Nate Milkosky Reference Strings Lab Report

**Abstract**

This lab is an examination of some popular page-replacement strategies. In examining this, sample reference strings had to be generated in order to simulate the page-replacement strategies. For the first part of this lab, a program that would generate reference strings would need to be created, and it would have to generate various types of strings with different localities. For the next part, the optimal page replacement algorithm for each type of locality would have to be described.

Part three required that the simulator programs would be created, and it would implement first-in-first-out (FIFO), true least recently used (LRU), and second chance page replacement strategies.

The final part included the simulation of a round-robin scheduler using a clock, which would then be analyzed using the simulation techniques in part three.

**Problem Statement**

Part One – For part one, it was necessary to create a program that would generate reference strings using a multitude of parameters. These reference strings should exhibit a different types of locality, such as spatial, temporal, a combination of both, or no specific locality (through psuedorandom generation).

Part Two – In part two, the optimal page replacement algorithms had to be described for the four types of locality generated by the program above.

Part Three – Part three asked that a simulator program to be created for the three different types of page-replacement algorithms described in this lab: FIFO, LRU, and second chance. The simulation had to produce statistics on the number of page faults generated for each one.

Part Four – Part four required that a scheduler would be created, that would simulate multiprogramming using a round-robin scheduler. It would simulate different job loads by using different types of locality after context switching. Then, statistics would be generated by using the different page-replacement algorithms.

**Methodology (contains Part Two solution)**

Part One – In creating part one, I decided that I would create a function that would generate a reference string exhibiting a specific type of locality. Each of these functions would be passed a few parameters: the array that would contain the reference string, the length of the reference string, the bounds of the pages for each process, and the page list. In spatial locality, I simply iterated through a number of pages as though the process was moving from one page to the next. For temporal locality, I had it use one or two pages repeatedly, simulating a loop or function calls. For the combination of both, I simply switched back and forth between spatial and temporal locality. Finally, generating a pseudorandom reference string was easy, as I just generated a bunch of random numbers and put them into the reference string.

Part Two –

a. The best algorithm for spatial locality is an algorithm that will allow a the pages to expire when not used, but if some pages are frequently used, they may want to be kept in memory. As such, LRU would be good for this.

b. The best algorithm for temporal locality is an algorithm that will not spend a lot of time doing complex operations, as the pages won't change as much as the other algorithms. An algorithm that keeps track of frequent pages, such as second chance would be good here.

c.The best algorithm for both is an algorithm that has the capability to keep frequented pages in memory as well as allow infrequent pages to expire. LRU keeps frequently used pages away from replacement, so it would be best here.

d.For pseudorandom reference strings, a simple algorithm is best. Because there are no patterns and it is not guaranteed that a page will be visited again, simple FIFO page replacement is best. Very little overhead is needed and the frequency of pages will be (on average) about the same for every page.

Part Three *–* In part three, I had to implement a simulator function for each different type of page-replacement. For the FIFO function, I made a data structure that would house a node in the queue. It contained the page number, and the next page in the queue. Once the queue was full (indicating that our memory ran out), a page had to be replaced. The page was replaced by removing the oldest page and putting the new one at the end of the queue. For LRU, I had to use a data structure like a stack or a doubly-linked list. This structure contained the page number, and the list items next to this list item. This was more complicated as I had to move a item to the top of the stack when it was referenced, and if it was full I had to dequeue the least recently used item. The second chance algorithm wasn't much different than the FIFO algorithm, except that if an item had its reference bit set (indicating that it was recently used), it wouldn't be removed from the front of the queue. It instead would move to the end of the queue, and un-set the reference bit. This process would repeat until a process without a set reference bit was found, and then it was dequeued and the new page could come in.

Part Four – In part four, I simulated the round robin scheduler by implementing a clock that would track what 'time' the program is at, and a list of processes and when they would be arriving. This scheduler would first check if any processes are arriving, and if they were, it would add them to the end of the round-robin queue. After checking, the program would then start simulating process execution by decrementing a timer that kept track of the processes time quantum. Once the timer ran out, a reference string for that execution would be generated, and then the next process in the round robin queue would execute (after the previous process was moved to the end of the queue, and it's reference string was added to the whole reference string for the entire time). This process repeated until the time ran out (the length of the simulation was specified by a parameter), after which the final reference string was complete. Once the reference string was complete, the simulation functions in part three could be applied and performance could be analyzed.

