Brian Bowles

Assignment 8

Memory Scheduling

April 1, 2015

Memory scheduling is how the computer determines what pages will be replaced if the current requested page of memory is not in the cache. There are four main algorithms for scheduling memory in such a way and they are as follows, simple first in first out, second chance, optimal, and least recently used, one of which will be examined here and that is FIFO, first in first out. The way FIFO works is the first page of memory in the cache is the first page to be replaced, take for example if a process is requested page two of memory and page one, and six were put in cache in that order, then page 1 would be replaced with page two. When a page gets added into the cache, whether it’s replaced or simply added, that is referred to as a page fault, or a cache miss and there are several factors to the number of page faults a program will experience, only one of which is the choice of algorithm.

Cache is where memory pages are placed when they are currently in use, the size of cache, which is directly related to the number of CPU’s, determines how many pages of memory can be used at a single time. Another thing to consider is the size of memory, if memory is small, for example two pages, then the likelihood of getting a page fault significantly decreases, especially if there are two CPU’s, then the least possible number of page faults is achieved. The least possible number of page faults that may be experience is the number of pages in memory, because when that memory is accessed and put into cache it is still a page fault, but if it’s never replaced and continually used then those will be the only page faults. Along with minimum amount of page faults, there is also a maximum and that is the number of times memory is being accessed, this usually will not occur.

When studying the FIFO algorithm I discovered that even if you have large memory the size of cache will help the number of page faults decrease. Also if you have a small cache size then the size of memory is the biggest determining factor in how many page faults will be experienced, if memory is large a large number of page faults will occur, if it’s small then a small number of page faults will occur. Another thing to keep in mind is the type of scheduling algorithm that is being used, if it’s round robin you will experience thrashing which is getting little to no work done, most of the time is spent in IO. I found that the biggest determine factor of thrashing and when it occurs is the time penalty, the larger the time penalty the sooner thrashing will occur. After implementing this algorithm one can see that it isn’t the best choice of the memory scheduling algorithms, but FIFO never is due to its simplicity.