Execution Date: 20/11/2020
Submission Date: 26/11/2020

Assignment A-4

Title: Parallel implementation of Search algorithms.

Problem Statement: Design and implement parallel algorithms for:
1) Binary search for sorted array
2) Best first search ~~that travels~~ for traversal of graph to reach a target in the shortest possible path.

objectives: Understand parallel implementation of search algorithms.
- Implement parallel algorithms for binary search and best first search

Outcomes: To understand and implement parallel algorithms for searching

Software and Hardware Requirements: Google Colab

Theory:

Parallel Best First Search:

- In sequential BFS, the most promising node based on f-value is expanded.
- In parallel BFS multiple processors concurrently expand different nodes from the open list.
- Depending on data structure used for open list, multiple strategies exist for parallel BFS.
- Simplest strategy is to assign each processor the then current best node from the "global" open list. which is shared

by all processors. This is called "centralized" strategy.

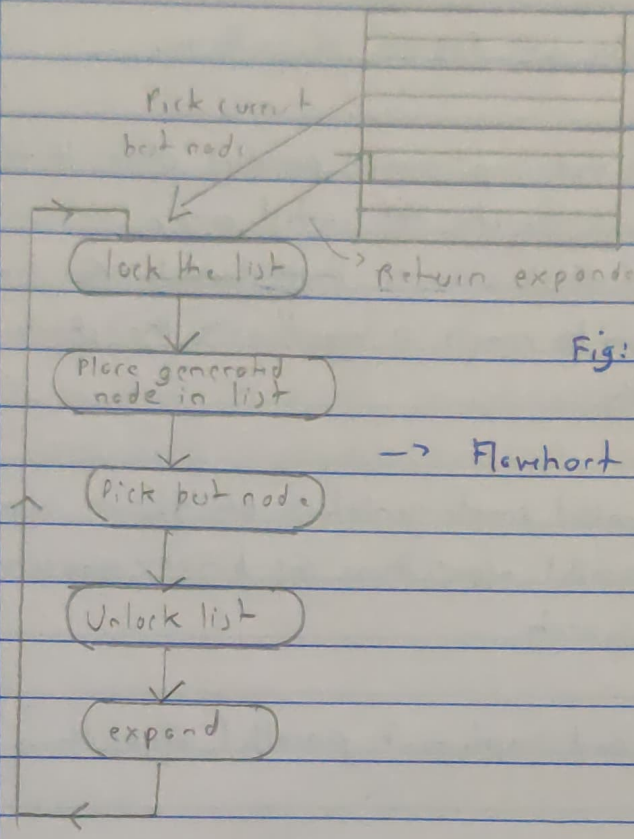


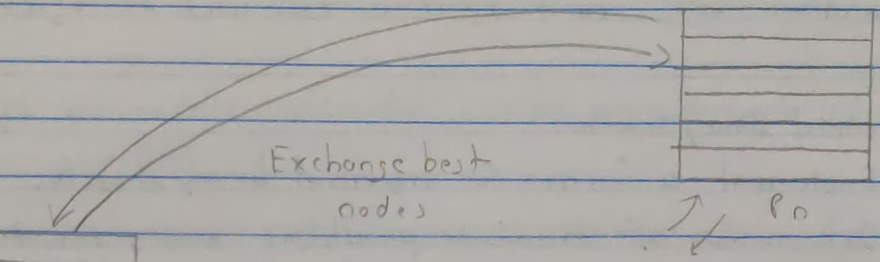
Fig: Schematic Representation

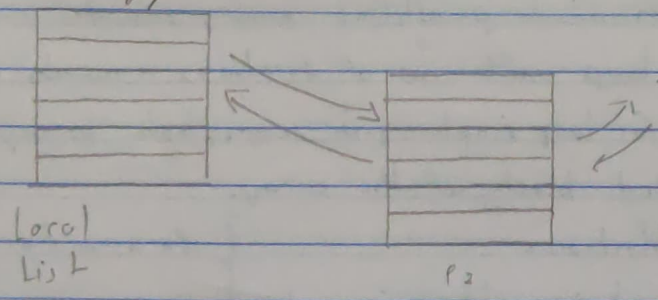
→ Flowchart of processors.

- To avoid contention due to centralized open list. Each processor should maintain its local open list.
- Multiple processors involved should communicate to avoid high search overhead factor and poor speed-up.
- Nodes are expanded and corresponding state space is explored by a processor. If communication is absent then unnecessary exploration may take place, slowing the parallel implementation.

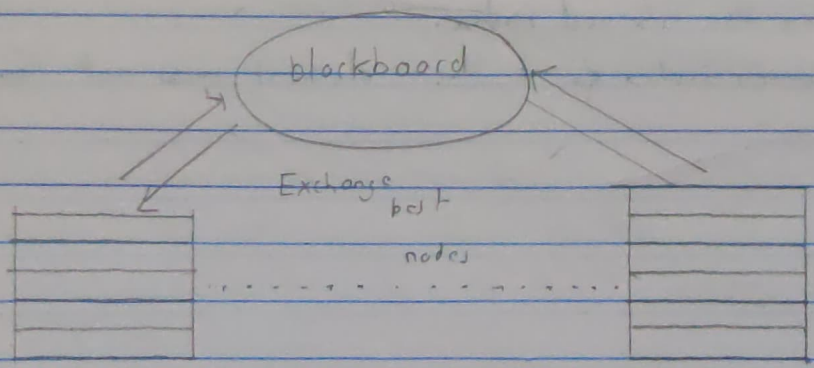
Communication Strategies
for parallel BFS (tree)

Random	Ring	Blackboard
processors randomly message few processors ensures that others get a good part of search space if shared by processor	processors are mapped to a virtual ring and neighbours periodically exchange best open list	a shared blackboard is maintained which enables switching of nodes across processors based on f-value.





Ring communication



blackboard communication

→ Parallel Binary Search:

- In parallel binary search the array can be divided into multiple subarrays and one thread can be allocated for each of the subdivided arrays.

Algorithm:

→ Parallel BFS:

- start with the root node ~~and~~
- Mark the root node as visited
- Set ~~num~~ threads to parallelize implementation of BFS.
- Implement a parallel for loop which shows the visited array
- For each thread check if the current node has been visited.
  If not mark it as visited.
- Continue until search space is exhausted or target is reached.

→ Parallel Binary Search

- Implement a function for sequential binary search.
- Set number of threads to parallelize binary search.
- Implement a for loop with no of iterations = number of threads.
- For Inside the for loop implement sequential binary search on each sub divided block of the array.
  start = i * blocksize
  end = i * blocksize + blocksize - 1
  } Params for binary search.
- Display the returned index.
- If ~~element~~ index is -1 then element was not found.

Test Cases:

	Input	Output	Expected Output	Remarks
①	Binary Search → Array to be searched: 0 – 100000000 Value to be searched → 67557522	Number of threads = 4 Indexes for each subblock = 67557522, -1, -1, -1	Number of threads = 4 Indexes for each subblock = 67557522, -1, -1, -1	Pass
	Matrix → 0 1 1 1 1 0 0 1 1 0 1 0 1 1 0 0 0 0 0 1 0 1 1 0 0	Start node: 2 Path: 2, 3, 0, 4, 1	Start node: 2 Path: 2, 3, 0, 4, 1	Pass

Conclusion: Successfully implemented parallel algorithms for Binary Search and BFS