Cache Simulator Results

Jacob Haynes

ECEN 4593 Final Project

Fall 2012: Prof. Pleszkun

1 FIGURES

1.1 RUNNING TIME BY CACHE CONFIGURATION

In order of increasing total time (Fastest First More explanation in other figures to come):

- L1: 16K Fully Associative + L2: 32K Fully Associative **[All-FA]**
- L1: 16K 2Way Set Associative + L2: 64K Direct Mapped **[L2-Big]**
- L1: 16K 2 Way Set Associative + L2: 32K 4 Way Set Associative **[2-4 Way]**
- L1: 16K 2 Way Set Associative + L2: 32K 2 Way Set Associative **[All-2Way]**
- L1: 16K Direct Mapped + L2: 32K 2 Way Set Associative **[L2-2way]**
- L1: 16K 2 Way Set Associative + L2: 32K Direct Mapped **[L1-2Way]**
- L1: 16K Direct Mapped + L2: 32K Direct Mapped **[Base]**

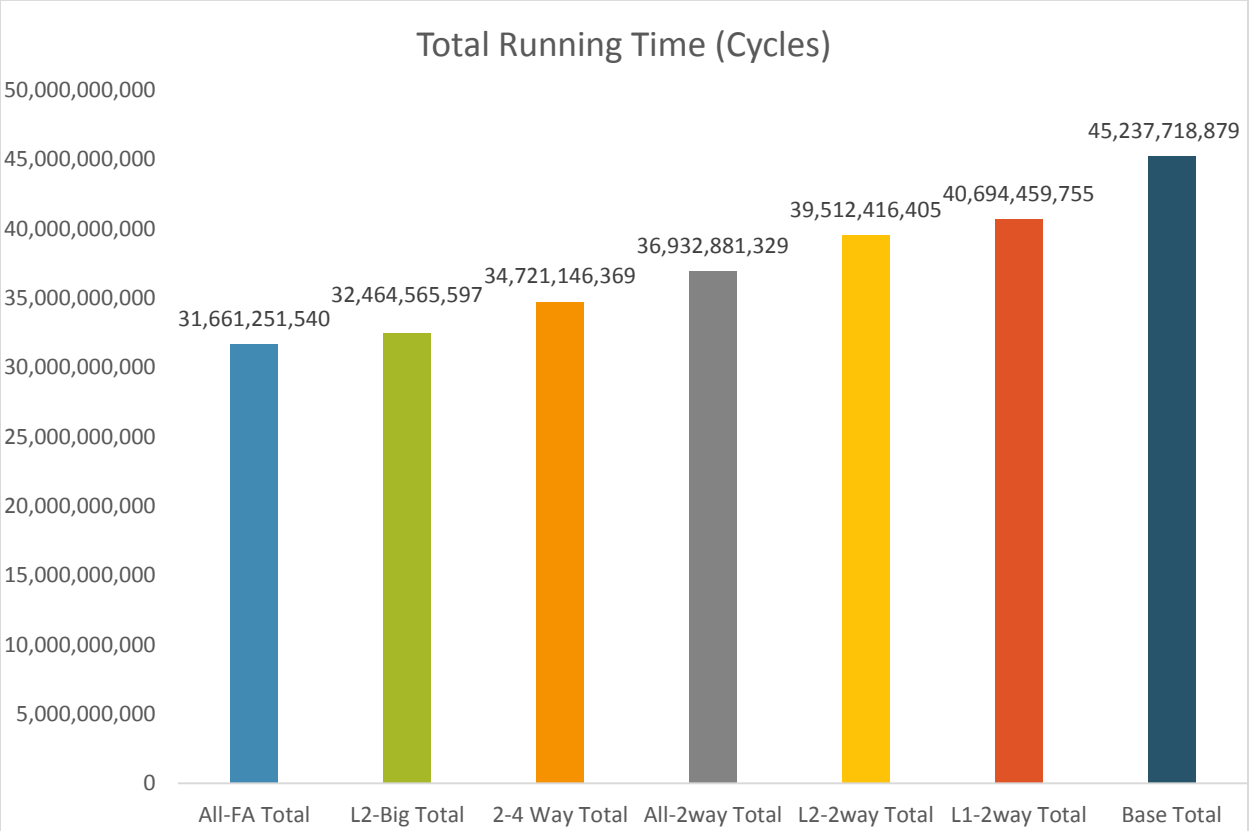

Figure 1

1.2 Cycles Per Instruction By Cache Configuration

The Big L2 cache configuration has an interesting patter. It has a higher CPI for store cycles than 2-4 Way, and a lower Load CPI than All-FA. It does not follow the pattern of the rest of the types (that CPI for all is lower than the slower configs and higher than the faster configs). The reason, as seen in 1.3, is that there is a lower eviction rate for the L2 cache than any other configuration, so it does not have to go to main memory as often, but the L1 cache is not as efficient as a FA, so it ends up slower overall.

