Data Fragmentation

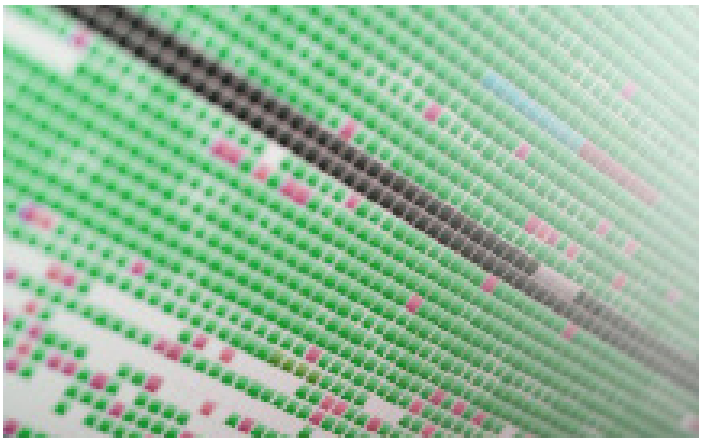
Nisha Balabhadra

Data fragmentation is a process by which files and data are broken apart into small blocks of memory in order to be stored by the computer. When breaking the data into these blocks, often times the computer will allocate more space than is needed to store the data, and this space will remain empty and unusable by the system. This process by which wasted space is created is called fragmentation. Fragmentation doesn't seem like a huge problem with newer computers and systems, however as time passes the wasted memory space builds up and the computers become our worst nightmares, slow, lagging, and mere shadows of the machines that they were originally.

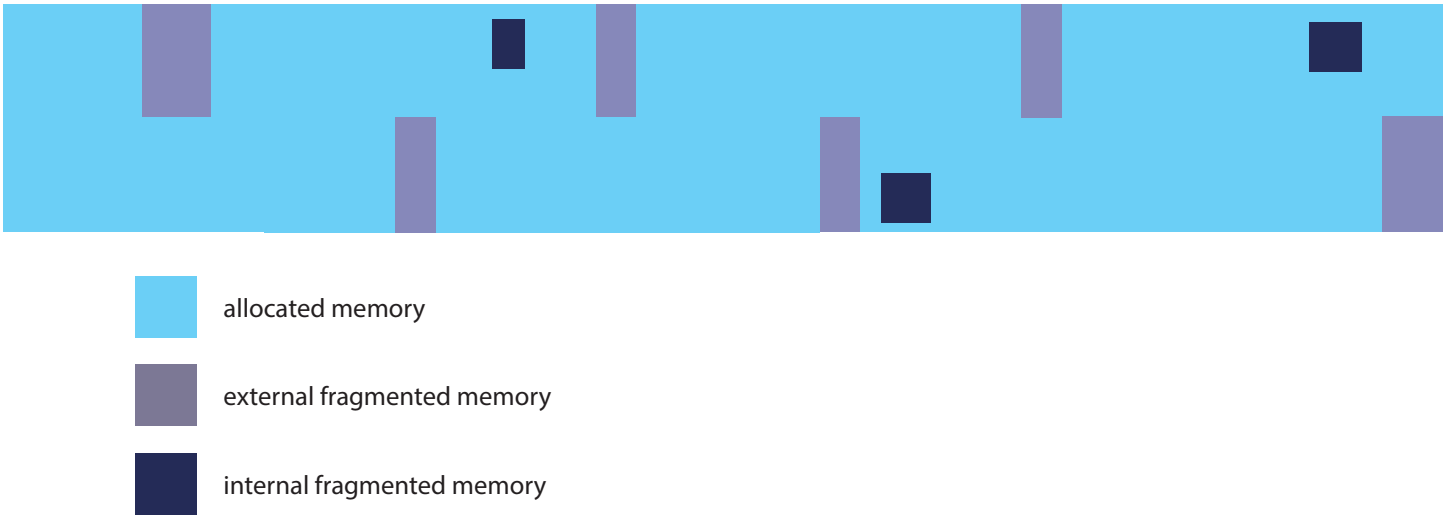
There are two main types of memory fragmentation, internal and external. Internal fragmentation occurs when a certain amount of memory is allocated for a specific job that the computer has to do. The computer overestimates the amount of space needed for that job and within the allocated memory there are blank spots of unused, wasted space that the computer can no longer use (Samanta, 2004). Since the wasted space is within a block of allocated memory it is referred to as internal fragmentation. External fragmentation happens when there are blocks of unused memory in-between already allocated memory blocks (Samanta, 2004). With external fragmentation the unused blocks are still available for use, however they cannot be used for larger jobs as the memory space is no longer continuous (Samanta, 2004).

Files stored on the computer are called fragmented when they are not stored in a continuous block of memory (Pal, 2008). This commonly occurs when there is not enough space in the computer's disk (Pal, 2008). It can also occur due to appending and editing the file (Pal, 2008). File types that are most often fragmented are PST files (that come from outlook emails), Microsoft Word documents and JPEG (image) files (Pal, 2008). Fragmented files are harder for the computer to retrieve as their memory is stored in many different places, and are therefore slower to retrieve (Peterson, 2002).

