An Evaluation of Parallel Sorting Algorithms Using IMAN1 Supercomputer

PhD Students: Maha Saadeh and Huda Saadeh
Supervisor: Prof. Mohammed Qatawneh
The University of Jordan, Computer Science Department, King Abdullah II School for Information Technology

Abstract: Efficient sorting implementation is important for optimizing the use of other algorithms such as searching, load balancing, etc. In this work, parallel Quicksort, parallel Merge sort, and parallel Merge-Quicksort algorithms are evaluated and compared in terms of running time, speedup, and efficiency . These algorithms divide the data and distribute it among processors in the same way, see Figure 1. The three sorting algorithms are implemented using Message Passing Interface (MPI) library and the experiments are conducted using IMAN1-Zaina cluster.

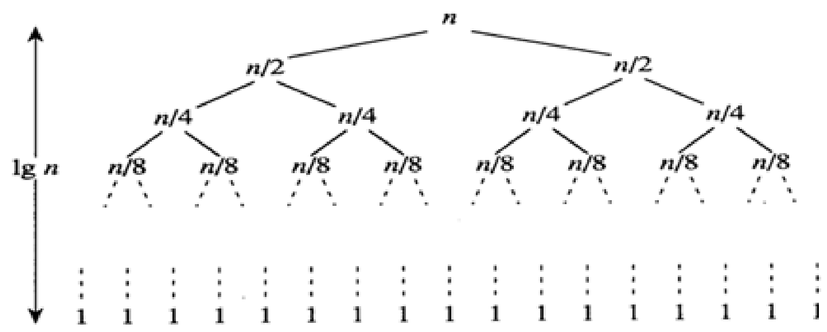


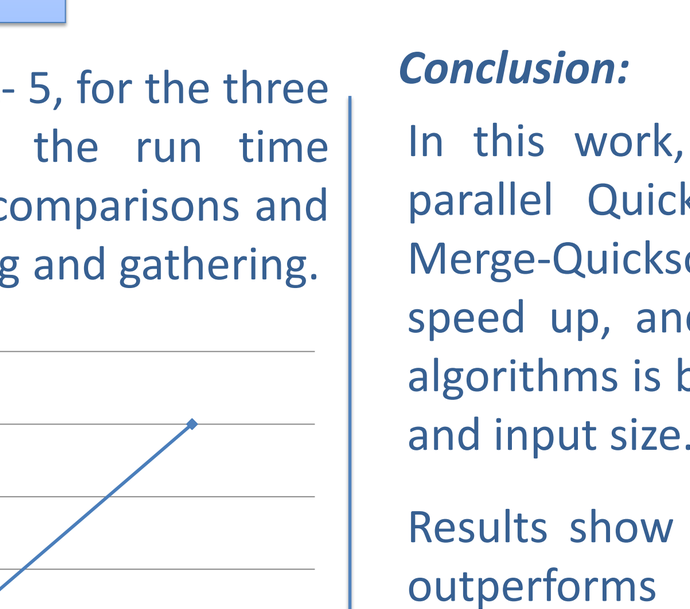
Figure 1: Data Splitting Among Processors

Results show that parallel Quicksort outperforms the other algorithms especially for large data size .

Evaluation Results: Table I shows the experimental parameters that are used.

Table I: Experimental Parameters.

Hardware specification	Dual Quad Core Intel Xeon CPU with SMP, 16 GB RAM
Software Specification	Scientific Linux 6.4 with open MPI 1.5.4, C and C++ compiler.
Data Size	2^{22}, 2^{23}, 2^{24}, 2^{25}, 2^{26}
Number of Processors	8, 16, 32, 64, 128



Run Time Results: As illustrated in Figures 2- 5, for the three algorithms, as the data size increases, the run time increases due to the increased number of comparisons and the increased time required for data splitting and gathering.