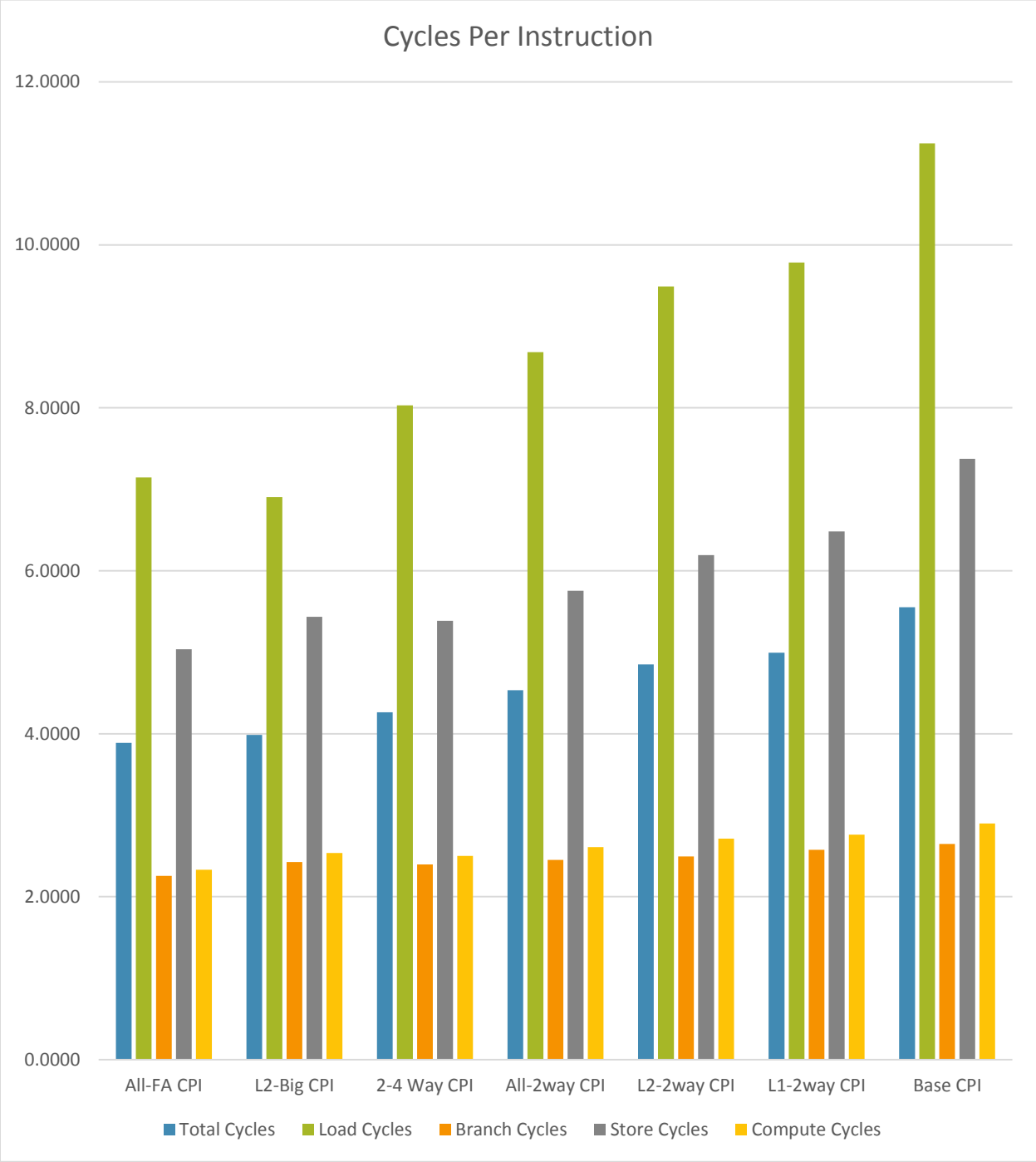

Figure 2

1.3 Misses and Evictions By Cache Configuration

See 1.2 for more in depth explanation of this data.

