

LPI → HPC - 3

* Title : Parallel Sorting Algorithm

* Problem Statement :

For bubble sort and merge sort, based on existing sequential algorithms, design and implement parallel algorithm utilizing all resources available.

* Objectives:

To learn and understand parallel programming using OpenMP.

* Outcomes:

Design parallel sorting algorithm - bubble sort and merge sort.

* Pre-requisites:

- i) Multithreading
- ii) basics of OpenMP

* Theory:

Sorting is a process of arranging elements in a group in a particular order i.e. ascending, descending, alphabetic, etc.

Char. of Sorting are-

- i) Arrange elements of a list in a certain order.
- ii) Make the data easier to access.
- iii) Speed up other operations such as searching

and merging.

Many sorting algorithms exist, with different time and space complexities.

* Parallel sorting:

A sequential sorting algo. may not be efficient enough when we have to sort a huge volume of data. Thus, parallel algo. are used in such cases.

Design methodology:

It is based on an existing sorting algorithm.

Try to utilize all the resources available.

Possible to turn a poor sequential algorithm into a reasonable parallel algorithm.

Bubble sort

The idea of bubble sort is to compare two adjacent elements. If they are not in the right order, swap them.

Do this comparison and keep switching until the end of the array.

Repeat this process from beginning of array.

* Parallel bubble sort:

① Implemented as a pipeline.

② Let local-size = $n / \text{no-proc.}$ Each process executes the bubble sort on its part, including comparing last element with the first element belonging to the next thread.

- ③ Implement for $(j=0; j < n-1; j++)$
- ④ For every iteration of i , each thread needs to wait until the previous thread has finished that iteration before starting.

* Algorithm:

for $k=0$ to $n-2$

if $k \% 2 == 0$

for $i=0$ to $(n/2)-1$ do in parallel

if $A[2i] > A[2i+1]$

swap $(A[2i], A[2i+1])$

else

for $i=0$ to $(n/2)-2$ do in parallel

if $A[2i+1] > A[2i+2]$

swap $(A[2i+1], A[2i+2])$

$k++$

* Merge Sort (Parallel)

Parallelize processing of sub-problems. Max.

parallelization is achieved with one processor per node.

Algorithm:

- ① create processors P_i where $i=1$ to n .
- ② if $i > 0$, then receive the size and parent from the root.
- ③ Receive the list, size and parent from the root.
- ④ $Midval = list_size / 2$

- ⑤ If both the children are present
- Send mid value, first child
 - Send $(\text{list_size} - \text{mid})$, second child
 - Send list, mid value, first child
- send list from mid value, listsize - mid value, second child
- call merge(list, 0, midval, midval + 1, listsize, temp, 0, listsize)
- store temp in another array list2.
- ⑥ else
- call parallelMergeSort(list, 0, listsize) \rightarrow current algo.
- ⑦ if $i > 0$
- send list, listsize, parent

End

* Conclusion:

From this assignment, I was able to understand the basics of parallel programming, sorting algorithms and hence implement this assignment.