CS231 Lab 4 Part 2

Shreyas Katdare (22B0636)

Question 1

To make fifo, I just updated the current cycle value when the line first enters the cache. When it gets the hit, I do not update it's current cycle value. In case of LFU, I started counting frequency of cache line when it enters the cache. And every time it gets hit, I update its frequency count by 1. In case of BIP, I tagged incoming line as LRU (0) or MRU (1) according to given probability. And then evict one of the LRU lines. If all lines become MRU then I tag all of them as LRU.

Benchmark	**Miss Rate (%)**	**IPC**	**Speed_up**
602 (gcc)	73.09	2.554	1
603 (bwaves)	85.19	0.9992	0.9942
619 (lbm)	34.48	0.2201	1.0023
bc-0	79.666	0.1458	1.0014
sssp-3	73.47	0.4072	1.0005

Table 1: FIFO Policy

Benchmark	**Miss Rate (%)**	**IPC**	**Speed_up**
602 (gcc)	73.10	2.554	1
603 (bwaves)	85.24	1.005	1
619 (lbm)	35.58	0.2196	1
bc-0	79.08	0.1456	1
sssp-3	71.92	0.407	1

Table 2: LRU Policy

Benchmark	**Miss Rate (%)**	**IPC**	**Speed_up**
602 (gcc)	73.38	2.559	1.0019
603 (bwaves)	87.34	1.001	0.9960
619 (lbm)	85.23	0.2202	1.0027
bc-0	85.82	0.1544	1.0604
sssp-3	79.14	0.438	1.0762

Table 3: LFU Policy

Benchmark	**Miss Rate (%)**	**IPC**	**Speed_up**
602 (gcc)	73.27	2.555	1.0004
603 (bwaves)	85.83	1.005	1
619 (lbm)	84.17	0.2193	0.9986
bc-0	81.23	0.155	1.06456
sssp-3	74.39	0.4298	1.0560

Table 4: BIP Policy (eps=0)

Benchmark	**Miss Rate (%)**	**IPC**	**Speed_up**
602 (gcc)	73.20	2.553	0.9996
603 (bwaves)	85.58	1.005	1
619 (lbm)	58.56	0.2194	0.9991
bc-0	82.53	0.1466	1.0069
sssp-3	72.45	0.41	1.0074

Table 5: BIP Policy (eps=0.25)

Benchmark	**Miss Rate (%)**	**IPC**	**Speed_up**
602 (gcc)	73.17	2.553	0.9996
603 (bwaves)	85.47	1.004	0.9990
619 (lbm)	46.7	0.2198	1.0009
bc-0	81.76	0.1452	0.9973
sssp-3	72.46	0.4066	0.9990

Table 6: BIP Policy (eps=0.5)

Benchmark	**Miss Rate (%)**	**IPC**	**Speed_up**
602 (gcc)	73.13	2.553	0.9996
603 (bwaves)	85.32	1.005	1
619 (lbm)	40.54	0.2199	1.0014
bc-0	80.45	0.1452	0.9973
sssp-3	72.21	0.4066	0.9990

Table 7: BIP Policy (eps=0.75)

Benchmark	Miss Rate (%)	IPC	Speed_up
602 (gcc)	73.09	2.553	0.9996
603 (bwaves)	85.25	1.005	1
619 (lbm)	37.25	0.2193	0.9986
bc-0	78.96	0.1455	0.9993
sssp-3	71.83	0.4072	1.0005

Table 8: BIP Policy (eps=1)

