The Alpha AXP, epilogue: A correction about file system compression on the Alpha AXP

devblogs.microsoft.com/oldnewthing/20170831-00

August 31, 2017

Raymond Chen

Some time ago, I claimed that Windows file compression had to be dumbed down in order for the Alpha AXP to hit its performance targets.

I have since been informed by somebody who worked on file system compression for the Windows NT that information was badly sourced. (My source was somebody who worked on real-time file compression, but not specifically on the Windows NT version.)

This is a bit of a futile correction because the wrong information has already traveled around the world [citation needed], posted some selfies to Instagram, and renewed its passport.

Windows NT file compression worked just fine on the Alpha AXP. It probably got a lot of help from its abundance of registers its ability to perform 64-bit calculations natively.

We regret the error.

Bonus chatter: Real-time file compression is a tricky balancing act.

Compression unit: If you choose a small compression unit, then you don't get to take advantage of as many compression opportunities. But if you choose a large compression unit, then reads become more inefficient in the case where you needed only a few bytes out of the large unit, because you had to read the entire unit and decompress it, only to get a few bytes. Updates also become more expensive the larger the compression unit because you have to read the entire unit, update the bytes, compress the whole unit, and then write the results back out. (Possibly to a new location if the new data did not compress as well as the original data.) Larger compression units also tend to require more memory for auxiliary data structures in the compression algorithm.

Compression algorithm: Fancier algorithms will give you better compression, but cost you in additional CPU time and memory.

What makes twiddling the knobs particularly difficult is that the effects of the knobs aren't even monotonic! As you make the compression algorithm fancier and fancier, you may find at first that things get slower and slower, but when the compression ratio reaches a certain level, then you find that the reduction in I/O starts to dominate the extra costs in CPU and memory, and you start winning again. This crossover point can vary from machine to machine because it is dependent upon the characteristics of the hard drive as well as those of the CPU. A high-powered CPU on a slow hard drive is more likely to see a net benefit, whereas a low-powered CPU may never reach the breakeven point.

Raymond Chen

Follow