CSC369 A3 (Task 3) by Tash-had Saqif and Labib Chowdhury

tr-simpleloop.ref

Memory Size = 50

	RAND	**FIFO**	**LRU**	**CLOCK**
Hit Rate	70.8236	71.0290	72.7993	72.7700
Hit Count	7241	7262	7443	7440
Miss Count	2983	2962	2781	2784
Overall Eviction Count	2933	2912	2731	2734
Clean Eviction Count	343	319	208	207
Dirty Eviction Count	2590	2593	2523	2527

tr-simpleloop.ref

Memory Size = 100

	RAND	**FIFO**	**LRU**	**CLOCK**
Hit Rate	72.9656	73.0243	73.7480	73.7285
Hit Count	7460	7466	7540	7538
Miss Count	2764	2758	2684	2686
Overall Eviction Count	2664	2658	2584	2586
Clean Eviction Count	180	165	119	121
Dirty Eviction Count	2484	2493	2465	2465

tr-matmul.ref

Memory Size = 50

	RAND	**FIFO**	**LRU**	**CLOCK**
Hit Rate	65.5236	60.9656	63.9450	63.9450
Hit Count	1,892,248	1,760,618	1,846,660	1,846,659
Miss Count	995,640	1,127,270	1,041,228	1,041,229
Overall Eviction Count	995,590	1,127,220	1,041,178	1,041,179
Clean Eviction Count	956,116	1,083,361	1,040,201	1,040,202
Dirty Eviction Count	39,474	43,859	977	977

tr-matmul.ref

Memory Size = 100

	RAND	**FIFO**	**LRU**	**CLOCK**
Hit Rate	88.8166	62.4796	65.1491	65.3104
Hit Count	2,564,925	1,804,340	1,881,433	1,886,092
Miss Count	322,963	1,083,548	1,006,455	1,001,796
Overall Eviction Count	322,863	1,083,448	1,006,355	1,001,696
Clean Eviction Count	315,564	1,061,347	1,005,395	1,000,735
Dirty Eviction Count	7299	22101	960	961

Discussion

From the tables presented above we can see that for simple loop, the clock and LRU algorithms were the best, with the LRU algorithm achieving just barely more hits than the Clock algorithm. This is as expected, since the Clock algorithm is meant to emulate the behaviour of LRU while sacrificing accuracy for simplicity (and less space usage if you remove the *next* and *prev* pointers from the frame structs for the clock algorithm). FIFO comes in last with the lowest hit rate, which is reasonable considering the fact that it does not use any data to help it make a decision. (We ignore the random algorithm here because it produces unpredictable behaviour).

For LRU, the odds of getting a clean eviction as opposed to a dirty one is higher due to the fact that we move any referenced frame to the head of our linked list (note that pages are also referenced when they are modified and in turn, made dirty). Since we evict only the last frame in our linked list, it is easy to see why this is since pages that are repeatedly written to are not repeatedly evicted. For clock, pages are only evicted when they have a reference bit of zero, which minimizes dirty evictions, so there's a lower chance of the page being dirty when evicted. FIFO on the other hand, sticks to a predefined rule (first in, first out) which, as mentioned previously, does not take into whether or not pages are being frequently referenced. So, it is indeed very possible for a page to be evicted dirty several time than need be.

Custom Tracefile Results

trace1.ref

	FIFO	**LRU**	**CLOCK**	**Optimal**
Hit Rate	26.6667	30	36.6667	50
Hit Count	8	9	11	15
Miss Count	22	21	19	15

trace2.ref

	FIFO	**LRU**	**CLOCK**	**Optimal**
Hit Rate	75	75	75	75
Hit Count	24	24	24	24
Miss Count	8	8	8	8

trace3.ref

	FIFO	**LRU**	**CLOCK**	**Optimal**
Hit Rate	0	0	0	0
Hit Count	0	0	0	0
Miss Count	48	48	48	48