OPERATING SYSTEM

Lecture 2

Early Memory Management Systems

Dr Victoria Mukami

INTRODUCTION

This lecture is an introduction to early memory management systems. We will review the different types of memory allocation schemes: single-user systems, fixed partitions, dynamic partitions and relocatable dynamic partitions. Additionally, we review the types of memory allocation systems: best-fit and first-fit. Finally, we review two critical areas by reviewing the role of the memory list and the memory deallocation.

Learning objectives

By the end of this topic, you should be able to:

1. Identify the various memory management allocation schemes
2. Understand how memory allocation works in early systems
3. Identify the role of the memory list during allocation and deallocation

OVERVIEW

During the last lecture, we did a review of operating systems including a review of the evolution of operating systems while describing the functions of the operating systems. The evolution was a preamble of the lecture this week where we concentrate on how the early operating systems used to be able to manage memory. We review four types of memory allocation schemes and two types of memory systems. When we talk about memory allocation, we are looking at main memory or Random Access Memory.

MEMORY ALLOCATION SCHEMES

There are four memory allocation schemes. These are schemes from early operating systems and focused on how memory was allocated. These schemes include single-user schemes, fixed partitions, dynamic partitions, and relocatable dynamic partitions.

Single-User Contiguous Scheme [1]

The scheme was used in the earliest operating system. This scheme worked by loading the entire job into memory [1]. If the job or the program under process needed a specific amount of memory, then it was allocated the entire memory that is required. If the program was too large for the available memory, then it could not be loaded therefore execution couldn’t begin [1]. Each job was allocated contiguous space. Contiguous in this case means memory allocated is next to each other or is in sequence. The following is a diagram that explains how the allocation is done.

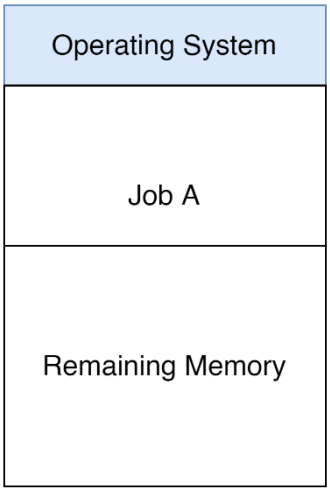

Figure 1: Single-user contiguous memory allocation scheme (Adapted [1]).

As shown in figure 1, the operating system reserved a specific space in memory. The incoming job is then allocated memory that will be used in its execution. One disadvantage of this scheme is that the memory could only hold one job or program at a given time. This meant that even with sufficient resources to support another job, only one job could be executed at a given time. As with the early operating system, this scheme did not multiprogramming [1].

Fixed Partitions Scheme

The fixed partitions scheme was developed to deal with the challenges of the single-user system which is a lack of multiprogramming. The fixed partition scheme allowed for multiprogramming where several jobs, so long as they could fit could be allocated memory resources [1]. This scheme used already created partitions to allocate a job space in memory. These partitions were known as fixed partitions [1] and were configured by the systems administrator. A job received the entirety of its partition, so long as it could fit within the space. If a job existed that was bigger than the biggest partition, the systems administrator would have to switch off the entire system and recreate the partitions [1]. As with the single-user allocation scheme, the operating system reserved specific space. Figure 2a shows the partition while Figure 2b shows the job allocation based on a 128 Mb memory.

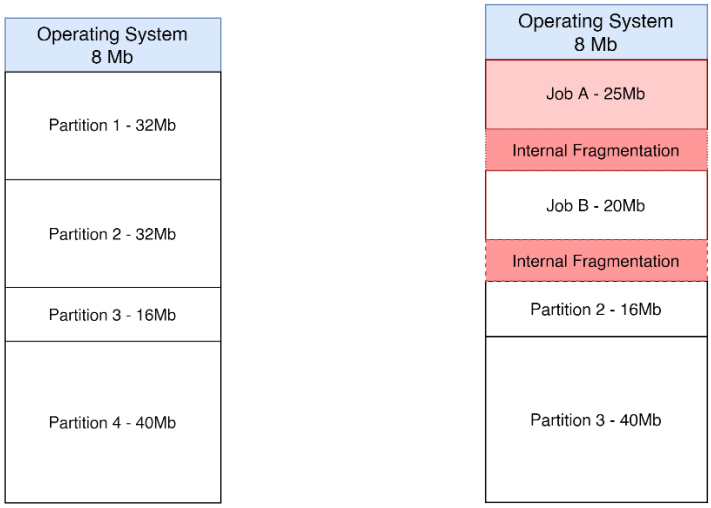

Figure 2: Partition creation, job allocation with internal Fragmentation

This scheme had two main disadvantages:

1. The memory needed to be partitioned every time a job was too big to fit into the partitions. This means that the system administrator needed to create optimum partition sizes that would allow for most jobs to be processed.
2. The system wasted a lot of space inside the partition. This was known as internal fragmentation [2] and it referred to the situation where a job would be fed into a partition so long as the partition was bigger than the job. This then left a situation where the space inside the partition was not used and led to memory wastage as shown in Figure 3.

