

HOST DISPATCHER DOCUMENTATION

Author: Ashwin Ramesh

Student ID: 311254012

1. Introduction

The second assignment in the Operating System Internals course focuses on the implementation of a hypothetical host dispatcher system. The main inspiration is to try and mimic the actual process dispatching mechanism used by the various operating systems in today's computers.

This design document will, in detail, explain the reasoning behind the memory allocation algorithm used, the various data structures created to make the dispatching algorithm work and possible issues stemming from this implementation and potential improvements in the next iteration of the program.

2. Memory Allocation Algorithms

The specification for the Host Dispatcher Assignment was to ensure that the memory allocation algorithm used was implemented using contiguous allocation. This means that memory is consecutively allocated to each process. The size of the process is compared to the amount of contiguous main memory remaining and if there is a block of memory, the process is placed in memory otherwise it is placed in queue awaiting memory. [1] This constraint resulted in a subset of memory allocation algorithms that could be implemented for this assignment.

2.a. First Fit

The First Fit Algorithm allocates the first large enough contiguous block of memory found to the process. The algorithm uses a linked list type data structure, where each block of memory is connected to its neighbours. It starts at the first block, iterating through each of the next blocks until it finds a large enough block of memory. If that block of memory is too large, it will be split into two separate blocks such that the extra memory portion is left free for other processes. [2]

The obvious advantages of the First Fit algorithm is that it allocates memory very quickly and manages to merge smaller, neighbouring blocks to become a larger block. However, as the algorithm always starts at the front of the linked list, smaller holes tend to accumulate at the beginning of memory, increasingly fragmenting the whole memory block. [3] This is also known as internal fragmentation.

2.b. Next Fit

As mentioned above, the First Fit algorithm has the downfall of heavy fragmentation at the beginning of the memory block. An improvement would be to leave the pointer to the last allocated location rather than moving back to the start of the list. This would better distribute the allocation process. This method of memory allocation is known as the Next Fit algorithm. However, the Next Fit algorithm does create the issue of external fragmentation, such that the memory allocation causes many small blocks of free memory to be scattered throughout the total memory space. This results in larger processes not being able to run due to a lack of contiguous memory. [4]

2.c. Best Fit

The Best Fit algorithm is much slower than both the First Fit and Next Fit algorithms. Instead of allocating to the first largest block of free contiguous memory, it will search through the whole memory space to find the smallest block which will fit the process memory required. [5] This means that the memory is efficiently allocated. The performance of this algorithm is the worst among the various algorithms described in this report.

2.d. Worst Fit

The Worst Fit algorithm will allocate the largest remaining block to the next process, which will allow future processes to use the remaining memory from that block. The idea is to maximise the split to increase the probability of allocating memory to other processes. [6] This reduces the chances of external fragmentation but is also very slow to run.

2.e. Buddy System

Unlike the various “Fit” systems described above, which use a linear linked list data structure to store contiguous memory blocks, the buddy system uses a binary tree type data structure to allocate memory between processes. The allocator will only assign blocks of specific sizes, and has many free lists one for each permitted size. Usually these sizes are powers of two.

If the requested memory size is not exactly available, then the algorithm will split a larger block in two again and again until a block of the correct size is created. The main advantage of this system is that merging (coalescence) of adjacent blocks is cheap since the neighbour of any free block can be calculated from its address.

This type of system can either react well to incoming processes or can cause severe internal fragmentation due to the rounding of block sizes. Typically, the process of freeing and merging of

memory is extremely fast, maxing out at $\log\left(\frac{\text{max. block size}}{\text{min. block size}}\right)$. [7]

2.f. Personal Choice

For this assignment, my personal choice was to use the First Fit Algorithm. Although the Buddy System does offer a better memory management scheme in the long run, it is far too complex for this type of assignment. On the other hand, the other “Fit” algorithms were not as efficient as the First Fit or resulted in a far more fragmented memory allocation. Finally, the First Fit algorithm is extremely easy to implement, allowing me to focus on the actual dispatcher and queuing system in more detail.

3. Data Structures used in the Host Dispatcher

3.a. Queue Structure

3.b. Memory Structure

3.c. Resource Structure

3.d. Dispatcher Structure

Program Structure and Modules**Issues with the Dispatching Scheme and Possible Improvements****References**

1. <http://www.techopedia.com/definition/3769/contiguous-memory-allocation>
2. <http://www.memorymanagement.org/articles/alloc.html>
3. <http://www2.cs.uregina.ca/~hamilton/courses/330/notes/memory/segmentation.html>
4. http://wiki.answers.com/Q/What_is_the_difference_between_external_and_internal_fragmentation
5. <http://courses.cs.vt.edu/csonline/OS/Lessons/MemoryAllocation/>
6. <http://research.cs.vt.edu/AVresearch/MMtutorial/WorstFit.php>
7. <http://www.memorymanagement.org/articles/alloc.html>
- 8.