Adaptive Dictionary Sharing Method for Re-Pair Algorithm

Kei Sekine*, Hirohito Sasakawa*, Satoshi Yoshida*, Takuya Kida*
*Graduate School of Information Science and Technology, Hokkaido University.

1. Background

Re-Pair [1]

- Lossless compression algorithm for text data
- Good compression ratio on repetitive text
  (Approximately 0.1% in repetitive text)
- Huge memory consumption
  (20 times as large as text length)

Difficult to apply several gigabyte of text

- Easy solution: divide into small blocks and compress one by one. This solution works well but makes compression ratio worse.

Re-Merge [2]

- A Variant of Re-Pair for large texts.
  It merges blockwise dictionary recursively.
  It achieves very good compression ratio, but the compression speeds are sacrificed.

Re-Use [3]

- Simple extension of Re-Pair for large texts.
  To reduce the total dictionary size,
  it shares a part of blockwise dictionaries among all blocks

In this work, we developed more sophisticated method for sharing the entries of dictionary.

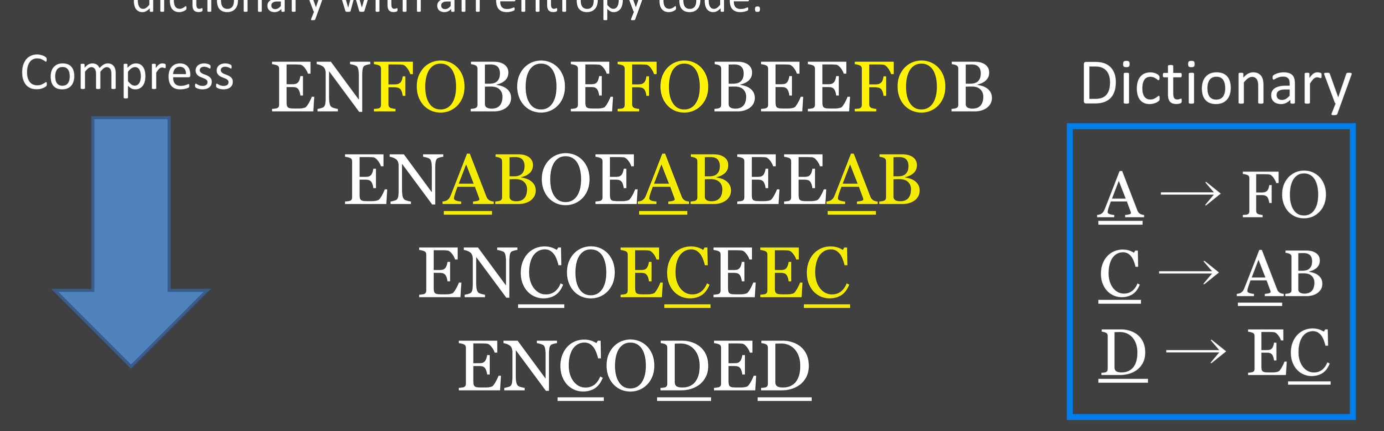
[1] N. Jesper Larsson and Alistair Moffat, "Off-Line Dictionary-Based Compression", Proceedings of the IEEE, vol.88, Issue 11, pp. 1722 – 1732 Nov. 2000.

[2] R. Wan and A. Moffat, "Block merging for off-line compression", Journal of the American Society for Information Science and Technology, vol. 58, Issue 1, pp. 3 – 14, 1 Jan. 2007.

[3] Kei Sekine, Hirohito Sasakawa, Satoshi Yoshida, Takuya Kida, Variable-to-Fixed Encoding for Large Texts Using Re-Pair Algorithm with Shared Dictionary, In Proc. of Data Compression Conference (DCC 2013), Snowbird, Utah, USA, pp. 518 Mar. 2013.

