Artsiom Skarakhod

November 23, 2020

Lab3

For this lab, we had to create three different paging replacement algorithms, to observe and learn how they are constructed as well as how they work from the inside in general. We had to make: First in First Out (FIFO), Least Recently Used (LRU), and last but not least, the Optimal algorithm.

With us having three different algorithms, there is always going to be a difference in performance while running them. FIFO and LRU are very fast, they run in less than a second, but a huge downside for them is that when you run the programs, you get a huge amount of page faults. The optimal algorithm, on the other hand, is a lot more complicated, takes a long time to run (in my case it takes about 4-5 minutes on my machine), but it gets a lot fewer page faults.

If you take a look at the graph below, called “Frame Size vs Different Rate faults in placement algorithms” we can see that LRU and FIFO performed almost identically, where the Optimal has outperformed them every single time by 20-30% fewer page faults, meaning that all of that extra time calculating the optimal page to replace was not lost for nothing, and it shows us a giant leap in page fault reduction. Although the algorithms are all different, by looking at the graph below, it is possible to notice a trend that is displayed by the points on the graph. The trend is: the higher the frame size, the fewer page faults will happen during the placing algorithms. Every time we raise the frame size of the program, the lower the page fault count seems to go. It also seems like the higher the fame size, the bigger the difference page faults between the algorithms are. For the frame size of 128, the difference between LRU and Optimal is 1880, at 256 is 2400, at 512 is 2843 and at 1024 the difference is 2869.

After taking a look at how well different replacement algorithms do in comparison to the frame size of the program, it is also going to be interesting to take a closer look at when and where some of the algorithms perform worse and better with our data input given to us by the professor. If you take a look below, you can see multiple different patterns going on in our page fault rates in comparison to what our frame size is as well as how many data elements that we have gone through in the file. The first pattern is how close our LRU and FIFO are located in performance. The graphs for the 4 different frame sizes look almost identical. To be fair that does make sense because LRU and FIFO algorithms are quite similar to each other and work similarly as well as have similar logic behind them. As expected, for every single algorithm, as the size of the frame goes up, the fault rate will go down. We can see that clearly in every single graph presented to us by the data. After comparing all of the three graphs we can also notice how inefficient FIFO and LRU are compared to Optimal. The Optimal replacement algorithm overall does much better with the smaller frame size, as well as improves a lot more as the size of the frame goes up.

Optimal is also the only algorithm that consistently has fewer page faults as we get to a higher number of pages. This tends to happen because the Optimal algorithm tries looking into the pages that will need to be replaced in the future. Because of that, it replaces a page that will be used the furthest away in the future/ won’t be used in the future at all, therefore as the time goes by, the more pages we go through more pages we will slowly get rid of which will cause less and fewer replacements as a whole. It is also interesting to look at the shape of the Optimal graph because the higher number of pages we have, the lower the replacement rate gets, because again, we replace all of the pages we won’t use in the future and that will be useless to us, as we get to the end of the input document there are going to be less and less different pages, and all of the upcoming pages should already be in the frame, therefore making it a lot more accurate. But as I have stated above, it takes much longer to run the Optimal scheduling algorithm on my computer compared to the LRU and FIFO ( Optimal takes about 5 minutes to finish running and FIFO and LRU are done in about a second each).