Breadth First Search(BFS)

Given a graph G=(V, E) where V is set of verices and E is the set of edges and a source vertex 's' BFS explores all the vertices in that graph reachable from s. A path from vertex s to t contains a sequence of edges (u1, v1), ...,(ui, vi) where u, v ϵ V. The length of the path is the number of edges along the path. BFS implies that all the vertices of level k (vertices has path length k from s) should be visited before vertices of level k+1. The output of the algorithm is the level of all the vertices reachable from s.

Serial BFS Algorithm

In serial BFS algorithm starting with the start vertex we explore its neighbours and update their level as cur_level+1.

Remove all the elements from current frontier and add the next set in frontier. We repeat the same step until frontier set is empty.

BFS Algorithm

```
//Initialize level of the vertices by INF
    for_each( srcVec.begin(), srcVec.end(),
    Initialize()); cur_level = 0;
/  Level of start vertex
    is 0 Level[ src] =
    cur_level;
    FS.push_back( src)
    ;

   //repeat until FS is
                empty
 while( !FS.empty()){

     //Find neighbours of frontier set
          for_each( FS.begin(), FS.end(), findNeighbour());

     //update level of the vertices in the neighbour set which has level=INF
          for_each( NS.begin(), NS.end(), updateLevel());

//update level by 1
      cur_level++;

    //clear current frontier set
          FS.clear();

   //make newly expolred vertex as frontier set for next iteration
          FS = NNS;

   //clear Next sets
          NS.clear();
          NNS.clear();
    }
```

Parallel BFS Algorithm

In parallel distributed BFS algorithm we are using 1D block partitioning method. If the total number of vertices of the graph is |V| and total number of processes is p then each processor is given |V|/p vertices at the starting of the algorithm to explore.

The graph is loaded into memory and master process co-ordinates the vertices to each processes for the elements it will be incharge for all the passes of the algorithm.

BFS Parallel Algorithm

```
1. Each process intialize the vertices level to INF
      for_each( srcVec.begin(), srcVec.end(), Initialize());
      cur_level = 0;
2. Master process initialize the src level to 0
      if( myRank == 0){
            //set the level of root = 0
            Level[ src] = cur_level;
            FS.push_back( src);
      }

      do{

3.          If frontier set of all processes is empty then break

4.          Explore the neighbour from frontier set
              for_each( FS.begin(), FS.end(), findNeighbour());

5.          Pack the explored vertex for correct processor


6.          Put the data in buffer for sending to corresponding process

7.          MPI_Barrier( MPI_COMM_WORLD);

8.          Pairwise exchange method to do all-to-all communication between processes

9.          MPI_Barrier( MPI_COMM_WORLD);

10.         Put recieved data in NS

11.         Now update level of explored vertices
             for_each( NS.begin(), NS.end(), updateLevel());

12.         Update level by 1
             cur_level++;

13.         Clear current frontier set
             FS.clear();

14.         Make newly expolred vertex as frontier set for next iteration FS =
             NNS;
```

```
15.         Clear Next sets
              NS.clear();
              NNS.clear();
        }while( 1);
```