CSC369H1 - Operating Systems
Thursday, December 5th, 2019
Shahzil Lakhani, Abhishek Kapoor
Prof. Karen Reid

README for A3

A recurring solution to a variety of OS problems seen in this course appears to be to divide physical resources into fixed-size “chunks”; this was a solution we used for File Systems as well (i.e. dividing the disk space into equal sized chunks known as data blocks). Now, with Virtual Memory, we saw how physical resources (like the RAM/memory) can be divided into equal sized chunks in order to add a layer of indirection and solve a variety of problems with memory allocation. In this assignment, we implemented various Page Eviction Algorithms - algorithms that are used to determine a “victim page” (i.e. a page to get removed from the physical memory and swapped into the swapfile). The following few sentences briefly compare several page eviction algorithms, including FIFO, Least-Recently Used (LRU), and Clock.

The three algorithms may be similar, but often perform differently. The FIFO algorithm has a low hit rate for simpleloop, possibly because the number of pages accessed within the loop of simpleloop may be large compared to the memory size of 8 and so pages that will be re-accessed in the next loop iteration are written to the swap file and subsequently need to be reloaded upon the next loop iteration. With matmul and especially blocked files, the FIFO algorithm has a much higher hit rate - this is likely because when computing products of matrices, there are many repeated instructions. Hence there may be less total distinct instructions, resulting in fewer pages having to be loaded to memory.

The LRU algorithm is slightly better than FIFO in most tests as shown by the greater hit rate. This may be because the LRU evicts the least recently used page. In the simpleloop for example, the algorithm may result in a lesser likelihood of certain pages being written to swap such as the page containing the loop counter increment instruction. In matmul and blocked files, it would be less likely that the page containing the multiplication formula gets written out to swap (since it is used often) thus resulting in a higher performance.

Finally, the clock algorithm has a quite similar hit rate to LRU for the most part, likely for the same reasons as LRU - pages that are used often enough, such as those containing repeated instructions likely get written out to swap less often. Hence LRU and Clock perform similarly, while FIFO performs worse by comparison of the hit rates.

CSC369H1 - Operating Systems
Thursday, December 5th, 2019
Shahzil Lakhani, Abhishek Kapoor
Prof. Karen Reid

In conclusion, the various page eviction algorithms we studied in CSC369 (FIFO, LRU, and Clock) all achieve a common goal, yet differ significantly in terms of performance - as can clearly be seen from running the different algorithms on different trace files. This assignment entailed thoroughly understanding virtual memory concepts, closely studying the various page eviction algorithms, and carefully implementing the algorithms - all of this has given us a deeper understanding of the performance and roles of page eviction algorithms, and how it fits in with the greater goals of the Operating System: to provide convenience and abstraction for the user.

Trace Files Output - Sample Traces

Algorithm	File(memsize)	Hit Rate	Hit Count	Miss Count	Overall Eviction Count	Clean Eviction Count	Dirty Eviction Count
FIFO	simpleloop(50)	22.4822	759	2617	2567	60	2507
	simpleloop(100)	23.7855	803	2573	2473	38	2435
	matmul(50)	62.9391	1913768	1126896	1126846	1104475	22371
	matmul(100)	64.3709	1957302	1083362	1083262	1071730	11532
	blocked(50)	99.8245	3507538	6166	6116	4142	1974
	blocked(100)	99.8817	3509549	4155	4055	2759	1296
LRU	simpleloop(50)	25.2073	851	2525	2475	14	2461
	simpleloop(100)	25.2073	851	2525	2425	14	2411
	matmul(50)	65.7665	1999739	1040925	1040875	1039933	942

CSC369H1 - Operating Systems
Thursday, December 5th, 2019
Shahzil Lakhani, Abhishek Kapoor
Prof. Karen Reid

	matmul(100)	66.9061	2034389	1006275	1006175	1005234	941
	blocked(50)	99.8618	3508848	4856	4806	2623	2183
	blocked(100)	99.8971	3510089	3615	3515	2587	928
CLOCK	simpleloop(50)	25.1481	849	2527	2477	15	2462
	simpleloop(100)	25.0889	847	2529	2429	16	2413
	matmul(50)	65.7663	1999733	1040931	1040881	1039937	944
	matmul(100)	65.7699	1999841	1040823	1040723	1039782	941
	blocked(50)	99.8616	3508841	4863	4813	2678	2135
	blocked(100)	99.8908	3509868	3836	3736	2584	1152

Trace file output - our traces

Algorithm	File(Memsize)	Hit Rate	Hit Count	Miss Count	Overall Eviction Count	Clean Eviction Count	Dirty Eviction Count
FIFO	trace1(8)	22.2222	10	35	27	21	6
	trace2(8)	75.0000	30	10	2	1	1
	trace3(8)	0.0000	0	36	28	25	3
LRU	trace1(8)	33.3333	15	30	22	18	4

CSC369H1 - Operating Systems
Thursday, December 5th, 2019
Shahzil Lakhani, Abhishek Kapoor
Prof. Karen Reid

	trace2(8)	75.0000	30	10	2	2	0
	trace3(8)	0.0000	0	36	28	25	3
CLOCK	trace1(8)	26.6667	12	33	25	20	5
	trace2(8)	75.0000	30	10	2	1	1
	trace3(8)	0.0000	0	36	28	25	3