# Task 3:

### Simple loop Size 50

| Algorithm | Hit Rate | Hit Count | Miss count | Overall Eviction Count | Clean eviction count | Dirty eviction count |
|---|---|---|---|---|---|---|
| FIFO | 22.4822 | 759 | 2617 | 2612 | 45 | 2567 |
| LRU | 25.2073 | 851 | 2525 | 2475 | 0 | 2475 |
| Clock | 25.1481 | 849 | 2527 | 2477 | 0 | 2477 |
| Random | 22.4822 | 759 | 2617 | 2612 | 45 | 2567 |

### Simple loop Size 100

| Algorithm | Hit Rate | Hit Count | Miss count | Overall Eviction Count | Clean eviction count | Dirty eviction count |
|---|---|---|---|---|---|---|
| FIFO | 23.7855 | 803 | 2573 | 2,496 | 23 | 2473 |
| LRU | 25.2073 | 851 | 2525 | 2425 | 0 | 2425 |
| Clock | 25.0889 | 847 | 2529 | 2430 | 1 | 2429 |
| Random | 23.8744 | 806 | 2570 | 2,486 | 16 | 2470 |

## Matmul Size 50

| Algorithm | Hit Rate | Hit Count | Miss count | Overall Eviction Count | Clean eviction count | Dirty eviction count |
|---|---|---|---|---|---|---|
| FIFO | 62.9391 | 1913768 | 1126896 | 2,231,301 | 1104455 | 1126846 |
| LRU | 65.7665 | 1999739 | 1040925 | 2,080,789 | 1039914 | 1040875 |
| Clock | 65.7663 | 1999733 | 1040931 | 2,080,798 | 1039917 | 1040881 |
| Random | 67.2663 | 2045343 | 995321 | 1,970,006 | 974735 | 995271 |

## Matmul Size 100

| Algorithm | Hit Rate | Hit Count | Miss count | Overall Eviction Count | Clean eviction count | Dirty eviction count |
|---|---|---|---|---|---|---|
| FIFO | 64.3709 | 1957302 | 1083362 | 2,154,972 | 1071710 | 1083262 |
| LRU | 66.9061 | 2034389 | 1006275 | 2,011,390 | 1005215 | 1006175 |
| Clock | 65.7699 | 1999841 | 1040823 | 2,080,485 | 1039762 | 1040723 |
| Random | 89.3805 | 2717762 | 322902 | 641,416 | 318614 | 322802 |

### Blocked Size 50

| Algorithm | Hit Rate | Hit Count | Miss count | Overall Eviction Count | Clean eviction count | Dirty eviction count |
|---|---|---|---|---|---|---|
| FIFO | 99.8245 | 3507538 | 6166 | 10,237 | 4121 | 6116 |
| LRU | 99.8618 | 3508848 | 4856 | 7,410 | 2604 | 4806 |
| Clock | 99.8616 | 3508841 | 4863 | 7,470 | 2657 | 4813 |
| Random | 99.7716 | 3505680 | 8024 | 13,736 | 5762 | 7974 |

### Blocked Size 100

| Algorithm | Hit Rate | Hit Count | Miss count | Overall Eviction Count | Clean eviction count | Dirty eviction count |
|---|---|---|---|---|---|---|
| FIFO | 99.8817 | 3509549 | 4155 | 6,793 | 2738 | 4055 |
| LRU | 99.8971 | 3510089 | 3615 | 6,083 | 2568 | 3515 |
| Clock | 99.8908 | 3509868 | 3836 | 6,300 | 2564 | 3736 |
| Random | 99.8563 | 3508654 | 5050 | 8,330 | 3380 | 4950 |

## Analysis of results from the tables

### Simple loop:

The program Simple loop has this property that it accesses the elements of an array of structs one by one (in which case we have a spatial locality) but then in each iteration it accesses the first element of the array field of the current struct (in which case we do not have temporal and spatial locality at all over all iterations - since we have to bring in a different page corresponding to the current field array each time). Because this program has such property, we see that overall the hit rate is low for all algorithms since the lack of locality for accessing the field array of each struct in each iteration causes the soon-to-be-needed pages to be evicted. Among all of the algorithms, LRU does the best as it is the algorithm that makes use of temporal locality of accessing the elements of the array of structs as much as possible. FIFO is not so good, as close as random, since lack of locality in accessing the elements of the field array of structs causes it to evict the soon-to-be-needed pages much earlier.