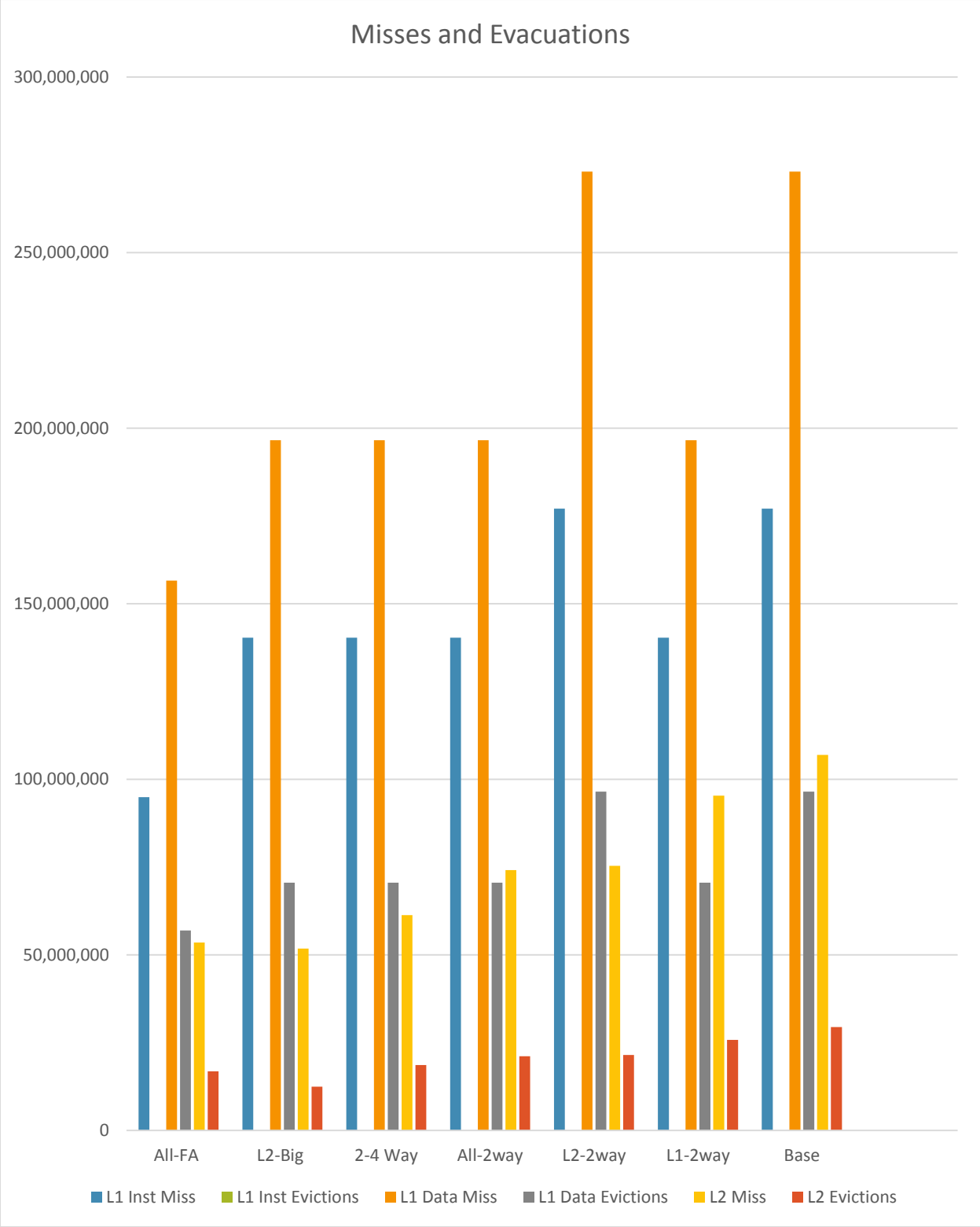

Figure 3

1.4 Hits By Cache Configuration

See 1.2 for more in depth explanation.

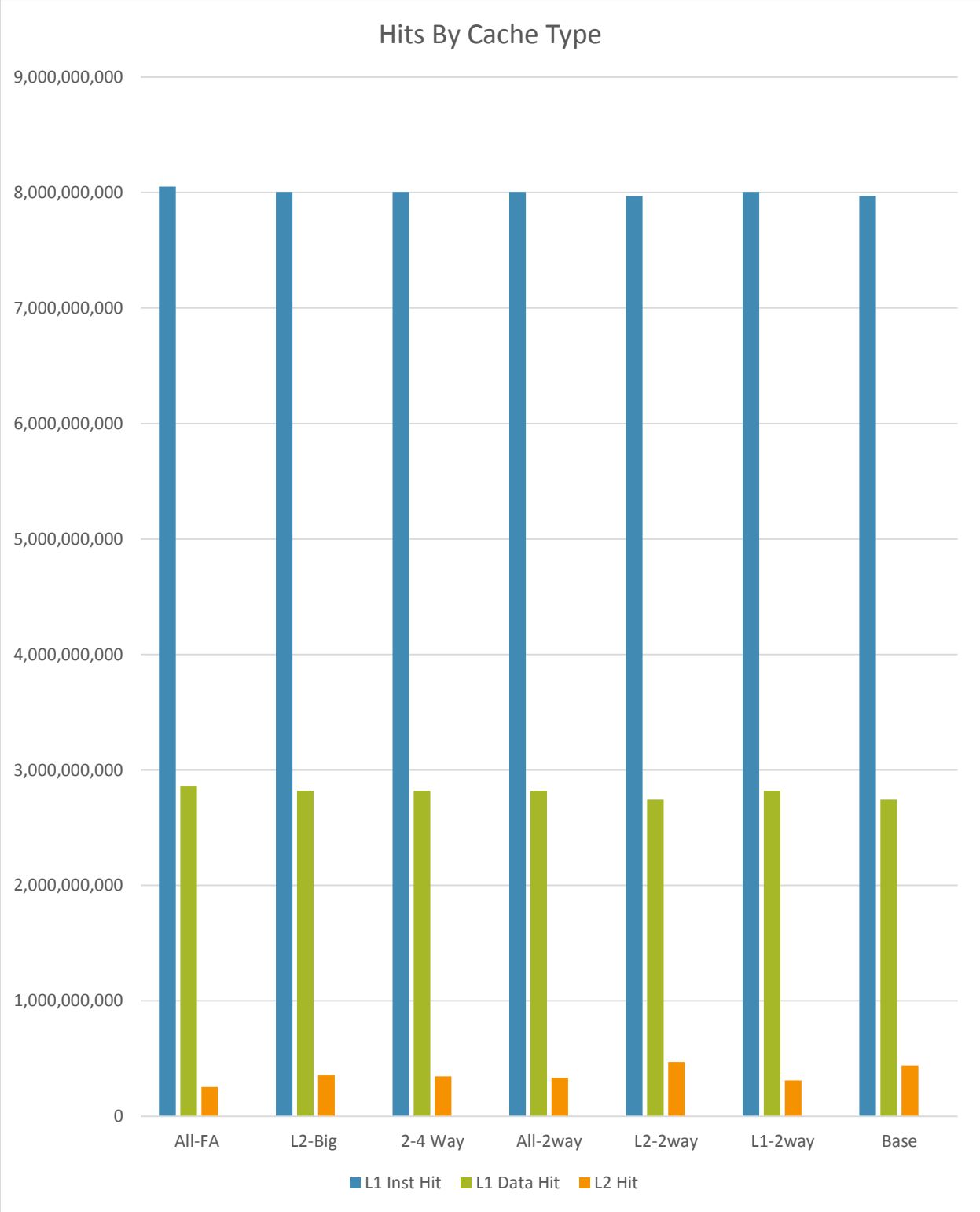

Figure 4

1.5 CPI BY L1 CACHE TYPE

Figure 5, below, shows—as expected—that Fully Associative is more efficient than either 2 way or direct mapped. There is a slight issue because this data is also dependent on the speed of the L2 cache and main memory, so these figures are not absolute. As seen in 1.6, the Evacuations and Misses for Fully Associative are lower, confirming that it is indeed the best cache type.

