

# A new approach of DCA by using BWT

Bo Zhao, Ken-ichi Iwata, Shuichi Itoh, and Toshihiko Kato  
University of Electro-Communications  
1-5-1 Chofugaoka, Chofu, Tokyo, 182-8585 Japan  
{zhaobo, iwata, itoh, kato}@net.is.uec.ac.jp

Crochemore et al. introduced a two-pass lossless data compression scheme called data compression using antidictionaries (DCA)[1]. DCA finds some words (minimal forbidden words) that never appear in the text to be compressed, and chooses a subset of minimal forbidden words as an antidictionary (AD). Let  $w$  be a minimal forbidden word with length  $|w|$ . Every time prefix of  $w$  of length  $|w| - 1$  appears in the text, the compression procedure can predict that the next bit cannot be the last bit of  $w$ . Hence we can take advantage of such predictability by using AD to (de)compress the binary text. DCA enables very fast decompressors, and allows a parallel processing in encoding or decoding. The compressed text by DCA has the property that we can implement a search engine easily. DCA achieves good compression rates for the output from binary balanced sources which are Markov sources with two output symbols of equal probabilities or of probability 1 and 0.

In this work, we develop improved DCAs in the following two points:

- Firstly, we propose an improved DCA using predictable dictionaries instead of the AD. The predictable dictionary is made up of the words whose last symbols are uniquely determined by their prefixes. With this convention we can extend the size of source alphabet in DCA, for example from binary to ternary. The improved compression method is the asymptotically optimal coding for the output from balanced sources of larger alphabet size. The above good properties of the original DCA also hold in this extension.
- Secondly, we present an algorithm to construct the AD (or predictable dictionary) by using suffix array of text. Although DCA constructs AD by using suffix tree in [1], we can reduce the amount of memory necessary to construct AD by using suffix array. DCA and the above improved DCA achieve good compression rates for the output from binary balanced sources and balanced sources, respectively. *Bzip2* can, however, achieve better compression rate for the output from other sources. Moreover, by applying suffix array instead of suffix tree, we can exploit the relationship between suffix array and BWT, then it gives us a chance of choice between DCA and Burrows-Wheeler Compression Algorithm (BWCA).

Through our discussion of DCA, by limiting the source class, we can think that DCA has some intermediate property of BWCA and Prediction by Partial Match because DCA achieves good compression rate for the output from some limited class of sources, i.e. balanced sources, and keeps the fast decompression property of *bzip2*.

[1] M. Crochemore, F. Mignosi, A. Restivo, S. Salemi, “Data Compression Using Antidictionaries”, *Proceedings of IEEE*, Vol.88, No.11, pp.1756–1768, (2000).