

Evaluating lossless compression methods

Matt Powell

February 2001

Abstract

This paper describes the work being done in maintaining the Canterbury Corpus website, and in particular the process of automating results generation. We investigate the popularity and usefulness of the Canterbury Corpus as a data compression standard, and propose several areas for further research and development of the current system.

Keywords: data compression; lossless compression; Calgary Corpus; Canterbury Corpus

1 The Calgary and Canterbury Corpora

The Canterbury Corpus (and its predecessor, the Calgary Corpus), are collections of “typical” files for use in the evaluation of lossless compression methods. The Canterbury Corpus consists of 11 files, shown in Table 1; an explanation of how the files were chosen, and why it is difficult to find “typical” files, can be found in [1]. Previously, compression software was tested using a small subset of one or two “non-standard” files. This was a possible source of bias to experiments, as the data used may have caused the programs to exhibit anomalous behaviour.

Several criteria were identified for choosing the Canterbury Corpus, including that it should be *representative* of the files that are likely to be used by a compression system in the future; that it should be *widely available* and *contain only public domain material*; that it *not be larger than necessary* (to limit distribution costs); and that it should be *perceptibly and actually valid and useful*. With these criteria in mind, about 800 candidate files were identified as being acceptable and relevant for inclusion in a corpus, and compressed using a variety of compression techniques. For each group of files, a scatter plot of file size before and after compression was produced, and a line of best fit calculated using ordinary regression techniques. The “best” file in each category was identified as the file that was most consistently close to the regression line, and was then deemed suitable for inclusion.

These two corpora¹ have become *de facto* standards for evaluation of lossless compression methods; in fact, [2] goes so far as to state that the Calgary corpus is “traditionally used to test data compression programs”. Certainly both corpora have gained a great deal of support in the compression community. The data in Table 2 was obtained by surveying the files used in papers and posters presented at the Data Compression Conference in 1998 and 1999, in a similar way to

¹Yes, “corpora” is the plural of “corpus”

File	Category	Size
alice29.txt	English text	152089
asyoulik.txt	Shakespeare	125179
cp.html	HTML source	24603
fields.c	C source	11150
grammar.lsp	LISP source	3721
kennedy.xls	Excel spreadsheet	1029744
lcet10.txt	Technical writing	426754
plravn12.txt	Poetry	481861
ptt5	CCITT test set	513216
sum	SPARC Executable	38240
xargs.1	GNU manual page	4227

Table 1: Files in the Canterbury Corpus

[1]. The Canterbury and Calgary Corpora were by far the most common data sets used for lossless compression testing (the next most popular would be the *Jefferson* data set, which was used a total of only three times).

2 A system for maintaining the corpus results website

The web page at <http://corpus.canterbury.ac.nz/> provides information about the corpora, as well as links to download the files and a list of results for various compression algorithms and software. The pages include results, discussion, analysis, and links to other benchmarks. Previously this results database was maintained by hand, but more recently it has been updated to allow automatic maintenance of the pages.

A series of scripts has been developed using Python and Unix shell scripts to provide a modular approach to maintaining the website, as shown in Figure 1. Each method/file combination is run several times by the `run` script, and the results are then “crunched” by the `mkcrunch` script, which calculates the mean results over all runs to avoid errors. The speed statistics are also “normalised” against a run of the standard UNIX `compress` utility, to avoid bias due to abnormal loads on system resources.

Once the “crunched” data has been produced, the `mkreport` script is used to massaged into a more useful format (`mkcrunch` produces code readable by Python, but not by many humans). Again, a modular approach has been applied, and `mkreport` passes its input to several `Formatter` objects, each specialised to a particular output format. Current formatters include HTML (which produces HTML tables), \LaTeX (`tabular` format), and comma-delimited text (suitable for importing into spreadsheets such as Microsoft Excel).

