DLDCA Lab4 Part2 Report

Jaladhi Joshi : 22B0994

6th November, 2023

0.1 Replacement Policies

The graph comparing the speedup of LFU,LRU,FIFO and BIP is as follows:

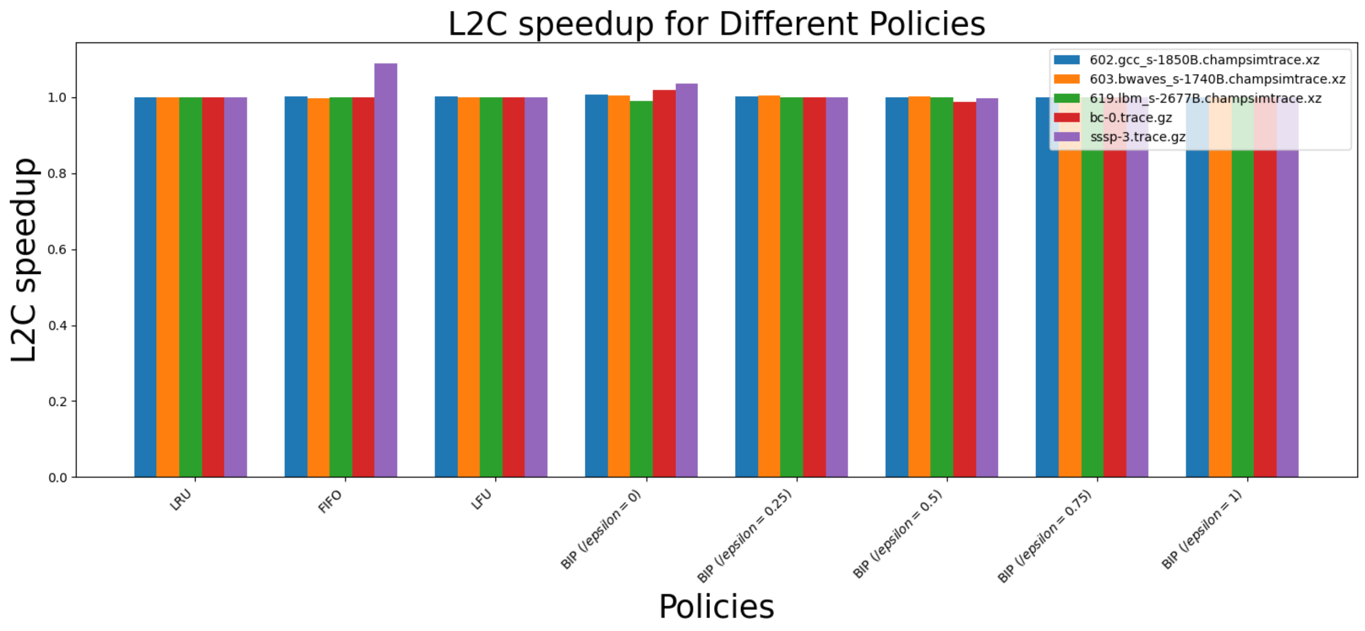

Figure 1: Speedup comparison

The graph comparing the L2C Miss Rate of LFU,LRU,FIFO and BIP is as follows:

