Report on Project 3, Part 1

Parallelization of sorting algorithms

CS415

March 30, 2017

Evan Su

Version 1.0

Speedup of Sorting Algorithms through Bucket Sort.

Overview:

Most sorting algorithms are based on sequential code. In other words, each instruction is executed in order. The speed of sorting depends greatly on the runtime of the algorithm and clock speed of the computer. To achieve greater speeds, parallelization of algorithms is used. The goal of the experiment is to demonstrate the speedup achieved by parallelization.

Test Methodology:

Two different programs are written to test the sorting time, sequential and parallel. The sequential code will be used as a controlled variable. The code will be tested at various sizes starting at 100 unsorted items. For more detail of the sequential and parallel code, please see below.

Sequential

The sequential code first makes the buckets. Then, the buckets are filled with numbers within a certain range. Then, the bucket is sorted using insertion sort. When all the buckets are sorted, the data is collected in to a single array which can be displayed. The time starts when the sorting starts and ends when the sorting stops.

Parallel

The parallel code has the master send each slave portion of unsorted numbers. Then all the processors put the numbers in small buckets. Then, each of the small buckets placed in large buckets with matching ranges. Each large bucket is given to a processor which sort using insertion sort. When the sort is finished, the slaves send all the data back to the master. The master puts all the data into an array. The time starts when the little buckets are filled and ends when the master receives the last set of sorted numbers.

Data Analysis:

Sequential

The run time of the sequential code exhibits a runtime of O(m*(n/m)^2) as show in graph 1.1. The runtime can be simplified to O(n^2 / m). The n^2 runtime is shown as the list size increases. The 1/m runtime effect is shown as the number of buckets increases.

Parallel

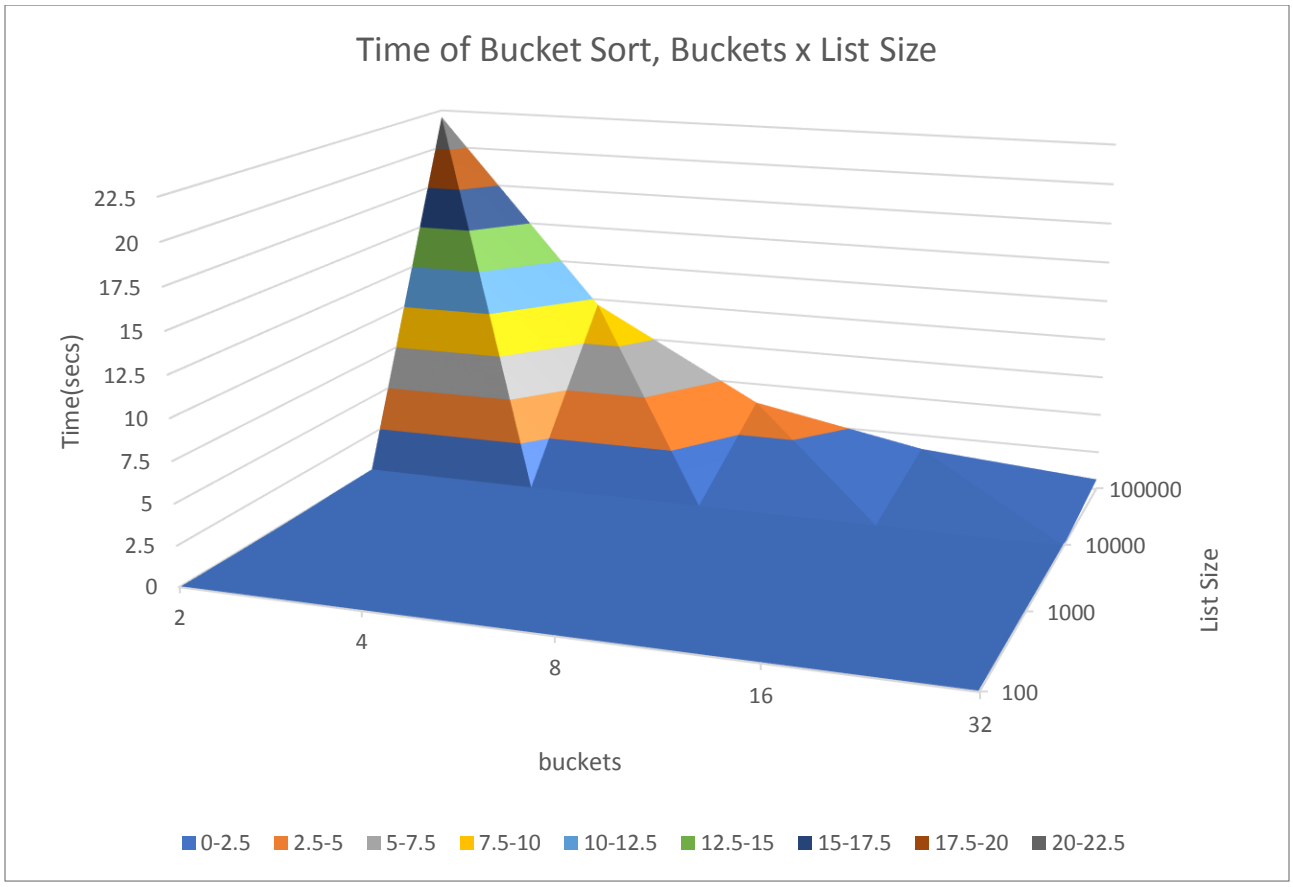
TODO

Conclusion:

TODO

Graphs and Tables

Graph 1.1



The Sequential time of bucket sort. The graph demonstrates the runtime of bucket sort is O(m*(n/m)^2). The time of the graph deceases linearly as the number of buckets used increases. The time of the graph increases exponentially as the list size increases.