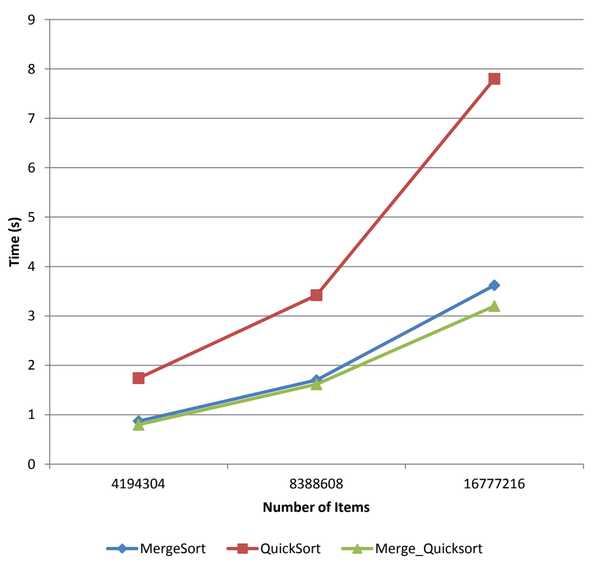


Figure 2: Comparison of the Three Algorithms According to Small Data Size

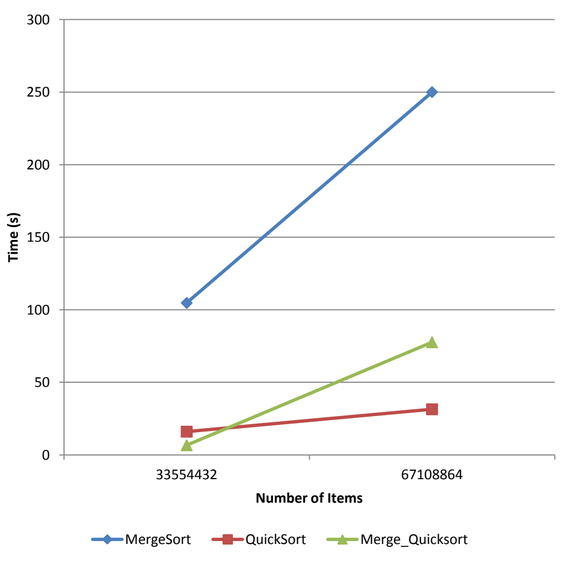


Figure 3: Comparison of the Three Algorithms According to Large Data Size

Regarding the performance of the three algorithms according to different number of processors, for large data size Quicksort outperforms both Merge sort and Merge-Quicksort algorithms.

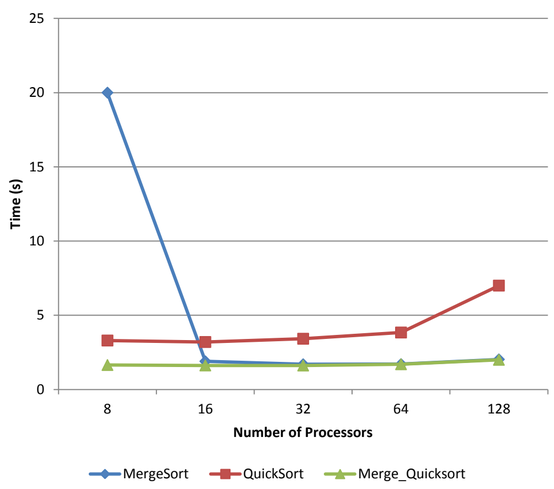


Figure 4: Comparison of the Three Algorithms According to the Number of Processors with 2^{23} Data Size.

