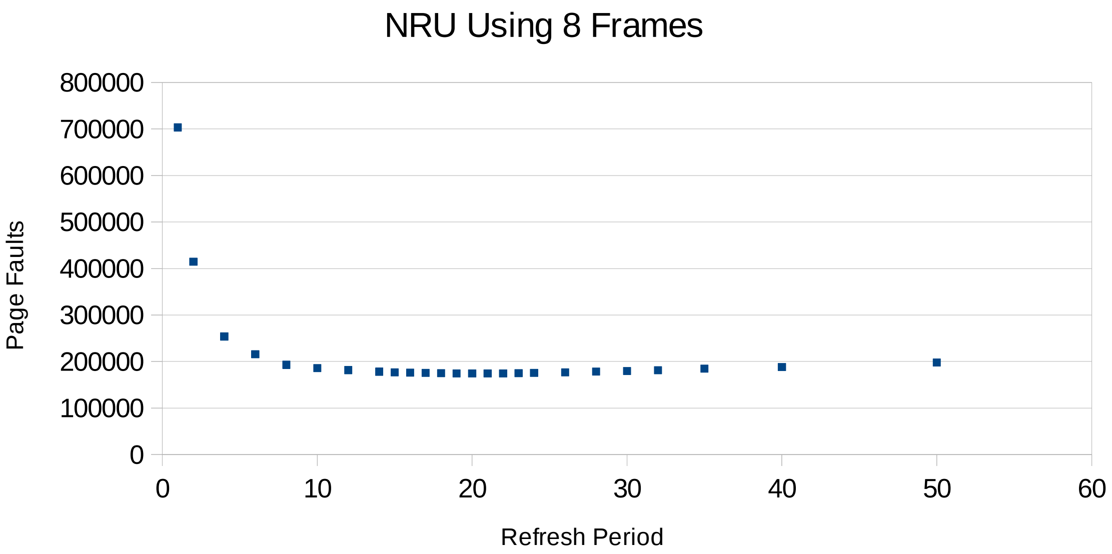
Forbes Turley
CS 1550
Project 3



Shown above is the graph depicting the number of page faults when using 8 frames in NRU on the gcc.trace file, versus the refresh period. The absolute minimum refresh time was found to be 21. That is, the referenced bits of all pages in the table are reset every 21 memory operations. Note that this was the minimum found for 8 frames. This value is not the minimum for higher numbers of frames. The higher the number of frames, the higher the optimal refresh period. In fact, by the time 64 frames is reaches, the optimal refresh period has climbed to approximately 500. This is because the more frames present, the greater the number of memory operations that can occur before the table starts to become saturated with referenced pages, which was the impetus for implementing a refresh period in NRU.

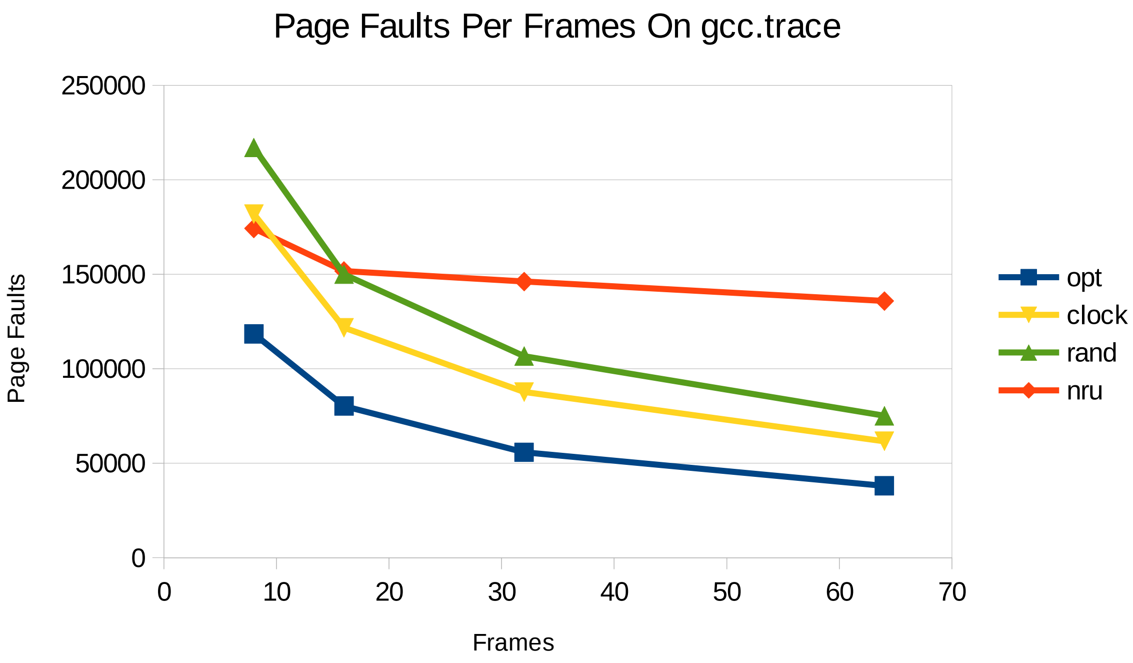
With a refresh period of 21 chosen to be utilized across 8,16,32, and 64 frames, NRU really performs quite poorly. These number can be significantly enhanced by finding the optimal refresh period for each frame count, but this was not what the project description specified.