Some ways to reduce the occurrence of file fragmentation, as well as speed up the time to retrieve files is to store files in larger blocks of memory, ensuring their continuity (Peterson, 2002). This however raises the problem of internal fragmentation, where memory is deemed unusable by the computer. Other solutions involve storing series of files sequentially so that operations can be performed sequentially as well (Peterson, 2002). This begets problems when the clusters of files that are stored together become too large and must be allocated elsewhere (Peterson, 2002). Some even better solutions to file fragmentation include what is called the buddy system (Samanta, 2006). In the buddy system, blocks of memory are split into two "buddy blocks" until one of the blocks is an appropriate size for the operation being performed (Samanta, 2006). Whenever blocks of memory are returned (i.e. are no longer in use) they can recombine with their buddy blocks, and create more new space for new files and jobs to be stored or performed (Chang, 1996). The buddy system significantly improves retrieval speeds of files and improves memory storage in computers (Chang, 1996).



Fragmentation not only reduces speed of file retrieval, but it also reduces performance of systems such as SSD, Solid State Drives, which are similar to hard drives in computers, but utilize flash memory like a USB drive will, and are faster. Fragmentation reduces speed of SSDs by up to fourteen times



(Chen, 2009). In order to prevent aging of the system the SSD needs to be defragmented, or organized and cleaned regularly as a short time fix (Chen, 2009). In addition to reducing performance, fragmentation contributes to software aging. After fragmentation, empty blocks of memory are not used freely until certain processes are finished executing (Macêdo, 2010). This causes higher processing costs, and contributes to aging of the computer (Macêdo, 2010). Additionally this can cause the system to exit user mode to retrieve blocks of unused memory and then return back to it, further increasing process size and therefore aging the computer more (Macêdo, 2010).

Fragmentation can also be applied to resources for the computer to use in executing an operation or a job. Job scheduling involves allocating resources to each job, as well as determining the order in which jobs are executed (Huang, 2009). Resource fragmentation affects job scheduling by reducing performance (Huang, 2009). This problem can be alleviated by algorithms that either work to fit in jobs in a particular order so that the smallest jobs are finished first, or by which jobs need the least resources (Huang, 2009). With proper management of resource fragmentation, systems can run up to five times better than before (Huang, 2009).

Although fragmentation is typically negative, especially for the average user who simply wants a fast computer to store their work, and other files, it has interesting applications in fields such as encryption. Fragmentation can be used to enable privacy over data collections (Ciriani, 2007). In this instance, data can be fragmented over different servers and encrypted where necessary (Ciriani, 2007). This allows data to be highly visible, but at the same time keeping information classified (Ciriani, 2007).

Fragmentation is a pain when it comes to personal computing, but there are new ways of avoiding it being found everyday. And with the advent of cloud technologies, we no longer need to concern ourselves too much with storing and allocating memory for our files on our computer's hard drives and risk diminishing their performance.

References

Bocharov, J. A., Zhang, G., Virdi, G., & Sood, V. (2008). U.S. Patent Application 12/262,593

Chang, J. M., & Gehringer, E. F. (1996). A high performance memory allocator for object-oriented systems. Computers, IEEE Transactions on, 45(3), 357-366.

Chen, F., Koufaty, D. A., & Zhang, X. (2009, June). Understanding intrinsic characteristics and system implications of flash memory based solid state drives. In ACM SIGMETRICS Performance Evaluation Review (Vol. 37, No. 1, pp. 181-192). ACM

Ciriani, V., Di Vimercati, S. D. C., Foresti, S., Jajodia, S., Paraboschi, S., & Samarati, P. (2007). Fragmentation and encryption to enforce privacy in data storage. In Computer Security–ESORICS 2007 (pp. 171-186). Springer Berlin Heidelberg.

Hua, B., Hua, H. L., Michel, D., & Xiong, W. (2010). U.S. Patent No. 7,739,422. Washington, DC: U.S. Patent and Trademark Office.

Huang, K. C. (2009, December). On Effects of Resource Fragmentation on Job Scheduling Performance in Computing Grids. In Pervasive Systems, Algorithms, and Networks (ISPAN), 2009 10th International Symposium on (pp. 701-705). IEEE.

Macêdo, A., Ferreira, T. B., & Matias, R. (2010, November). The mechanics of memory-related software aging. In Software Aging and Rejuvenation (WoSAR), 2010 IEEE Second International Workshop on (pp. 1-5). IEEE.

Pal, A., Sencar, H. T., & Memon, N. (2008). Detecting file fragmentation point using sequential hypothesis testing. digital investigation, 5, S2-S13

Peterson, Zachary Nathaniel Joseph. Data placement for copy-on-write using virtual contiguity. Diss. UNIVERSITY OF CALIFORNIA SANTA CRUZ, 2002.

Samanta, D. (2004). Classic data structures. PHI Learning Pvt. Ltd

Image Sources

http://thecreatorsproject.vice.com/blog/fields-ienergy-flowi-the-boundless-technological-landscapes-of-infinity

http://www.google.com/imgres?imgurl=http://stockarch.com/files/11/07/defragmentation.jpg&imgrefurl=http://stockarch.com/images/technology/file-fragmentation-3328&h=2008&w=3000&tbnid=ly20buNZcQiwEM:&zoom=1&docid=DBv8LeeQMUAn5M&ei=_PsZVfjHCMHgoASq4YCoBQ&tbm=isch&client=safari&ved=0CGIQMygyMDI

Layout by Henry Hammel