Computing the size of a directory is more than just adding file sizes

devblogs.microsoft.com/oldnewthing/20041228-00

December 28, 2004

Raymond Chen

One might think that computing the size of a directory would be a simple matter of adding up the sizes of all the files in it.

Oh if it were only that simple.

There are many things that make computing the size of a directory difficult, some of which even throw into doubt the even existence of the concept "size of a directory".

Reparse points
We mentioned this last time. Do you want to recurse into reparse points when you are computing the size of a directory? It depends why you're computing the directory size. If you're computing the size in order to show the user how much disk space they will gain by deleting the directory, then you do or don't, depending on how you're going to delete the reparse point.
If you're computing the size in preparation for copying, then you probably do. Or maybe you don't – should the copy merely copy the reparse point instead of tunneling through it? What do you if the user doesn't have permission to create reparse points? Or if the destination doesn't support reparse points? Or if the user is creating a copy because they are making a back-up?

Hard links
Hard links are multiple directory entries for the same file. If you're calculating the size of a directory and you find a hard link, do you count the file at its full size? Or do you say that each directory entry for a hard link carries a fraction of the "weight" of the file? (So if a file has two hard links, then each entry counts for half the file size.)
Dividing the "weight" of the file among its hard links avoids double-counting (or higher), so that when all the hard links are found, the file's total size is correctly accounted for. And it represents the concept that all the hard links to a file "share the cost" of the resources the file consumes. But what if you don't find all the hard links? It it correct that the file was undercounted? [Minor typo fixed, 12pm]

If you're copying a file and you discover that it has multiple hard links, what do you do? Do you break the links in the copy? Do you attempt to reconstruct them? What if the destination doesn't support hard links?

Compressed files

By this I'm talking about filesystem compression rather than external compression algorithms like ZIP.
When adding up the size of the files in a directory, do you add up the logical size or the physical size? If you're computing the size in preparation for copying, then you probably want the logical size, but if you're computing to see how much disk space would be freed up by deleting it, then you probably want physical size.

But if you're computing for copying and the copy destination supports compression, do you want to use the physical size after all? Now you're assuming that the source and destination compression algorithms are comparable.

Sparse files

Sparse files have the same problems as compressed files. Do you want to add up the logical or physical size?

Cluster rounding

Even for uncompressed non-sparse files, you may want to take into account the size of the disk blocks. A directory with a lot of small files requires up more space on disk than just the sum of the file sizes. Do you want to reflect this in your computations? If you traversed across a reparse point, the cluster size may have changed as well.

Alternate data streams

Alternate data streams are another place where a file can occupy disk space that is not reflected in its putative "size".

Bookkeeping overhead

There is always bookkeeping overhead associated with file storage. In addition to the directory entry (or entries), space also needs to be allocated for the security information, as well as the information that keeps track of where the file's contents can be found. For a highly-fragmented file, this information can be rather extensive. Do you want to count that towards the size of the directory? If so, how?

There is no single answer to all of the above questions. You have to consider each one, apply it to your situation, and decide which way you want to go.

(And copying a directory tree is even scarier. What do you do with the ACLs? Do you copy them too? Do you preserve the creation date? It all depends on why you're copying the tree.)

Raymond Chen

Follow