Date:

Assignment No-A4

Title:-

Parallel Search Algorithm.

Problem Statement:-

Design and implement parallel algorithm utilizing all available resources for:
i) Binary search for sorted array.
ii) Best first search (traversal of graph to react target in shortest possible path).

Objectives:-

To understand the parallel search algorithm specifically binary & best first search.

Outcomes:-

Understood parallel search algorithms and implemented them successfully.

Software & Hardware Requirements:-

CUDA, nvcc, gcc, 8 GB RAM, GPU, 64 bit ; 128 GB SSD ; Google Colaboratory.

Theory:-

BINARY SEARCH (SORTED ARRAY).

- Binary search is a fast search algorithm with a runtime complexity of O(log n).

- It works on the principle of divide & conquer. It also requires a sorted array.

- Binary search looks for a particular item by comparing the middle most item of collection. If a match occurs then the index of item is returned.

- If the middle item is greater than the item then the item is searched for in the subarray to the left of the middle item.

- Otherwise the item is searched for in the subarray to the left of the middle of item. This continues until the size of the subarray reduces to 0.

- For the ordered array that is the input and 'x' processors (usually 2), we part our array in x+1 parts.

- For x < n processors, split the array into n/k groups and assign a processor to each group and run binary search on that group.

- The time complexity is thus $O(\log n/k)$.

BEST FIRST TRAVEL.

Best First Search is an algorithm that traverses a graph to reach a target in the 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.
  1) Start with the root node, mark it visited
  2) Find the next appropriate node, mark it visited
  3) Go the next level and find the appropriate node and mark it visited.
  4) Continue this process until the target is reached.

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

• For Best-first search, a priority queue is the core data structure. Each processor locks in the queue, extracts the best node, then unlocks it. Successors of this node are generated; their heuristic functions estimated; and inserted into the 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 and Best-First search.