**Data**

Part One – Sample Reference Strings (length 250)

*Spatial Locality*

|  |
| --- |
| nate@nate-ubuntu:~/OSReferenceStrings$ ./gen\_d 250 10 100 1  0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 73 17 18 19 20 21 22 23 24 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 55 56 57 58 59 60 61 62 63 64 0 1 2 3 4 5 6 7 8 73 74 75 76 77 78 79 80 81 82 83 84 85 25 26 27 28 29 30 31 32 33 34 73 74 75 76 77 78 79 80 81 82 83 9 10 11 12 13 14 15 16 17 18 35 36 37 38 39 40 41 42 43 44 45 46 47 48 49 50 25 26 27 28 29 30 31 32 33 0 1 2 3 4 5 6 7 8 9 10 9 10 11 12 13 14 15 16 17 18 19 0 1 2 3 4 5 6 7 8 9 10 11 12 13 14 |
| nate@nate-ubuntu:~/OSReferenceStrings$ ./gen\_d 250 10 100 1  18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 44 45 46 47 48 49 50 51 52 44 45 46 47 48 49 50 51 52 53 54 55 56 35 36 37 38 39 40 41 42 43 9 10 11 12 13 14 15 16 26 27 28 29 30 31 32 33 34 35 36 37 38 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 35 36 37 38 39 40 41 42 43 44 45 0 1 2 3 4 9 10 11 12 13 14 15 16 17 18 19 20 21 0 1 2 3 4 5 6 7 8 9 10 11 61 62 63 64 65 66 67 68 69 70 71 0 1 2 3 4 5 6 7 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 26 27 28 29 30 31 32 33 34 35 9 10 11 12 13 14 15 16 69 70 71 72 73 74 75 69 70 71 72 73 74 75 76 77 78 79 80 81 82 83 84 9 10 11 12 13 14 15 16 17 18 19 61 62 63 64 65 66 67 68 69 70 71 72 73 74 75 76 77 78 18 19 20 21 22 23 24 25 26 27 28 29 0 1 2 3 |

These two are indicative of spatial locality because they spend a lot of time moving around the same area.

*Temporal Locality*

|  |
| --- |
| nate@nate-ubuntu:~/OSReferenceStrings$ ./gen\_d 250 10 100 2  13 13 13 13 13 41 42 41 42 41 42 42 42 42 42 42 42 42 42 42 42 42 42 16 16 16 16 16 16 16 30 30 30 30 30 30 30 30 43 44 43 43 43 43 43 35 36 35 36 35 36 35 36 35 36 35 36 35 15 15 15 15 54 55 54 55 54 55 54 55 54 50 50 50 50 50 50 50 50 50 50 50 50 50 50 54 54 54 54 54 54 54 54 54 54 54 23 23 23 23 23 23 23 23 23 23 23 23 54 54 54 54 54 54 54 54 54 54 54 54 39 39 39 39 39 39 39 19 20 19 20 57 58 57 58 57 74 75 74 75 74 5 5 5 5 5 5 5 5 11 11 11 11 11 11 11 64 65 64 2 3 2 3 2 3 2 3 2 3 25 26 25 26 25 26 25 26 25 26 9 10 9 10 9 10 9 10 9 10 9 10 58 58 58 0 1 0 57 57 57 57 57 57 57 57 57 57 65 65 65 65 65 65 65 65 65 65 65 65 65 65 81 81 81 81 81 81 81 81 80 80 80 80 56 56 56 56 56 31 32 31 32 31 32 31 14 15 14 15 14 |
| nate@nate-ubuntu:~/OSReferenceStrings$ ./gen\_d 250 10 100 2  14 14 14 14 14 14 67 68 67 68 67 68 67 68 67 68 67 68 67 56 57 56 57 56 57 56 57 56 57 56 32 32 32 32 32 32 32 0 0 0 0 0 0 0 0 0 0 0 0 0 78 79 78 79 78 79 78 79 78 79 78 47 47 47 47 45 45 45 86 87 86 87 86 87 86 87 86 87 86 87 86 30 31 30 7 7 7 7 7 7 29 29 29 29 29 29 29 29 29 84 85 84 25 25 25 90 90 90 90 90 90 90 90 90 30 30 30 80 80 80 80 80 80 80 80 52 52 52 52 89 90 89 90 89 5 6 5 6 5 6 5 6 58 58 58 58 58 58 58 58 58 16 17 16 17 16 17 16 17 16 17 16 17 16 17 0 1 0 1 0 1 70 71 70 71 70 71 70 71 70 71 70 91 91 91 91 91 91 91 91 91 91 91 91 91 91 62 62 62 62 62 86 87 86 87 50 51 50 51 50 51 50 51 50 71 72 71 72 71 11 12 11 12 11 12 11 12 11 12 11 12 11 89 90 89 90 89 90 55 56 55 56 55 56 55 56 55 77 77 77 |

