9

Conclusion

The story of the Burrows-Wheeler Transform is a valuable case study in the process of research and discovery. The idea was conceived in 1978, but it didn't see the light of day until published through a collaboration that led to the seminal technical report number 124 at Digital Equipment Corporation, published in 1994. Despite this being Burrows and Wheeler's only publication relating to the idea, a few people recognized its value, picked it up, evaluated it, and gave it enough exposure that it grew a large community that brought it to a level of maturity where it could be used in a general purpose compression utility. In the following dozen or so years, literally hundreds of papers had been published by scores of researchers based on the single technical report. The most visible application coming out of the BWT is the BZIP2 compression program, which will have saved many terabytes of disk space and weeks of download time. And with the number of papers produced that are built on the method, no doubt many careers have been greatly helped by the fertile research ground that it provided.

Through this book we have covered many of the directions that the BWT has led people in: new algorithms and data structures for performing the transform, fine-tuning the coding that can be done with the transformed file, new ways to sort the strings for the transform, a better understanding of other compression methods by relating them to the BWT, using the BWT data structures to aid searching and pattern matching, and applying its algorithms and data structures in contexts ranging from image analysis to computational biology. Yet there is a strong sense that we are only just beginning to understand the transform and its potential, as new variants and applications continue to be published regularly. The BWT is a powerful idea, and in the process of decoding generates a collection of useful arrays (R, V, W and so on) which can be used to provide a variety of indexes and views of the original text. The transform process thus provides us with data structures that have opened up novel possibilities, and may yet hold more opportunities for future applications.

As a compression system, the BWT is quite mature, having experiments done on it with that purpose in mind for over 10 years. It has been related to a broad range of the main lossless compression methods, including Ziv-Lempel methods, PPM, and DMC; the obvious relationship is that they all are using longest matches in some sense as part of their processing, but the literature has also established more subtle relationships where these apparently different methods gain their compression power by exploiting variable size contexts. The BWT has forced the issue of considering forward and backwards contexts, and has led us through a different path to the suffix array as a structure for identifying contexts, performing longest matches, and supporting compressed full-text indexing.

The promising theory underpinning the BWT for compression is backed up by practice. Seward's free open source system, BZIP2, is one of the best general purpose compression systems implemented, and is available for Linux, Windows and Macintosh systems. In contrast to the maturity of work relating to compression with the BWT, for other applications, especially pattern matching and text indexing, important new ideas are still being published, and we are only now starting to understand how to exploit it.

Even in the area of compression new developments are being made; for example, it has only recently come to light that the MTF list may not be necessary despite having been very closely associated with the BWT since its first publication. These developments (like most in lossless compression) are generally to do with the tradeoff between compression and speed. Squeezing extra compression out of existing systems seems to have hit very firm limits, and only small gains are made by applying even large amounts of computing power. However, researchers are still finding faster data structures and coding techniques that are not so demanding, which provide more options for applications where a balance must be struck between computing effort and compression.

Another issue that has serious implications for the future of the BWT is how it interacts with real computer hardware, particularly cache memory systems that may not be able to work well given the random nature of the transform's accesses to the large block of data that is being processed in memory. Other general-purpose compression methods (such as the widely used derivatives of Ziv-Lempel coding) appear to work better with cached systems because of the repeated use of a smaller section of memory, and potentially this may give them an advantage over BWT methods in the future, if speed is the issue. Balanced against this are two important trends in hardware: larger caches, and parallelization. In general the BWT benefits from having as large a block size as possible, but there are diminishing returns on this, and it is possible that caches will grow to the size that the BWT is able to exploit them and yet still give some of its best compression performance. Perhaps more significantly, multiprocessor systems with a large amount of memory parallelism have the potential to, for example, process different parts of a BWT decoding at the same time, which offers significant speed improvements.

In addition to applying parallel systems to existing BWT-based methods, variants of the BWT may develop that are more amenable to working in a parallel environment, or with whatever architectures develop in the future. There is also a growing body of work on hardware-based BWT implementations which are specifically designed for this kind of processing. Again, there is considerable potential here for improvement.

Because this area is such an active area of research, we have included in Appendix B a list of web sites that have up-to-date information about the Burrows-Wheeler Transform, and its connection with compression, suffix arrays and pattern matching.

We look forward to a promising future for this transform as it goes through its second decade; it is being applied in an environment of new data structures, more powerful computers with new models of computation, increasing amounts of data to be processed for storage, indexing and pattern matching, and new theory to help us better understand how we can exploit a powerful technique that is based on simply muddling up the contents of a file.