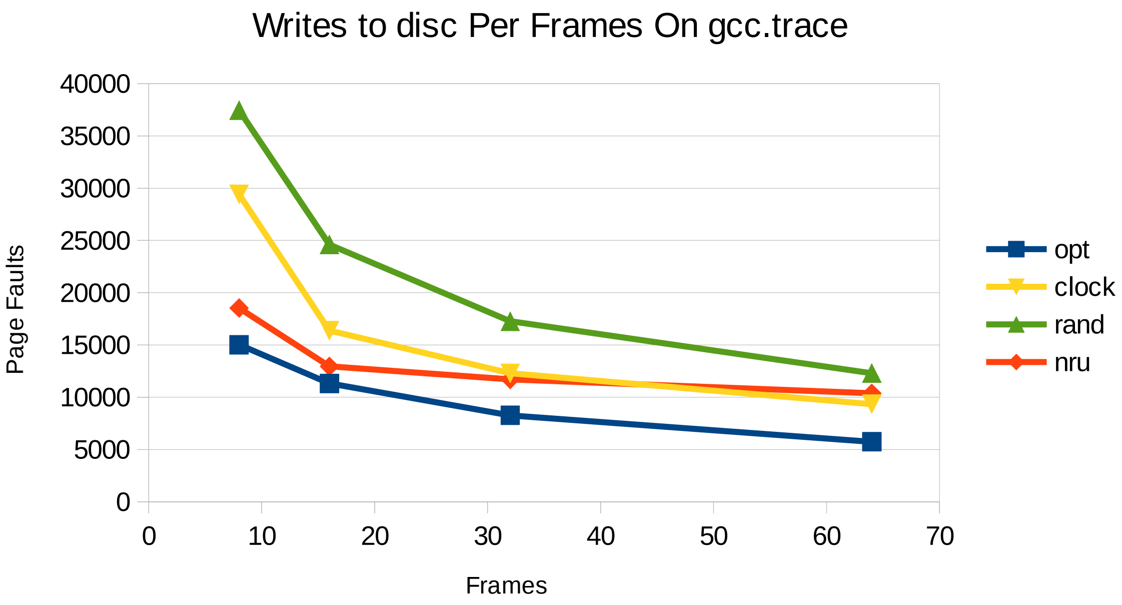
Based on the numbers given in the table below, it is obvious that opt is the best algorithm by both page faults and writes to disc under every testing condition. However, obviously opt could not be implemented on a real computer since it requires knowledge of the future. **Therefore, it would appear that clock is the best algorithm that could be implemented on a real computer.** Clock outperforms rand under every tested condition for both gcc and swim, and is better than NRU except when using only 8 frames. This is because NRU’s refresh period was optimized using 8 frames. If NRU were to have it’s optimal refresh period calculated for every number of frames, it may be the case that NRU would more consistently outperform clock.

