Lecture 6

Parallel Algorithms

Institute of Computer Science & Information Technology,
Faculty of Management & Computer Sciences,
The University of Agriculture, Peshawar, Pakistan.

Parallel Algorithms

- Parallel Algorithm, is an algorithm which can do multiple operations in a given unit time.

- Parallel algorithms executed concurrently most of the time.

- Which aspect of an algorithm is parallel and which is concurrent, not being clearly distinguished as both are correlated.

- Algorithms vary significantly in how parallelizable they are, ranging from easily parallelizable to completely un-parallelizable.

Parallel Algorithms (2)

- Some problems are easy to divide up into pieces so those could be solved as parallel problems.
- Some problems cannot be split up into parallel portions, as they require the results from a preceding step to effectively carry on with the next step, these are called inherently serial problems.
- Parallel algorithms on Personal Computers have become more common since the early 2000s because of extensive improvements in multiprocessing systems and the rise of multi-core processors.

Structure

- To apply any algorithm properly, it is very important that you select a proper data structure.

- It is because a particular operation performed on a data structure may take more time as compared to the same operation performed on another data structure.

Structure (1)

- Therefore, the selection of a data structure must be done considering the architecture and the type of operations to be performed.

- The following data structures are commonly used in parallel programming:

1. Linked List
2. Arrays
3. Hypercube Network

Structure (2)

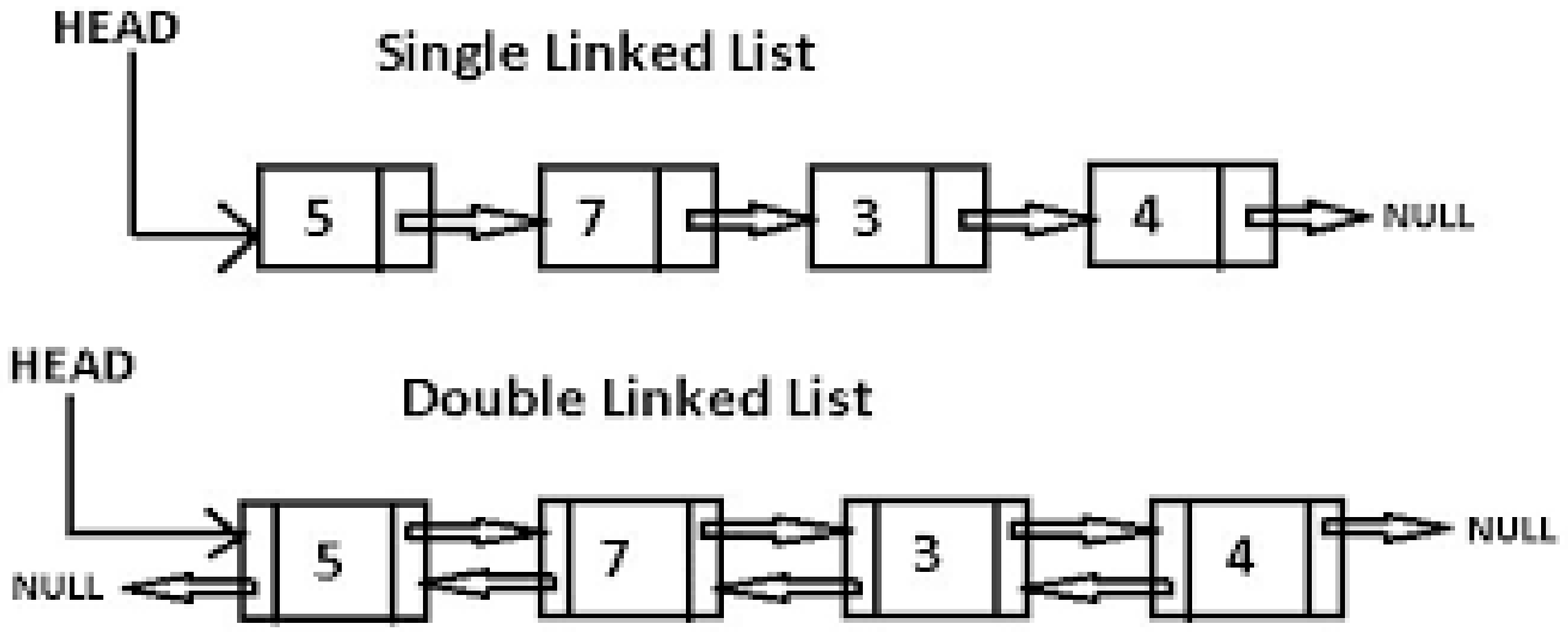
- A linked list is a data structure having zero or more nodes connected by pointers.
- Nodes may or may not occupy consecutive memory locations.
- Each node has two or three parts – one data part that stores the data and the other two are link fields that store the address of the previous or next node.