These exhibit temporal locality because they spend the majority of their time in the same pages.

*Combination*

|  |
| --- |
| nate@nate-ubuntu:~/OSReferenceStrings$ ./gen\_d 250 10 100 3  55 56 57 58 59 60 58 59 58 59 58 25 25 25 25 55 56 57 58 59 60 61 62 63 66 67 68 69 70 71 72 73 74 75 76 77 17 18 19 20 21 22 23 24 25 42 42 42 42 42 42 42 42 80 47 48 49 50 51 52 53 55 45 46 45 46 45 46 45 46 45 47 48 47 48 47 48 66 67 68 69 70 71 72 73 47 48 55 56 57 58 59 60 61 62 63 64 65 76 77 78 79 80 81 82 83 84 85 86 87 66 67 68 69 70 71 72 73 74 78 79 78 79 78 55 56 57 58 59 60 61 62 63 64 65 66 82 83 82 83 82 83 72 72 72 72 72 72 0 0 0 39 45 45 45 45 68 69 47 48 49 50 51 52 53 54 55 64 65 64 65 64 2 3 2 3 2 47 48 49 50 51 52 53 54 17 18 19 20 21 22 23 24 25 26 27 28 0 1 2 3 4 5 6 7 8 9 10 66 47 48 49 50 51 52 53 54 55 76 77 78 79 80 0 1 0 1 0 1 0 47 48 49 50 51 52 53 54 55 56 57 37 38 76 77 78 79 80 81 82 |
| nate@nate-ubuntu:~/OSReferenceStrings$ ./gen\_d 250 10 100 3  40 40 40 53 54 53 54 53 54 53 10 11 12 13 14 21 22 23 24 25 26 27 28 50 50 50 50 50 50 0 0 0 0 0 0 0 0 0 75 75 75 2 3 2 3 60 61 79 80 79 80 79 80 79 80 59 60 61 62 63 64 65 66 59 60 61 62 63 64 61 61 61 78 78 78 78 78 78 10 11 12 13 14 15 16 17 68 69 70 71 72 73 74 75 76 77 78 4 5 4 5 4 5 4 5 59 60 61 62 63 64 0 1 2 30 31 32 33 34 35 36 37 10 11 12 13 14 15 16 17 41 42 43 44 45 44 45 44 45 44 45 78 79 80 81 82 83 84 85 86 78 79 80 81 82 78 79 80 81 82 83 84 85 0 1 2 21 22 23 24 25 26 27 28 68 69 70 71 72 73 51 52 53 54 55 56 59 60 61 62 63 64 65 66 67 68 69 70 46 47 46 47 41 42 43 44 45 46 47 48 49 21 22 23 24 25 26 27 79 79 79 79 79 79 79 79 79 79 14 14 14 14 14 14 14 14 14 36 37 36 37 36 79 79 79 79 79 79 79 79 |

These exhibit characteristics of both spatial – the spatial sections are slanted while the temporal sections are flat.