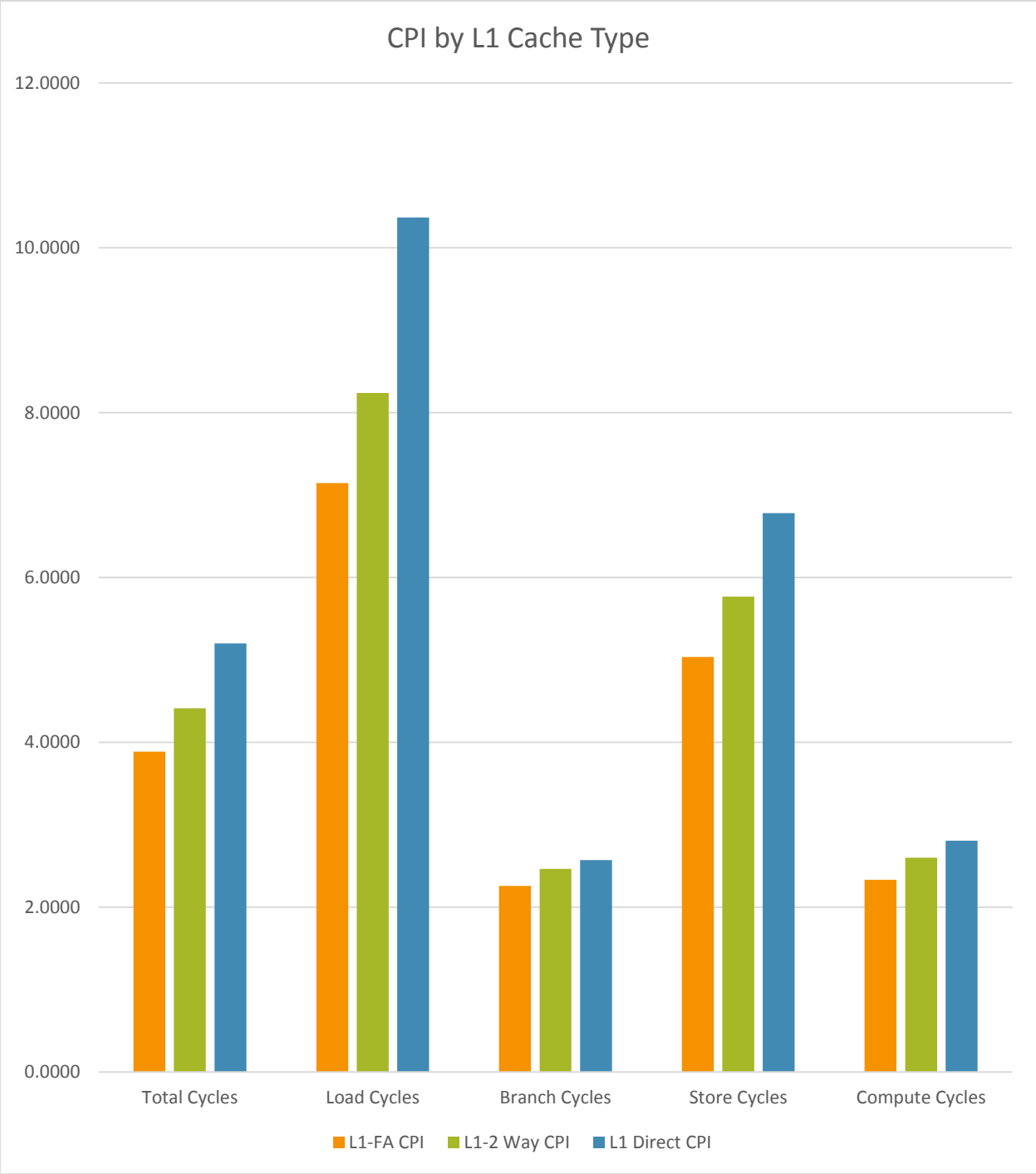

Figure 5

1.6 Misses and Evacuations Rate By L1 Cache Type

See in depth analysis in 1.5

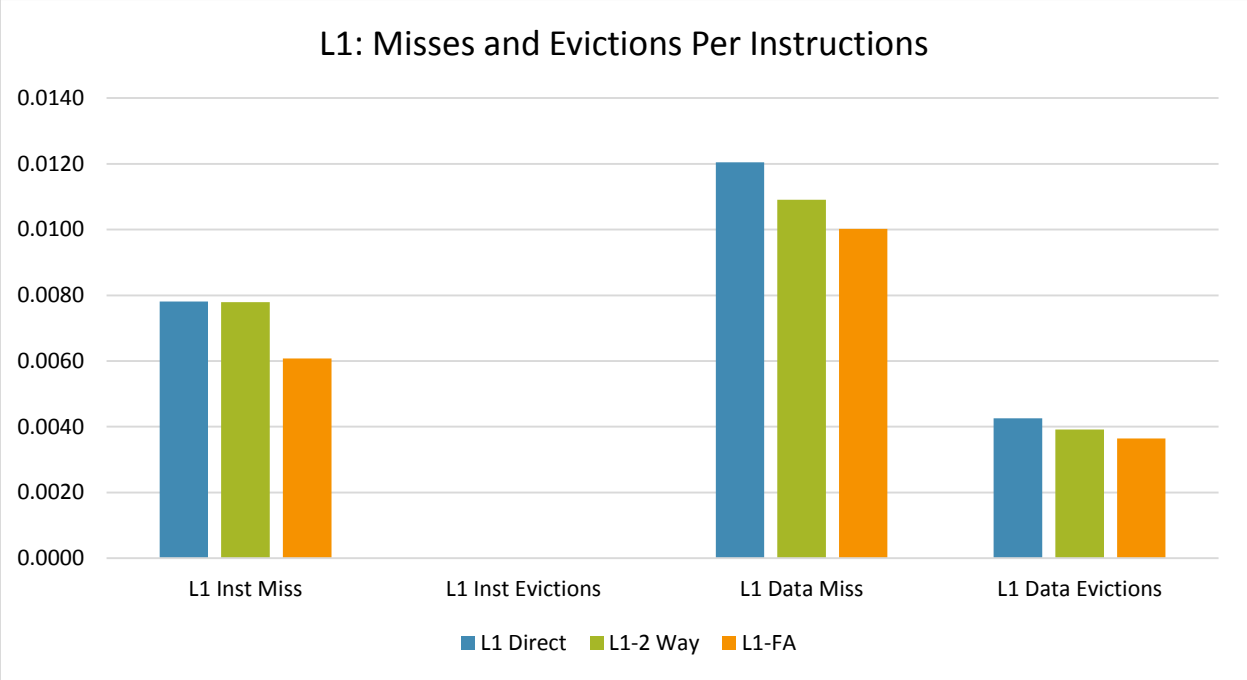

Figure 6

1.7 Hits Per Instruction By L1 Cache (Data Cache Only)

See in depth analysis in 1.5.