Linked Lists

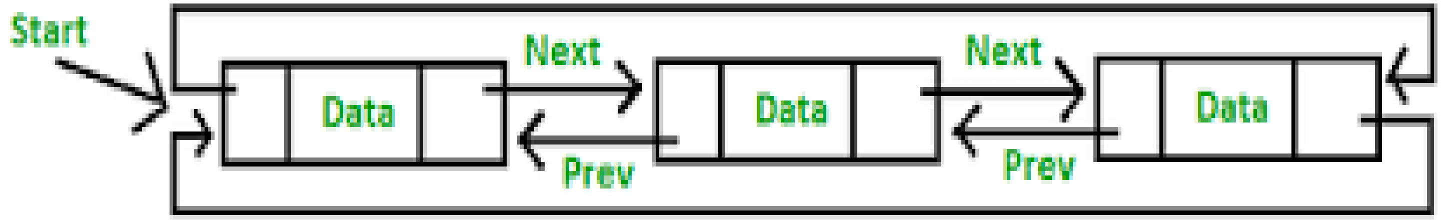
- The first node's address is stored in an external pointer called head. The last node, known as tail, generally does not contain any address. There are three types of linked lists:

1. Singly Linked List
2. Doubly Linked List
3. Circular Linked List

Linked Lists (I)



Circular Linked List



Arrays

- An array is a data structure where we can store similar types of data.
- It can be one-dimensional or multi-dimensional.
- Arrays can be created:

1. Statically
2. Dynamically

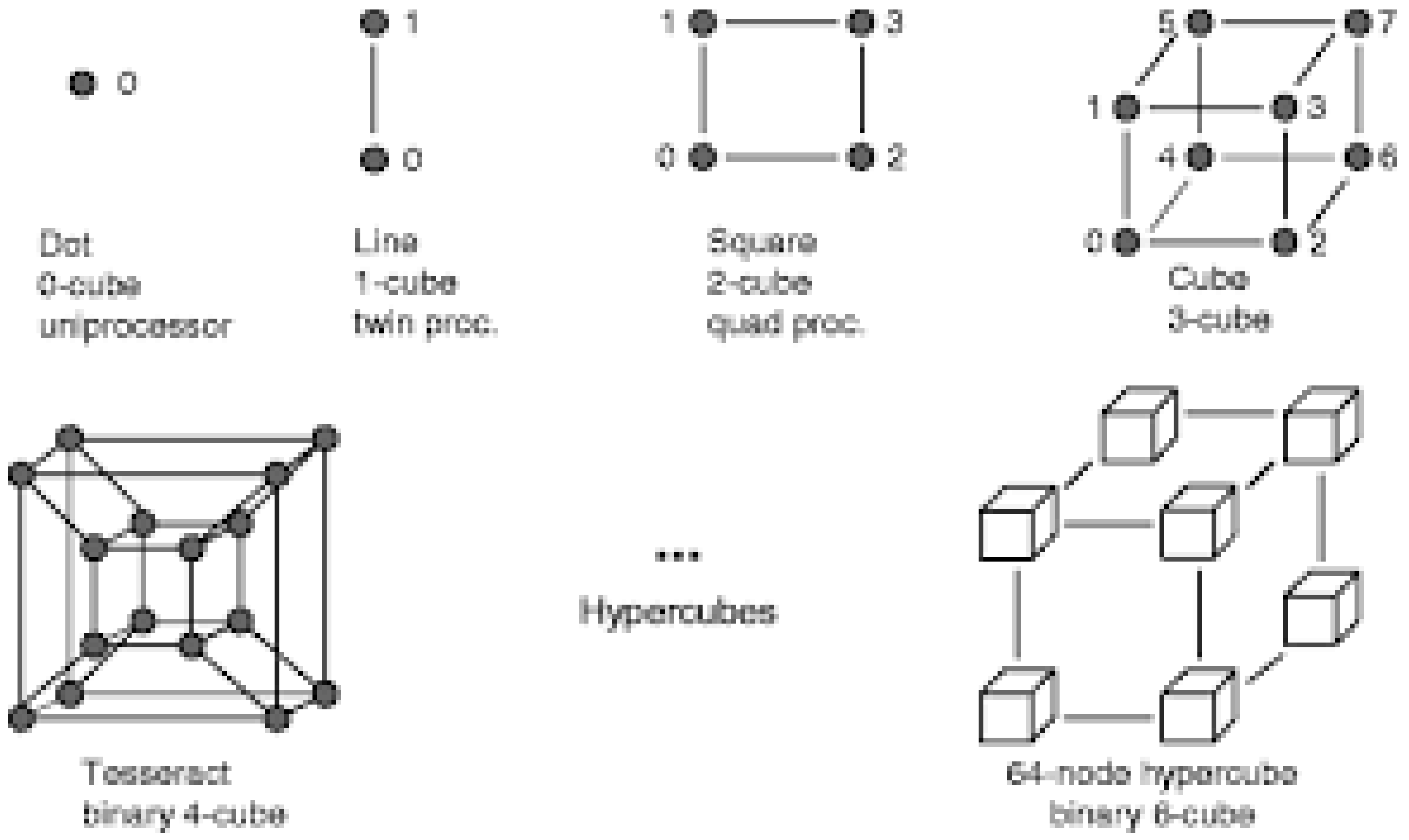
Arrays (1)

