Precise RDO in AV1

Thomas Daede

03/27/2018

1 Goal

x264, x265, libtheora, and Daala implement an accurate rate estimation scheme for RDO (rate-distortion optimization). This scheme involves creating a checkpoint, writing all symbols to the bitstream, measuring the real number of bits consumed by the entropy coder, and then restoring to the checkpoint. In contrast, libaom only estimates the number of bits used for each symbol by looking at the CDF. In addition, although the probabilities dynamically change, the probabilities used for estimation are only updated at the end of the superblock. The goal is to create an AV1 encoder that can use an accurate rate estimation scheme instead.

2 Prior art

2.1 x264

H.264's state consists of 460 8-bit bins, plus a couple of bytes of entropy coder state, totalling to 488 bytes. There is no spatial context. COPY_CABAC is used to create a temporary copy of the state, which is then written to and thrown away.

2.2 x265

H.265's state is a mere 160 8-bit bins, totalling to 188 bytes of state. However, H.265 also has spatial state. x265 keeps one master state. Many checkpoints (backups) of this state are made, at various depths of recursion. After writing to the master state and counting the bits, the checkpoint is restored. Spatial state is treated separately - it is part of the master state but neither saved nor restored, but is kept consistent by the higher level search.

3 Challenges with AV1

AV1's CDF state is much larger, about 30KB. In addition, it has a large amount of spatial context. Copies that were insignificant with x264 become a major overhead with rav1e.

4 Solutions for the CDF context

4.1 Checkpoints (like x265)

This is what rav1e currently does, however it is more expensive than it needs to be.

4.2 Temporary copies (like x264)

This involves half as many copies. Because the encoder only operates on a temporary copy rather than the master, it can simply throw away the temporary state rather than do a second copy to restore the master state.

4.3 Temporary partial copies

Not every block touches every CDF. Knowing what CDFs will be touched ahead of time could shrink the size of the temporary state. For example, intra mode CDFs will not be used if a partition is being searched for inter modes.

4.4 Temporary copy-on-write

Instead of statically describing the copies made, the bitstream writer could keep track of which CDFs have been copied, e.g. in a bitmap. CDFs that have not yet been touched are copied on-demand from the master that the temporary was created from. This has the least amount of copying but the greatest bookkeeping overhead.

5 Solutions for the spatial context

5.1 Checkpoint/restore of entire spatial context

This is what rav1e currently does, but is entirely impractical as it involves copying a huge amount of data twice for every decision.

5.2 No save/restore of spatial context (like x265, libaom)

The RDO loop must be designed in such a way that the correct values are always maintained spatially, e.g. by ensuring that only one temporary partial copy exists at a time, and that spatial contexts written only affect the context of symbols being written in that partial copy. This works best with depth-first searches.

5.3 Temporary spatial context

A temporary copy of a spatial context is made and operated on. This is half as many copies of a checkpoint/restore method, but still not good enough.

5.4 Temporary partial spatial context

A superblock or block sized context is created and operated on, then thrown away. It needs to create copies of not only the mutable parts (those within the block) but immutable copies of the neighbors of the block that are used for prediction. Alternatively, it could maintain a reference to the master spatial context for this purpose.