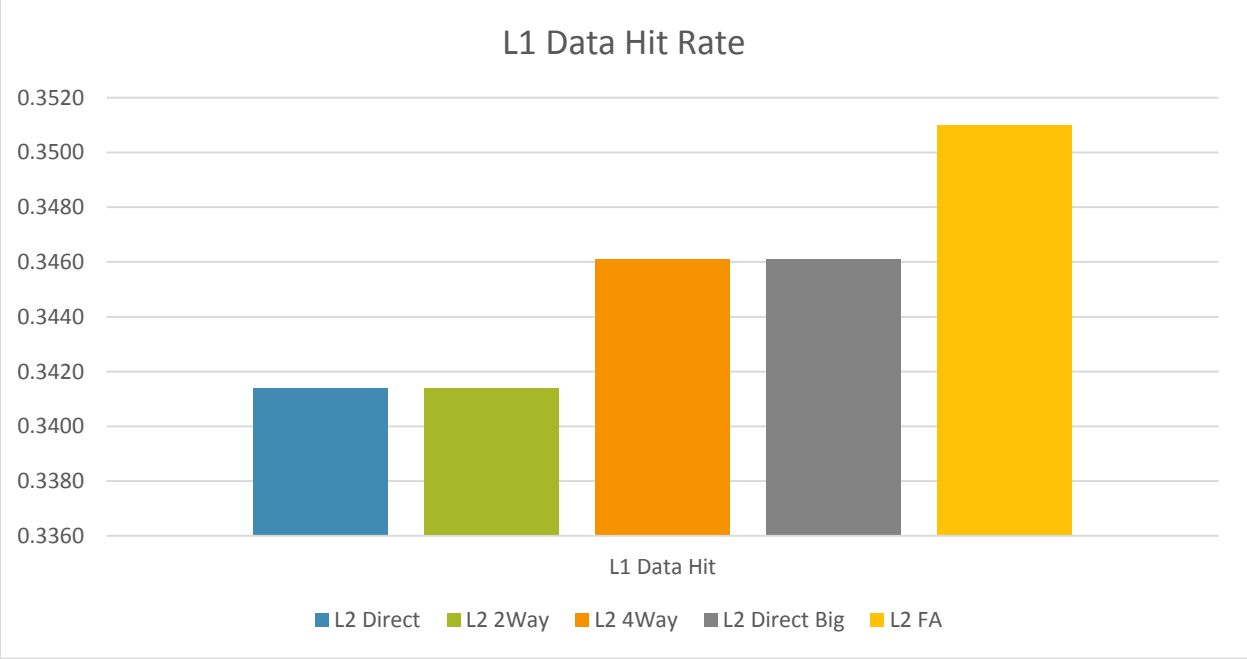

Figure 7

1.8 CPI BY L2 CACHE TYPE

Figure 8, below, shows the CPI by L2 cache type. The data in this figure and in 1.9/1.10, confirm that the Fully Associated Cache and Big L2 cache do a very good job, having low eviction and miss rate. The miss rate on the small direct mapped cache and 2 way set associative cache are not efficient.