Also the change of size of the memory from 50 to 100 does not have a special effect on LRU and clock as they use the localities possible regardless of the size of the memory. However, increase in size does improve FIFO as this increase delays eviction of those soon-to-be-needed pages.

### Matmul:

The program Matmul has this property that it does matrix multiplication in a naive (not memory aware) way. In such approach for matrix multiplication, when we do A * B = C, access to A has temporal and spatial locality(since each row of that will end up to be in almost one page or more pages close together), while access to B does not have any locality(since we get each column of that for each entry computation and this means the columns are in different pages each time). Overall the lack of locality for B causes all algos to not have very high hit rates. Because this program has such property, LRU uses that small locality that exists for A and does a better job here, and so LRU is the best with clock after that, but FIFO has a bit worse hit rate. Increase in size to 100, made FIFO to be impacted more and get a higher hit rate since it delays evicting soon-to-be-needed pages. RLU increased a little too but the clock remained the same. RLU does noticeably better in every size.

### Blocked:

The program Blocked has this property that we do matrix multiplication in a more memory aware way. In this approach we try to make use of spatial and temporal locality of the matrices A and B both plus we do computations in blocks. That means the algorithms can make use of such localities to do a much better job specifically LRU should outperform among all. However, We see FIFO, LRU and clock all have a high hit rate in this program. This is because computations are done in blocks, the number of unique pages needed at a time are small and all needed pages happen to exist in memory when needed most of the time. This leads all algorithms to converge to a hit rate of 100%. Increase in size of memory does not have any noticeable effect and the reason is we have this locality and the nature of the code.

From the data we gathered in the tables, we can see that LRU and clock perform almost exactly the same, as clock attempts to simulate what LRU does without the massive overhead. They perform almost the same with only tiny differences in the hit rate and eviction count. LRU and clock will do best if the program has locality. In all cases, FIFO performed worse than LRU and Clock. However, if the program has the property such that the num of pages it needs during the time of a computation will fit in memory(like in the block program), or the program has no locality at all, then all the algorithms will perform nearly the same.

**Trace 1**

| Algorithm | Hit Rate | Hit Count | Miss count | Overall Eviction Count | Clean eviction count | Dirty eviction count |
|---|---|---|---|---|---|---|
| FIFO | 43.3333 | 13 | 17 | 9 | 0 | 9 |
| LRU | 33.3333 | 10 | 20 | 13 | 1 | 12 |
| Clock | 46.6667 | 14 | 16 | 9 | 1 | 8 |
| Random | 56.6667 | 17 | 13 | 5 | 0 | 5 |

**Trace 2 -** FIFO and Clock are the same as OPT.

| Algorithm | Hit Rate | Hit Count | Miss count | Overall Eviction Count | Clean eviction count | Dirty eviction count |
|---|---|---|---|---|---|---|
| FIFO | 62.5000 | 20 | 12 | 4 | 0 | 4 |
| LRU | 62.5000 | 16 | 15 | 7 | 0 | 7 |
| Clock | 54.8387 | 20 | 12 | 4 | 0 | 4 |
| Random | 59.3750 | 19 | 13 | 5 | 0 | 5 |

**Trace 3**

| Algorithm | Hit Rate | Hit Count | Miss count | Overall Eviction Count | Clean eviction count | Dirty eviction count |
|---|---|---|---|---|---|---|
| FIFO | 0 | 0 | 31 | 32 | 9 | 23 |
| LRU | 0 | 0 | 31 | 32 | 9 | 23 |
| Clock | 0 | 0 | 31 | 32 | 9 | 23 |
| Random | 51.6129 | 16 | 15 | 7 | 0 | 7 |