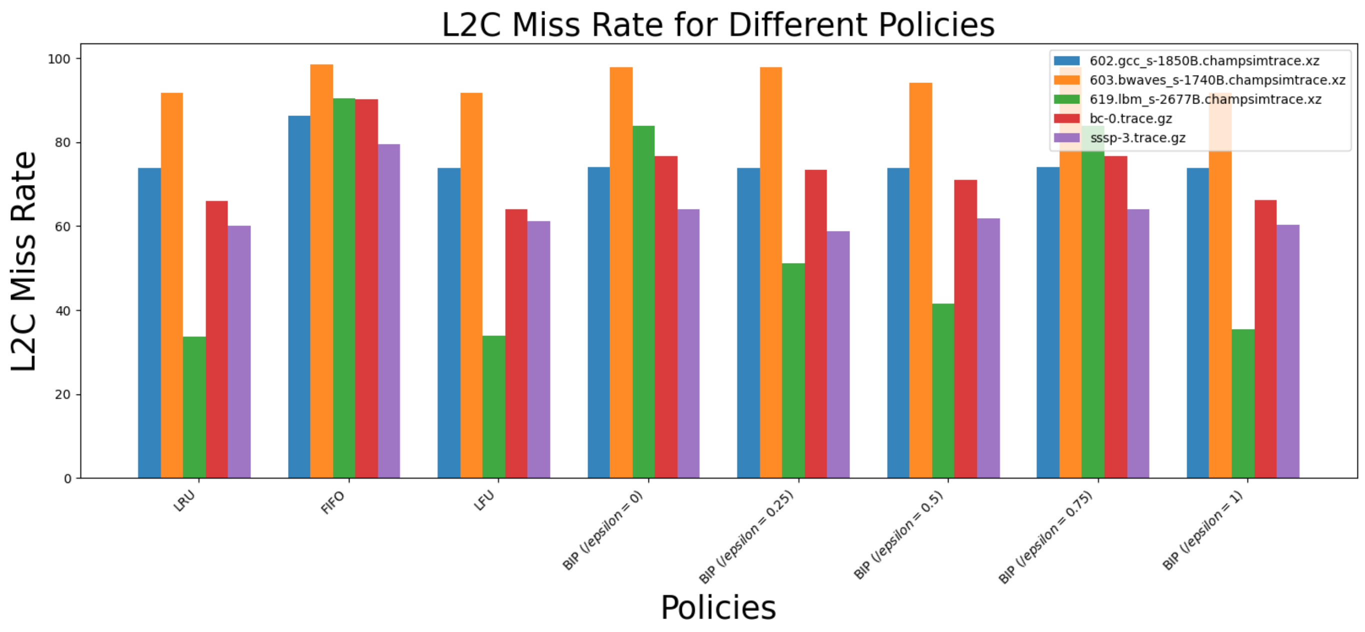

Figure 2: L2C Miss Rate Comparison

0.1.1 Analysis of patterns

- LRU is effective because it keeps the most recently used data in the cache and that can be needed again. It is very useful whenever we have loops or any cyclic pattern,this method saves a lot of time

- LFU has a higher miss rate compared to others as it may lead to early eviction of data that might be required in some patterns or cyclic order.
- FIFO is unaffected by frequency and timings of accesses . As the name suggests it is just considering the eviction of the first one to come into the cache . This becomes inefficient when there are multiple patterns but still this is somwhat consistent
- BIP can work with linear and cyclic patterns according to the values of epsilon.As the value of epsilon increases it can easily recognise linear and cyclic patterns hence becomes more efficient and has less miss rates

0.2 Stream Prefetcher

I have used the consecutive misses approach that consecutivity of misses is necessary to define a monitoring region.

I have implemented a table that stores the monitoring regions, a map from an index to a bool, bool when true indicates it has already defined a monitoring region and false indicates otherwise.

Whenever I get a miss, I check if its already in a monitoring region or not ,if yes I prefetch else I check if it can define another monitoring region or not, if yes I update that region to the table.

If table is full,I use lru concept to replace one of the entries of the table, which I find by the map last_used_cycles

I have improved the performance by trying and testing different values of Prefetch Degree and Prefetch Distance.

The best values I got are Prefetch Distance=10 and Prefetch Degree=100.Other values giving me decent results were Prefetch Distance=7 and Prefetch Degree=81 and Prefetch Distance=12 and Prefetch Degree=121

The performance became better or deteriorated because there are multiple things happening here either you prefetch less and every time access through the clock cycles or you prefetch enough that the table is filled easily and you have to apply the lru removal again and again.

So we had to find a perfect balance between both of these things somewhat through trial and error and somewhat through common sense and observing patterns.