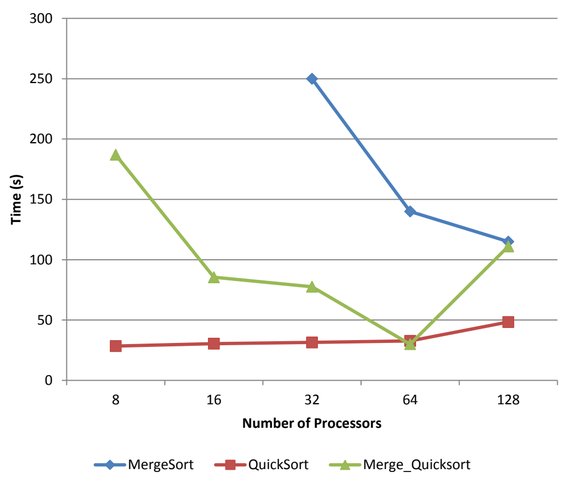


Figure 5: Comparison of the Three Algorithms According to the Number of Processors with 2^{26} Data Size.

Efficiency Results: Table II shows the results of parallel efficiency achieved by the three algorithms.

Table II: Parallel Efficiency Results.

Data Size	Merge Sort	Merge-Quicksort	Quicksort
2^{22}	43%	41.41%	278%
2^{23}	26.08%	20.78%	119%
2^{24}	51.32%	41.44%	52.45%
2^{25}	failure	40.48%	56.75%
2^{26}	failure	3.58%	62.91%

Conclusion:

In this work, we present performance evaluation of parallel Quicksort, parallel Merge sort, and parallel Merge-Quicksort algorithms in terms of running time, speed up, and efficiency. The evaluation of the three algorithms is based on the varying number of processors and input size.

Results show that for large data size parallel Quicksort outperforms the other algorithms using fixed and different number of processors. Moreover, Quicksort achieves the best speedup of up to 22 times comparing with the sequential algorithm, while Merge and Merge-Quicksort achieve up to 16.98 and 3.52 speedups, respectively.

[1] Basel A. Mahafzah, "Performance assessment of multithreaded Quicksort algorithm on simultaneous multithreaded architecture", J Supercomput, Vol. 66, pp.339–363, 2013.
[2] http://en.wikipedia.org/wiki/Sorting_algorithm. Accessed on 16/9/2013.
[3] Concurrent Programming, Addison wisely http://www.nondot.org/sabre/Mirrored/AdvProgLangDesign/finkel07.pdf
[4] Ananth grama, Anshul gupta, George kerypis, Vipin kumar, "Introduction to Parallel Computing", Second Edition, Addison wesley, 2003.
[5] http://www.iman1.jo/iman1/. Accessed on 7/10/2014.
[6] Minsoo Jeon and Dongseung Kim, "Load-Balanced Parallel Merge Sort on Distributed Memory Parallel Computers", Proceedings of the IEEE International Parallel and Distributed Processing Symposium, IPDPS.02, 2002.
[7] Minsoo Jeon and Dongseung Kim1, "Parallel Merge Sort with Load Balancing", International Journal of Parallel Programming, Vol.31, No.1, pp 21-33, 2003.
[8] Ishwari Singh Rajput, Bhawnesh Kumar, Tinku Singh, "Performance Comparison of Sequential Quick Sort and Parallel Quick Sort Algorithms", International Journal of Computer Applications, Vol.57, No.9, pp.14-22, 2012.
[9] Charles U. Martel and Dan Gusfield "A fast parallel Quicksort algorithm ", Information Processing Letters, Vol.30, No.2, pp.97–102, 1989.
[10] Philippas Tsigas and Yi Zhang, "A Simple, Fast Parallel Implementation of Quicksort and its Performance Evaluation on SUN Enterprise 10000", 16th Euromicro Conference on Parallel, Distributed and Network-Based Processing, PDP, 2003.
[11] Vivek Kale and Edgar Solomonik, "Parallel Sorting Pattern", the 2nd Annual Conference on Parallel Programming Patterns (ParaPLoP), March 30 - 31st, 2010, ACM 978-1-4503-0127-5.
[12] Davide Pasetto and Albert Akhriev, "A Comparative Study of Parallel Sort Algorithms", IBM Dublin Research Lab, Mulhuddart, Dublin 15, Ireland.
[13] Luis Quiles, " Quick Sort", Florida Institute of Technology.