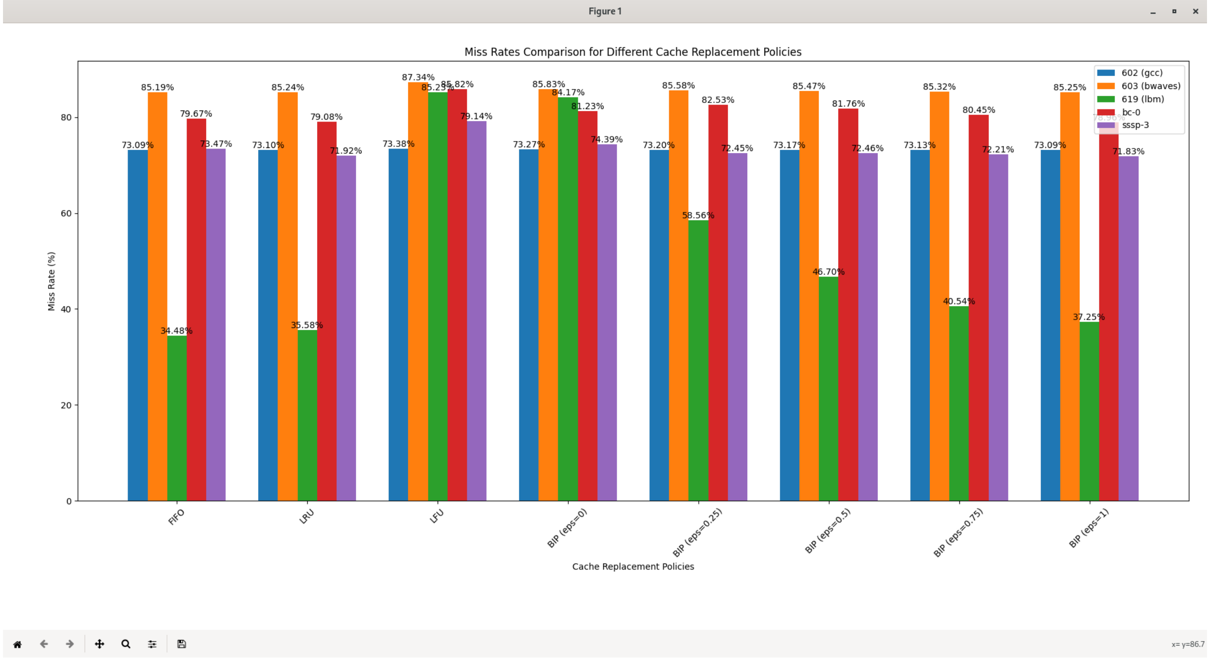

Figure 1: Miss Rates Comparison

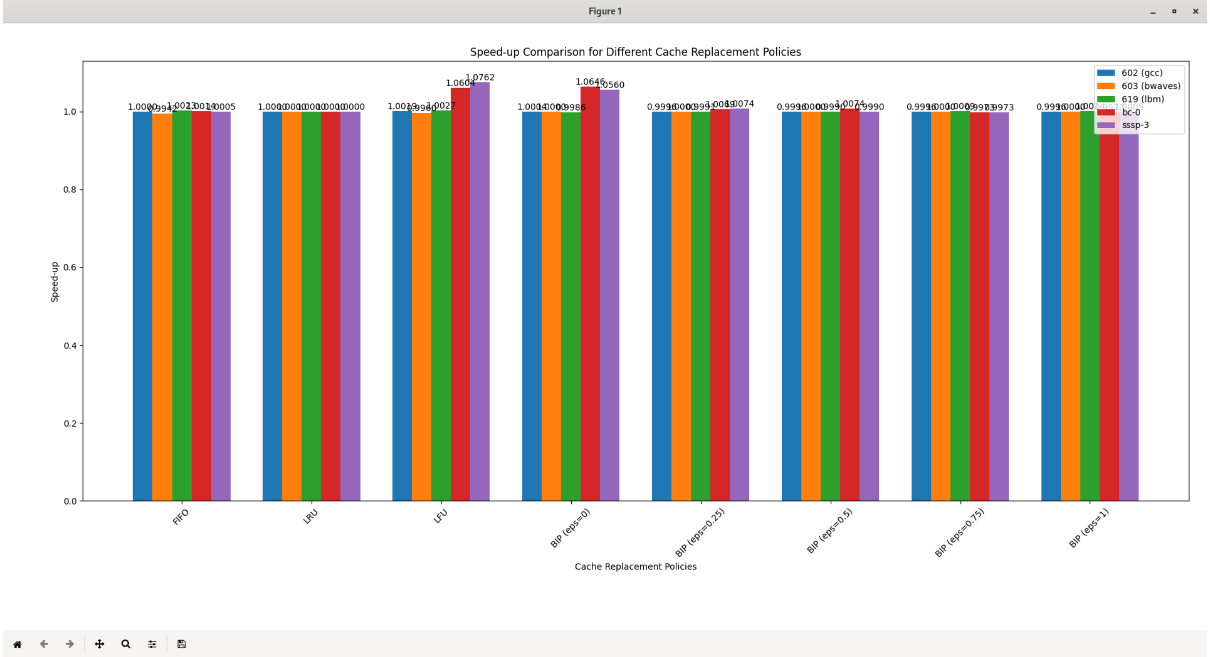

Figure 2: Speed up Comparison

Explanation :

We can see that FIFO gives the lowest miss rate for almost all the traces. Whereas LRU is also very close to FIFO. But LFU performs poorly. I think the performance of these policies are highly dependent on trace files. The performance of replacement policies depends on the pattern of how processor is requesting for addresses. In BIP policy, we can see that the miss rate decreases as epsilon increases. That means as the probability to place cache in MRU increases the miss rate decreases. The potential reason for this could be that the processor is requesting for nearby addresses so if line enter MRU with high probability, it is evicted with less probability. Thus less miss rate if processor requests for nearby addresses of that line.

