y) = \Theta(x) + S(x, y/2)$$
$$T(n,m) = O(n) + T\left(n, \frac{m}{2}\right) = O(n) + T\left(n, \frac{m}{2}\right)$$
$$S(c,y) = \Theta(y)$$
$$S(x,y) = \Theta(y) + T\left(\frac{x}{2}, y\right)$$
$$T(n,n) = \Theta(n) + S(n, n/2)$$
$$= \Theta(n) + \Theta\left(\frac{n}{2}\right) + T\left(\frac{n}{2}, \frac{n}{2}\right)$$
$$= \Theta\left(3\frac{n}{2}\right) + T\left(\frac{n}{2}, \frac{n}{2}\right) = \Theta\left(\frac{3}{2}n\right) + \Theta\left(\frac{3}{2}\frac{n}{2}\right) + \Theta\left(\frac{3}{2}\frac{n}{4}\right) + \cdots$$
$$= \Theta\left(\frac{3}{2}n\left(1 + \frac{1}{2} + \cdots\right)\right) = \Theta(3n) = \Theta(n)$$

Problem 1-3

(a) Algorithm1 is the one descirbed in class: starts at column $m/2$ finds the maximum on the column, then checks the neighbors of the maximum, if it isn't a maxium recurses on half of the original problem. The algorithm is corrent and takes $O(n\log m)$, where m is the number of rows and m is the number of columns.

(b) Algorithm2 is greedy algorithm which checks for a better neighbor, if there is recurses on the problem. The algorithm is correct but is $O(nm)$.

(c) Algorithm3 checks the maximum on a cross which center is in the middle of the grid, chooses the best sub-problem and recurses. This algorithm is incorrect since, by jumping back to the center of the subproblem we can miss the peak

(d) Algorithm4 is similar to Algorithm1 but alternates between splitting rows and splitting columns. The algorithm is correct.

Problem 1-4

For size of $n \times n$

(a) The worst case runtime of Algorithm1 is $O(n \log n)$.

(b) The worst case runtime of Algorithm2 is $O(n^2)$.

(c) The algorithm is incorrect..., worst case runtime of Algorithm3 is (same solution as 1-2 (a))

$$T(n, n) = O(n) + T\left(\frac{n}{2}, \frac{n}{2}\right) = O(n)$$

(d) The worst case runtime of Algorithm4 is $O(n)$, following the same alysis of problem 1-2 (c)

Problem 1-6

An example of a counter example for Algorithm3 is

$$\begin{array}{cccccc}
4 & 3 & 2 & 0 & 0 & 0 \\
 & 2 & 1 & 0 & 0 & 0 \\
3 & 2 & 1 & 0 & 0 & 0 \\
 & 0 & 0 & 0 & 0 & 9 \\
 & 1 & 1 & 0 & 1 & 8
\end{array}$$

where all the empty enteries are zeros.