Inside Windows NT Disk Defragmenting

Mark Russinovich

(Preprinted From WindowsItPro Magazine)

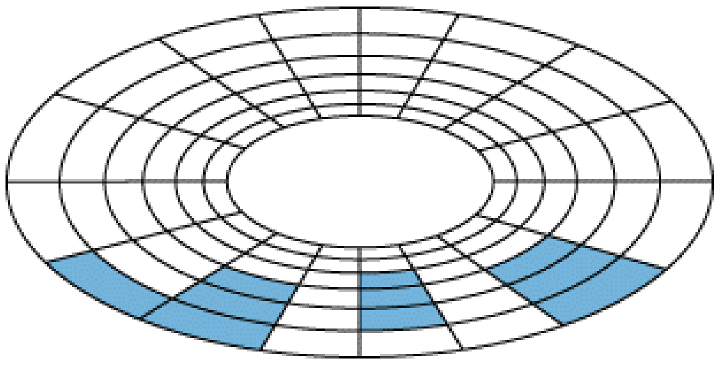
Welcome to the first installment of my regular column that explores the "guts" of Windows NT. I plan to cover topics ranging from high-level issues such as the architecture of NT to low-level issues such as details of algorithms NT employs. If you have a particular NT internals topic you'd like to see this column demystify, please drop me an email.

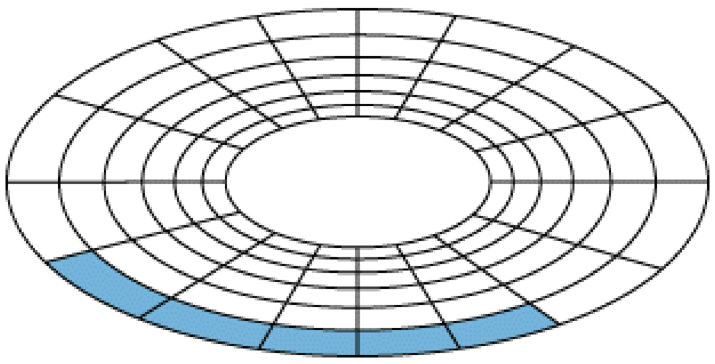
This month's topic is NT's disk defragmentation support. Disk defragmentation products are hot sellers, and many people want to know how these products work. After defining disk defragmentation, I'll provide a synopsis of the history of NT defragmentation products, and describe the support Microsoft added to NT 4.0's file systems specifically for use by defragmentation tools.

Disk Defragmentation

NT's file systems allocate the disk sectors of logical drives (e.g., C, D) in units known as clusters. When you create a file on a drive, the file system assigns clusters of storage for the file's data. As time passes and you expand and shrink the file and create and delete other files, both the free clusters on the drive and the clusters assigned to the files become fragmented (i.e., non-contiguous).

For example, suppose you create a Word document and the file system assigns six clusters to hold the document's contents. Later, you add a few pages to the document, and the file grows by six more clusters. If another file or directory uses the clusters that immediately follow the document's original clusters, the file system assigns the document's new file clusters from somewhere else on the drive--your file now consists of at least two fragments. Figure 1A, shows a simplified diagram of a fragmented file; Figure 1B depicts a defragmented file.





Fragmentation can negatively affect performance. When the disk head reads contiguous data, the level of throughput is high. When the system reads a fragmented disk file, the disk head must move from fragment to fragment. On a disk with many fragmented files, moving the disk head across fragments can consume significant time.

Disk defragmenters are products that aim to optimize system performance by reorganizing a drive's clusters to make file clusters contiguous, and in some cases, to make the free clusters contiguous. Defragmenters use heuristic algorithms that implement a fragment-shifting strategy to defragment the drive's files as quickly as possible.

The History of Disk Defragmenting on NT

In April 1995, Executive Software released the first defragmenter for NT: Diskeeper for NT 3.5 with Service Pack 1 (SP1) or Service Pack 2 (SP2). The company announced an update for NT 3.51 in July of the same year.

