Assignment - HPC4.

- TITLE & PROBLEM STATEMENT: Parallel Search Algorithm Design and implement parallel algorithm utilizing all available resources for:
  i) Binary search for sorted array
  ii) Best first Search (traversal of graph to reach target in shortest possible path).

- OBJECTIVES: To understand the parallel search algorithms, specifically binary & best-first search.

- OUTCOMES: Understood parallel search algorithms and implemented successfully

- SOFTWARE AND HARDWARE REQUIREMENTS: CUDA, NVCC, GCC, 8GB RAM, GPU, 64 bit; 128 GB SSD; Google Colab.

- THEORY:

Binary Search (Sorted Array)

- It is a fast search algorithm with runtime complexity of O(log n). It works on the principle of divide & conquer.
- Binary search looks for a particular item by comparing the middle most item of collection. If element found index is returned
- If the middle item is greater than element then item is searched for in the sub-array to the left of middle item else to right. This continues until sub-array size reduces to 0.
- For the ordered array that is input, and 'x' processors (usually 2), we part our array in x+1 parts. For k≠x processors, split the array into n/k groups and assign a processor to each group, & run binary search.
- The time complexity is thus O(log n/k).

- Best First Search:

It is an algorithm that traverses a graph to reach a target in shortest possible path. Unlike BFS, DFS, Best first search follows an evaluation function to determine which node is the most appropriate to traverse next.

- Steps of Best First Search:
  - start with root node, mark it visited.
  - Find the next appropriate node, mark it visited
  - go to next level and find the appropriate node & mark visited.
  - Continue this process until target is reached.

- In the parallel formulations of BFS, different processors concurrently expand the nodes in the open list. However, in this case, sequential termination criterion fails, & the open list access issues severely limit performances.

- For BFS, a priority queue is the core data structure. Each processor locks the queue, extracts the best node, then unlocks it. Successors of this node are generated; heuristic functions estimated & inserted into open list (queue). Termination is signaled when a solution is found that has better cost than the best heuristic value in the open list.

- CONCLUSION:

Successfully implemented parallel Binary Search & Best First Search.