Efficient Computation of the Burrows-Wheeler Transform

Kunihiko Sadakane
Department of Computer Science and Communication Engineering,
Kyushu University, Japan
sada@csce.kyushu-u.ac.jp

The Burrows-Wheeler Transform [1] (BWT) is the core technique that unifies text search and compression. Many self-indexing text indices use this technique [2, 7, 3]. In spite of its importance, there has been no time and space efficient algorithm to compute the BWT. Though it can be computed in linear time by using linear time algorithms for constructing suffix trees or suffix arrays, they need much more space than the output. On the other hand, the BWT can be computed using $O(\log n)$-bit extra space although it will require $O(n^3)$ time for a text of length n.

In this talk, we review the development of time and space efficient algorithms for computing the BWT. The algorithms runs in optimal space, i.e., $O(n \log |\mathcal{A}|)$-bit working space for a text of length n on alphabet $\mathcal{A}$. The time complexities are $O(n|\mathcal{A}| \log n)$ [6], $O(n \log n)$ [4], and $O(n \log \log |\mathcal{A}|)$ [5].

References

[1] M. Burrows and D. J. Wheeler. A Block-sorting Lossless Data Compression Algorithms. Technical Report 124, Digital SRC Research Report, 1994.

[2] P. Ferragina and G. Manzini. Opportunistic Data Structures with Applications. In *41st IEEE Symp. on Foundations of Computer Science*, pages 390–398, 2000.

[3] R. Grossi, A. Gupta, and J. S. Vitter. When indexing equals compression: experiments with compressing suffix arrays and applications. In *Proc. ACM-SIAM SODA 2004*, pages 636–645, 2004.

[4] W. K. Hon, T. W. Lam, K. Sadakane, and W. K. Sung. Constructing Compressed Suffix Arrays with Large Alphabets. In *Proc. of ISAAC*, pages 240–249. LNCS 2906, 2003.

[5] W. K. Hon, K. Sadakane, and W. K. Sung. Breaking a Time-and-Space Barrier in Constructing Full-Text Indices. In *Proc. IEEE FOCS*, pages 251–260, 2003.

[6] T. W. Lam, K. Sadakane, W. K Sung, and S. M Yiu. A Space and Time Efficient Algorithm for Constructing Compressed Suffix Arrays. In *Proc. COCOON*, pages 401–410. LNCS 2387, 2002.

[7] K. Sadakane. New Text Indexing Functionalities of the Compressed Suffix Arrays. *Journal of Algorithms*, 48(2):294–313, 2003.