

Ant Colony Heuristics in File Compression: An Exploratory Study

Vijesh Mehta

Abstract

In compressing files, conventional algorithms use mathematical patterns to reduce file size. It is our thesis to test and see if conventional file compression can be improved by altering the bits in a file.

Using Ant Colony algorithms (Dr. Gargano – Ant Colony Heuristics), we were able to create a testing environment for this thesis. Software was developed which puts individual instances of ants in a file, where each ant moves in a predictable pattern while changing the bits in its path. Once the algorithm is finished executing, the altered file is written to disk in order to see if further compression can be achieved using a normal ZIP compression.

The results showed minimal changes in file compression. It is probabilistically equally likely to change a bit from 1 or 0. So the starting number of zeros and ones in a file remain somewhat equal to the ending number. In future tests it would probably be prudent to make more intelligent ant motions which move based on understanding zip's pattern search mechanism.

Introduction

Traditional file compression (ZIP) uses pattern searching to compress files. In order to exploit this process, it is our thesis to develop a method to change bits in a file which can lead to increased compression by traditional methods.

Ant colony heuristics as proposed by Dr. Gargano involves the use of individual ants which move through a file in a predictable motion. This method allows for the changing of bits and still being able to reverse the action to attain the original file.

It is our thesis that using an ant colony algorithm to traverse a file will change the bits enough to be able to improve compression.

Methods

In order to test this thesis, an application was developed to simulate the ant motion. The application reads in a file and separates each bit into a separate entity. Individual instances of ants are placed randomly inside a file and then allowed to move in a predefined method.

The file is organized in a 2 dimensional grid. An ant starts by facing north and turns right 90 degrees if it is standing on a 0, or left 90 degrees if it is standing on a 1. The ant changes the bit from a 0 to 1 or 1 to 0 and then moves forward 1 square. If the ant is at any of the edges of the grid, it simply wraps around to the other side. In addition, we tested ants which moved 2 squares also.

The ant is stopped when one of three conditions is met. Either the ant has traversed the whole file, moved 25,000 bits or has returned to the same location 10 times. These cues were setup in order to limit processing time and to prevent repetitious movement by the ants.

In the testing application, we used 500 instances of ants to move around the file. The test was performed on 4 different file types: DOC, PDF, BMP, and TXT. The tests which we performed were as follows. The original file was zipped using standard compression. A second test was to first move ants in the original file and then use zip compression on the altered file. A third test was to zip the original file, move ants in the zip file and then compress the altered zip file. These tests were performed on both single moving ants and double moving ants.

Results

The results (Table 1) showed that in only one case the ant algorithm had better compression (Zipped-Ant-Zipped BMP). The savings in file size was only 1 byte, so it is nearly negligible.

An important thing to notice is that the ants traversed approximately 3-11 Kilobytes. The DOC file had consistent 4 Kilobyte traversal. PDF showed that the 2Ant (ants which moved 2 positions every time) moved 11 Kilobytes and 7 Kilobytes with the single square moving ants.

Table 1: Data which shows the file size, percent traversed and bytes traversed after each method of testing.

Method	Bytes	Ant Trav. %	Bytes Trav.
DOC - Original	16.5 KB		
Zipped	4.99 KB		
Ant-Zipped	8.75 KB	19.01%	3.13 KB
Zipped-Ant-Zipped	5.11 KB	74.08%	3.69 KB
2Ant-Zipped	9.10 KB	20.90%	3.44 KB
Zipped-2Ant-Zipped	5.11 KB	76.69%	3.82 KB
PDF - Original	166 KB		
Zipped	157 KB		
Ant-Zipped	160 KB	4.61%	7.65 KB
Zipped-Ant-Zipped	157 KB	4.59%	7.21 KB
2Ant-Zipped	162 KB	6.84%	11.35 KB
Zipped-2Ant-Zipped	157 KB	7.10%	11.15 KB
BMP - Original	516 KB		

Zipped	165 KB		
Ant-Zipped	171 KB	0.88%	4.54 KB
Zipped-Ant-Zipped	164 KB	5.02%	8.28 KB
2Ant-Zipped	173 KB	1.07%	5.52 KB
Zipped-2Ant-Zipped	164 KB	6.84%	11.29 KB
TXT - Original	62.5KB		
Zipped	980 bytes		
Ant-Zipped	10.7 KB	9.00%	5.63 KB
Zipped-Ant-Zipped	980 bytes	94.00%	.92 KB
2Ant-Zipped	20.1 KB	14.00%	8.75 KB
Zipped-2Ant-Zipped	1.07 KB	95.00%	.93 KB

Discussion

The results showed that compression was not improved by changing the bits. Only in one case (BMP) a 1 Byte improvement was made. It is just as likely to change a bit from a 0 to 1 or 1 to 0 depending on the percentage the original file starts with. So the file which has been altered by the ants ends up with mostly equal percentage of 0's to 1's as it started with. This in turn will not lead to better compression. It would be more prudent if the percentage of 0's to 1's was drastically changed by the ants thus making it more feasible to higher compression.

New questions can be asked from the results. In general, the ants moved approximately 3-11 Kilobytes. This range questions how efficiently the ants moved in a file. The stop circumstances for the ants (25,000 bits traveled, 10 times repeated, or the whole file) cause the ants to traverse a limited area of the file. Thus only 3-11 Kilobytes traversals are made even when used in different file sizes and types.

In future research it would be prudent to use more ants for larger files and to move the ants based on a more complex algorithm rather than moving 1 square or 2. An important thing to devise the algorithm is to look at zip algorithm and try to make the ants alter the file attempting to make it more compressible.

Conclusion

Traditional file compression uses patterns to compress files. By using ant colony heuristics to alter the file it was hypothesized that file compression can be improved.

An application to explore this theory was developed. Testing involved a combination of zipping and altering the original file with the ant algorithm. Results showed that no improvement in compression was gained by altering the file. A very minor gain of 1 Byte was made on a BMP file type.

The minimal change is attributed to the fact that the ants moved a limited area of the file – usually 3 to 11 Kilobytes. In that space, ants changed bits blindly – thus resulting in equal percentage of 0's to 1's as in the original file. Compression cannot be improved if the ant algorithm does not create patterns which zip can take advantage of.

In future tests it would be more prudent to have ants which traverse more efficiently through a file – based on a more complex algorithm which can take advantage of zip's pattern searching.

Acknowledgements

I want to thank Dr. Michael Gargano for asking me to do this project as an independent learning experience. He has been the advisor for this project.