

Parallel Entropy Coding Using Multiple Coders

Martin Boliek, James D. Allen, Edward L. Schwartz
RICOH California Research Center, Menlo Park, California
boliek@crc.ricoh.com

Presented here is a method that allows for parallel compression and decompression in hardware. This parallelism can result in extremely high rates, often 100 million symbols per second or higher. The system can be generalized to any lossless or lossy system with deterministic decompression or decoding. The focus is on high-speed entropy coding. Efforts to build high-speed hardware versions of many different entropy coders are limited by fundamental feedback loops. Dividing the data into multiple streams that are fed into parallel coders is an design option. For this design to be feasible, however, the problem of efficient transmission of multiple streams of variable-length coded data must be solved.

Imagine image data delivered in raster scan data divided into multiple parallel streams. There are many choices of how this is done. For example, each parallel stream could be the data associated with a context bin or bins or it could have the same or similar predictor value. The data could be divided into probability classes. Or it could be parallelize arbitrarily by regions or tiles or every Nth pixel.

These parallel streams are encoded by separate variable-length entropy encoders in parallel. The codewords are placed in parallel buffers. If a decoder has the same structure (satisfying causality) and access to all the streams simultaneously, it is clear that the parallel codewords could be decoded and the original data stream could be regenerated. Note that the parallel decoders take codewords from the appropriate code stream in some, perhaps complicated, deterministic order, called codeword order. Because of the parallelization, this codeword order is very different than the original raster order.

In a real system it is unrealistic to have several channels for the parallel code streams. If the codewords from the parallel code streams are interleaved in the codeword order then a single channel will suffice. This interleaving can be performed by having a "snooper" decoder at the encoder to determine the proper order. A system such as this has variable-length codewords. Thus, it requires a bit shifter at the decoder to align the codeword and deliver them to the proper parallel decoder. This is the bottleneck of the system and mitigates the virtue of parallelism.

The solution offered here is to group the codewords from each independent code stream into fixed-length words, called interleaved words. The interleaved words can contain many codewords and parts of codewords. These words are interleaved according to the demand at the decoder. Each independent decoder receives an entire interleaved word. The shifting operation is now done locally at each decoder, maintaining the parallelism of the system.

An important feature of this interleaving is that it allows codes that are single context to be used in multi-context system. Experiments with run length codes, such as Golomb or Rice codes, show coding efficiency comparable to the multi-context QM-coder. Experiments with a prototype hardware suggest that a run length code system could run 10 to 20 times faster in a VLSI implementation with the same gate count as the QM-coder.