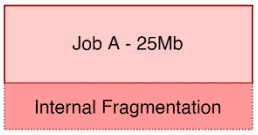

Figure 3: Internal Fragmentation

Job A is 25 Mb and was allocated to a 32Mb partition. This meant that the internal fragmentation is 7Mb.

Dynamic Partitions scheme

One of the biggest disadvantages of the fixed partitions scheme was the lack of optimum partitions that did not generate a lot of internal fragmentation. The dynamic partitions scheme was developed to deal with this challenge. The dynamic partition scheme used a somewhat similar system to the single-user system with a couple of

exceptions. First, space was allocated contiguously to each job. Second, since the system supported multiprogramming, several jobs could be executed at a go. The partitions created were of a variable length and number [2]. Allocation was done on the go as job needs arose. Figure 4 shows a step-by-step of the job allocation as they need resources.

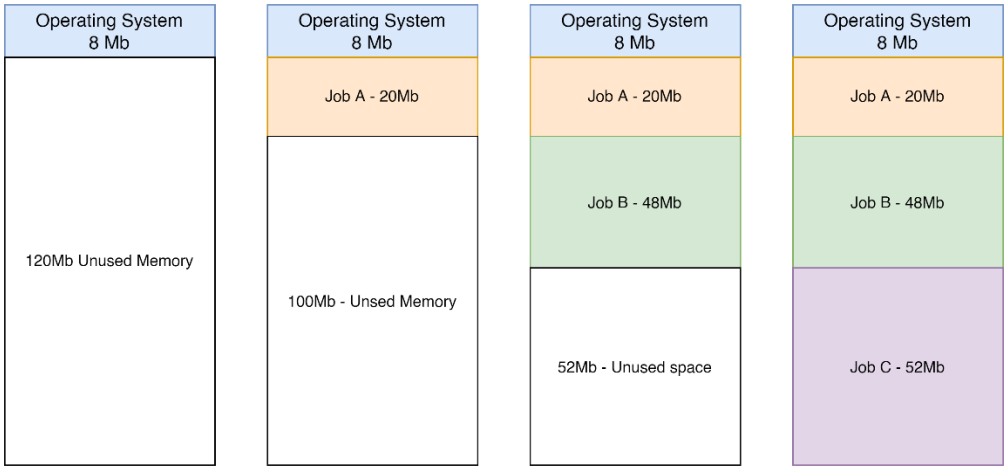

Figure 4: Dynamic partitions allocation

As seen in Figure 4, the entire memory was allocated three jobs (Job A, B and C). When the jobs are removed and swapped a new challenge is brought up and that is external fragmentation. Reviewing Figure 5 you can see that some space is left when Job B is swapped for D and when Job A is swapped for Job E. External fragmentation is memory wastage but, in this case, outside of the partition.

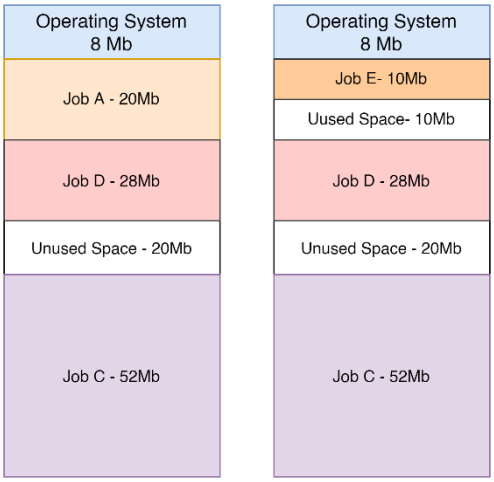

Figure 5: Swapping of Jobs (External Fragmentation)

If other smaller jobs are swapped once existing ones finish, then more external fragmentation will be created. It may reach a point where the existing partitions, cannot support the incoming jobs.