*Pseudorandom*

|  |
| --- |
| nate@nate-ubuntu:~/OSReferenceStrings$ ./gen\_d 250 10 100 4  41 86 4 91 71 30 12 22 98 33 66 17 45 8 39 18 38 28 27 10 37 39 99 0 80 36 92 83 66 27 84 59 13 89 50 36 71 15 11 69 48 29 86 94 37 77 12 28 6 91 38 95 82 90 95 63 78 39 46 44 66 82 4 31 71 6 67 42 21 30 63 70 60 50 16 49 79 80 77 37 72 68 32 6 10 27 21 88 66 67 84 84 50 40 15 73 99 83 16 72 65 79 94 25 81 10 27 13 91 4 50 15 24 35 21 34 14 43 74 33 62 11 17 64 51 33 38 2 68 6 75 33 37 69 11 71 32 90 36 23 46 86 90 71 73 11 57 88 6 32 73 21 43 90 37 46 75 27 1 43 85 76 29 23 97 92 46 29 82 82 4 28 20 94 51 46 58 9 34 16 93 59 37 36 1 75 34 77 54 35 72 40 63 1 63 61 45 61 90 27 95 47 8 15 41 59 61 99 68 47 16 13 58 5 49 60 32 36 37 87 71 9 27 87 63 42 48 8 55 90 88 50 37 96 17 79 7 79 30 28 78 98 41 37 4 43 97 36 79 86 |
| nate@nate-ubuntu:~/OSReferenceStrings$ ./gen\_d 250 10 100 4  4 59 26 81 43 13 58 18 78 85 50 30 35 98 9 4 39 28 17 6 14 34 20 50 41 2 86 76 97 40 13 1 99 40 35 94 5 93 12 84 30 62 66 17 13 76 74 52 56 91 10 70 25 30 73 66 84 11 94 81 51 8 34 50 0 69 44 5 14 9 41 45 23 8 14 36 36 88 40 92 79 50 15 56 80 88 23 64 99 69 97 50 77 32 52 29 1 48 35 16 57 28 13 33 88 27 21 24 16 62 69 47 12 84 4 45 72 79 61 71 48 59 73 78 43 25 59 44 73 94 12 83 75 25 68 63 5 89 88 73 51 57 20 16 41 76 13 65 55 74 36 56 85 9 86 28 86 45 25 11 92 37 46 67 15 14 82 20 56 70 45 59 79 65 75 72 94 40 37 1 15 25 9 0 86 95 81 72 93 6 36 37 95 82 56 62 97 38 34 5 61 79 64 40 97 92 65 91 32 2 92 47 28 2 0 66 49 33 39 94 91 75 83 86 9 39 49 6 30 83 11 91 15 28 83 12 20 48 3 52 3 47 52 83 49 4 49 51 37 40 |

These have pretty much no patterns to identify, and move all over the place, which is characteristic of randomness.

Part Three – Page Faults – rows are string types, and columns are page-replacement algorithms

Reference String was of length 250, and the maximum frames in memory was 10.

|  |  |  |  |
| --- | --- | --- | --- |
|  | FIFO | LRU | Second Chance |
| Spatial # 1 | 228 page faults | 229 page faults | 225 page faults |
| Spatial # 2 | 239 page faults | 239 page faults | 240 page faults |
| Temporal # 1 | 47 page faults | 46 page faults | 46 page faults |
| Temporal # 2 | 35 page faults | 36 page faults | 35 page faults |
| Combination # 1 | 138 page faults | 142 page faults | 140 page faults |
| Combination # 2 | 171 page faults | 171 page faults | 171 page faults |
| Pseudorandom # 1 | 229 page faults | 228 page faults | 228 page faults |
| Pseudorandom # 2 | 221 page faults | 220 page faults | 220 page faults |

Spatial Reference String # 1: ./sim\_d 250 10 100 1 10

18 19 20 21 22 23 24 29 30 31 32 33 34 35 36 37 38 39 40 41 42 43 44 45 37 38 39 40 41 65 66 67 68 69 45 46 47 48 49 50 51 52 53 54 55 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 71 72 0 1 2 3 4 5 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 62 18 19 20 21 22 23 29 30 31 32 33 10 11 12 13 14 15 16 17 18 19 20 18 19 20 21 22 23 18 19 20 21 22 23 24 25 26 27 28 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 73 74 75 76 77 78 79 65 66 67 68 69 70 71 72 73 74 75 76 77 78 79 80 0 1 2 3 4 5 6 7 8 37 38 39 40 41 42 43 44 45 46 47 48 37 38 39 40 41 42 43 44 45 46 47 48 49 50 45 46 47 48 49 50 51 52 53 54 55 56 57 58 59 60 61 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 45 46