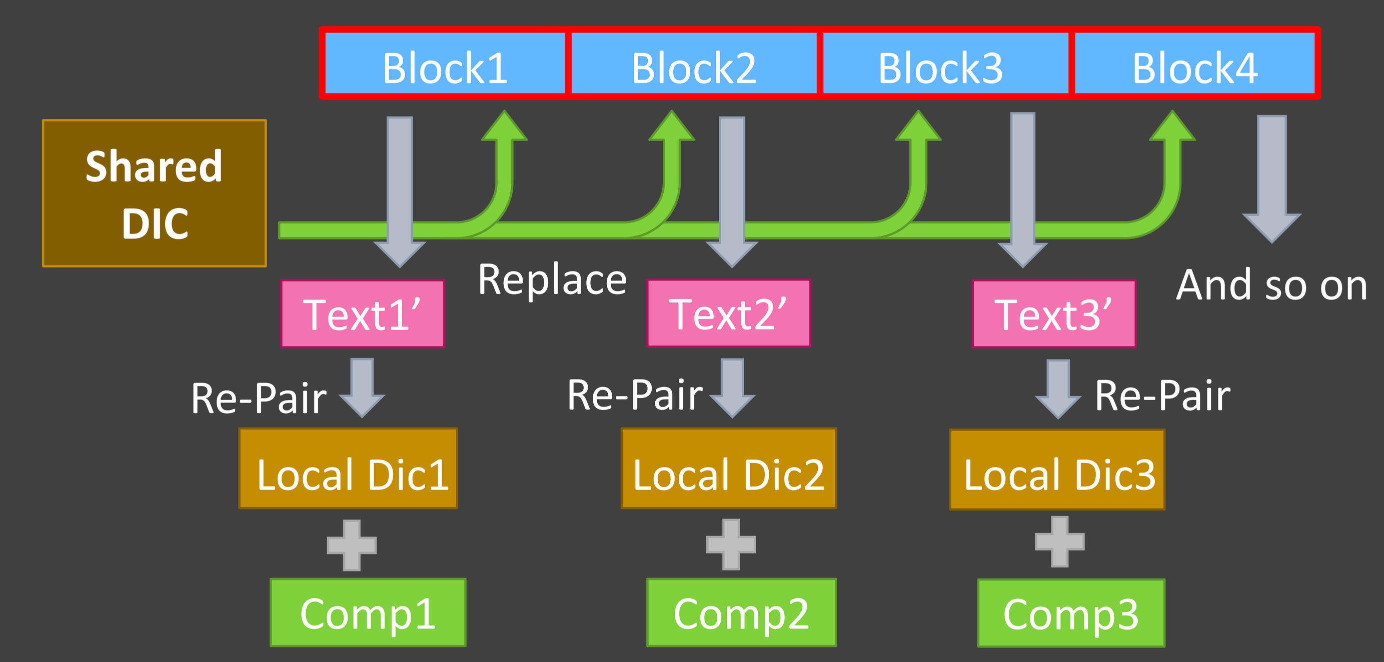
2. Re-Pair [1]

Substitute the most frequent bigram into a new symbol until all the bigrams are unique. Then, encode the compressed text and dictionary with an entropy code.



3. Re-Use [3]

It constructs dictionary which shared in all blocks (shared dictionary) by sampling from the input text.



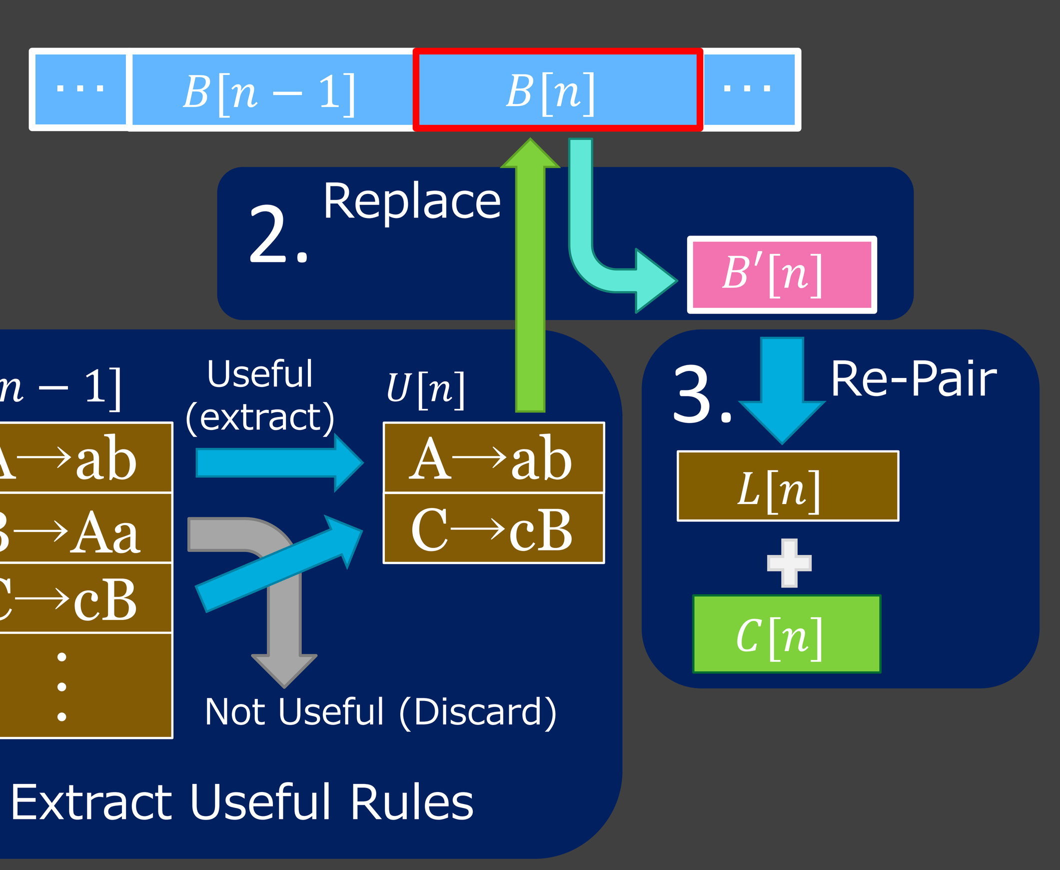
Problem: Shared dictionary is static, some useless rules are possible to make compression ratio worse.

