

12. Explain how to modify the all-pairs shortest path algorithm for a CREW PRAM that was given in class so that it runs in time  $O(\log^2 n)$  on a EREW PRAM with  $n^3$  processors.

We need to first create  $n$  copies of the input using the algorithm described in 3a. This requires  $n^3$  to do this copying process in  $\log n$  time. The processor requirement is  $n^3$  since we need to iterate over all pairs of nodes ( $n^2$  processors) and then perform the actual copy of the edge from the first to the second node using  $n$  processors.

```
Repeat log n times
  ParFor i = 1 to n do
    ParFor j = 1 to n do
      ParFor m = 1 to n do
        each processor has it's own copy of D to work with (created above)
        T[i,m,j] = min{D[i,j], D[i,m]+D[m,j]}
        D[i,j]   = min{T[1,1,j] ... T[1,n,j]}
```

13. Explain how to modify the all-pairs shortest path algorithm for a CREW PRAM that was given in class so that it actually returns the shortest paths (not just their lengths) in time  $O(\log^2 n)$  on a EREW PRAM with  $n^3$  processors.

We need to first create  $n$  copies of the input using the algorithm described in 3a. This requires  $n^3$  to do this copying process in  $\log n$  time. The processor requirement is  $n^3$  since we need to iterate over all pairs of nodes ( $n^2$  processors) and then perform the actual copy of the edge from the first to the second node using  $n$  processors.

Repeat  $\log n$  times

```

  ParFor i = 1 to n do
    ParFor j = 1 to n do
      ParFor m = 1 to n do
        each processor has it's own copy of D to work with (created above)
        T[i,m,j] = min{D[i,j], D[i,m]+D[m,j]}
        D[i,j]   = min{T[1,1,j] ... T[1,n,j]}
        M[i,j]   = index of min{T[1,1,j] ... T[1,n,j]}
        This new table stores the best intermediate vertex between i and j

```

We then create  $n$  copies of both  $D$  and  $M$  using  $n^3$  using the algorithm in 3a.

```

  ParFor i = 1 to n do
    ParFor j = 1 to n do
      We have n processors remaining to find the shortest path between i and j.
      From this point we perform a binary search where one processor in  $O(1)$  time
      find the shortest path from i to j which may route through  $m = M[i,j]$ .

```

Then in  $O(1)$  time two processors will concurrently find the shortest paths from i to m and m to j and so on and so on. This search is  $O(\log n)$  time.

When the processor returns the results are concatenated by the parent processor.

14. Explain how to solve the longest common subsequence problem in time  $O(\log^2 n)$  using at most a polynomial number of processors on a CREW PRAM.

HINT: One way to do this is to reduce the longest common subsequence problem to a shortest path problem. Note that the shortest path algorithm works for any graph for which there are not cycles whose aggregate weight is negative.

16. Design a parallel algorithms that merges two sorted arrays into one sorted array in time  $O(1)$  using a polynomial number of processors on a CRCW PRAM.