Spatial Reference String # 2: ./sim\_d 250 10 100 1 10

37 38 39 40 41 42 43 44 45 46 47 48 49 50 51 72 73 74 75 76 77 78 79 80 81 82 83 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 46 47 48 49 50 51 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 46 47 48 49 50 51 52 53 54 18 19 20 21 22 23 24 25 26 27 28 29 30 31 32 33 34 35 36 62 63 64 65 66 67 68 69 70 71 54 55 56 57 58 59 60 61 62 63 64 65 66 67 68 69 70 83 84 85 86 87 88 89 90 91 92 93 94 95 96 97 72 73 74 75 76 77 78 79 80 81 82 83 84 85 86 87 88 54 55 56 57 58 59 60 61 62 63 64 65 0 1 2 3 4 5 6 7 8 0 1 2 3 4 5 6 7 8 9 72 73 74 75 76 77 78 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 62 63 64 65 66 67 68 69 70 71 72 73 74 75 18 19 20 21 22 27 28 29 30 31 32 33 34 35 36 37 38 54 55 56 57 58 59 60 61 62 63 64 0

Temporal Reference String # 1: ./sim\_d 250 10 100 2 10

11 11 11 11 11 11 11 11 45 45 45 45 45 45 45 45 5 5 5 5 5 5 6 5 6 5 6 5 6 5 1 1 1 1 1 1 1 1 1 1 1 1 1 1 56 57 56 57 56 57 56 57 18 19 18 19 18 19 18 19 54 55 54 12 13 12 13 12 13 12 13 12 13 12 13 30 31 30 31 30 31 30 31 30 31 0 0 0 0 0 0 4 4 4 4 4 4 4 13 14 13 14 13 14 13 14 13 19 20 19 20 19 20 1 1 1 1 1 1 1 1 1 1 1 33 34 33 34 33 34 33 34 33 34 33 34 5 5 5 5 5 5 5 5 44 44 44 44 44 44 44 44 67 68 67 68 93 94 93 94 56 56 56 33 33 33 33 33 33 82 82 82 82 82 82 82 82 82 82 82 82 82 9 10 9 10 9 71 72 71 72 71 72 71 72 71 72 71 72 15 15 15 15 15 15 15 15 15 15 40 41 40 41 40 41 40 41 58 58 58 58 58 25 25 25 84 85 84 85 84 85 84 85 84 76 77 76 77 76 6 7 6 7 6 7 6 7 6 54 54

Temporal Reference String # 2: ./sim\_d 250 10 100 2 10

62 63 62 63 62 63 62 63 62 76 77 76 77 76 77 76 77 76 77 76 77 76 77 93 94 93 94 93 94 93 94 93 94 93 94 93 63 64 63 64 63 64 60 60 60 60 60 60 60 60 60 77 78 77 78 77 78 77 78 77 78 77 78 77 78 79 79 79 79 79 79 79 79 79 79 79 79 79 94 94 94 94 94 94 94 94 94 68 68 68 68 68 68 68 77 77 77 6 7 6 93 93 93 93 93 93 93 93 93 93 93 87 87 87 87 87 87 87 87 87 87 87 87 87 51 51 51 51 51 51 51 51 51 51 51 51 9 10 9 10 9 10 9 10 9 10 9 10 9 16 17 16 21 22 21 22 21 22 21 26 26 26 26 26 26 26 26 26 26 26 26 26 84 85 84 85 84 85 84 85 84 85 84 85 15 16 15 16 15 16 15 16 50 50 50 50 87 87 87 15 15 15 15 15 15 35 36 35 36 56 57 56 57 56 57 56 57 56 57 56 57 56 57 13 13 13 13 13 13 13 13 13 13 88 88 88 88 88 88 88 88 88 88 88 88 40 41 40 41 40

Combination Reference String # 1: ./sim\_d 250 10 100 3 10

57 58 59 60 61 62 63 64 65 66 67 68 45 46 45 46 45 46 45 87 87 87 35 36 35 36 77 77 77 77 30 30 90 90 90 90 90 89 90 89 87 87 87 87 87 87 87 31 32 31 32 31 7 8 7 8 13 13 13 13 27 27 27 66 66 66 10 11 12 13 14 15 16 17 4 5 4 5 4 5 4 5 4 64 65 64 65 64 65 64 65 64 20 21 22 23 24 57 20 21 22 23 24 25 26 27 28 29 30 31 32 30 31 32 33 34 27 27 27 76 77 78 79 80 30 31 32 33 53 53 53 0 1 2 3 4 5 6 7 8 9 38 39 40 41 42 57 58 59 60 61 38 39 40 41 42 43 44 15 15 15 10 11 12 13 14 15 16 17 18 30 31 69 69 69 69 69 69 69 69 69 69 17 18 17 18 17 18 17 55 55 55 55 55 55 55 55 62 63 38 39 40 41 42 43 44 45 46 47 48 49 8 8 8 8 8 8 0 14 14 14 10 11 12 13 14 15 20 21 22 23 24 25 26 27 68 69 68 69 68 10 11 12 13 14 15 20 21 22 23

