Part I: Performance Table:

- **Program of our choice:**

We choose to use a depth first search program as the fourth program. The program is used to tell if a Graph is connected. It shows interesting memory usage pattern, such that each new value is calculated recursively before been assigned to specific array location. It tends to first execute multiple I and S operations, and then execute M operation periodically after the S and I operations. Although the total reference count is as large as 24256, yet there are only 122 distinct virtual pages, which is calculated from the exercise program wrote in tutorial practice. The virtual memory usage pattern is interesting, as the hit rate will keep stationary after the memsize exceed 122.

SIMPLELOOP								
rand.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	7239	3001	366	2585	10240	70.6934	29.3066
	100	7477	2763	169	2494	10240	73.0176	26.9824
	150	7526	2714	137	2427	10240	73.4961	26.5039
	200	7530	2710	135	2375	10240	73.5352	26.4648
fifo.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	7274	2966	321	2595	10240	71.0352	28.9648
	100	7484	2756	162	2494	10240	73.0859	26.9141
	150	7524	2716	134	2432	10240	73.4766	26.5234
	200	7532	2708	130	2378	10240	73.5547	26.4453
clock.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	7438	2802	222	2530	10240	72.6367	27.3633
	100	7552	2688	121	2467	10240	73.75	26.25
	150	7557	2683	117	2416	10240	73.7988	26.2012
	200	7557	2683	117	2366	10240	73.7988	26.2012
lru.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	7457	2783	207	2526	10240	72.8223	27.1777
	100	7556	2684	119	2465	10240	73.7891	26.2109
	150	7558	2682	117	2415	10240	73.8086	26.1914

	200	7558	2682	117	2365	10240	73.8086	26.1914
opt.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	7572	2668	111	2507	10240	73.9453	26.0547
	100	7598	2642	42	2500	10240	74.1992	25.8008
	150	7598	2642	2	2490	10240	74.1992	25.8008
	200	7598	2642	2	2440	10240	74.1992	25.8008
BLOCKED								
rand.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	2409621	8515	6044	2421	2418136	99.6479	0.3521
	100	2412821	5315	3590	1625	2418136	99.7802	0.2198
	150	2413804	4332	2849	1333	2418136	99.8209	0.1791
	200	2414329	3807	2410	1197	2418136	99.8426	0.1574
fifo.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	2411656	6480	4305	2125	2418136	99.7320	0.2680
	100	24137991	4337	2880	1357	2418136	99.8206	0.1794
	150	2413910	4226	2775	1301	2418136	99.8252	0.1748
	200	2414960	3176	2000	976	2418136	99.8687	0.1313
clock.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	2409621	5263	3001	2212	2418136	99.7824	0.2176
	100	2414098	4038	2744	1194	2418136	99.833	0.167
	150	2414193	3943	2697	1096	2418136	99.8369	0.1631
	200	2414946	3190	2048	942	2418136	99.8681	0.1319
lru.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	2412918	5218	2948	2220	2418136	99.7842	0.2158
	100	2414350	3786	2726	960	2418136	99.8434	0.1566
	150	2414367	3769	2679	940	2418136	99.8441	0.1559

	200	2414440	3696	2556	940	2418136	99.8472	0.1528
opt.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	2414427	3709	2699	960	2418136	99.8466	0.1534
	100	2415126	3010	1961	949	2418136	99.8755	0.1245
	150	2415608	2528	1427	951	2418136	99.8955	0.1045
	200	2415858	2278	1136	942	2418136	99.9058	0.0942
MATMUL								
rand.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	1892034	995878	956483	39345	2887912	65.5156	34.4844
	100	2563681	324231	316810	7321	2887912	88.7728	11.2272
	150	2791474	96438	94061	2227	2887912	96.6606	3.3394
	200	2831383	56529	54811	1518	2887912	98.0426	1.9574
fifo.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	1760646	1127266	1083356	43860	2887912	60.9661	39.0339
	100	1804361	1083551	1061344	22107	2887912	62.4798	37.5202
	150	2853503	34409	33064	1195	2887912	98.8085	1.1915
	200	2854024	33888	32554	1134	2887912	98.8266	1.1734
clock.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	1846651	5263	1040227	984	2887912	63.9442	36.0558
	100	1881457	4038	1005395	960	2887912	65.1494	34.8506
	150	2854705	3943	32094	963	2887912	98.8501	1.1499
	200	2855011	3190	31741	960	2887912	98.8607	1.1393
lru.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	1846683	1041229	1040200	979	2887912	63.9453	36.0547
	100	1881457	1006455	1005395	960	2887912	65.1494	34.8506
	150	2855026	32886	31776	960	2887912	98.8613	1.1387