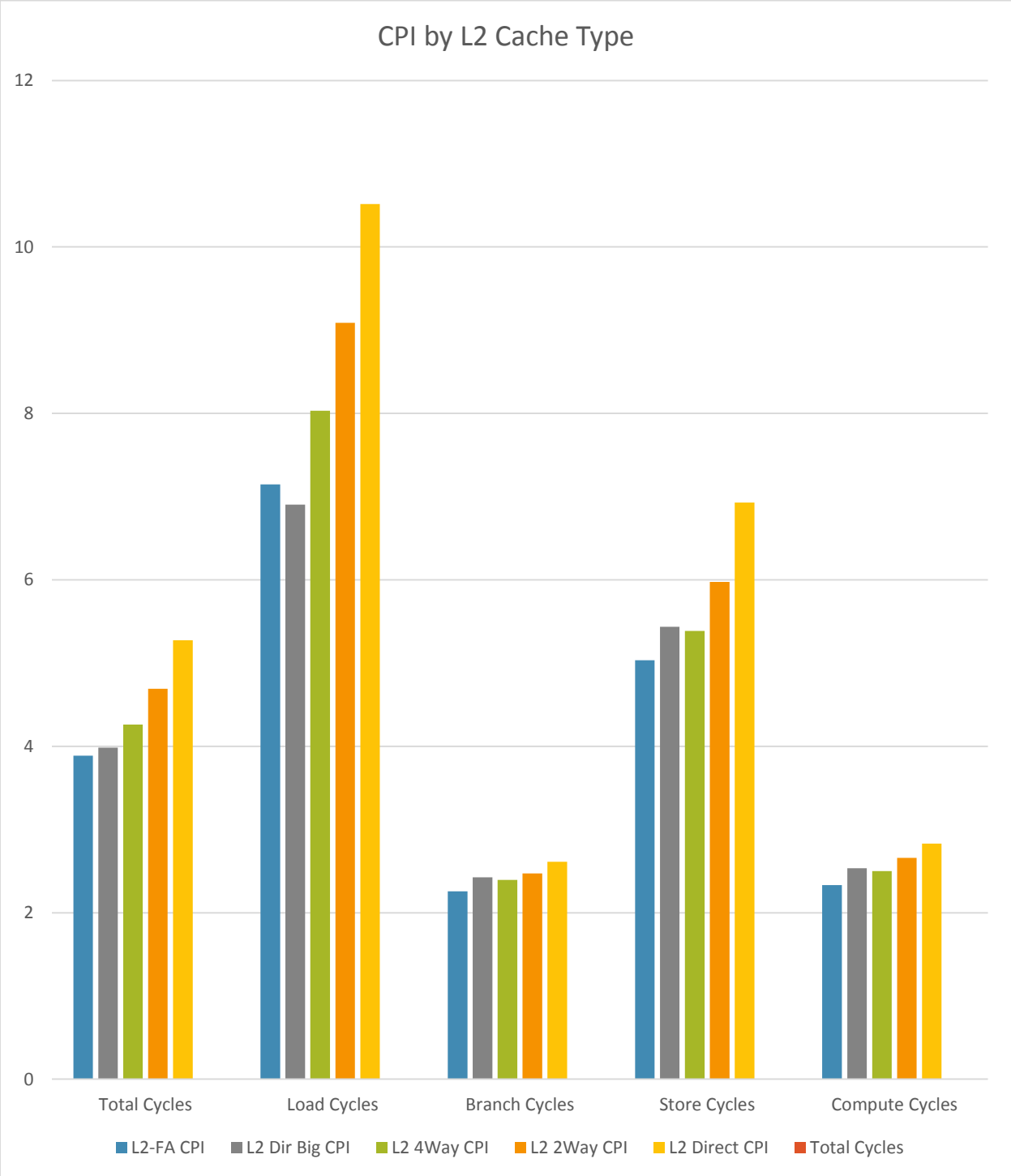

Figure 8

1.9 Misses and Evacuations Rate By L2Cache

Note: Load Cycles and Store Cycles on Big L2 Cache

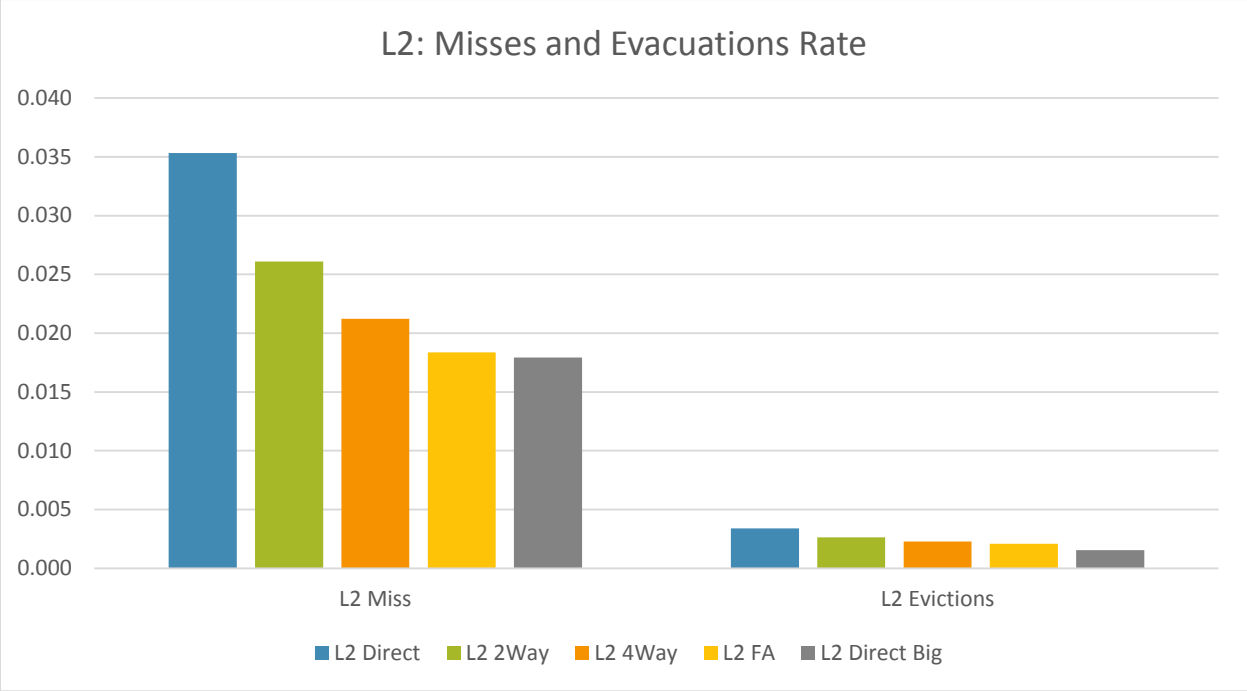

Figure 9

1.10 Hit Rate By L2Cache

The hit rate on the fully associative L2 cache is very low compared to what might be expected. The most likely reason for this is that the L1 Fully Associated Cache is very efficient at storing data and is more likely to ask for data that has not yet loaded.

