Statistics: simpleloop.c

	Hit Rate	Hit Cnt	Miss Cnt	Eviction Cnt	Clean Evi	Dirty Evi
	Memory Size = 50					
RAND	71.0974%	7360	2992	2942	285	2657
FIFO	71.0781%	7358	2994	2944	274	2670
Exact LRU	73.0197%	7559	2793	2743	96	2647
CLOCK	72.9328%	7550	2802	2752	102	2650
OPT	74.1886%	7680	2672	2622	21	2601
	Memory Size = 100					
RAND	73.2805%	7586	2766	2666	79	2587
FIFO	73.3192%	7590	2762	2662	70	2592
Exact LRU	74.0340%	7664	2688	2588	2	2586
CLOCK	74.0147%	7662	2690	2590	4	2586
OPT	74.4494%	7707	2645	2545	0	2545
	Memory Size = 150					
RAND	73.6573%	7625	2727	2577	39	2538
FIFO	73.7249%	7632	2720	2570	32	2538
Exact LRU	74.0533%	7666	2686	2536	0	2536
CLOCK	74.0437%	7665	2687	2537	0	2537
OPT	74.4494%	7707	2645	2495	0	2495
	Memory Size = 200					
RAND	73.8698%	7647	2705	2505	22	2483
FIFO	73.8022%	7640	2712	2512	24	2488
Exact LRU	74.0533%	7666	2686	2486	0	2486
CLOCK	74.0533%	7666	2686	2486	0	2486
OPT	74.4494%	7707	2645	2445	0	2445

Statistics: matmul.c

	Hit Rate	Hit Cnt	Miss Cnt	Eviction Cnt	Clean Evi	Dirty Evi
	Memory Size = 50					
RAND	65.5388%	1892786	995254	995204	975056	20148
FIFO	60.9668%	1760746	1127294	1127244	1104706	22538
Exact LRU	63.9467%	1846805	1041235	1041185	1040078	1107
CLOCK	63.9466%	1846802	1041238	1041188	1040080	1108
OPT	76.5139%	2209752	678288	678238	677148	1090
	Memory Size = 100					
RAND	88.7962%	2564470	323570	323470	319073	4397
FIFO	62.4814%	1804488	1083552	1083452	1071789	11663
Exact LRU	65.1508%	1881580	1006460	1006360	1005278	1082
CLOCK	65.3122%	1886242	1001798	1001698	1000616	1082
OPT	92.9262%	2683746	204294	204194	203114	1080
	Memory Size = 150					
RAND	96.6742%	2791989	96051	95901	94104	1797
FIFO	98.8084%	2853627	34413	34263	33115	1148
Exact LRU	98.8611%	2855149	32891	32741	31659	1082
CLOCK	98.7982%	2853332	34708	34558	33476	1082
OPT	99.0177%	2859671	28369	28219	27143	1076
	Memory Size = 200					
RAND	98.0408%	2831457	56583	56383	54975	1408
FIFO	98.8265%	2854148	33892	33692	32552	1140
Exact LRU	98.8615%	2855161	32879	32679	31597	1082
CLOCK	98.8611%	2855148	32892	32692	31610	1082
OPT	99.2722%	2867021	21019	20819	19743	1076