Speedupzaccuracy,L2C Load MPKI and L1D MPKI Comparison is as follows: Prefetcher Accuracy comparison: L2C Load MPKI Comparison: L1D MPKI Comparison:

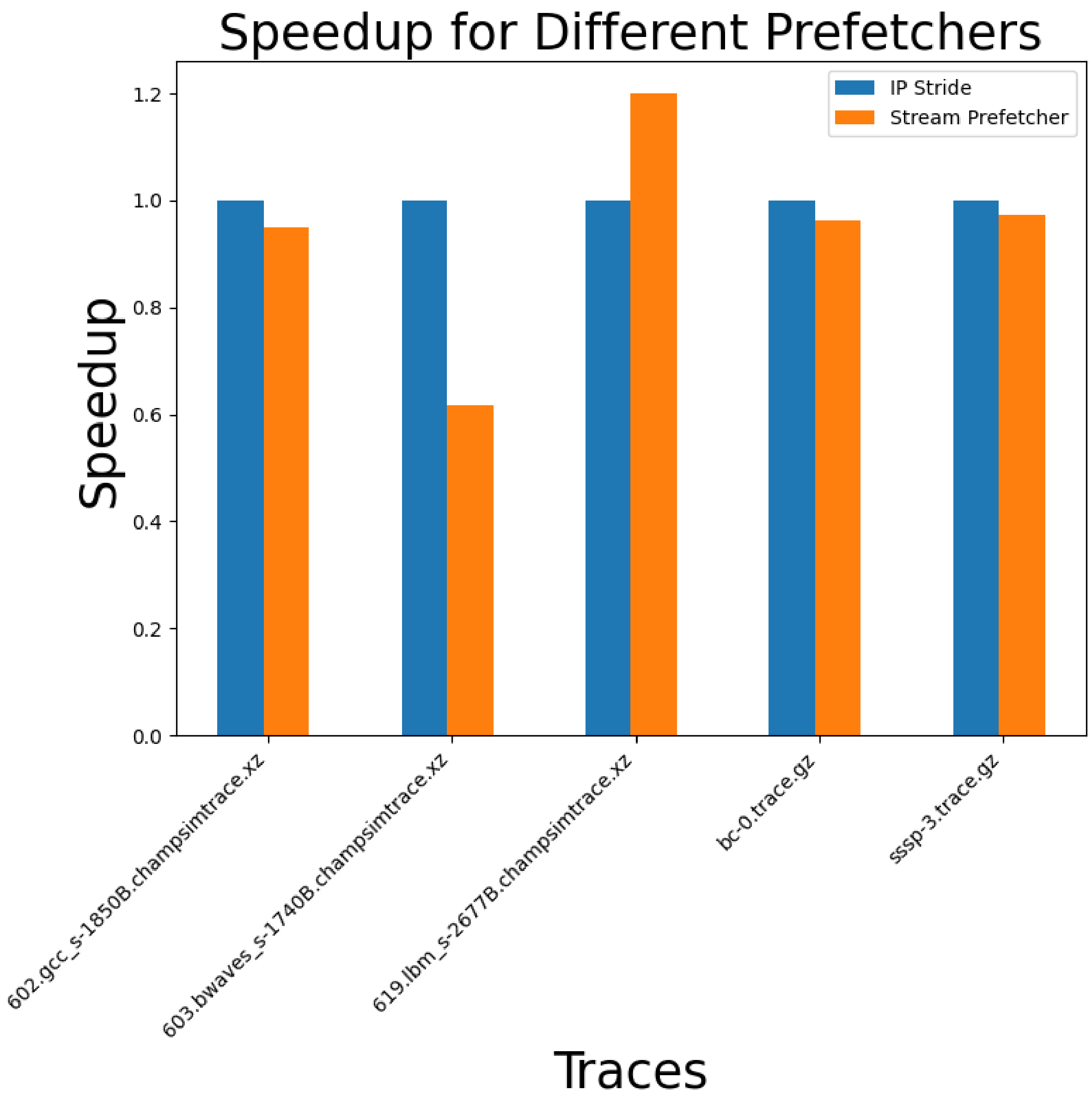

Figure 3: Speedup comparison

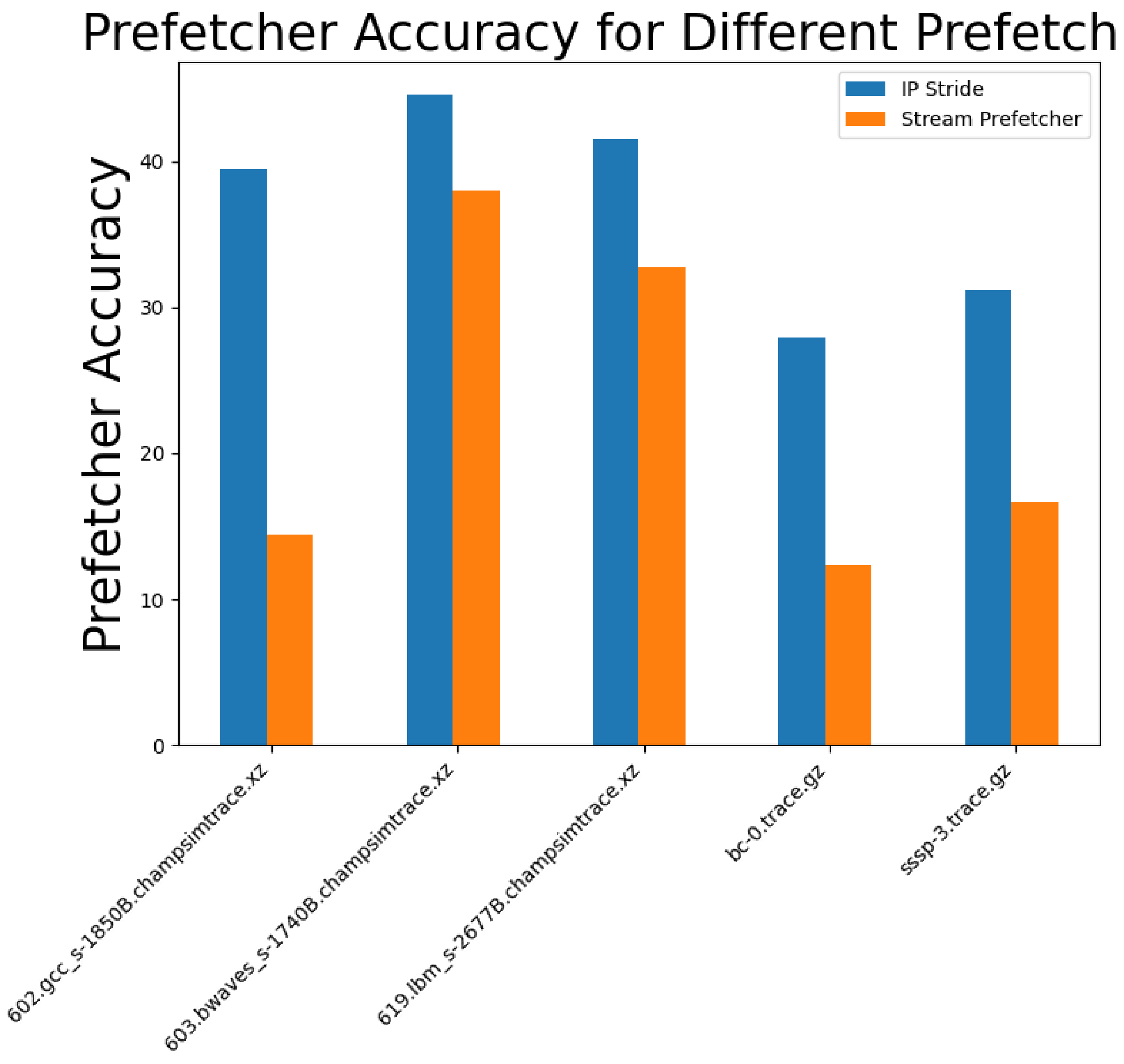

