CSCC69 Assignment2: Qi Lin(linqi9), Jialiang Lin(linjial2) Tianji Liu(liutia59)

Date: 2020-02-29

Table Report

1) SimpleLoop

M = 50

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	71.0834	7296	2968	2918	320	2598
LRU	72.9053	7483	2781	2731	205	2526
CLOCK	72.8956	7482	2782	2732	204	2528
OPT	74.0062	7596	2668	2618	110	2508
RAND	71.2295	7311	2953	2903	322	2581

M = 100

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	73.1781	7511	2753	2653	158	2495
LRU	73.8991	7585	2679	2579	113	2466
CLOCK	73.8796	7583	2681	2581	114	2467
OPT	74.3083	7627	2637	2537	38	2499
RAND	73.1586	7509	2755	2655	161	2494

M = 150

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	73.5776	7552	2712	2563	129	2433
LRU	73.9088	7586	2678	2528	112	2416
CLOCK	73.8893	7584	2680	2530	112	2418
OPT	74.3083	7627	2637	2487	2	2485
RAND	73.6068	7555	2709	2559	134	2425

M = 200

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	73.6555	7560	2704	2504	125	2379
LRU	73.9088	7586	2678	2478	112	2366
CLOCK	73.8991	7585	2679	2479	112	2367
OPT	74.3083	7627	2637	2437	2	2435
RAND	73.6458	7559	2705	2505	130	2375

2) Blocked

M = 50

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	99.7322	2411708	6476	6426	4299	2127
LRU	99.7844	2412971	5213	5163	2944	2219
CLOCK	99.7821	2412915	5269	5219	3002	2217
OPT	99.8438	2414407	3777	3727	2764	963
RAND	99.6562	2409871	8313	8263	5872	2391

M = 100

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	99.8208	2413850	4334	4234	2876	1358
LRU	99.8436	2414403	3781	3681	2720	961
CLOCK	99.8339	2414167	4017	3971	2731	1186
OPT	99.8658	2414938	3246	3146	2203	943
RAND	99.7839	2412959	5225	5124	3501	1624

M = 150

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	99.8254	2413962	4222	4072	2770	1302
LRU	99.8443	2414419	3765	3615	2674	941
CLOCK	98.8372	2414246	3938	3788	2691	1097
OPT	99.8951	2415648	2536	2386	1441	945
RAND	99.8161	2413737	4447	4297	2902	1395

M = 200

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	99.8688	2415012	3172	2972	1995	977
LRU	99.8473	2414492	3692	3492	2551	941
CLOCK	99.8675	2414979	3205	3005	2063	942
OPT	99.9049	2415884	2300	2100	1149	951
RAND	99.8404	2414324	3860	3660	2436	1224

3) Matmul

M = 50

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	60.966	1760686	1127266	1127216	1083352	43864
LRU	63.9459	1846728	1041224	1042202	1041224	978
CLOCK	63.9453	1846709	1041243	1042228	1041243	985
OPT	79.2528	2288782	599170	559120	598154	966
RAND	65.5241	1892306	995646	995596	956401	39195

M = 100

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	62.4806	1804410	1083542	1093442	1061337	22105
LRU	65.1500	1881502	1006450	1006350	1005389	961
CLOCK	63.9533	1846940	1041012	1040912	1039949	963
OPT	96.4191	2784536	103416	103316	102352	964
RAND	88.7958	2564381	323571	323471	316133	7338

M = 150

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	98.8087	2853547	34405	34255	33059	1196
LRU	98.8614	2855069	32883	92733	91772	961
CLOCK	98.8503	2854748	33204	33054	32090	964
OPT	99.0081	2859306	28646	28496	27534	962
RAND	96.6816	2792128	95824	95674	93427	2247

M = 200

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	98.8267	2854068	33884	33684	32549	1135
LRU	98.8618	2855081	32871	32671	31710	961
CLOCK	98.8609	2855055	32897	32697	31736	961
OPT	99.1874	2864485	23467	23267	22337	930
RAND	98.0349	2831200	56752	56552	55046	1506

4) Fourth Algorithm (avlinsertion.c created by Jialiang Lin on 2019-02-29)

M = 50

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	95.6548	6428	292	242	194	48
LRU	96.8899	6511	209	159	145	14
CLOCK	96.6667	6496	224	174	152	22
OPT	97.7976	6572	148	98	92	6
RAND	95.9524	6448	272	222	184	38

M = 100

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	97.8720	6577	143	43	34	9
LRU	98.1399	6595	125	25	22	3
CLOCK	98.0655	6590	130	30	26	4
OPT	98.1399	6595	125	25	25	0
RAND	98.0060	6586	134	34	31	3

