# The Implementation and Comparison of the BCCBT Data Compression Algorithm

Aiden Taylor - B.Sc. in Computer Science
Noah Pinel - B.Sc. in Computer Science
Ty Irving - B.Sc. in Computer Science

***Abstract*—Elaborate upon each major aspect of the paper.**

## I. Introduction

Provides the context and motivates the topic. It introduces the paper/research that is the basis of the project. [**?**]

## II. Proposed Work

Describes the proposed work and evaluation framework (e.g. experiments, data analysis, statistical tests, mathematical proof..).

## III. Evaluation

Implementation and Experiments: what you actually did (objectives, choices, data, and final analysis of results). Base comparison: (what is a comparable work, and relating your work to existing works). Developed software and how it can be evaluated.

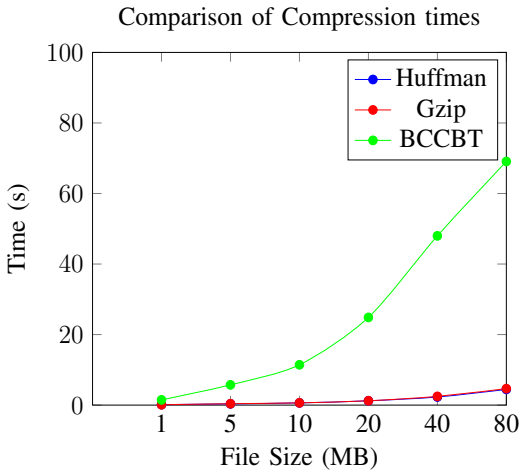
### A. Implementation

### B. Experiments

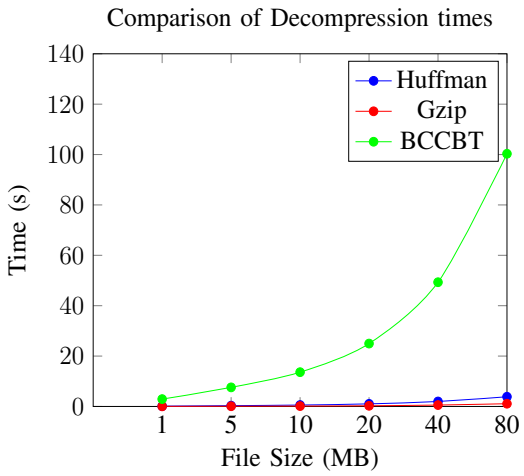
In order to effectively be able to analyze the effectiveness of the BCCBT algorithm we had to find two other comparable open source compression algorithms to test against. As a refresher for the factors that we had used to compare the 3 algorithms against eachother we used the 4 following factors

1) Compression Time
2) Decompression Time
3) Saving % = $\frac{Orig\ File\ Size - Compressed\ File\ Size}{Orig\ File\ Size}$
4) Compression Ratio = $\frac{Compressed\ File\ Size}{Original\ File\ Size}$

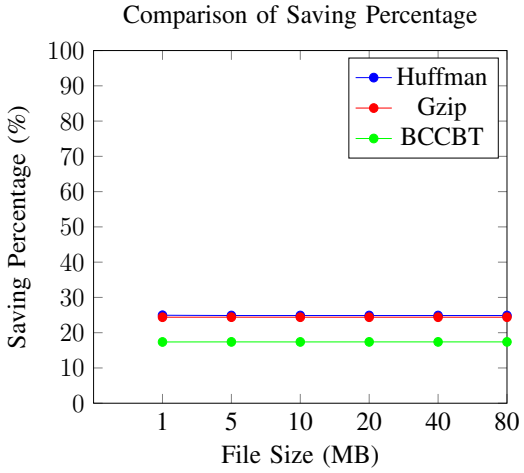
Using these factors will enable us to effectively test whether the BCCBT algorithm is both effective for compressing files of differing sizes along with showing whether or not it is practical and where the use cases for this compression algorithm.



