

Project Proposal

Authors:

Joshua Hernandez [umhern23@myumanitoba.ca]

Paymahn Moghadasian [umpaymah@myumanitoba.ca]

Instructor: Dr. S. Durocher

COMP 4420

Advanced Design and Analysis of Algorithms

Due : February 24, 2014

Project Proposal

Sorting is a well studied topic in algorithms, in particular quicksort has been extensively studied. The quicksort is particularly interesting because it is a comparatively simple algorithm which has an average asymptotic runtime equal to that of the mergesort and heapsort without any additional memory requirements; quicksorts can be done in place without additional time or code complexity. A major drawback of the quicksort is the it has a worst case complexity of $O(n \log n)$ unlike the mergesort or heapsort.

Aumuller and Dietzfelbinger published a paper in ICALP 2013 titled "Optimal Partitioning for Dual Pivot Quicksort" where they explored the asymptotic runtime of various quicksort along with their respective partitioning algorithms. Aumuller et al. mathematically demonstrated that dual pivot quicksorts are asymptotically lower bound by $1.8n \log n + o(n \log n)$ time where n is the number of elements to be sorted. They generalized the math to describe any dual pivot quicksort algorithm and verified the asymptotic lower bounds of the classic quicksort and Yaroslavskiy's variation.

We plan to first verify Aumuller's results and secondly to expand the scope of the original paper by mathematically describing all multi-partition variations. We will implement quicksort using various partitioning algorithms, again using varying number of partitions. We hope to determine if varying the number of partitions will yield to a more efficient quicksort algorithm.

The goal of this project is to investigate multi-partitioned quicksort, determine an optimal number of partitions and bring to light any of the challenges that arise from the mathematical analysis of multi-partition quicksorts. We're particularly interested in determining whether having more than two partitions provides any theoretical or experimental benefits.

- Week 1
 - Implement basic quicksort
 - * Pivot Picking
 - first element as pivot
 - median of first, middle, and last
 - Find others
 - * Partitioning algorithm
 - The basic one
 - * Each version has an insertion sort flag on arrays with small size
 - * Test functionality on small arrays
 - Implement 2 partition quicksort
 - * Pivot picking
 - first and last elements

- two middle elements of entire from 5 entries
 - Test with 4 entries as well
 - * Partitioning Algorithm
 - Basic Partitioning
 - make smalls then bigs
 - flag partitioning algorithm
 - * Each version has an insertion sort flag on arrays with small size
 - * Test functionality on small arrays
- Week 2
 - Implement 3 and 4 partition quicksorts
 - * Partition algorithm
 - * Basic Partition
 - * Look for partition algorithms
 - * Each version has an insertion sort flag on arrays with small size
 - * Test functionality on small arrays
- Week 3
 - Run several experiments using all the quicksort algorithms implemented
 - Run several arrays sizes
 - Array 'types'
 - * sorted arrays
 - * reverse sorted arrays
 - * random arrays
 - * partially random arrays
 - Preliminary analysis of data
- Week 4
 - Analyze data
 - Write paper
 - Make Presentation