Oliver has said that many of the algorithms we have chosen are too simple to have effective parallelism out of. Because of this, we should find something thats a little more complex, yet more easily parallelizable.

We should just pick two and then go from there, because we do not have that much time!!!

https://towardsdatascience.com/data-scientists-the-five-graph-algorithms-that-you-should-know-30f454fa5513

https://www.cs.cmu.edu/~scandal/nesl/algorithms.html

Graph traversal
Graph property checking
Graph minimisation

Connected Components
Trees
Shortest paths
Closest pairs

BFS
Seems alright to do, can be parallelized

Prim's Algorithm seems alright to do, although unsure on Parallel gains

- Just need to see how to implement in DPC

Delta Stepping

- A full parallelizable graph traversal algorithm for graph traversal
- Seems complex
- https://en.wikipedia.org/wiki/Parallel_single-source_shortest_path_algorithm#Delta_stepping_algorithm

All pairs shortest path:

Generally these have large time complexities, so i'm thinking of tackling one of these ones.

Floyd-Warshall

- Can be parallelised
- https://www.tutorialspoint.com/all-pairs-shortest-paths
- One that he mentioned
- Already been done :(- But does he know this????

Dijkstra algorithm for All pairs shortest path

- Can be parallelised
- Just dijkstra done multiple times
- TOO EASY!!!!!

Edmonds–Karp algorithm

- Uses BFS
- USed to find Network FLow
- Not sure how to parallise

Kruskal's Algorithm

- Minimal spanning tree
- Cant be parallised
- The Sorting step can be however

Hopcroft-karp
https://en.wikipedia.org/wiki/Hopcroft%E2%80%93Karp_algorithm

- We could parallise the For loops inside the main method
- USes both DFS and BFS
    - Maybe BFS can be parallised???
- Cannot do Parallel,
- REquires recursive call for DFS
- BFS can only be parallelised.

Hungarian Method!!

Strongly Controll

Lets pick:

Hopcroft-Karp
    Basically BFS

Floyd-Warshall
Dijkstra all pairs

BruteForce??