Figure 4: Accuracy comparison

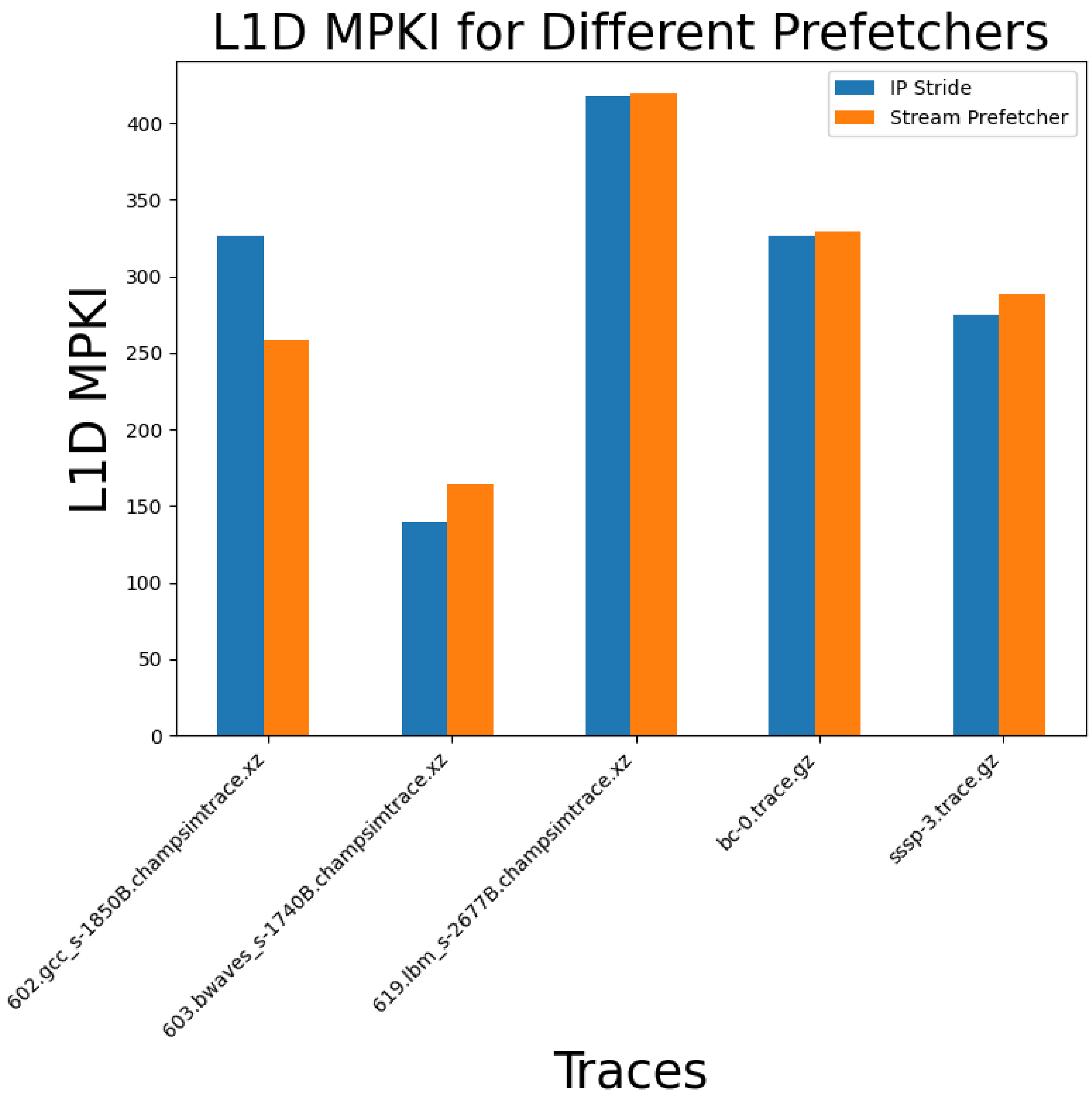

Figure 5: L2C Load MPKI comparison

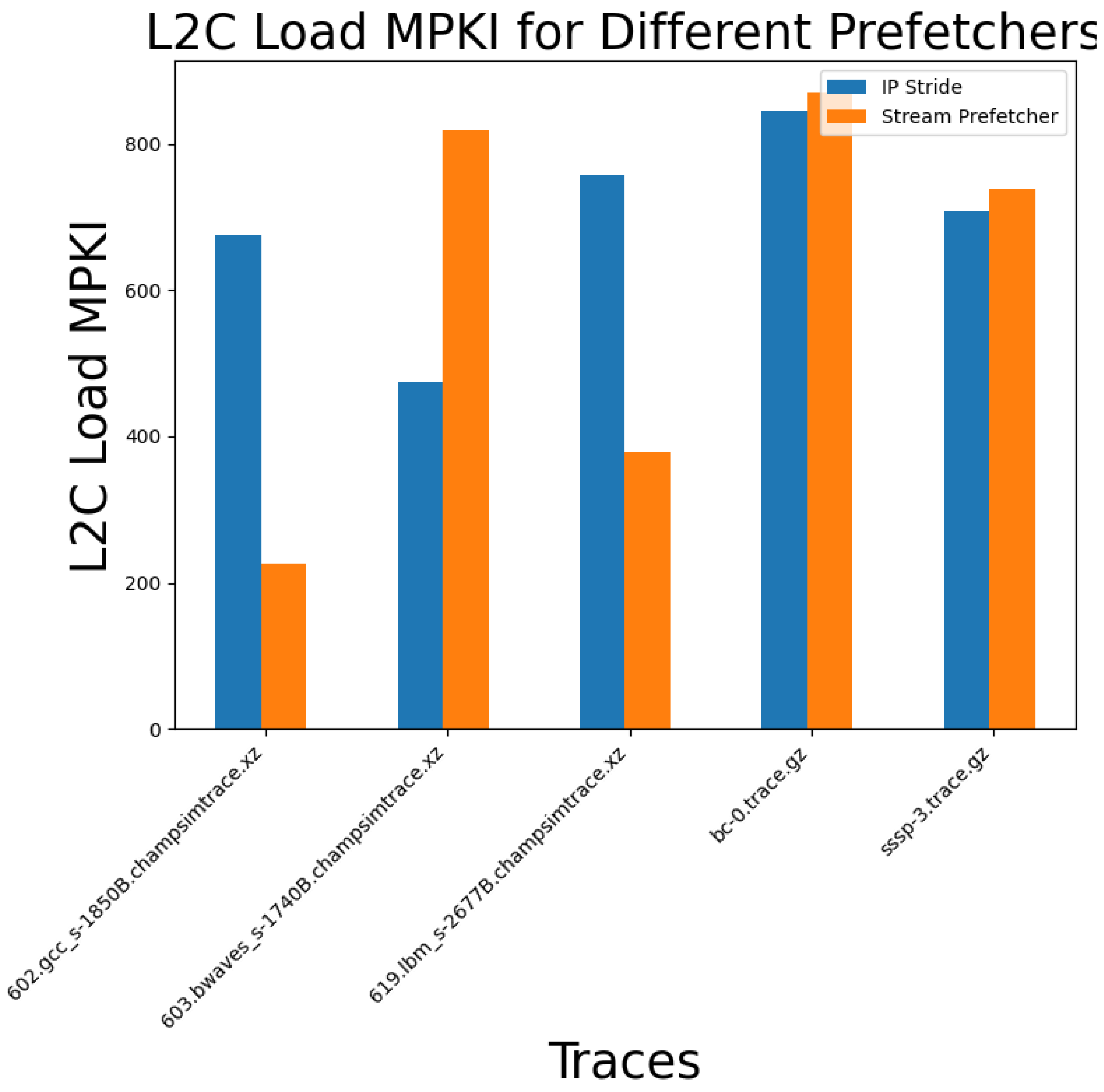

Figure 6: L1D MPKI Comparison

0.2.1 Analysis of Paterns

- The speedup of IP Stride and Stream Prefetcher are not very different for all the traces and it is calculated by $\frac{\text{IPC of Prefetcher}}{\text{IPC Of IP Stride Prefetcher}}$
- IP stride has significantly larger accuracy than Stream Prefetcher for almost all traces which is calculated by $100 * \frac{\text{Useful}}{\text{Issued}}$
- L1D MPKI pattern is somewhat same for both the prefetchers and is calculated by $1000 * \text{L1D}\frac{\text{Miss}}{\text{Total}}$
- L2C MPKI pattern is very irregular and is calculated similar to the L1D pattern $1000 * \text{L2C}\frac{\text{Miss}}{\text{Total}}$