Contiguous Memory Allocation:

Contiguous memory allocation is a memory allocation method that allocates a single contiguous section of memory to a process or a file.

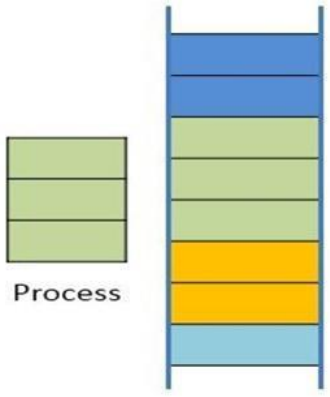
a) Contiguous memory allocation

In **contiguous memory allocation**, all the available memory space remains together in one place. It means freely available memory partitions are not scattered here and there across the whole memory space.

In the **contiguous memory allocation**, both the operating system and the user must reside in the main memory. The main memory is divided into two portions one portion is for the operating and other is for the user program.

In the **contiguous memory allocation** when any user process request for the memory a single section of the contiguous memory block is given to that process according to its need. We can achieve contiguous memory allocation by dividing memory into the fixed-sized partition.

A single process is allocated in that fixed sized single partition. But this will increase the degree of multiprogramming means more than one process in the main memory that bounds the number of fixed partition done in memory. Internal fragmentation increases because of the contiguous memory allocation.



Contiguous Memory Allocation

→ Fixed sized partition

In this method, the system divides memory into fixed size partition (may or may not be of the same size) here entire partition is allowed to a process and if there is some wastage inside the partition is allocated to a process and if there is some wastage inside the partition then it is called internal fragmentation.

Advantage: Management or book keeping is easy.

Disadvantage: Internal fragmentation

→ Variable size partition

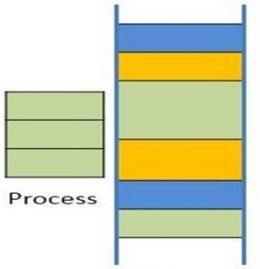
In the variable size partition, the memory is treated as one unit and space allocated to a process is exactly the same as required and the leftover space can be reused again.

Advantage: There is no internal fragmentation.

Disadvantage: Management is very difficult as memory is becoming purely fragmented after some time.

b) Non-contiguous memory allocation

In the non-contiguous memory allocation the available free memory space are scattered here , and all the free memory space is not at one place. So this is time-consuming. In the non contiguous memory allocation, a process will acquire the memory space but it is not at one place it is at the different locations according to the process requirement. This technique of non-contiguous memory allocation reduces the wastage of memory which leads to internal and external fragmentation. This utilizes all the free memory space which is created by a different process.



Noncontiguous
Memory Allocation

Non-contiguous memory allocation is of different types,

1. Paging
2. Segmentation
3. Segmentation with paging

i) Paging

A non-contiguous policy with a fixed size partition is called paging. A computer can address more memory than the amount of physically installed on the system. This extra memory is actually called virtual memory. Paging technique is very important in implementing virtual memory. Secondary memory is divided into equal size partition (fixed) called pages. Every process will have a separate page table. The entries in the page table are the number of pages of a process. At each entry either we have an invalid pointer which means the page is not in main memory or we will get the corresponding frame number. Size of a page table is generally very large so cannot be accommodated inside the PCB, therefore, PCB contains a register value PTBR(page table base register) which leads to the page table.

Advantages: It is independent of external fragmentation.

Disadvantages:

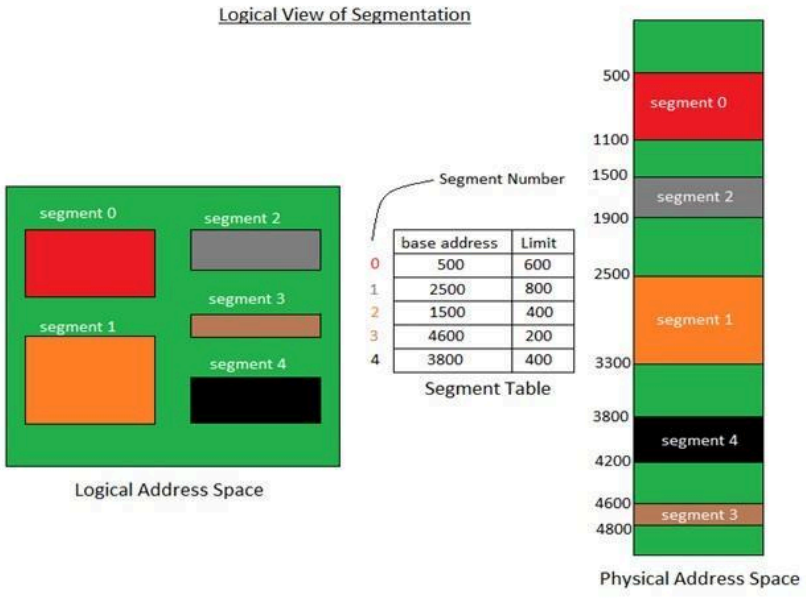
1. It makes the translation very slow as main memory access two times.
2. A page table is a burden over the system which occupies considerable space.