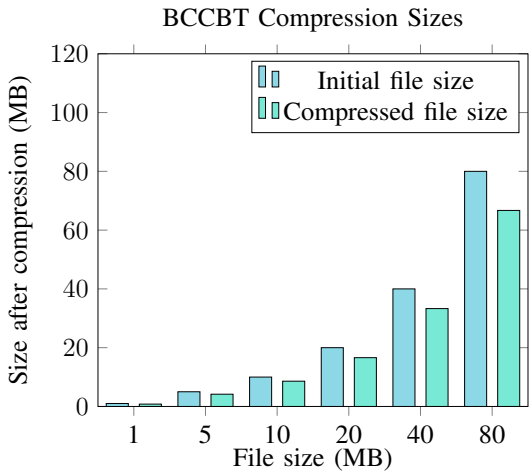

These results indicate that although for low file sizes the compression time is quite similar as the file size gets bigger the compression time exponentially gets bigger and the gap between both Huffman and GZip gets bigger and bigger this result indicates that for the implementation that we made it would not be practical or in your interest to use this on file sizes above 1MB however the compression time of this algorithm can be cut down in further implementations of the algorithm. Through our tests we were able to find out why the compression times of our implementation of the BCCBT algorithm were slower than GZip and Huffman and it was due to the population of the complete binary tree and traversing through the tree recursively rather than iteratively which had a heavy impact on the compression time making the implementation much slower when compared to the Huffman and Gzip compression algorithms.

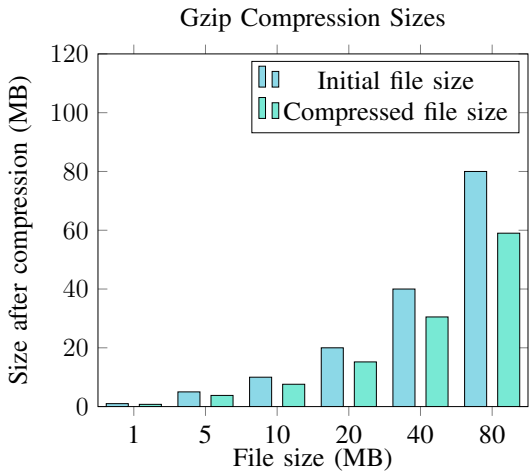


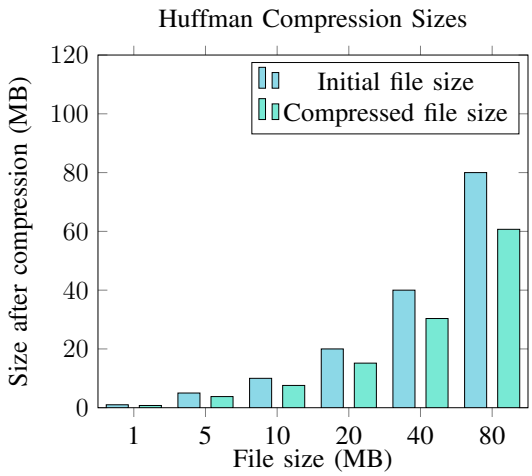

The results for the comparision of the decompression times are quite similar to the compression times mainly because while decompressing we use the similar functions to the compression for the files this graph also represents the problem with the optimization of the compression and decompression of the implementation of the BCCBT algorithm while at higher file sizes.




The graph above demonstrates the saving percentage in % which shows the comparision of the different algorithms for compression and the overall effectiveness of each one depending on file size, this demonstrates that the saving percentage is linear and the gap between the algorithms do not end up becoming bigger unlike the compression/decompression times.




The compression size graphs for the Huffman, Gzip, and BCCBT all show that no matter what the file size is the amount the file size is being reduced by stays linear. Another important part to note about the graph is that the BCCBT does not make the compressed file size as small as the other compression algorithms further indicating how the algorithm is not as efficient as the opensource algorithms that we had compared our implementation against. In conclusion based on the experiment results that have been presented the BCCBT algorithm is a efficient

compression algorithm however the implementation that compresses and decompresses the bitcodes, lvls, and frequency files make the distinction of usability clear as the implementation of the BCCBT algorithm performs much slower making it less practical for high file sizes.

## IV. Planning and Execution

Compares the final work with the project proposal: comments on if the stated goals have been achieved, and if not, outlines the challenges. Outlines contributions of group members including, ideas, implementation, research, writing report.