Statistics: blocked.c

	Hit Rate	Hit Cnt	Miss Cnt	Eviction Cnt	Clean Evi	Dirty Evi
	Memory Size = 50					
RAND	99.6536%	2409768	8376	8326	5951	2375
FIFO	99.7308%	2411634	6510	6460	4318	2142
Exact LRU	99.7838%	2412917	5227	5177	2827	2350
CLOCK	99.7617%	2412382	5762	5712	3292	2420
OPT	99.8283%	2413991	4153	4103	2698	1405
	Memory Size = 100					
RAND	99.7848%	2412939	5205	5105	3448	1657
FIFO	99.8203%	2413799	4345	4245	2801	1444
Exact LRU	99.8433%	2414354	3790	3690	2607	1083
CLOCK	99.8192%	2413772	4372	4272	2625	1647
OPT	99.8695%	2414988	3156	3056	1986	1070
	Memory Size = 150					
RAND	99.8190%	2413768	4376	4226	2787	1439
FIFO	99.8251%	2413914	4230	4080	2679	1401
Exact LRU	99.8440%	2414371	3773	3623	2560	1063
CLOCK	99.8435%	2414359	3785	3635	2572	1063
OPT	99.8934%	2415566	2578	2428	1364	1064
	Memory Size = 200					
RAND	99.8420%	2414324	3820	3620	2311	1309
FIFO	99.8685%	2414964	3180	2980	1892	1088
Exact LRU	99.8470%	2414444	3700	3500	2437	1063
CLOCK	99.8671%	2414931	3213	3013	1939	1074
OPT	99.9037%	2415816	2328	2128	1071	1057

Statistics: test.c (own program)

	Hit Rate	Hit Cnt	Miss Cnt	Eviction Cnt	Clean Evi	Dirty Evi
	Memory Size = 50					
RAND	95.7696%	720103	31809	31759	20125	11634
FIFO	95.9437%	721412	30500	30450	18800	11650
Exact LRU	97.2494%	731230	20682	20632	10336	10296
CLOCK	97.2229%	731031	20881	20831	10532	10299
OPT	97.3002%	731612	20300	20250	9991	10259
	Memory Size = 100					
RAND	96.6951%	727062	24850	24750	13835	10915
FIFO	96.7311%	727333	24579	24479	13563	10916
Exact LRU	97.2593%	731304	20608	20508	10222	10286
CLOCK	97.2574%	731290	20622	20522	10231	10291
OPT	97.3087%	731676	20236	20136	9929	10207
	Memory Size = 150					
RAND	96.9132%	728702	23210	23060	12328	10732
FIFO	96.9350%	728866	23046	22896	12205	10691
Exact LRU	97.2594%	731305	20607	20457	10171	10286
CLOCK	97.2591%	731303	20609	20459	10173	10286
OPT	97.3154%	731726	20186	20036	9881	10155
	Memory Size = 200					
RAND	97.0092%	729424	22488	22288	11656	10632
FIFO	97.0244%	729538	22374	22174	11552	10622
Exact LRU	97.2601%	731310	20602	20402	10117	10285
CLOCK	97.2591%	731303	20609	20409	10123	10286
OPT	97.3220%	731776	20136	19936	9831	10105

Why choose this program:

This program allocates space both in heap and in stack. For demonstration purposes, I set the space allocated each time to 4096 bytes, same as page size. Then, the program keeps allocating space in heap and stack, meanwhile fetching instructions from code.

Comparison of Algorithms:

- RAND

A trivial algorithm that is easy to implement. Relatively poor performance.

- OPT

For a given set of program and memory size, we can clearly see that OPT page replacement algorithm outperforms all other algorithms. Actually, the OPT algorithms gives a theoretical upper bound for page replacement algorithms. Only when given the memory traces can we implement such an algorithm, otherwise it merely serves as a benchmark.
Our implementation takes O(nlogn) to process the traces, where n is the length of the trace.

- FIFO

As we can observe, FIFO algorithm outperforms RAND algorithm most of the time. However, it can sometimes be observed that it suffers from Belady's anlmaly, that is, as the memory sizes increases, the hit rate declines.

- Exact LRU

LRU algorithm also outperforms RAND algorithm most of the time. However, no Belady's anomaly has been observed for the given parameters.

- CLOCK

CLOCK algorithm is an variation of FIFO. Instead of remembering the time of which page was first swapped in, it determines that a page should be swapped out if it hasn't been referenced for a very long time. It also outperforms RAND algorithm in most cases.

What happens in LRU as memory increases?

As we can observe, the hit rate of LRU increases as memory size increases, a.k.a. it does not suffer from Belady’s anomaly. In contrast, the FIFO performs differently, as memory increase undermines hit rates in some cases.

This is probably because LRU is an “stack algorithm”, which guarantees better performance as memory increases.