Finally, the “.phtml” (“Python HTML”) files produced by `mkreport` are run through `mkhtml` to add formatting changes like heading styles and footer data. This is so that the formatting of the site may be kept separate from the data it contains, and the look and feel of the site can be updated without re-running the compression tests (a lengthy process!)

	1998		1999		Total	%
	papers	posters	papers	posters		
No test data used	26	20	20	27	93	32.6
Data not identified:						
—lossless	1	11			12	4.2
—lossy	5	13	2	3	23	8.1
Data not used by others	16	10	25	13	64	22.5
Lena	4	5	10	2	21	7.4
Calgary Corpus	5	4	2	2	13	4.6
Canterbury Corpus	2		3	3	8	2.8
Barbara	1	2	3	1	7	2.5
Goldhill	1	1	4	1	7	2.4
Peppers	1	2	2		5	1.8
Girl		1	2		3	1.1
Jefferson	2			1	3	1.1
Austen	2				2	0.7
Bike	1		1		2	0.7
Bridge			2		2	0.7
Brown	2				2	0.7
Coastguard	1		1		2	0.7
JPEG2000	1	1			2	0.7
King James Bible	2				2	0.7
LOB	2				2	0.7
Mandrill/Baboon		2			2	0.7
Palette Image Corpus	1		1		2	0.7
Shakespeare	2				2	0.7
Stefan		1	1		2	0.7
Wall Street Journal	2				2	0.7
Total	80	73	79	53	285	100.0

Table 2: Test data used in Data Compression Conference papers and posters

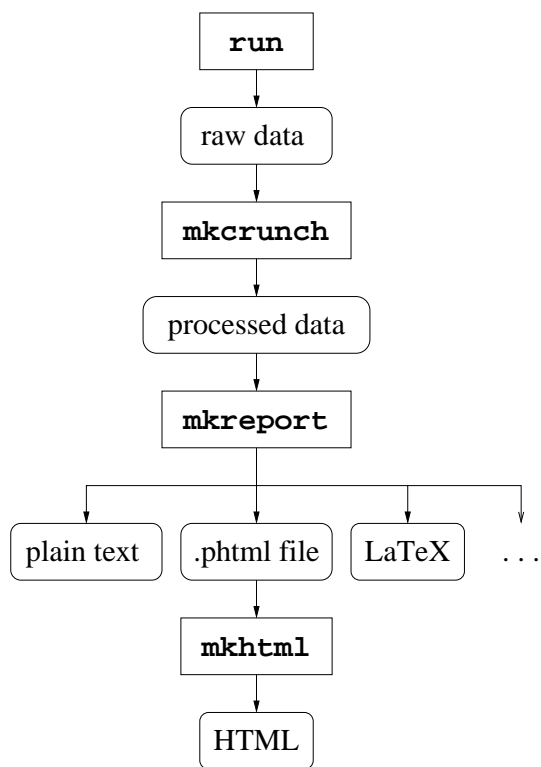


Figure 1: A typical maintenance run

It is possible to add results to the database without their being automatically generated by `run`. In such a case, all that is needed is a set of results that complies with the format of the `.crunch` files produced by `mkcrunch`. Each `.crunch` file contains results for each file in a given collection, as well as details about the run, including when it was completed, by whom, and on what system.

3 Issues with the current system

The Canterbury Corpus was primarily intended to replace the Calgary Corpus, which was showing its age. Unfortunately, the new collection of files has aged even more quickly than its predecessor, and may no longer fully reflect the range of files that need to be compressed. In particular, large files (in the order of tens or even hundreds of megabytes) are no longer uncommon, and there are many new file formats which could be investigated. To this end, several new “collections” have been added to the project, including a collection of large files.