Because Microsoft did not design NT 3.5x's native file systems, FAT and NTFS, with disk defragmenting in mind, FAT and NTFS have no support for moving clusters around a disk. Executive Software purchased an NT source license from Microsoft to modify NT and the FAT and NTFS drivers to support disk defragmenting. As a result, Executive Software shipped its custom version of NT with Diskeeper for NT 3.51.

But as Diskeeper users upgraded NT 3.51 with SPs, they experienced problems. Installing an SP overwrote some of Diskeeper's files and left others alone, causing incompatibility problems for upgraded systems--and technical support problems for Microsoft and Executive Software.

As development of NT 4.0 began in 1995, Microsoft invited a Diskeeper developer to Redmond, Washington, to participate in the design and implementation of NT 4.0's defragmentation support. Basing Diskeeper on NT's built-in support let Executive Software avoid shipping custom versions of NT and meant Microsoft's technical support didn't have to troubleshoot non-standard versions of NT. One month before NT 4.0's public release, Executive Software offered a trial version of Diskeeper 2.0--the version for NT 4.0--on its Web site. Symantec has since entered the NT disk defragmenting market with its Norton Utilities Speed Disk, which also uses the NT 4.0 defragmentation support.

NT 4.0 Support for Disk Defragmenting

The defragmentation support introduced in NT 4.0's file systems consists of five commands: GetVolumeBitmap, GetRetrievalPointers, and MoveFile (common to both FAT and NTFS), and GetVolumeData and ReadMFTRecord (specific to only NTFS). Microsoft doesn't document these commands or officially acknowledge that they exist.

TABLE 1: Example File Mapping Array Returned by GetRetrievalPointers

File Cluster	Drive Cluster
0	1200
4	1000
7	1300

12	

TABLE 2: Example Compressed File Mapping Array Returned by GetRetrievalPointers

File Cluster	Drive Cluster
0	1200
4	-1
16	

GetVolumeBitmap

The GetVolumeBitmap command obtains a map of the free and allocated clusters of a logical drive. NT file systems maintain an array of bits that track the drive's clusters. An "on" bit signals an allocated cluster; an "off" bit signals a free cluster.

GetRetrievalPointers

The GetRetrievalPointers command returns an array of mapping information about the clusters allocated to a file. Each fragment of the file corresponds to an array entry that consists of a logical cluster number within the file and a drive cluster number. A final array entry contains the file cluster number of the end of the file. For example, consider a file with three fragments: The first fragment starts at drive cluster 1200 and continues for 4 clusters, the second fragment starts at drive cluster 1000 and continues for 3 clusters, and the third fragment starts at drive cluster 1300 and continues for 5 clusters. Table 1 shows the mapping array that GetRetrievalPointers returns for the file.

Sometimes GetRetrievalPointers returns a mapping array that contains a drive cluster entry of -1, as shown in Table 2. This entry signals a compressed file. In Table 2's example, the compressed data starts at drive cluster 1200 and continues for 4 clusters. The final file cluster entry of 16 means that the uncompressed file will require 12 more clusters.

MoveFile

The MoveFile command is the heart of NT's defragmentation support. MoveFile requires a handle to both the file that contains segments to be moved and the file's drive. Additional parameters track the starting cluster (relative to the start of the file) of the segment to be moved, the target drive cluster number, and the length of the segment. If the target clusters are free, MoveFile relocates the original clusters in a way that prevents data loss in case the system crashes during the move.

GetVolumeData

NT includes the two NTFS-specific defragmenting commands because the NTFS driver uses clusters differently from the FAT driver. The GetVolumeData command obtains detailed information about an NTFS drive, including its cluster size (in bytes), the size of the drive (in clusters), and the amount of free space on the drive (in clusters).

Defragmenters use GetVolumeData to identify a reserved portion of the disk (known as the MFT-Zone) that NTFS uses for expanding the Master File Table (MFT). The MFT is NTFS's index to all the files and directories on a drive.