	200	2855037	32875	31715	960	2887912	98.8616	1.1384
opt.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	2300456	587456	586444	962	2887912	79.6581	20.3419
	100	2795115	92797	91737	960	2887912	96.7867	3.2133
	150	2861298	26614	25504	960	2887912	99.0784	0.9216
	200	2868648	19264	18104	960	2887912	99.3329	0.6671
Depth First Search								
rand.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	23933	323	202	71	24256	98.6684	1.3316
	100	24107	149	32	17	24256	99.3857	0.6143
	150	24134	122	0	0	24256	99.4970	0.5030
	200	24134	122	0	0	24256	99.4970	0.5030
fifo.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	23973	283	175	58	24256	98.8333	1.1667
	100	24108	148	35	13	24256	99.3898	0.6102
	150	24134	122	0	0	24256	99.4970	0.5030
	200	24134	122	0	0	24256	99.4970	0.5030
clock.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	24024	232	138	44	24256	99.0435	0.9565
	100	24119	137	30	7	24256	99.4352	0.5648
	150	24134	122	0	0	24256	99.4970	0.5030
	200	24134	122	0	0	24256	99.4970	0.5030
lru.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	24043	213	121	42	24256	99.1219	0.8781
	100	24129	127	25	2	24256	99.4764	0.5236
	150	24134	122	0	0	24256	99.4970	0.5030

	200	24134	122	0	0	24256	99.4970	0.5030
opt.c	**Memsize**	**Hit Count**	**Miss Count**	**Clean Evictions**	**Dirty Evictions**	**Total Reference**	**Hit Rate**	**Miss Rate**
	50	24091	165	91	24	24256	99.3198	0.6802
	100	24134	122	22	0	24256	99.4970	0.5030
	150	24134	122	22	0	24256	99.4970	0.5030
	200	24134	122	22	0	24256	99.4970	0.5030

Part II: Performance Comparison:

Five eviction algorithms were tested, and the performance were shown in the above table. The algorithms are: Random, First In First Out, Clock, Least Recently Used, and Optimal. In our experiment, all algorithms tends to have better performance as the memsize increasing. In other words, Belady's Anomaly is not observed for all algorithms, including FIFO. With random algorithm as the benchmark, there is a clear trend that the hit rate performance of the FIFO algorithm tends to be higher than Random. The LRU algorithm has slightly better performance than Clock algorithm, while they both outperform Random and FIFO. Meanwhile, the Optimal algorithm consistently has the best performance all the time.

It's worth noticing that, although random tends to have the lower bound performance, it's not guaranteed. For example, when experimenting with Matmul trace with memsize=100, the Random algorithm has the 2nd best performance, which is only worse than the Optimal algorithm.

Part III: LRU Performance Analysis:

In all above experiments with four distinct trace file, the performance of LRU increased consistently with the increasing memory size. In our implementation, exact LRU was implemented by keeping a stack of pages been used. For any page in memory, it will be moved to the top of the stack on reference. With the above described approach, the LRU is likely to keep most frequently used pages in the memory, so that to well exploit the locality of the progress's memory usage. As a result, with increasing memory size, the most frequently used pages are likely to stay longer in the memory, and the fault rate of LRU will decrease.