Additionally, the files in the Canterbury Corpus were carefully chosen to examine “typical” behaviour on the part of the compression software. However, it is often useful to investigate “worst case” behaviour. An “artificial” collection has been added, consisting of files for which the compression methods may exhibit pathological or worst-case behaviour—for example, files full of randomly-generated characters, a file consisting of a single letter ‘a’, or a file with many repetitions of a short sequence.²

Perhaps a more serious concern is the issue of compression speed. For the corpus to provide reliable results, the compression tests should ideally be run under identical test conditions. In practice, this is never possible. Issues such as processor speed and available system resources can have a dramatic effect on the running time of a program. To counter this, all speed measurements are given relative to the time taken by the standard UNIX `compress` utility, as mentioned above. This ensures that the times listed in the results remain consistent.

However, if an author of a new compression program wishes to add his or her results to the list, there may be genuine reasons why the software cannot be tested in this way. Perhaps the software was written for a different architecture, or the program is unable to be released for reasons of commercial sensitivity. In such cases it will be necessary to assess the speed of such software *absolutely*—that is, without respect to the system on which the tests were run. Possible solutions to this problem are discussed in Section 4.3.

It will also be necessary to be able to verify that the results submitted for inclusion on the website were actually generated by the compression software. Such verification may be possible by means of testing software which outputs an encrypted version of the results.

²Such artificial files proved very useful in the investigation of a compression scheme brought to the curators of the corpus for testing. Its author claimed that the program was capable of compressing any file to a nearly constant ratio of around 7%—which is clearly impossible. Suspiciously, it was even able to compress completely random files to this remarkable ratio. The only drawback to the system was a mysterious loss of free hard drive space, roughly equal to the size of the file being compressed. . .

4 Proposed research

4.1 User interface

To simplify the process of maintaining the results website, a graphical user interface has been developed which allows the user to perform various mundane or complex tasks with relative ease. For example, by selecting a menu item it is possible to generate the reports, process the HTML files, move them onto the server, set correct file permissions and remove “artifact” files created in the process. By automating such tasks, the capacity for inadvertent error is greatly reduced.

There is still substantial room for improvements to the interface, including adding the facilities to add new files or compression methods.

4.2 Data analysis

At present the data presented on the results website is relatively rudimentary—only the raw results and the statistical mean are given. However, given the automated nature of the maintenance process, it would be a simple matter to add other statistics (for example, the standard deviation of the compression ratios for a given algorithm) to the site.

4.3 Cross-platform benchmarking

As mentioned above, the question of performance measurement poses a definite problem. The current system, which uses `compress` to measure the relative speeds of compression software, is a good solution for many existing methods, but may lack wider applicability because current software tends to be geared more and more towards the Windows/Intel platform, and as such will be incompatible with the current UNIX-based benchmarking system.

Although `compress` is available for a number of different platforms, it is difficult to tell whether the UNIX version will be “the same” as the DOS version, and so on. In fact, even when compiled from the same source code, two versions of `compress` may differ in performance because of optimisation routines built into the source code by means of conditional `#defines`. Thus the end result is a program that runs as fast as possible on each target platform, but whose internal behaviour may vary to take advantage of each platform.

It is proposed that research be undertaken to investigate whether such optimisations make any real difference to the performance of `compress` (or any other compression software we may wish to test). One possible method of testing this would be to write a program that compiles on as many platforms as possible, and whose behaviour does not depend on idiosyncracies of the target platform. Such software need not necessarily *be* a compression program, as long as it performs the same *sort* of tasks, e.g. complex array processing, file input and output and sub-byte processing.

5 Conclusions

Although the Canterbury and Calgary Corpora are already widely used, there is significant opportunity to make results more accessible via the Canterbury

Corpus Website. In particular, more accurate time measurements are called for, as well as methods for verifying results remotely.

References

- [1] Ross Arnold and Tim Bell. A corpus for the evaluation of lossless compression algorithms. In *Data Compression Conference*, pages 201–210. IEEE Computer Society Press, 1997.
- [2] David Salomon. *Data Compression: the complete reference*. Springer-Verlag, 1998.