How NTFS Reads a File

On NTFS there is metadata (data which "lives above" your data).

Metadata are special files that the NTFS file system driver uses to manage an NTFS volume. The most famous piece of metadata is the MFT (Master File Table), which is a special file typically consisting of 1024-byte records. Each file or directory on the volume is described by at least one of these MFT records. It may take several MFT records to fully describe a file...especially if it is compressed. (A 271MB compressed file can require over 450 MFT records!)

Below is an example of what occurs when NTFS goes to read in the 1-cluster file \Flintstone\Barney.txt.

1. The volume's boot record is read to get the cluster address of the first cluster of the MFT.
2. The first cluster of the MFT is read, which is used to locate all of the pieces of the MFT.
3. MFT record 5 is read as it is predefined to be the MFT record of the root directory.
4. Data cluster 0 of the root directory is read in and searched for "Flintstone".
5. If "Flintstone" is not found, then at least one other data cluster of the root directory needs to be read to find it.
6. The MFT record for the "Flintstone" directory is read in
7. Data cluster 0 of the "Flintstone" directory is read in and searched for "Barney.txt".
8. If "barney.txt" is not found, then at least one other data cluster of the "Flintstone" directory needs to be read to find it.
9. The MFT record for the "Barney.txt" file is read in
10. Data cluster 0 of the "Barney.txt" file is read in.

This is a worst-case scenario. It presumes the volume is not yet mounted, so the NTFS cache is empty at step 1, and the MFT itself needs to be located. But it shows how many I/Os[1] are required to get at a file that is only one level removed from the root directory: 10. Each one of those 10 I/Os requires a head movement.

As these I/Os are to different pieces of data on the volume (different MFT records, directories, and files); the contiguity of the pieces would only matter if the different segments were in close proximity to the MFT records representing them.

If you follow the step-by-step I/O sequence outlined above, you'll see that every time a new directory is encountered in the path is an additional two or three I/Os. For obvious performance reasons it is beneficial to keep the depth of your directory structure at a minimum.

[1] I/O is an acronym for input/output; the communication operation between computer components and/or users.