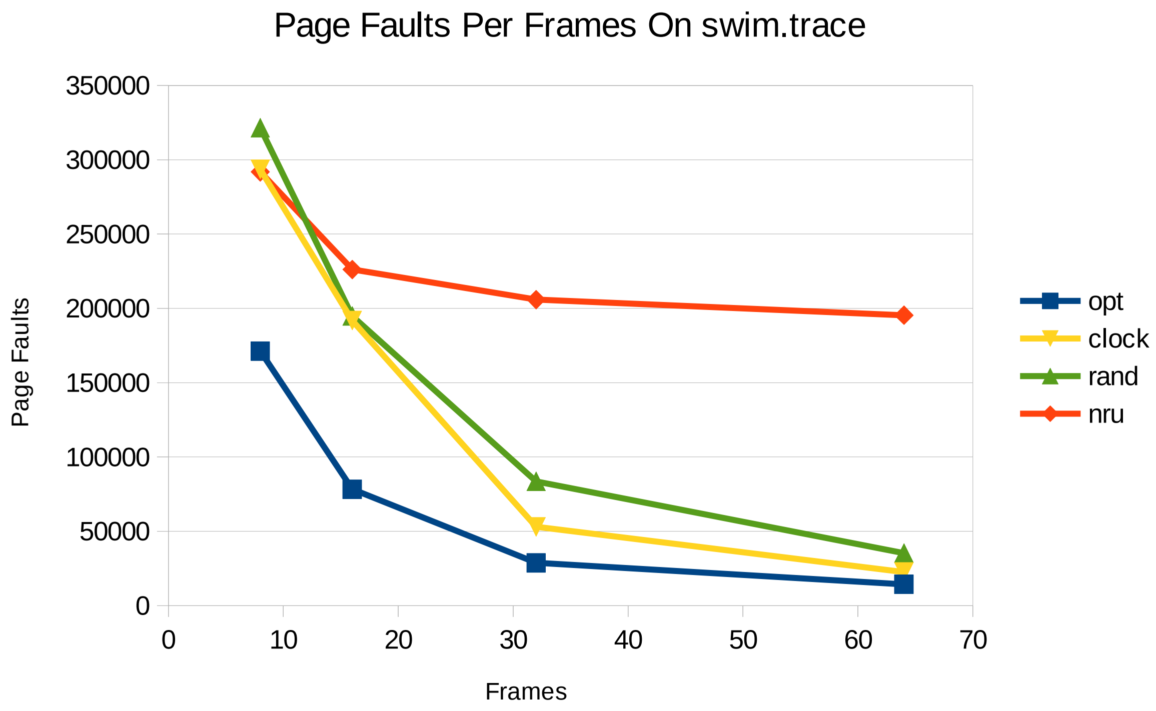
It is worth noting the rand is not a totally unreasonable choice. Rand performed reasonably closely to clock under most tests, and outperformed NRU on many tests. However, rand does involve the overhead of generating random numbers, which could be fairly significant. Nonetheless, this could potentially be offset by the fact that rand will (after calculating the random number) immediately know which page to evict and will not have to make numerous page checks as both clock and NRU will.