Question 2

For stream prefetcher, I first made 2 structures namely monitory region and stream. After that I applied the logic given in the problem statement. To optimize it further I tried with varying prefetch degree and prefetch distance. As I increased prefetch degree, the IPC significantly improved. Also the accuracy improved on all trace files. The IPC became constant after some value of degree. In my case I got it around 20. If we increase the prefetch distance further then the accuracy decreases. So I set prefetch distance to be about size of 3 catch lines. To optimize further I allowed multiple streams to form. If a miss does not lie in any of streams prefetch distance, then that miss can create its own stream. This improved my code a lot. For some traces especially bc-0, increasing the degree reduced my IPC value.

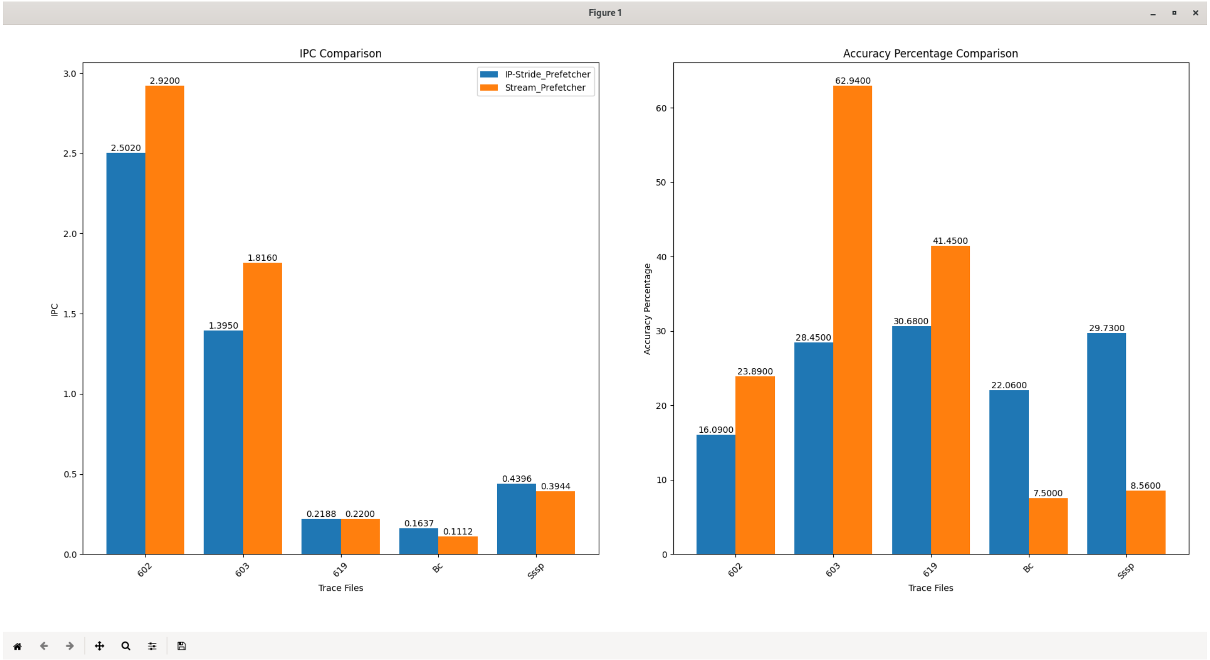

Figure 3: IPC and Accuracy Comparison

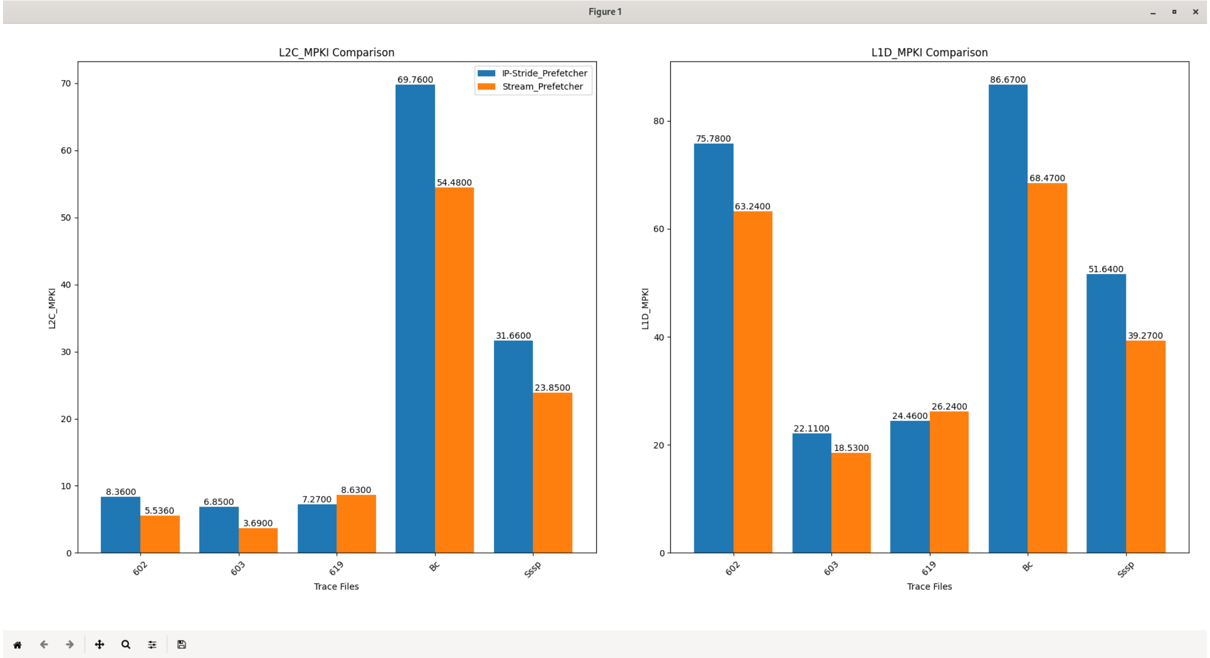

Figure 4: L1C-MPKI and L1D-MPKI Comparison

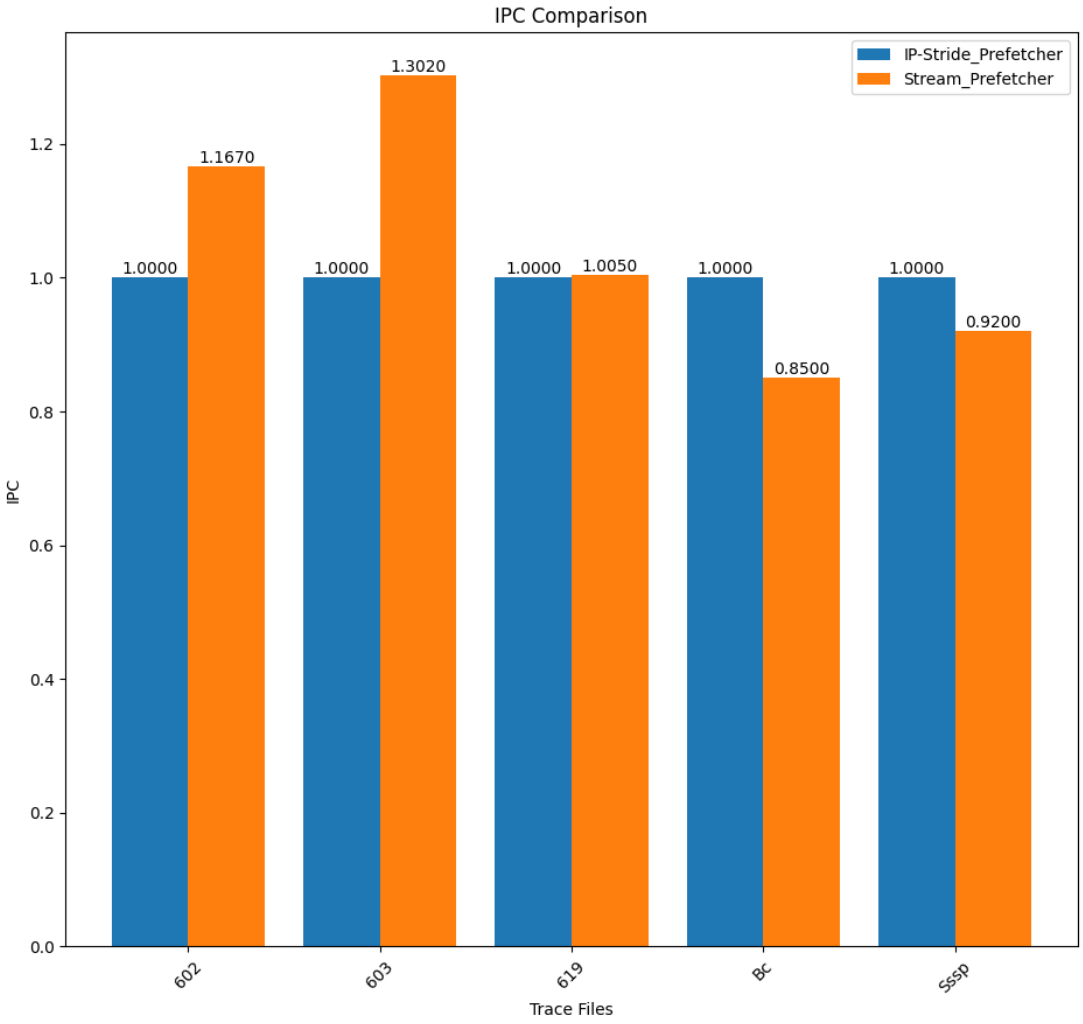

Figure 5: Speed-Up