CAB301 Assignment 2: Empirical Comparison of Two Algorithms for Finding the Minimum Distance between Two Elements in an Array

Shane Havers N9477756

Andrew Grant N

Date Submitted:

# Summary

The purpose of this report is to summarise the outcomes of the experiments to compare the executions of two algorithms that both find the smallest difference between two elements within a set of numbers. The tests were completed by writing the algorithms and a test driver in C++ within the Code Blocks IDE, and outputting the results to Comma Separated Value files. These outputs were plotted in MATLAB, and the results were

# Description of the Algorithms

The algorithms being compared in this report are algorithms that check the difference – or ‘distance’ – between each element of an input list of numbers, and seek to return the smallest difference. The first algorithm, MinDistance, was proposed by Levitin [1] and can be found in Appendix 1. MinDistance2, which can be found in Appendix 2, serves the same purpose as MinDistance but is proposed as a more efficient version.

The MinDistance algorithm works by first setting the variable to an arbitrarily large value, ideally infinity. Then, it loops over each element in the array with an index variable *,* and then again with an index variable *.* The algorithm then asks: first ensure the element is not being compared to itself (check ), and if not is the difference between the two elements smaller than the current minimum (check ). If this difference is the smallest, set to its value. After checking every permutation of and , the algorithm returns the minimum value and exits.

The MinDistance2 starts the same way, by setting to a large value. Then, it loops over every element of bar the last one (again with index ), and then loops over every value with an index larger than the index (i.e. the second loop goes from index to the end of the array). Then, it stores the difference between the values at indices and in a temporary variable, and if that variable is smaller than the current minimum, change to that value. After both loops have exited, the algorithm returns the minimum value and exits.

# Theoretical Analysis of the Algorithms

## Identifying the Algorithm’s Basic Operation

### MinDistance

### MinDistance2

## Choice of Problem Size

## Analysis of Theoretical Average Case Efficiency

# Methodology, Tools and Techniques

## Programming Environment

## Implementation of Algorithms

## Generating Test Data and Running Experiments

## Implementation of Basic Operations Counters

### MinDistance

### MinDistance2

## Implementation of Execution Timers

### MinDistance

### MinDistance2

# Experimental Results

## Functional Testing

## Number of Basic Operations

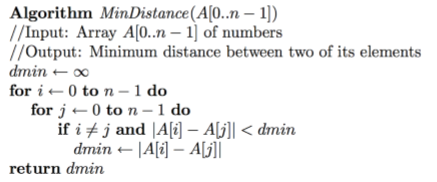
## Execution Time

# References

|  |  |
| --- | --- |
| [1] | A. Levitin, Introduction to the design and analysis of algorithms, 2nd ed., Pearson Addison-Wesley, 2007. |

# Appendix

1. MinDistance Algorithm



1. MinDistance2 Algorithm

