

Partially Decodable Compression with Static PPM

Daisuke Okanohara

Department of Information Science, University of Tokyo
Hongo 7-3-1, Bunkyo-ku, Tokyo 113-0033
hillbig@is.s.u-tokyo.ac.jp

We propose a novel compression method, *Static PPM* (SP), which supports *partial decode* (p-decode) with low memory and computation requirement. P-decode is a function that decodes a data from an arbitrary position without decoding the whole, which is critical to exploit a large data in compressed state. While conventional compression methods do not support p-decode, recent self-indexing data structures such as Compressed Suffix Arrays (CSA) [1] and FM-index [2] support p-decode. However, they have to store the whole compressed data in memory since the order of the data is not preserved after compression. This causes a serious problem when a compressed data is larger than memory size. In contrasts, SP does not have to store compressed data in memory, that is, SP is the first compression method that supports p-decode for very large compressed data.

SP is the same as PPM except that it exploits a fixed prediction model derived from the whole data and flushes the history information at each constant interval. Because the encoder builds a fixed prediction model, the decoder can use the same model for p-decode at an arbitrary position. SP can start p-decode from the positions which have empty history information. The fixed model reduces the memory and computation requirement because the model can be converted into a deterministic automaton beforehand. The decoder therefore only traces the automaton instead of using dynamically constructed model as PPM.

SP uses prefix arrays to efficiently construct the fixed prediction model where it minimizes the total description length of compressed data and the model itself. SP compresses the data as PPM and the model by using redundancy among similar histories.

In order to show SP's high compression performance inherited from PPM and its p-decode performance, we compared SP, CSA [1], FM-index [2], gzip (with option -9) and bzip (with option -9) in terms of speed and compression ratio (Table 1). These results show that SP achieves fast p-decode with high compression ratio.

Table 1: Comparison of compression performance with the test data (116,524,435 Bytes) synthesized by XMark (<http://monetdb.cwi.nl/xml/index.html>). The column of p-decode shows the average time for 1,000 trials of 1kB p-decode. *1 Although FM-index also supports p-decode, we could not examine its performance under the same condition with others.

Method	SP	CSA	FM-index	gzip	bzip
p-decode (msec./kB)	0.317	0.626	*1	-	-
Encode (sec.)	134.36	290.69	285.81	12.88	34.08
Decode (sec.)	30.51	45.78	52.61	1.74	14.56
Compression ratio (bpc)	1.717	4.391	1.576	2.604	1.754

Acknowledgement This work was supported in part by IPA's Exploratory Software Project in 2002, 2003. The author thanks Prof. Sadakane for many comments and suggestions.

References

- [1] K. Sadakane. New text indexing functionalities of the compressed suffix arrays. *Journal of Algorithms* 48 (2), 2003, pp. 294–313.
- [2] P. Ferragina and G. Manzini. An experimental study of an opportunistic index. *Proceedings of ACM-SIAM SODA*, 2001, pp. 269–278.