Combination Reference String # 2: ./sim\_d 250 10 100 3 10

31 32 33 34 35 36 37 38 79 80 10 11 12 13 14 15 16 17 32 33 32 33 68 68 68 68 68 68 68 68 46 47 77 78 77 78 0 0 0 0 0 0 0 18 19 18 19 18 19 18 79 31 32 33 34 35 36 37 38 39 40 41 42 43 44 31 32 31 32 31 32 31 32 31 95 96 95 96 95 96 40 41 42 43 44 45 63 64 63 64 63 64 63 64 63 64 89 90 91 92 93 94 95 79 80 81 82 83 84 85 86 10 11 12 13 65 66 65 66 65 66 65 66 65 66 79 80 81 82 83 84 85 10 11 89 90 91 92 93 94 95 96 97 98 99 9 51 52 51 52 51 52 51 52 51 52 20 21 22 23 24 25 40 41 42 43 44 79 80 81 82 83 84 85 86 87 88 89 90 70 70 70 70 70 70 70 70 0 1 2 3 4 5 6 7 8 9 89 90 91 92 93 94 95 59 60 61 62 63 64 65 89 90 91 92 93 94 95 96 97 98 99 89 90 33 34 33 34 33 52 53 52 53 52 53 52 89 90 91 92 93 94 59 60 61 62 63 64 0 1

Pseudorandom Reference String # 1: ./sim\_d 250 10 100 4 10

85 58 4 35 70 36 19 38 6 2 23 92 10 29 25 96 16 13 66 99 10 51 24 41 23 78 77 7 50 49 32 88 7 88 23 30 77 42 20 35 45 43 79 7 25 4 3 93 69 69 44 32 72 68 73 47 98 50 55 49 51 39 89 11 79 12 41 56 7 13 92 4 57 71 11 34 28 66 27 97 35 23 81 59 43 6 6 42 57 13 43 60 4 32 71 84 44 64 40 3 78 84 7 87 56 18 21 36 84 0 33 71 23 67 82 66 73 89 60 30 54 3 43 59 35 14 43 32 79 35 35 9 20 43 96 28 13 69 64 98 69 49 69 44 16 52 62 90 93 23 72 99 78 67 58 14 82 53 98 13 41 33 22 13 28 70 41 42 91 57 40 12 58 61 56 75 65 70 17 10 45 41 10 24 9 20 90 43 74 88 56 15 73 30 28 2 52 21 44 43 30 36 55 88 49 63 15 67 33 32 77 79 26 39 55 87 60 45 30 86 85 86 53 58 68 81 60 20 54 56 15 84 44 70 72 94 33 40 61 18 24 90 97 50 82 4 37 94 49 67

Pseudorandom Reference String # 2: ./sim\_d 250 10 100 4 10

54 24 69 70 31 94 92 30 40 32 2 53 81 77 89 45 36 71 59 90 60 75 25 70 86 66 24 31 53 93 78 59 69 99 29 1 93 21 83 85 6 85 38 87 14 80 32 51 3 92 41 63 19 66 34 57 84 10 89 90 56 19 49 25 70 31 78 63 4 61 1 10 99 91 97 13 23 82 64 26 26 57 90 45 76 76 3 60 86 44 2 42 15 4 20 85 35 50 0 91 12 53 2 11 45 51 76 68 85 41 47 63 98 89 61 26 65 16 87 3 60 41 98 27 97 70 64 84 20 64 76 84 70 30 47 15 33 24 35 19 17 82 82 15 71 43 42 88 59 81 92 71 74 42 50 72 12 14 56 32 31 84 69 1 14 16 16 48 40 3 19 57 38 1 25 9 97 19 50 8 52 94 32 26 36 82 50 48 49 59 32 80 43 1 81 58 18 49 58 10 4 77 68 42 30 45 4 27 64 54 88 68 48 20 46 84 54 97 84 3 8 16 83 51 70 16 61 88 17 19 50 22 96 18 16 27 15 20 54 79 74 94 47 74 14 94 10 21 43 94

