## CSCI 305 Assignment 3

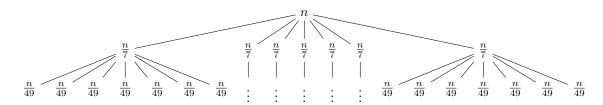
(Solo) Isaac Boaz

November 4, 2023

1. Provide a  $\Theta$  bound for the solution of each of these recurrences.

1.

$$T(n) = 7T(n/7) + n$$



We see each level has  $7^i \cdot \frac{n}{7^i} = n$  work done. Since n is being divided by 7 each level, this will be run  $\log_7 n$  times.

$$n \cdot \log_7 n \to \Theta(n \log n)$$

2.

$$T(n) = 9T(n/3) + n^{2}$$

$$(\frac{n}{3})^{2} \qquad \frac{n^{2}}{9} \qquad \frac{n^{2}}{9} \qquad \frac{n^{2}}{9} \qquad \frac{n^{2}}{9} \qquad \frac{n^{2}}{9} \qquad \frac{n^{2}}{9} \qquad \frac{n^{2}}{9}$$

$$(\frac{n}{9})^{2} \qquad (\frac{n}{3^{2}})^{2} \qquad \vdots \qquad \vdots \qquad \vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

$$(\frac{n}{3^{3}})^{2}$$

$$\vdots \qquad \vdots \qquad \vdots \qquad \vdots \qquad \vdots$$

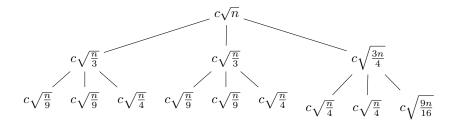
At each level we do  $n^2$  amount of work. Since we're dividing by 3 each time, we will do  $\log_3 n$  levels. Thus, our runtime is  $n^2 \cdot \log_3 n \to \Theta(n^2 \log n)$ 

3.

$$T(n) = 49T(n/25) + n^{3/2}\log n$$

2. Draw the recurrence tree for the following recurrence:

$$T(n) = 2T(n/3) + T(3n/4) + c\sqrt{n}$$



1. Give an asymptotic  $\Theta$ -bound for lines 1-3.

$$\Theta(1)$$

2. Give an asymptotic  $\Theta$ -bound for lines 6-9.

$$\Theta(n)$$

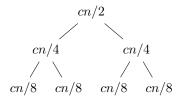
3. What size array is being input and output?

We see the first array takes the Fourier Transform of the even-indexed elements, and the second array takes the Transform of the odd-indexed elements. Thus, each array is n/2 in size.

4. Recurrence relation for the cost of T(n).

$$T(n) = 2T(\frac{n}{2}) + O(\frac{n}{2})$$
$$= 2T(\frac{n}{2}) + c\frac{n}{2}$$

5. Solve the above recurrence relation.



Going by the tree diagram, we see each level x has  $\frac{n}{2}$  work. Since we halve the size of the problem each level, there will be  $\log_2 n$  levels, amounting to a runtime of

$$O(n \log n)$$

Additionally, since the subtree is balanced, we can say it is both  $O(n \log n)$  and  $\Omega(n \log n)$ , and thus  $T(n) = \Theta(n \log n)$ .

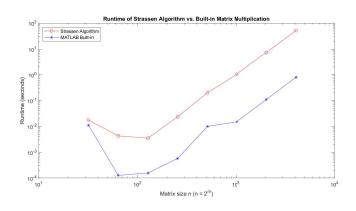
6. Find the  $\Theta$  cost of slowFT. Since the outer for loop is run n times, and the inner loop is consequently run  $n^2$  times, we know that Algorithm 1 is asymptotically faster.

$$\Theta(n^2) > \Theta(n \log n)$$

## 4. Strass

- 1. The base case for this algorithm is when it encounters matrices that are  $16 \times 16$  or smaller.
- 2. I have verified this.

	n	s(n)	m(n)
	32	0.0176	0.0112
	64	0.0043	0.0001
	128	0.0035	0.0002
3.	256	0.0234	0.0006
	512	0.1993	0.0100
	1024	1.0313	0.0150
	2048	7.3894	0.1085
	4096	51.6236	0.7990
		ı	ı



- trend lines seems to roughly match, with a constant factor making the Strass algorithm slower.
- 4. The naive algorithm for matrix multiplication does  $(n-1)n^2$  additions / subtractions for summing up each row/column product, whereas the Stras algorithm will always do more additions / subtractions. Thus, since the Strass algorithm is likely to do more subtractions, it is less stable.