Activity: Extracting Parallelism (Recursive)

Extracting dependency from code is an almost automatic process. You need to choose a granularity. But once that is chosen, the entire analysis follows.

In the whole activity, you should express the metrics in complexity notation as a function of the parameters of the functions.

1 Fast Exponentiation

Consider this function to compute x^n where n is a positive integer.

```
double expBySquaring(double x, int n) {
  if (n == 0)
    return 1;
  if (n == 1)                                          Task PreN
    return x;
  if (n % 2 == 0)
    return expBySquaring(x * x,  n / 2);
  else
    return x * expBySquaring(x * x, (n - 1) / 2);
}
```

Question: What is the complexity of this function? O(n)
Question: Extract the dependencies.
Question: What is the width? 1
Question: What is the work? log(n)
Question: What is the critical path? What is its length? log(n)

[illegible] extracting 3

1 Fast Exponentiation

$$\begin{tikzcd}
\text{Pre 5} \arrow[r, "RN{=}5"] \arrow[d, dash] & \text{Pre 2} \arrow[r, "RN{=}2"] & \text{Pre 1} \arrow[r, "RN{=}1"] & \text{End} \\
\text{Pre 10} \arrow[r, dash, "RN{=}10"] & {}
\end{tikzcd}$$

$$\begin{tikzcd}
\text{Pre 15} \arrow[r] & \text{Pre 7} \arrow[r] & \text{Pre 3} \arrow[r] & \text{Pre 1} \arrow[r] & \text{End}
\end{tikzcd}$$

2 Dense Matrix Matrix Multiplication Recursively

Consider this algorithm to compute $C = A * B$ when A, B, and C are $n \times n$ matrices where n is a power of 2.

```
Multiply(A, B):
  A11 = A[1..n/2][1..n/2]
  A12 = A[1..n/2][n/2..n]
  A21 = A[n/2..n][1..n/2]
  A22 = A[n/2..n][n/2..n]
                                              Task Pre-n
  B11 = B[1..n/2][1..n/2]
  B12 = B[1..n/2][n/2..n]
  B21 = B[n/2..n][1..n/2]
  B22 = B[n/2..n][n/2..n]

  C11 = A11*B11 + A12*B21                     Task D1n
  C12 = A11*B12 + A12*B22                     Task D2n
  C21 = A21*B11 + A22*B21                     Task D3n
  C22 = A21*B12 + A22*B22                     Task D4n

  return [[C11, C12],[C21, C22]]              Task Post-n
```

Note that the $*$ operation are done by recursively calling the Multiply function. And that the $+$ operation is a matrix operation.

Question: What is the complexity of this function? (Hint: use Master theorem) O(n^4)

Question: Extract the dependencies.