- In statically declared arrays, dimension and size of the arrays are known at the time of compilation.
- In dynamically declared arrays, dimension and size of the array are known at runtime.
- For shared memory programming, arrays can be used as a common memory and for data parallel programming, they can be used by partitioning into sub-arrays.

Hypercube Network

- Hypercube architecture is helpful for those parallel algorithms where each task has to communicate with other tasks.
- Hypercube topology can easily embed other topologies such as ring and mesh.
- It is also known as n-cubes, where n is the number of dimensions.

Hypercube Network (1)



Design Technique for Parallel Algorithms

- Selecting a proper designing technique for a parallel algorithm is the most difficult and important task.
- Most of the parallel programming problems may have more than one solution.

Design Technique (1)

- The following designing techniques can be used for parallel algorithms:

1. Divide and Conquer
2. Greedy Method
3. Dynamic Programming
4. Backtracking
5. Branch and Bound

Divide and Conquer

- In the divide and conquer approach, the problem is divided into several small sub-problems.

- Then the sub-problems are solved recursively and combined to get the solution of the original problem.

- The divide and conquer approach involves the following steps at each level:

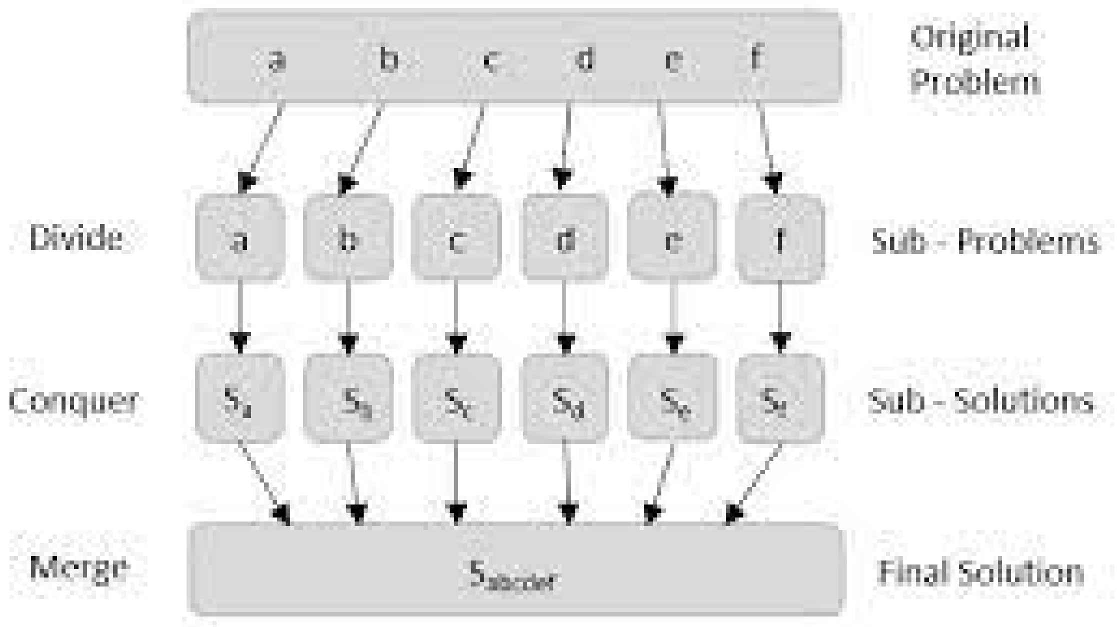
Divide and Conquer (1)