Relocatable Dynamic Partitions [2]

One way to the challenges of dynamic partitions is to perform **compaction.** Compaction is also referred to as defragmentation. Compaction is whereby the operating systems pick all the empty spaces, make the jobs memory to be contiguous and combines the unused space into a contiguous block as shown in Figure 6. This is drawn from Figure 5 swapping of jobs.

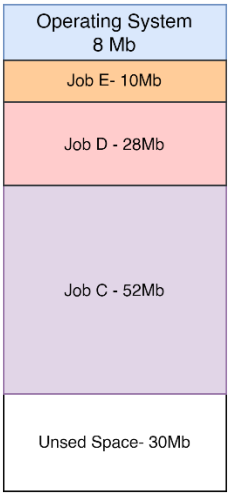

Figure 6: Compaction of Memory

Compaction can lead to a waste of the operating system as it must pause all jobs, compact, and then continue. If done too often then the computer overheads become high and if done too few times, then the process may not be useful. Compaction is also not an easy job and requires sufficient operating system resources.

MEMORY ALLOCATION SYSTEMS

Two main allocation systems were used within the early memory management systems. These allocation schemes are mainly used for the fixed partition and the dynamic partition scheme. These included the Best-Fit and the First-Fit system. We review these two systematically.

Best-Fit System

The best-fit system chooses the block that is closest in size to allocate the job [2]. The best fit will review based on a memory list of the number of free partitions. The memory list is a record of all partitions, their size and their status [1]. Based on this list, when an incoming job needs to be allocated a partition, the operating system will check which partitions fit which job best. Table 1 and Table 2 are a display of a job list and memory list allocation.

Table 1: Job List

Job Number	Resources requested
Job A	128Mb
Job B	300Mb
Job C	50Mb
Job D	256Mb

Table 2: Memory Allocation List

Memory Block size	Job Number	Job Size	Status	Internal Fragmentation
310Mb	Job B	300Mb	Busy	10Mb
220Mb	Job A	128Mb	Busy	92Mb
50Mb	Job C	50Mb	Busy	0
150Mb	--------	----------	Free	----------
Total: **730Mb**		**Total:** 478Mb		

As shown Job D has to wait until memory block 1 is freed for it to be processed. The best fit uses memory more efficiently as it reduces the amount of internal fragmentation. On the other hand, it is much slower to implement as the operating system needs to check against all available resources.

First – Fit System

The first-fit system chooses the first memory block that is available and can fit the incoming job. The first fit system will work on a job-by-job basis. Using a similar job list in Table 1 above, we will go ahead and allocate based on the first fit.

Table 3: Job List

Job Number	Resources requested
Job A	128Mb
Job B	300Mb
Job C	50Mb
Job D	256Mb

Table 4: Memory Allocation List

Memory Block size	Job Number	Job Size	Status	Internal Fragmentation
310Mb	Job A	128Mb	Busy	192Mb
220Mb	Job C	50Mb	Busy	170Mb
50Mb	--------	--------	Free	--------
150Mb	--------	--------	Free	--------
Total: **730Mb**		**Total:** 178Mb		

From the memory list, we can see that the scheme is not very efficient especially since the system could have supported additional jobs. The first-fit scheme ends up with a lot of internal fragmentation. This is one of the greatest challenges where memory is not used efficiently. A benefit is that it is a fast allocation scheme as it just checks for what is available and allocates it.

DEALLOCATION

This is the process by which memory is released after a job concludes being processed [1]. There are different techniques used depending on the type of allocation scheme used. For the fixed partition scheme, when the job execution is completed the memory list is set from busy to free and the job removed [1]. The dynamic system uses a complex algorithm like the compaction method discussed. It reviews whether the block being released is near another free block or next to busy blocks. This will then determine how compaction will be carried out.

SUMMARY

This lecture has been an introduction to early memory management systems. We have reviewed the different types of memory allocation schemes: single-user systems, fixed

partitions, dynamic partitions and relocatable dynamic partitions. Additionally, we reviewed the types of memory allocation systems: best-fit and first-fit. Finally, we reviewed two critical areas by reviewing the role of the memory list and the memory deallocation.

DISCUSSION TOPIC

Do web research and find a specific early operating system that used fixed partition systems. Identify what some challenges were especially based on its era.

REFERENCES

[1] McHoes, A., & Flynn, I., Understanding Operating Systems. Boston: Cengage Learning, 2018

[2] Stallings, W., Operating Systems: Internals and Design Principles. Harlow: Pearson Education Limited, 2018.