Rohit Kulkarni 41346

HPC ASSIGNMENT 3

PROBLEM STATEMENT:
For bubble sort & merge sort, based on existing sequential algorithms, design & implement parallel algorithm utilizing all resources available.

PRE-REQUISITS:

1] Multithreading
2] OpenMP basics.

OBJECTIVES:
Students will be able to learn OpenMP & parallel programming.

OUTCOMES:
Students will be able to design parallel sorting algorithm - Bubble ~~so~~ sort & Merge sort.

THEORY:
Sorting is a process of arranging elements in a group in a particular order, ie, ascending order, descending order, etc.

* PARALLEL SORTING:

1] A sequential sorting algorithm may not be efficient enough when we have to sort a huge volume of data.
2] Therefore, parallel algorithms are used in sorting.

Date:

* Parallel Bubble Sort:

i] Implemented as a pipeline.

ii] Let local-size = n/no-proc. We divide the array in no-proc parts and each process executes the bubble sort on its part, including comparing the last element with the first one belonging to next thread

iii] Implement with for loop (instead of $j<i$), do $j<n-1$

iv] For every iteration of i, each thread needs to wait until the previous thread has finished that iteration before starting.

* Algorithm for parallel sort (Bubble)

```
1] For k = 0 to n-2
2] If k is even then
       for i=0 to (n/2)-1 do in parallel
       if A[2i] > A[2i+1] then
           Exchange A[2i] & A[2i+1]
3] Else
       for i=0 to (n/2)-2 do in parallel
         if A[2i+1] > A[2i+2] then
           Exchange A[2i+1] & A[2i+2]

4] Stop after exiting for loop.
```

* Algorithm for parallel sort (Merge)

```
1] Begin.
2] Create processors i=1 to n.
3] if i>0, then receive size & parent from root
4] Receive the list, size, parent from root.
5] Endif.
```

```
6] If both children are present in tree, then
      send midvalue, first child
      send list_sizemid, second child.
      send list midvalue, first child.
      call mergelist.
      store temp in another array list 2.
7] Else
      call parallel Merge sort.
8] if i>0, then
      send list, listsize, parent
9] end if
10] end.
```

CONCLUSION:

Using the OpenMP parallel sorting techniques, task & data parallelism is implemented for sorting algorithms.