Part 4 – Scheduler simulation

|  |  |  |  |
| --- | --- | --- | --- |
|  | FIFO | LRU | Second Chance |
| Simulation # 1 | 142 page faults | 140 page faults | 144 page faults |
| Simulation # 2 | 127 page faults | 130 page faults | 129 page faults |

Reference String Simulation # 1: ./sched\_d

0 1 2 3 4 5 6 7 8 9 1 1 1 1 1 1 1 0 0 0 0 1 2 3 4 5 6 7 8 0 10 11 12 13 14 15 16 17 10 11 2 3 2 3 2 3 2 3 5 5 41 42 41 42 40 40 40 40 40 40 10 11 12 13 14 15 16 17 10 11 58 58 58 58 58 62 63 62 63 62 9 6 3 3 8 9 9 7 1 6 44 34 43 37 44 33 33 32 42 44 17 21 16 25 12 20 19 25 11 18 64 48 66 63 62 49 69 60 66 61 4 4 4 4 4 5 5 5 5 5 33 42 30 38 33 35 44 36 34 35 24 25 24 25 24 17 18 17 17 17 45 46 47 48 49 50 47 47 47 47 9 9 9 9 9 9 9 9 9 9 94 95 94 73 73 73 73 73 73 73 30 31 32 33 34 35 36 37 38 39 14 15 14 15 11 12 11 12 29 29 68 59 46 46 47 59 69 62 61 53 9 4 5 8 9 4 3 6 8 6 93 94 93 94 93 94 93 94 70 71 30 31 32 33 34 35 36 37 38 30 10 11 12 13 14 15 16 17 18 19

Reference String Simulation # 2: ./sched\_d

0 1 2 3 4 5 6 7 8 9 0 1 2 3 4 5 6 7 0 0 4 8 3 0 4 7 8 3 2 5 21 20 22 19 19 11 13 15 29 29 8 8 8 8 0 1 0 1 0 1 32 33 32 33 32 33 32 33 32 37 10 11 12 13 14 15 16 13 13 13 57 57 57 57 60 60 60 60 60 60 0 1 2 3 4 5 6 7 0 1 36 31 33 30 35 39 33 44 33 30 12 13 12 16 17 16 16 17 16 17 47 48 47 68 69 68 69 68 69 68 0 1 2 3 4 5 6 7 8 9 30 31 32 33 34 35 36 37 38 39 27 27 27 27 27 19 20 19 20 10 45 46 47 48 49 45 46 47 48 49 3 4 3 4 3 4 3 4 3 3 79 80 79 80 79 80 79 80 83 84 33 31 38 41 42 39 36 41 33 37 10 11 12 13 14 15 16 10 11 12 64 47 57 54 63 45 58 56 67 66 3 4 3 4 3 4 3 4 3 4 92 93 92 93 92 93 92 93 92 93 41 42 41 42 41 42 41 42 41 42 24 18 11 12 13 13 29 20 12 14

**Analyses**

After experimenting with the different page replacement algorithms, it appears to be that they all exhibit similar performance. This presents a problem for true LRU which is very computationally intensive – it doesn't do much better (or better at all) than the other, simpler algorithms. Because of this, I don't believe that using LRU is a good idea. Second chance, however, seems to work very slightly better in some cases and pretty much the same in others. Because of this (as well as its ease of implementation), I believe that second chance has the best opportunity to provide good results. All of this assumes that my implementations work as intended (which I believe to be the case).

**Conclusions**

This lab provided a good way to experiment with reference strings and work with C in a more intensive way than I had previously. After the experiments, I didn't feel as though the sample reference strings simulated a real environment well enough to draw any conclusions about which page replacement algorithm was the best – however, it was clear that true LRU would not work out in a real-life situation, and that second chance gave slightly better efficiency. In my testing, I found that the performance of each algorithm depended a lot on the reference string – one algorithm could work well with one string, and another could not work well with the same string.

**Appendix**

Program Descriptions

generator.c / .h: implements the functions that generate the types of reference strings. For Part 1.

generator\_driver.c: a driver for the generator program. For Part 1.

simulator.c / .h: implements the page replacement algorithms and their simulations. Part 3.

simulator\_driver.c: driver for the simulator. Part 3.

scheduler.c : implements the scheduling simulation, and also has the main method. Part 4.