ii) Segmentation

Segmentation is a programmer view of the memory where instead of dividing a process into equal size partition we divided according to program into partition called segments. The translation is the same as paging but paging segmentation is independent of internal fragmentation but suffers from external fragmentation. Reason of external fragmentation is program can be divided into segments but segment must be contiguous in nature.

iii) Segmentation with paging

In segmentation with paging, we take advantages of both segmentation as well as paging. It is a kind of multilevel paging but in multilevel paging, we divide a page table into equal size partition but here in segmentation with paging, we divide it according to segments. All the properties are the same as that of paging because segments are divided into pages.

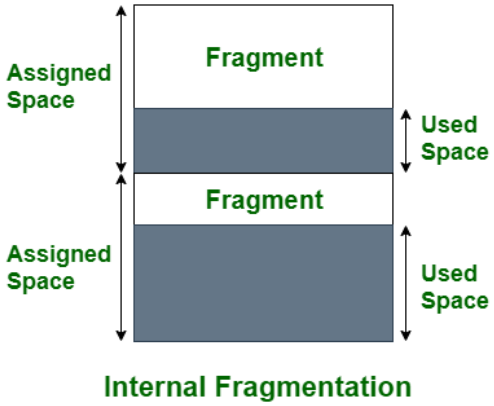


Difference between Internal and External fragmentation

There are two types of fragmentation in OS which are given as: Internal fragmentation, and External fragmentation.

Internal Fragmentation:

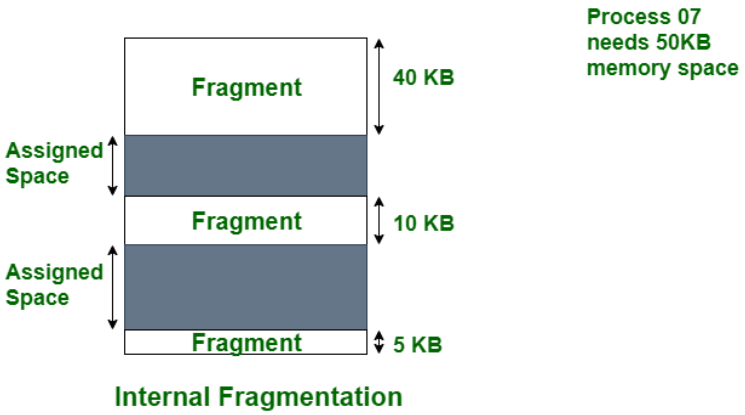
Internal fragmentation happens when the memory is split into mounted sized blocks. Whenever a method request for the memory, the mounted sized block is allotted to the method. Just in case the memory allotted to the method is somewhat larger than the memory requested, then the distinction between allotted and requested memory is that the internal fragmentation.



The above diagram clearly shows the internal fragmentation because the difference between memory allocated and required space or memory is called Internal fragmentation.

External Fragmentation:

External fragmentation happens when there's a sufficient quantity of area within the memory to satisfy the memory request of a method. however the process's memory request cannot be fulfilled because the memory offered is during a non-contiguous manner. Either you apply first-fit or best-fit memory allocation strategy it'll cause external fragmentation.



Internal Fragmentation

In above diagram, we can see that, there is enough space (55 KB) to run a process-07 (required 50 KB) but the memory (fragment) is not contiguous. Here, we use compaction, paging or segmentation to use the free space to run a process.

As processes are loaded and removed from memory, the free memory space is broken into little pieces. It happens after sometimes that processes cannot be allocated to memory blocks considering their small size and memory blocks remains unused. This problem is known as Fragmentation.