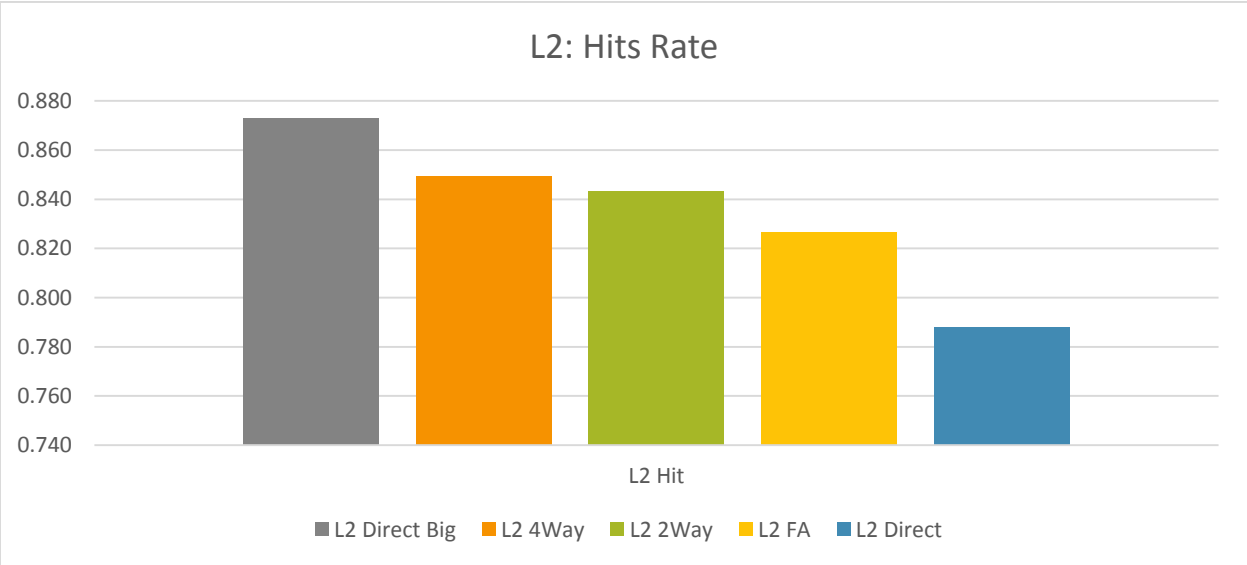

Figure 10

1.11 Total Cycles By Path Size

In figure 11, you can see the decrease in time when you increase the main memory path size and decrease the time it takes to transfer from L2 to Main Memory. In figure 1.12, you can see that a decreased main memory access time can make up for less efficient cache schemes, see conclusion for deeper discussion.

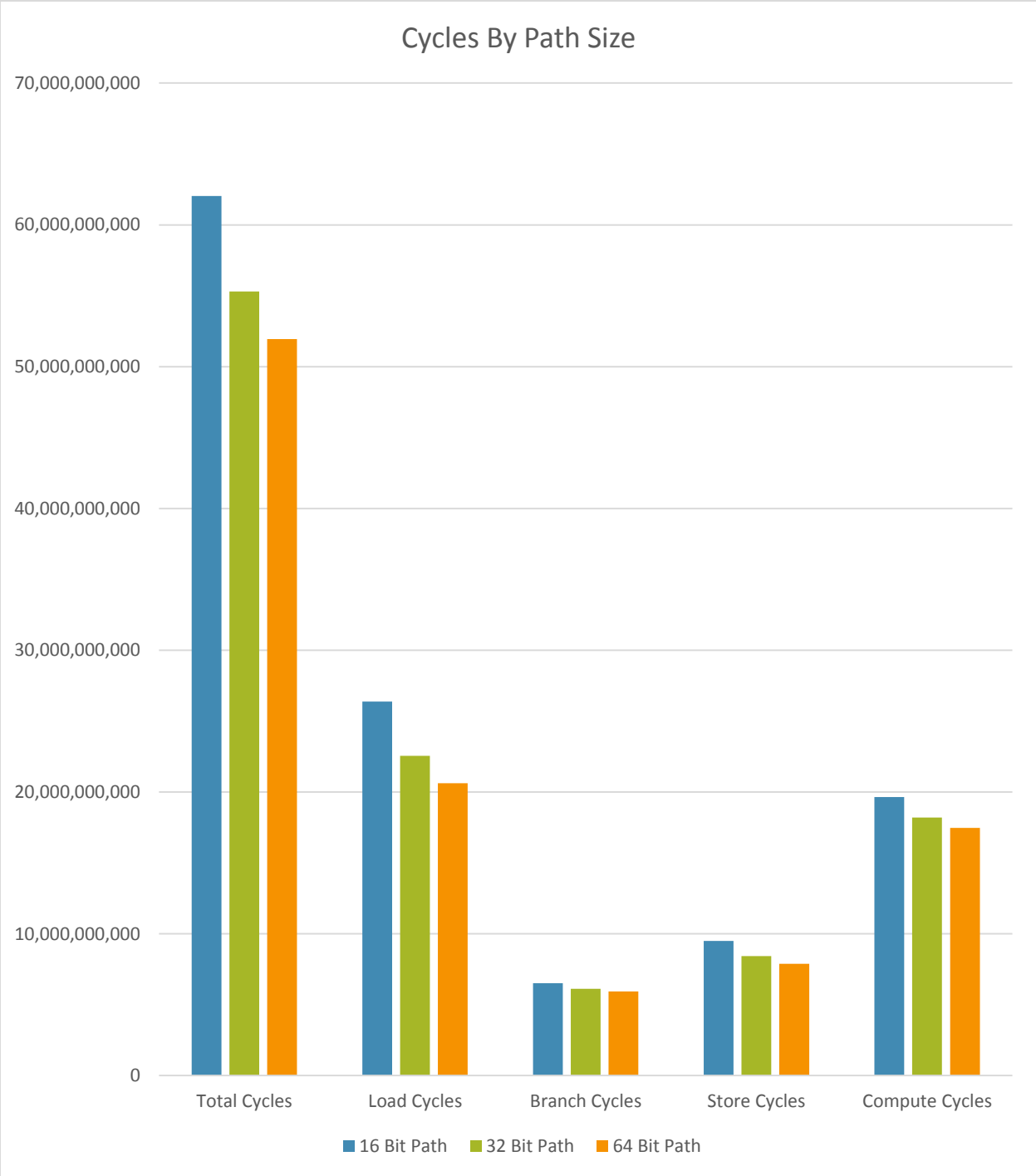

Figure 11

1.12 Misses and Evictions By Cache Configuration

See Conclusion for more in depth analysis.

