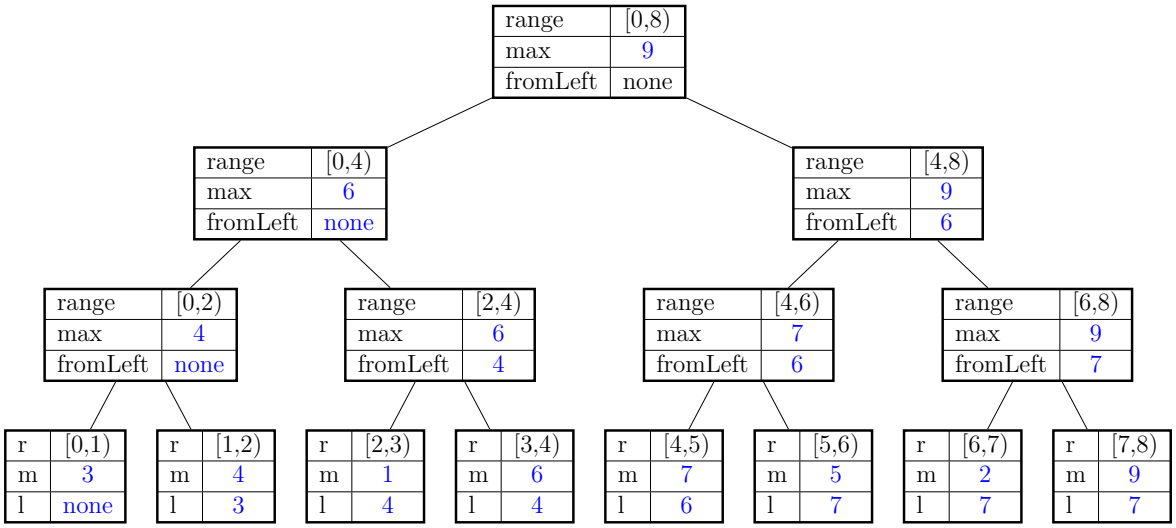
1 Parallel Prefix FindMax

a) Given input array [3,4,1,6,7,5,2,9], output an array such that each `output[i] = max(input[0], input[1], ...,input[i])`, using the idea of Parallel Prefix Sum algorithm from the reading. Show the intermediate steps. Fill the output array, and for each step, show the tree of the recursive task objects that would be created (where a node's child is for two problems of half the size) and the fields each node needs. Do **not** use a sequential cut-off. (Hint: run the up pass (first pass) to compute **`max`**, then run the down pass (second pass) to compute **`fromLeft`**.)



`Input:`	3	4	1	6	7	5	2	9
`Output:`	3	4	4	6	7	7	7	9

b) **(Optional)** Give formulas for the following values where **`p`** is a reference to a non-leaf tree node and **`leaves[i]`** refers to the leaf node in the tree visible just above the corresponding location in the `input` and `output` arrays above. Use `p.left` to refer to the left child of `p` and `p.right` to refer to the right child of `p`.

- `leaves[i].max = input[i]`
- `p.max = max(p.left.max, p.right.max)`
- `p.right.fromLeft = max(p.left.max, p.fromLeft)`
- `p.left.fromLeft = p.fromLeft`
- `output[i] = max(leaves[i].max, leaves[i].fromLeft)`

c) Is Parallel Prefix FindMax a map or a reduction? Why? ● map ○ reduction

We are mapping the problem from an input array of **size N** to an output array of **size N**.

Will you want to pick up your worksheet later? Circle one: Yes / No
How confident do you feel with the material this week? Circle one: 1 / 2 / 3 / 4

2 Parallel Algorithm Analysis

Work and Span

- **Work** (T_1): how long it takes to run on **one** processor.
- **Span** (T_∞): how long it takes to run on an **unlimited** number of processors.

Terms for performances

- **Speed-up** on P processors (T_1/T_P): how much faster the algorithm runs given the extra processors.
- **Perfect Linear Speed-up**: occurs when $\frac{T_1}{T_P} = P$; speed-up is P as we vary P.
- **Parallelism** (T_1/T_∞): maximum possible speed-up.

Amdahl's law

- Suppose we have an algorithm with S portion of sequential work and $(1 - S)$ portion of parallel work with perfect linear speed-up.
- **Speed-up**: $\frac{T_1}{T_P} = \frac{1}{S+\frac{(1-S)}{P}}$
- **Parallelism**: $\frac{T_1}{T_\infty} = \frac{1}{S}$

a) Choose an algorithm that its **best-case span** can be represented by the following recurrence: $T(N) = T\left(\frac{N}{2}\right) + c_1 N + c_0$.

- ○ Quicksort (Sequential sort & Sequential partition)
- ○ Quicksort (Sequential sort & Parallel partition)
- ● Quicksort (Parallel sort & Sequential partition)
- ○ Quicksort (Parallel sort & Parallel partition)

b) What are the **recurrences representing the worst-case span** of the following algorithms? Given, from the reading, that **the worst-case parallel merge span** is in $\Theta(log^2 N)$.

Mergesort (Sequential sort & Sequential merge)	$T(N) = 2T\left(\frac{N}{2}\right) + c_1 N + c_0$
Mergesort (Sequential sort & Parallel merge)	$T(N) = 2T\left(\frac{N}{2}\right) + c_1 log^2 N + c_0$
Mergesort (Parallel sort & Sequential merge)	$T(N) = T\left(\frac{N}{2}\right) + c_1 N + c_0$
Mergesort (Parallel sort & Parallel merge)	$T(N) = T\left(\frac{N}{2}\right) + c_1 log^2 N + c_0$

c) **(Optional)** What ***fraction of a program must be parallelizable*** in order to get 10x speedup on 30 processors?

Answer: $\frac{27}{29}$

Will you want to pick up your worksheet later? Circle one: Yes / No
How confident do you feel with the material this week? Circle one: 1 / 2 / 3 / 4

We have $P = 30$ and $\frac{T_1}{T_P} = 10$. Then, by Amdahl's law

$$\begin{aligned}
\frac{T_1}{T_p} &= \frac{1}{S + \frac{(1-S)}{P}} \\
10 &= \frac{1}{S + \frac{1-S}{30}} \\
10S + 10\frac{(1-S)}{30} &= 1 \\
30S + (1-S) &= 3 \\
29S &= 2 \\
S &= \frac{2}{29}
\end{aligned}$$

Hence, the portion of parallelizable work is $1 - S = \frac{27}{29}$.

d) **(Optional)** Is it possible to get 100x speedup on a program of which 19/20 is parallelizable? If so, how many processors would you need? ○ Yes _____ ● No

Since 19/20 of the program is parallelizable, $S = 1/20$ of the program is sequential work. By Amdahl's law, the parallelism of this program is

$$\begin{aligned}
\frac{T_1}{T_\infty} &= \frac{1}{S} \\
&= \frac{1}{\frac{1}{20}} \\
&= 20
\end{aligned}$$

That means 20x is the maximum possible speed-up of this program. Hence, it is not possible to get 100x speed-up on this program.