To optimize file lookups, NTFS tries to keep the MFT defragmented by not allocating clusters around the MFT to other files. GetVolumeBitmap reports free clusters in the MFT-Zone, but MoveFile will not relocate clusters to this area; defragmenters need to know the MFT-Zone's location to avoid it.

ReadMFTRecord

The other NTFS-specific command, ReadMFTRecord, obtains a record of information from the MFT that you can use to create a cluster map for a file. Alternatively, you can enumerate the files on the drive and use GetRetrievalPointers to obtain mapping information for each file. However, sequentially reading MFT records and interpreting the raw NTFS on-disk data structures can enhance performance.

How NT 4.0 Defragmenters Work

At press time, only Executive Software and Symantec offer defragmenters for NT 4.0. Both vendors provide downloadable versions of their products. The products perform the same basic operations, but they employ different defragmentation algorithms. Let's look at how the products work.

First, each product creates a map of the drive, which shows the file fragmentation to the user. Mapping a drive takes three steps: Get the map of free clusters on the drive with GetVolumeBitmap, enumerate all the files on the drive and obtain file cluster maps with GetRetrievalPointers or ReadMFTRecord, display the file mappings.

Some clusters in the bitmap appear to be in use but not allocated to a file. These clusters belong to system metadata files (i.e., files that store file system-related information), directories, or files accessed exclusively by a process other than the defragmenter. Both products designate these clusters as immovable in their GUIs. The products also identify the MFT-Zone if the drive is an NTFS volume.

Next, the products enter a defragmenting phase. Because the drive mapping information constantly changes as programs create, delete, grow, and shrink files, the products do not rely on the information displayed to the user. Instead, they again enumerate all the files on the drive, and perform the following steps for each file: Get the map of free clusters on the drive with GetVolumeBitmap; obtain a cluster map for the file in question using GetRetrievalPointers or ReadMFTRecord; move segments of the file with MoveFile in an attempt to defragment the file.

The logic behind the third step is different for each product, depending on whether the defragmenter tries to move files to defragment the drive's free space or make room for files in specific places. Defragmentation is an iterative process that can even be undone as other processes perform file operations, so the defragmenters often repeat the third step many times.

Because MoveFile requires a handle to the file to be moved, the defragmenters must open a file before moving it. Opening a file that another process has already opened for exclusive access is not possible. Neither product can move files such as the MFT, Registry files, and Paging files because the system opens these files for exclusive access.

NTFS Caveats

Some restrictions apply to the NTFS implementation of MoveFile because its cluster movement engine uses NTFS file compression code. NTFS file compression adds a twist to the way NTFS allocates clusters for files.

NTFS performs compression on 16-cluster segments of a file. If a 16-cluster segment of data compresses down to 5 clusters, for instance, NTFS stores the 5 clusters on disk and notes the remaining 11 clusters as virtual clusters. To read the compressed file, the system reads the 5-cluster compressed portion from the disk, allocates memory for the 11 virtual clusters, and fills those memory locations with 0s. The system passes this 16-cluster chunk to the decompression algorithm, which re-creates the original data.

On FAT volumes, MoveFile can move clusters individually. The NTFS MoveFile routine moves clusters in only 16-cluster blocks because NTFS file compression works with 16-cluster segments.
Furthermore, the NTFS MoveFile function does not work with clusters larger than 4KB (NTFS file compression buffers are 64KB in size: 64KB ÷ 16 = 4KB). On drives larger than 4GB, the FORMAT utility initializes NTFS partitions with cluster sizes greater than 4KB; consequently, large drives with FORMAT's default cluster sizes do not support defragmentation.

Finally, NTFS prevents deallocated clusters from being used again until NTFS checkpoints the drive's state. Once every few seconds, NTFS ensures that all its crash recovery data is safely on disk; only then can deallocated clusters be reused. This characteristic challenges defragmenters because they can't determine when they can reallocate free clusters without repeated calls to GetVolumeBitmap.