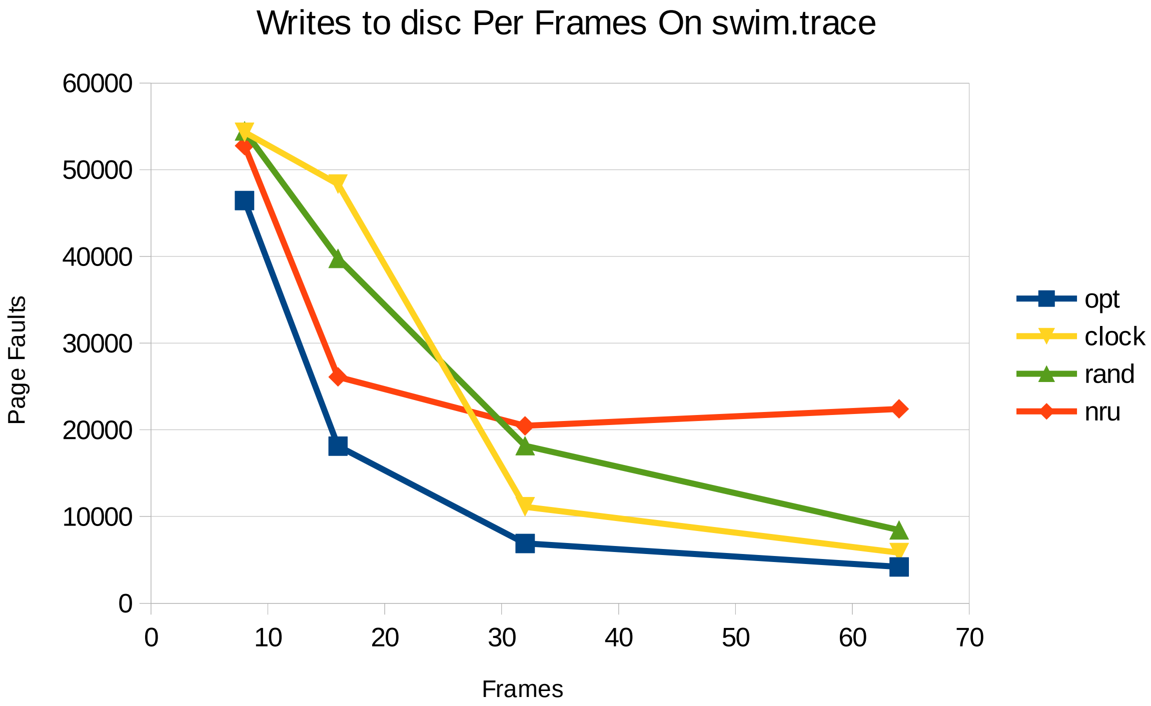
Finally, it should also be noted that if the number of pages written to disc was the primary goal, that is, you cared more about reducing the number of pages written to disc than reducing the number of page faults, it would be better to use NRU than clock. Even without optimizing NRU for each frame count, it still outperforms clock in terms of the number of pages written to disc. This is because NRU prioritizes evicting clean pages over dirty pages, while clock makes no distinction between the two when choosing which page to evict.

opt	gcc		swim	
Frame Count	Page faults	Writes to Disc	Page faults	Writes to Disc
8	118480	15031	171244	46450
16	80307	11319	78312	18132
32	55802	8278	28826	6911
64	38050	5742	14289	4193
clock	gcc		swim	
Frame Count	Page faults	Writes to Disc	Page faults	Writes to Disc
8	181856	29401	293519	54327
16	121682	16376	191848	48350
32	87686	12293	53025	11140
64	61640	9346	22611	5844
rand	gcc		swim	
Frame Count	Page faults	Writes to Disc	Page faults	Writes to Disc
8	217098	37445	321538	54482
16	149997	24607	194826	39788
32	106692	17269	83633	18169
64	75122	12306	35296	8454
nru	gcc		swim	
Frame Count	Page faults	Writes to Disc	Page faults	Writes to Disc
8	174243	18536	291787	52786
16	151713	12964	226158	26119
32	146170	11712	205928	20476
64	135837	10378	195409	22424


Page Faults Per Frames On gcc.trace
250000
200000
150000
100000
50000
0
Page Faults
0
10
20
30
40
50
60
70
Frames
opt
clock
rand
nru


Writes to disc Per Frames On gcc.trace
40000
35000
30000
25000
20000
15000
10000
5000
0
Page Faults
0
10
20
30
40
50
60
70
Frames
opt
clock
rand
nru


Page Faults Per Frames On swim.trace
Page Faults
350000
300000
250000
200000
150000
100000
50000
0
0
10
20
30
40
50
60
70
Frames
opt
clock
rand
nru


Writes to disc Per Frames On swim.trace
Page Faults
60000
50000
40000
30000
20000
10000
0
0
10
20
30
40
50
60
70
Frames
opt
clock
rand
nru