CS330: Assignment 3

Ayush Sekhari (Roll No: 12185)
Vicki Anand (Roll No: 12793
Kundan Kumar Roll NO: 12375)

November 14, 2014

1 General Observations

- We observe that as the number of physical pages decrease, the number of page-faults and consequently the number of timer ticks required increase.
- Random page replacement works randomly, sometimes good, sometimes bad as compared to other algorithms.
- In most of the cases, the LRU algorithms performs the best as expected and learned in class. We saw that the optimal algorithm i.e. longest forward distance can be approximated by LRU algorithm.
- The results of LRU and LRU are comparable(mostly). This verifies that LRU clock is a good approximation to the LRU algorithm.
- The FIFO algorithm generally performs the worst as expected.
- As the number of physical pages increase, the algorithms tend to give similar/same results. This is because the need/probability to call for replacement decreases as we have sufficient memory.

2 Statistics

For each table, under each algorithm, the first entry corresponds to the total ticks and the second entry corresponds to the number of page-faults observed during execution.

2.1 Application `queue`

- The queue program is a very complex program and does not have any such pattern in general.

- Thus here, LRU and LRU clock(which are similar algorithms) work efficiently as they are closest approximations to optimal longest forward distance algorithm.
- FIFO works poor in general.
- Here, random performs the worst as expected because it randomly chooses the page to be replaced without respecting/giving any importance to the replacing page. Here, it may be a code page or some page to be used in future (which may be respected by FIFO and LRU algorithm).

2.2 Application `vmtest1`

- Here LRU and LRU-CLOCK are comparable and FIFO performs bad w.r.t other two.
- We observe that in the program we run over the four arrays array1, array2, array3 and array4. First we iterate over the parts of the first array OUTER_LOOP times, then we iterate over second array and so on.
- Let us observe how each outer-loop works. If we divide the array into three parts 1, 2 and 3. One complete iteration of it calls the following array read sequence: 1 2 1 3 1 2 1 3 1 2 1 3 1 2 1 3
- This is done for each of the 4 arrays. We will assume that size of these blocks is such that only 2 of them can be fully accommodated in the memory. If that is not the case all the algorithms will perform comparable as we read in this sequence.
- Here, we observe that part 1 is read alternately and 2 and 3 are read in between 1's.
- LRU would replace out 2 when a 3 wants to enter and similarly 3 when 2 wants to enter, this would cause one page fault and 1 would remain in place and would not cause any page-faults.
- FIFO would cause many more page faults here. We can check via a dry-run. Every time a new read to 2 or 3 would replace out 1 and then when we read 1 next, it replaces out the older one of 2 or 3 and this is the one to be read next. Thus we observe that the trend to be bad for FIFO as compared to LRU.
- random would be something between FIFO and LRU and thus shows this trend.

2.3 Application `vmtest2`

- The trend that we observe is due to a very serious drawback of the FIFO algorithm. If the code-page is used again and again then FIFO may replace it (when memory is fully occupied) even thought it was used most recently because it may have been brought in first.

- This would cause many more page faults than in the case of LRU or LRU clock which respect the fact that this code-page was used recently so it may be important
- The pattern in which array parts (1, 2 and 3) are read is : 1 2 1 3 2
- Even though from the program dry run with the assumption that 2 array parts fill up my memory, it seems that FIFO should have performed better than LRU or LRU clock but that is not the case due to repeated replacement of code-block page in FIFO case.
- In the case of vmtest1, Both the factors(dry run sequence and code-block replacement) were against the FIFO policy and it performed the worst.
- This trend is much more evident when the number of physical pages are less in which case there is a huge competition between the code block pages and the data pages.
- As we can expect, Random performs randomly.

Table 1: statistics for queue

Number of Physical Pages	Random		FIFO		LRU		LRU-CLOCK	
16	1475945	1827	1315005	1513	1183561	1290	1209219	1329
32	949316	1011	996086	991	820862	803	815118	830
64	506723	213	506613	205	472571	166	479656	146
128	451050	118	451050	118	451050	118	451050	118

Table 2: statistics for vmtest1

Number of Physical Pages	Random		FIFO		LRU		LRU-CLOCK	
16	3962297	2387	3957506	2381	3566706	2021	3566706	2021
32	3519536	1981	3602935	2055	3430159	1896	3431670	1897
64	3057110	1552	3250295	1732	3174842	1663	3181692	1666
128	2475241	1017	2726782	1247	2704906	1225	2704906	1225
256	1945974	528	1788593	381	1786266	379	1785190	378
512	1783071	376	1783071	376	1783071	376	1783071	376

Table 3: statistics for vmtest2

Number of Physical Pages	Random		FIFO		LRU		LRU-CLOCK	
16	1246108	752	1225678	732	1116786	632	1116786	632
32	1118718	635	1141262	657	1093401	613	1093401	613
64	1052740	574	1075122	596	1057374	578	1055507	577
128	955710	484	987661	515	979383	509	979383	509
256	896574	431	842391	381	840739	380	839590	379
512	837263	377	837263	377	837263	377	837263	377