A Fast Block-Sorting Algorithm for Lossless Data Compression

Dipl.-Ing. Michael Schindler, Vienna University of Technology
Karlsplatz 13/1861, A–1040 Vienna, Austria. michael@eiunix.tuwien.ac.at

This poster introduces a new transformation for block-sorting data compression methods. The transformation is similar to the one presented by Burrows and Wheeler, but avoids the drawbacks of uncertain runtime and low performance with large blocks. The cost is a small compression loss and a slower backtransformation. In addition to that it is well suited for hardware implementation. Typical applications include real-time data recording, fast communication lines, on the fly compression and any other task requiring high throughput.

The difference between this transformation and the original block-sort transformation is that the original transformation sorts on unlimited context, whereas this transformation sorts on limited context (typically a few bytes) and uses the position in the input block to determine sort order in case of equal contexts. This maintains the two goals of the original transformation, which are backtransformability and grouping symbols in similar contexts together. At the same time it allows to use a very efficient sorting method for limited context, radix sort.

A typical compression implementation for order N first counts the number of occurrences for each possible order 2 context and calculates starting points for each bucket of the 65536 way radix sort. No matter how many sort passes follow the starting point calculation only needs to be done once. While N is greater than zero pointers to each input character are sorted into the buckets based on the Nth and N−1th context byte; after each pass N is decremented by two.

Decompression needs to know where each block of symbols sharing the same context starts. Due to the close relation to the original Burrows–Wheeler transform (only a permutation of symbols with the same limited context) a modification of the original backtransformation can be used to count how often each order N context appears in the original block. Having this information the start for each context block can be calculated. Then decoding gets simple: The first unused symbol of the current context is the correct continuation.

Related block-sorting topics presented:
- A very locally adaptive entropy coder
- How longer repetitions might be exploited

Further details and related topics can be found at: http://eiunix.tuwien.ac.at/~michael/st/