Question: What is the width? 1

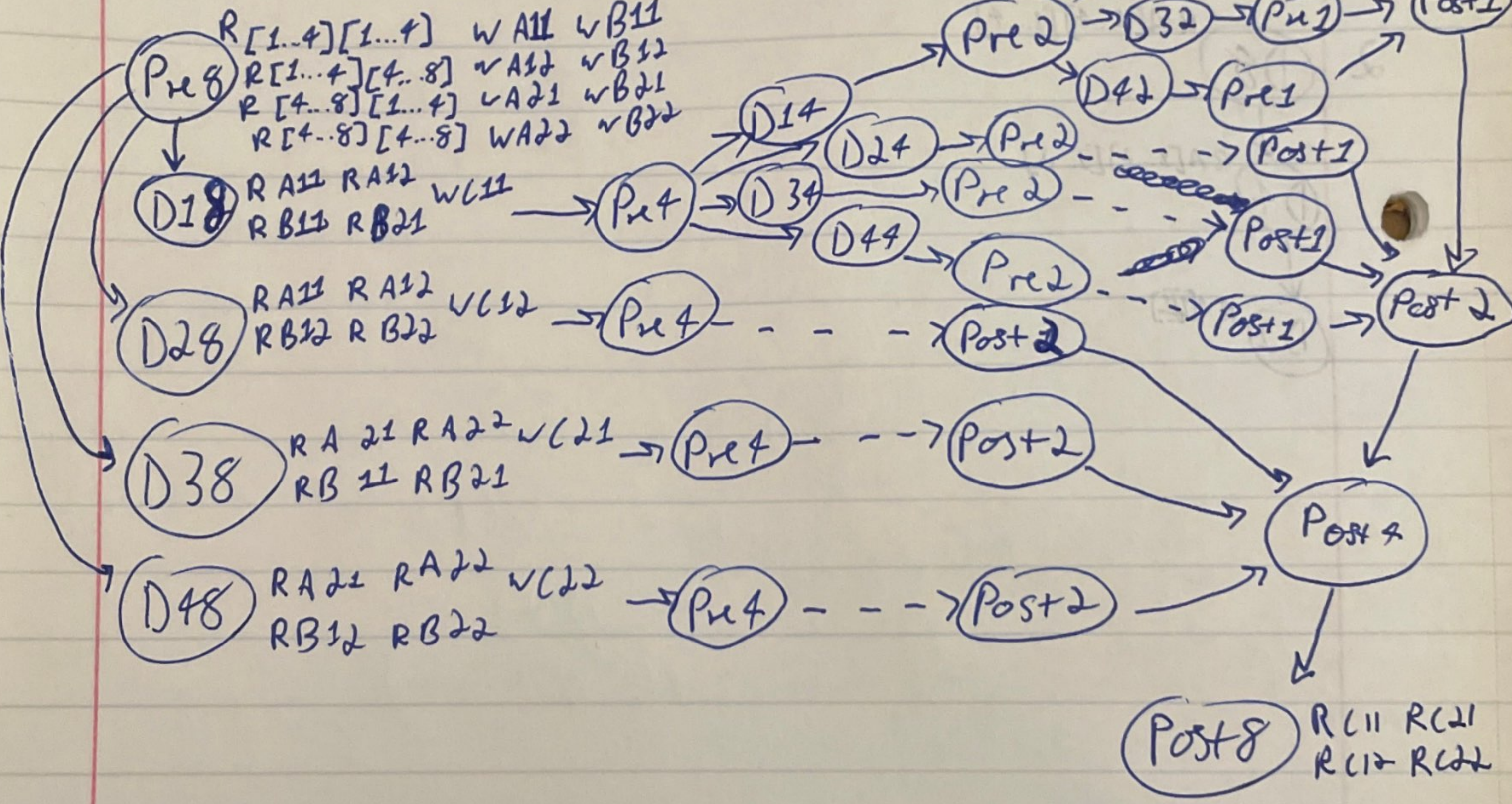
Question: What is the work? log(n/2)^4

Question: What is the critical path? What is its length? log(n/2)^4

2 ~~Dense~~ Dense Matrix Multiplication

~~Pre 8 RA[1...4][1...4] RB[1...4][1...4] n=8~~
~~RA[1...4][4...8] RB[1...4][4...8]~~
~~RA[4...8][1...4] RB[4...8][1...4]~~
~~RA[4...8][4...8] RB[4...8][4...8]~~

~~D14 RA[1...2][1...2]~~
~~RA[1...2][2...4] + B versions n=4~~
~~RA[2...4][1...4]~~
~~RA[2...4][2...4]~~



3 Merge Sort

Question: Recall the merge sort algorithm. (Give the algorithm.)
Question: What is the complexity of this function?
Question: Extract the dependencies.

(Hint: instead of using loop iterations as a task, you can use function calls and function return as tasks. Think that merge sort is recursive! Remember that when working with functions, a name in two different function can represent different underlying variable/memory location.)
Question: Do all tasks have the same processing time?
Question: What is the width?
Question: What is the work?
Question: What is the critical path? What is its length?
Question: How does the schedule of such an algorithm look like when $P = 4$? (What I mean is that what ever the values of n, the schedules have "shapes". What "shape" does any schedule for this problem have? The sketch of what a Gantt chart would look like answer the question.)

Split the Array in half, sort each had independently and merge back together as a post-process
O(nlogn)

1
nlog(n)
nlog(n)

extracting 3 Cont

3 Merge Sort

$$\begin{tikzcd}
 & & & & \text{M16} \arrow[dl] \arrow[drr] & & & \\
 & & & \text{M8L} \arrow[dl] \arrow[dr] & & & \text{M8R} \arrow[ddd, dashed] & \\
 & & \text{M4L} \arrow[dl] \arrow[dr] & & \text{M4R} \arrow[dd, dashed] & & & \\
 & \text{M2L} \arrow[dl] \arrow[d] & & \text{M2R} \arrow[dd, dashed] & & & & \\
\text{M1L} \arrow[dr] & \text{M1R} \arrow[d] & & & \text{Post2} \arrow[dd] & & \text{Post4} \arrow[d] & \\
 & \text{Post1} \arrow[dr] & & \text{Post1} \arrow[dl] & & & \text{Post8} \arrow[r] & \text{Post16} \\
 & & \text{Post2} \arrow[rr] & & \text{Post4} \arrow[urr] & & &
\end{tikzcd}$$