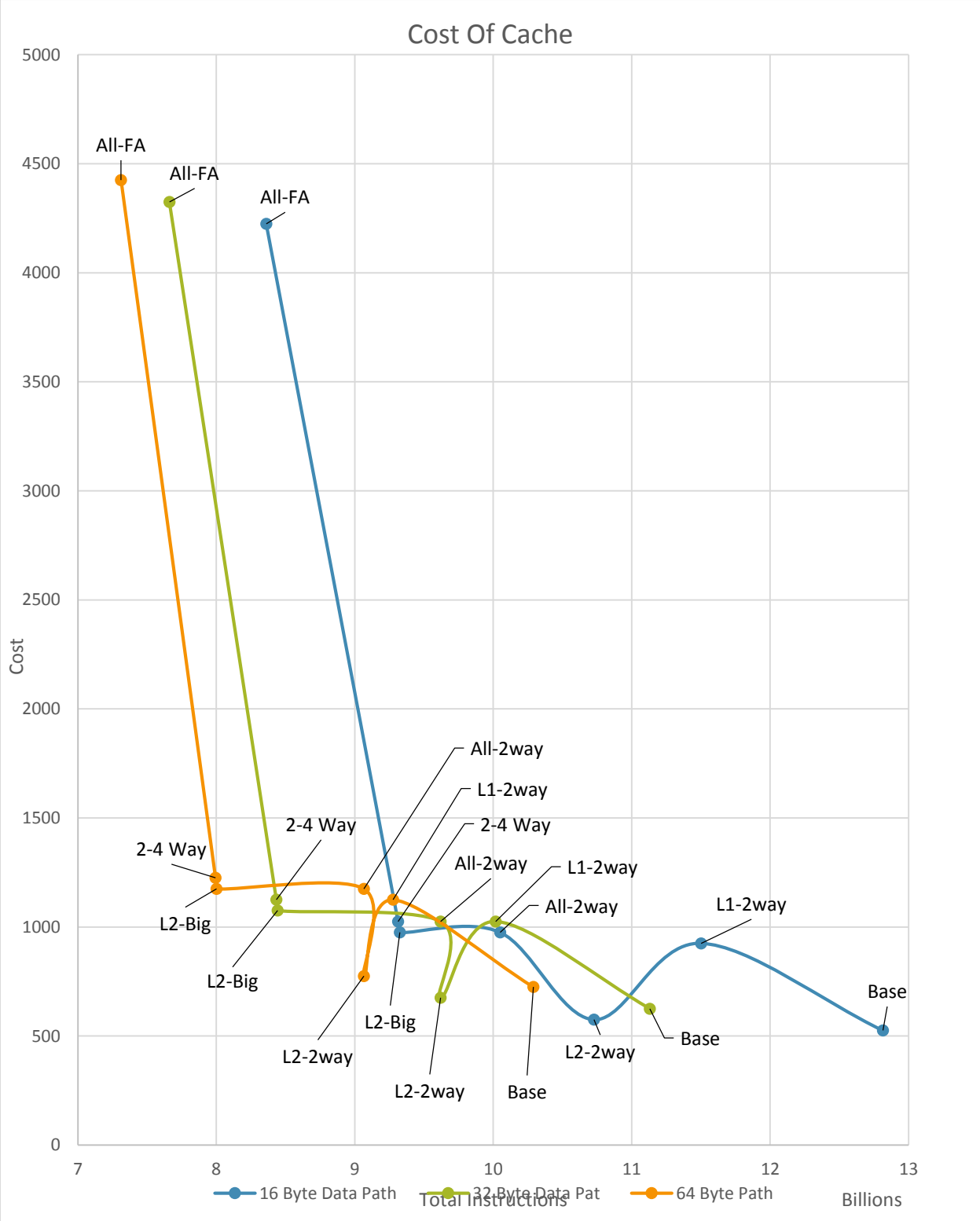

Figure 12

2 RAW DATA

See Attachments

3 CONCLUSION

Figure 12 shows the overall time based vs. the cost of each configuration. The first major conclusion that can be made from this figure is that it is very cost efficient to increase main memory size: increasing from a 16 byte path to a 64 byte bandwidth increase only costs $200, but can cut down total runtime by 20%. It does appear that the return on increasing the path width increase, so further increasing it may not provide benefits. According to the figure, it appears that 16KB 2 Way Set Associated Cache along with a 64KB Direct Mapped Cache and a 64 Byte Data Path seems to provide a fast cache for a reasonable amount of money (the 2-4-Way config is not far behind). In terms of the most cost effective, that would be the L1-2 Way with 64 byte data path. L1-2Way is not cost effective, and the Base is too slow for the amount of money.

If forced to choose one configuration to use, I would chose the L2-Big config.

There are some questions that I would have before building this cache: Would a 4 Way set associative L1 cache be more cost effective than a 2 Way set associative L1 Cache? Can the L1 cache size be increased? Could the Instruction Cache be implemented as Direct with Data as set associative to save money? Would 128 Byte bandwidth increase cost effectiveness? Can two separate types of caches be implemented for L1I and L1D? Would allocating 12KB (or around) to Data Cache and 4KB (or around) to Instruction Cache be faster? (The miss rate on Data is higher than on Instruction cache).

My best guess is that an L1 Data cache with a 4 Way Set associative cache and a 64KB 2 Way L2 cache or an 8 Way 32KB L2 cache would provide performance much closer to Fully Associative without the massive cost. I would run more simulations exploring more configurations before settling on a single cache to use.