- **Divide:** The original problem is divided into sub-problems.

- **Conquer:** The sub-problems are solved recursively.

- **Combine:** The solutions of the sub-problems are combined together to get the solution of the original problem.

Divide and Conquer (2)



Divide and Conquer (3)

- The divide and conquer approach is applied in the following algorithms:

1. Binary Search
2. Quick Sort
3. Merge Sort
4. Matrix Multiplication

Greedy Algorithm

- In greedy algorithm approach, solutions are made from the given solution domain.
- As being greedy, the closest solution that seems to provide an optimum solution is chosen.
- Greedy algorithms try to find a localized optimum solution, which may eventually lead to globally optimized solutions.
- However, generally greedy algorithms do not provide globally optimized solutions.

Greedy Algorithm (1)

- The problem is to count to a desired value by choosing the least possible coins and the greedy approach forces the algorithm to pick the largest possible coin.

- If we are provided coins of 1,2, 5 and 10 and we are asked to count 18 then the greedy procedure will be:

Greedy Algorithm (2)

1. Select one 10 coin, the remaining count is 8.
2. Then select on 5 coin, the remaining count is 3.
3. Then select one 2 coin, the remaining count is 1.
4. And finally, the selection of one 1 coins solves the problem.

Greedy Algorithm (3)

- If we slightly change the problem then the same approach may not be able to produce the same optimum result.

- For the currency system, where we have coins of 1, 7, 10 value, counting coins for value 18 will be absolutely optimum but for count like 15, it may use coins than necessary.

Greedy Algorithm (4)

- For example, the greedy approach will use 10+1+1+1+1+1, total 6 coins.
- Where as the same problem could be solved by using only 3 coins (7+7+1)
- Hence, we may conclude that the greedy approach picks an immediate optimized solution and may fail where global optimization is a major concern.

Dynamic Programming

- Dynamic programming approach is similar to divide and conquer in breaking down the problem into smaller and yet smaller possible sub-problems.
- But unlike, divide and conquer, these sub-problems are not solved independently.
- Rather, results of these smaller sub-problems are remembered and used for similar or overlapping sub-problems.

Dynamic Programming (1)

- Dynamic programming is used where we have problems, which can be divided into similar sub-problems, so that their results can be re-used.
- Mostly, these algorithms are used for optimization.
- Before solving the in-hand sub-problem, dynamic algorithm will try to examine the results of the previously solved sub-problems.
- The solutions of sub-problems are combined in order to achieve the best solution.

Backtracking

- Backtracking is an optimization technique to solve combinational problems.

- In backtracking, we start with a possible solution, which satisfies all the required conditions.

- Then we move to the next level and if that level does not produce a satisfactory solution, we return one level back and start with a new option.

Branch and Bound

- A branch and bound algorithm is an optimization technique to get an optimal solution to the problem.

- It looks for the best solution for a given problem in the entire space of the solution.

- The bounds in the function to be optimized are merged with the value of the latest best solution.

Branch and Bound (1)

- The bounds in the function to be optimized are merged with the value of the latest best solution.
- It allows the algorithm to find parts of the solution space completely.
- Once a solution is found, it can keep improving the solution.
- Branch and bound search is implemented in depth-bound search and depth-first search.