Abstract

The aim of this paper is to show that the great part of the execution time is consumed in computations. So as the number of processors increase, the amount of work done by each processor will be decrease regardless the effect of the number of physical cores used. Still the time taken to solve the computations dominates over the communication time as by increasing number of processors; tasks are more divided so overall time decreases. The total overhead generated from process initializations and inter-process communication negatively affects the execution time. Using MPI, parallelization on five sorting techniques which are selection sort, bubble sort, quick sort, insertion sort and shell sort have been implemented.

References

- Narayan Desai, Andrew Lusk, Rick Bradshaw, Ewing Lusk, “MPISH: A Parallel Shell for MPI Programs”, 19th IEEE International Parallel and Distributed Processing Symposium (IPDPS’05), pp. 1530-2075
- Fangfa Fu, Siyue Sun, Xin Hu, Junjie Song, Jinxiang Wang and Minyan Yu, "MMPI: A Flexible and Efficient Multiprocessor Message Passing Interface for NoC-Based MPSoC", IEEE, 2010, pp. 359-362
- Adeel Abbas, Affan Ahmad, "Object Oriented Parallel Programming", IEEE, 2002, pp. 89-93
- Sequential and parallel sorting algorithms http://www.iti.fh-flensburg.de/lang/algorithmen/sortieren/algoen.htm
- LINUX MAGAZINE-MPI in Thirty Minutes http://www.linux-mag.com/id/5759/
- Message Passing Interface (MPI) Author: Blaise Barney, Lawrence Livermore National Laboratory

**Index Terms**

Computer Science  
Communications

**Keywords**

MPI  Parallel Programming  Selection sort  Bubble sort  Quick sort  Insertion Sort

Shell Sort

Bucket sort

Sequential Programming.