Abstract

Sorting algorithms, their implementations and their applications in modern computing necessitates improvements for sorting large data sets quickly and efficiently. This paper will analyze the performance of a multi-threaded quick sort implemented using the thread pool pattern. The analysis will be done by comparing the time required to sort various data sets and their memory constraints, against the native sorting implementations of the Dual Pivot Quicksort and Merge Sort using the Fork-Join framework in the Oracle Java 8 programming language. Analysis is done of the effect of different number of processor (cores) of the test machine, as well as the performance barrier due to the initial time taken to create “p” threads, p being the number of processors. This paper also analyzes the limitations of the inbuilt Java method Arrays.parallelSort() and how the proposed system overcomes this problem. Finally, it also discuss possible improvements to the proposed system to further improve its performance.
Parallel Quick Sort using Thread Pool Pattern

1. “Parallel Programming in C with MPI and OpenMP”. M. J. Quinn, Tata McGraw Hill
Publications, 2003, p. 338
4. “Algorithms In C: Fundamentals, Data Structures, Sorting, Searching, Parts 1-4 (3
ed.)”. Sedgewick, Robet (1 September 1998).
8. “Introduction to Algorithms”. Cambridg. Cormen, Thomas H., Charles E. Leiserson,
Ronald L. Rivest, and Clifford Stein.
SUN Enterprise 10000.” Tsigas, P., and Yi Zhang. Eleventh Euromicro Conference on Parallel,
& Sharapov, Ilya. Prentice-Hall 2002,

Index Terms

- Computer Science
- Parallel Computing

Keywords

- Sorting, Multithreading, Object oriented programming, Parallel algorithms