M = 150

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	98.1399	6595	125	0	0	0
LRU	98.1339	6595	125	0	0	0
CLOCK	98.1399	6595	125	0	0	0
OPT	98.1399	6595	125	0	0	0
RAND	98.1399	6595	125	0	0	0

M = 200

	Hit Rate	Hit count	Miss Count	overall eviction count	Clean eviction count	Dirty eviction count
FIFO	98.1399	6595	125	0	0	0
LRU	98.1339	6595	125	0	0	0
CLOCK	98.1399	6595	125	0	0	0
OPT	98.1399	6595	125	0	0	0
RAND	98.1399	6595	125	0	0	0

Algorithm Comparison

According to the result shown in the table, the hit rate are ranked in this order:

FIFO < CLOCK <(or ≈) LRU < OPT

1. FIFO algorithm has the lowest hit rate and it is picking the page table entry that is first allocated in frames for eviction. It evicts frames that are oldest, but it doesn't care about the access time of each frame. Hence, it is ignoring the usage pattern of frames and treating all the frames equally in terms of their access time(A page that is referenced frequently is more likely to be referenced again). Thus it has a high miss rate and it is a bad replacement rule.

2. LRU algorithm has a hit rate that is very close to CLOCK's hit rate. Because CLOCK algorithm (second chance algorithm) is a combination of FIFO and LRU. The second chance gives the oldest pages that are recently visited a second chance to stay on the list. While LRU always picks the page which has not been referenced for the longest period as a victim. Because both algorithms take 'recentness of access time' as main factor in their consideration and thus they share a similar hit rate.

3. OPT algorithm is first read all the lines from the trace file as a linked list object with their address being stored. Whenever the reference function is being called, we will take a step forward from the current node of the linked list to represent a trace line in the trace file is being taken. After that, we are going to go through the linked list object to find the next trace action with the same address in the trace file. If such a trace line exists, we store the distance between the current action and the next address usage. Those distances are being used again in the evict function, where we chose the frame with the largest distance from its next usage with the current page table address. This algorithm has the highest hit rate because the frame with the largest distance from its next usage implies has the largest number of action calls in between the current frame address call and the next. Thus at the moment, we are actually picking the frame with an address that will be called in the furthest future(according to the trace file and current distance state, this frame is the frame that is going to be visited last in the future). This has a higher hit rate than the rest of all, because unlike others, by reading the trace

file of the program, we are kind of predicting the future of memory access sequence, hence we can pick the frame that is likely to the optimal choice to evict.

4. Random algorithm is very close to the OPT algorithm in the MATMUL test because MATMUL is created randomly and thus FIFO, LRU, and CLOCK all did badly when the number of simulated memory is small. However, the random algorithm is not competitive in terms of hit rate for the rest of the test. Generally, if one page has not been viewed for the longest time, it is likely that it is not going to be referenced in a short period. Therefore, FIFO, LRU, and CLOCK all find reasonable victims compared to the Random algorithm.

LRU Implementation and Comparison (Memory increase)

For the LRU implementation, we are implementing a linked list structure to represent each accessed frame. Whenever the reference function is being called, we are going through the linked list and if the input page table entry' frame number is not being found in the linked list, we will create a new node to represent the input frame and inserted on the front of the linked list. On the other hand, if the frame is being accessed before(meaning found in the list and still in use), then we are going to move that node representation to the front of the linked list. In this way, we can ensure the last node in the linked list is the node representation of the frame that is least recently visited. So, we can pick the last node as a victim frame to evict. This works well most of the time because the least recently used is usually less likely to be accessed again. However, if we compare it with OPT, then the hit rate of it is lower than OPT. This is due to the fact that the least recently visited is not necessary to be the frame that is the optimal choice for the victim. Since LRU does not predict the future, the following unexpected case could happen: the most recently evicted frame could be the frame that is going to be accessed next and increase the evicted count.

From the table above (blocked, matmul, and simple loop), we can see that the hit rate increases while total eviction decreases as the size of memory goes larger. For the number of eviction, it decreases because when the memory becomes larger, it allows us to store more frames. We are more likely to have frame already on the list when we have a larger memory. Therefore, the number of overall evictions decrease. Talking about the hit rate, according to the lecture, the least recently accessed frame address is less likely to be referenced again. With the help of larger memory size, the time of the least recently visited frame can stay in the memory longer. When the time period is long enough, we are more likely to find the address that is not being referenced recently. Therefore, we have a higher hit rate when the memory size becomes larger.