4. Proposed Method

Adaptive dictionary Sharing (ADS)

ADS shares the rules between two consecutive blocks adaptively. Details are as follows:

1. Extract rules which appear in the nth block $B[n]$ frequently from the dictionary for $B[n-1]$ ($D[n-1]$) and construct the set of extracted rules of $D[n]$ ($U[n]$).
2. Replace $B[n]$ by the rules in $U[n]$, then obtain half-compressed text of $B[n]$ ($B'[n]$).
3. Apply Re-Pair to $B'[n]$, and then obtain the compressed text of $B[n]$ ($C[n]$) and the set of local rules of $B[n]$ ($L[n]$).
4. Combine $L[n]$ and $U[n]$ into $D[n]$.



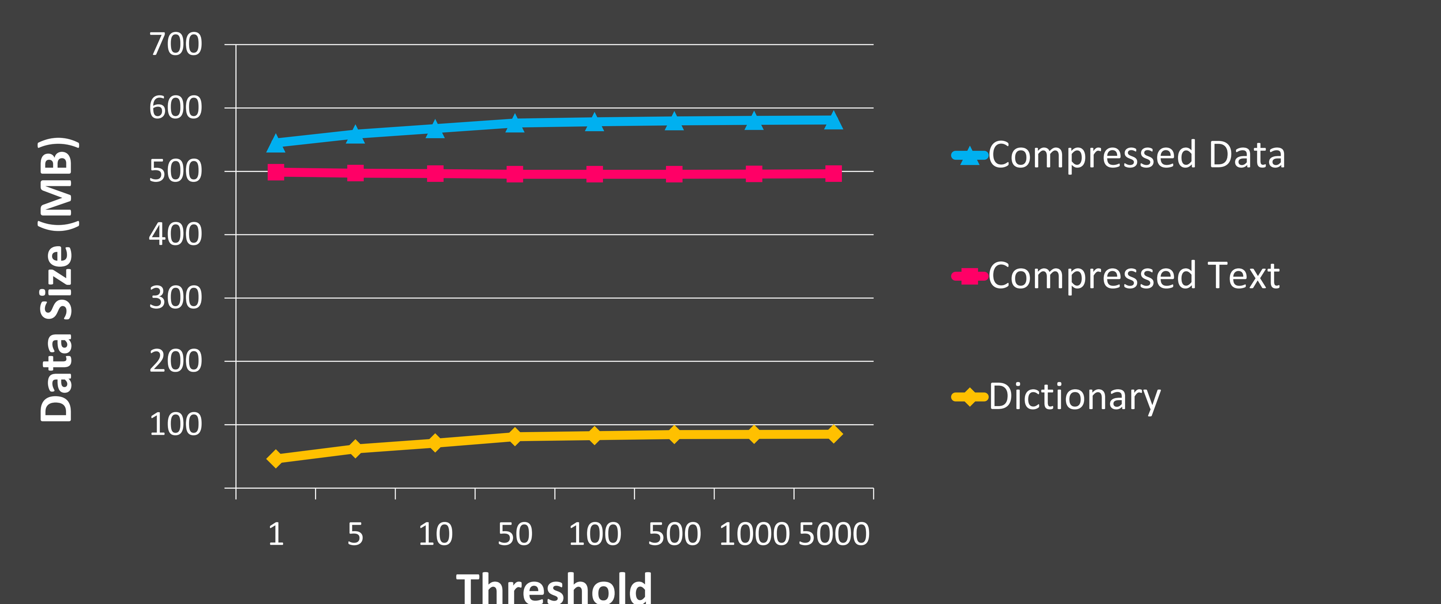
5. Experiments

Performance comparison

Algorithm	Comp. ratio	Memory	Comp. time
Proposed	**0.89 %**	**17,224 KB**	**67.5 sec**
Re-Use	0.97 %	16,432 KB	68.7 sec
Re-Merge	0.66 %	602,596 KB	188.6 sec
Re-Pair	0.11 %	5,065,924 KB	130.7 sec
Gzip	34.99 %	1,680 KB	24.3 sec
Bzip2	5.16 %	7,756 KB	55.4 sec

It shows the performance on highly repetitive text, *einstein* (650 MB) from *Pizza & Chili corpus* [3]. The proposed method reduces memory consumption without sacrifice of compression ratio.
The proposed method is three times as fast as Re-Merge in compression time, and memory consumption of ours is 1/37 of Re-Merge.

Compression data sizes for various threshold



It shows the variation of the compression data sizes for various threshold on natural language text, *english* from *Pizza & Chili corpus* [3].
We can see that by reducing threshold, the size of dictionary could be reduced without increasing the size of the compressed text data.

[3] *Pizza & Chili corpus* : http://pizzachili.dcc.uchile.cl/texts.html

6. Conclusion

We proposed a simple algorithm for dictionary sharing algorithm to reduce the memory consumption of Re-Pair algorithm. Experimental results show that proposed algorithm reduces memory consumption without sacrifice of compression ratio.

Our future work is to develop a method that determines the input parameters automatically.