NC State University

Department of Electrical and Computer Engineering

ECE/CSC 506/406: Architecture of Parallel Computers

Fall 2022

Project #2: Coherence Protocols

by

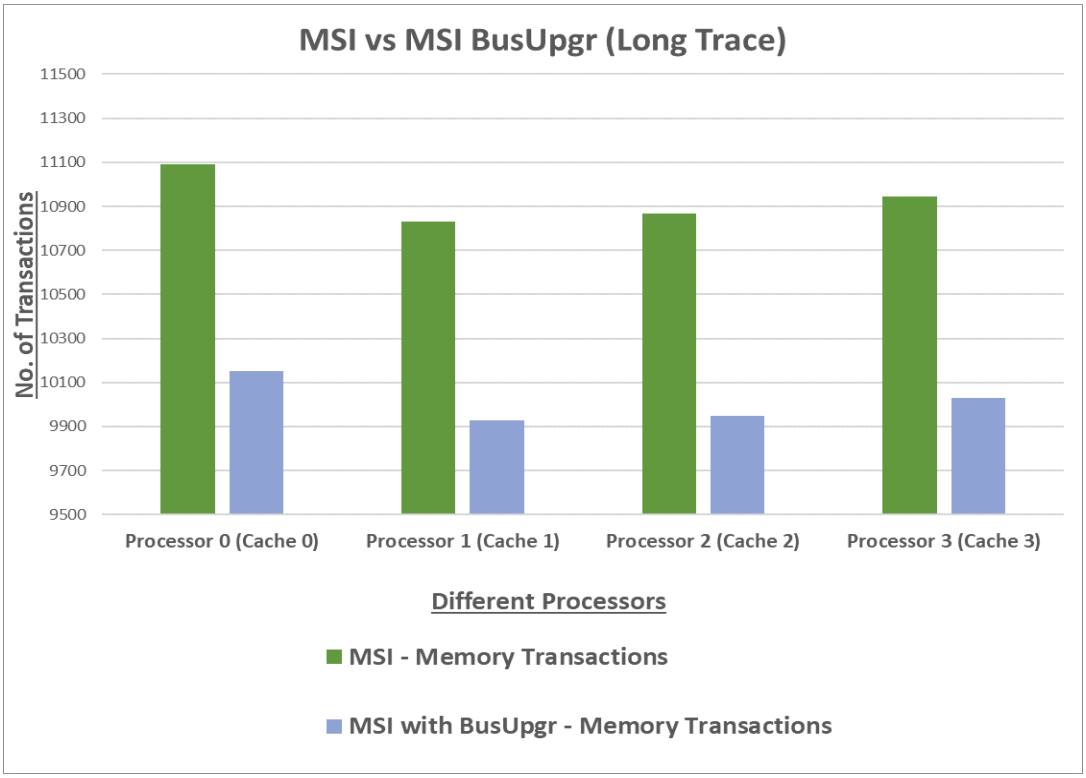
RAGHUL SRINIVASAN

(200483357)

GRAPH ANALYSIS:

Part 2. MSI vs MSI + BusUpgr

b. Comparison of MSI and MSI with BusUpgr optimization

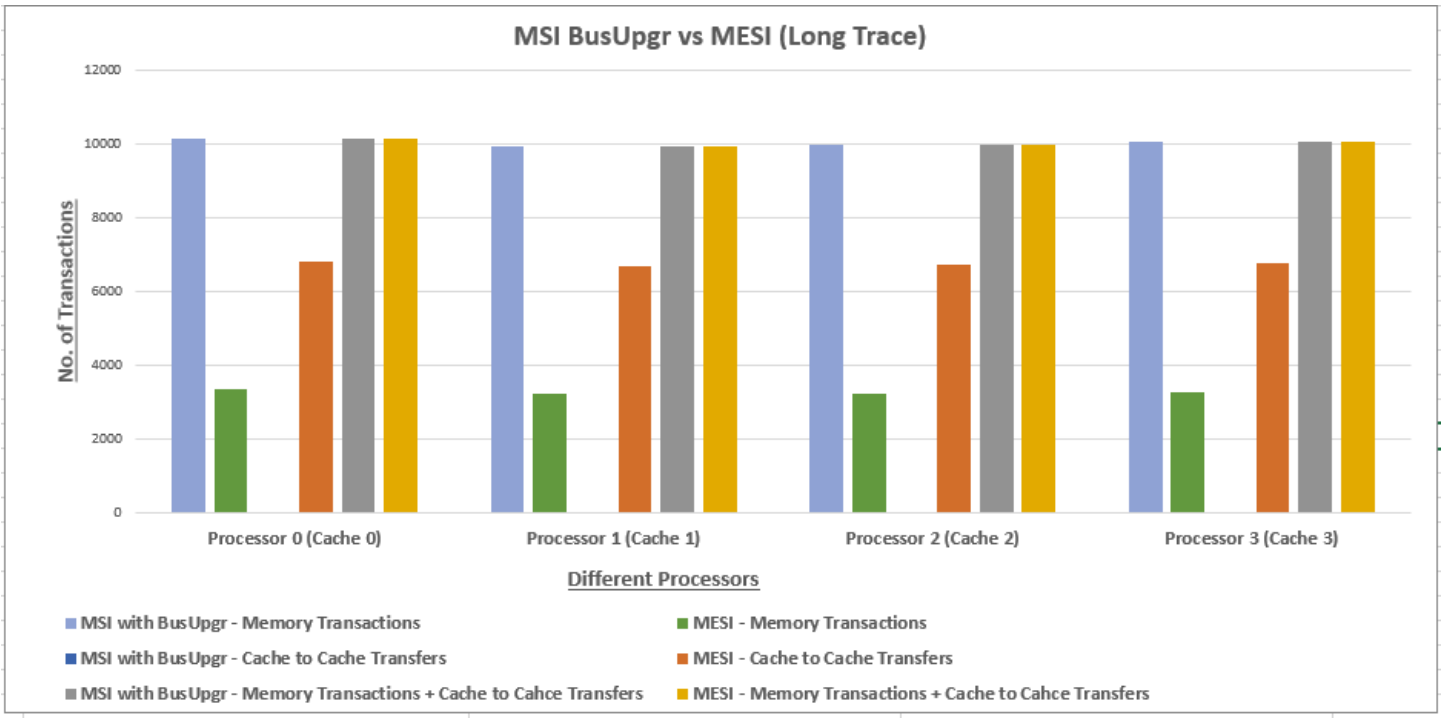


In the graph above, the number of memory transactions for **MSI with BusUpgr** Optimization is significantly lower than **MSI** for the same trace. This is because of the usage of the **BusUpgr** signal instead of the **BusRdx** signal when the cache block is changed from **Shared** to **Modified** as **BusUpgr** doesn't need the data to be transferred from Memory to cache hierarchy.

With reduced memory transactions for **MSI with BusUpgr** Optimization compared to **MSI** for a given trace, the introduction of the **BusUpgr** signal improves the performance to some extent.

Part 3. MESI

b. Comparison of MSI with BusUpgr optimization and MESI Protocols



From the above graph, the MESI protocol drastically reduces the number of memory transactions. This is because of the introduction of **Cache-to-Cache transfer** in the MESI protocol.

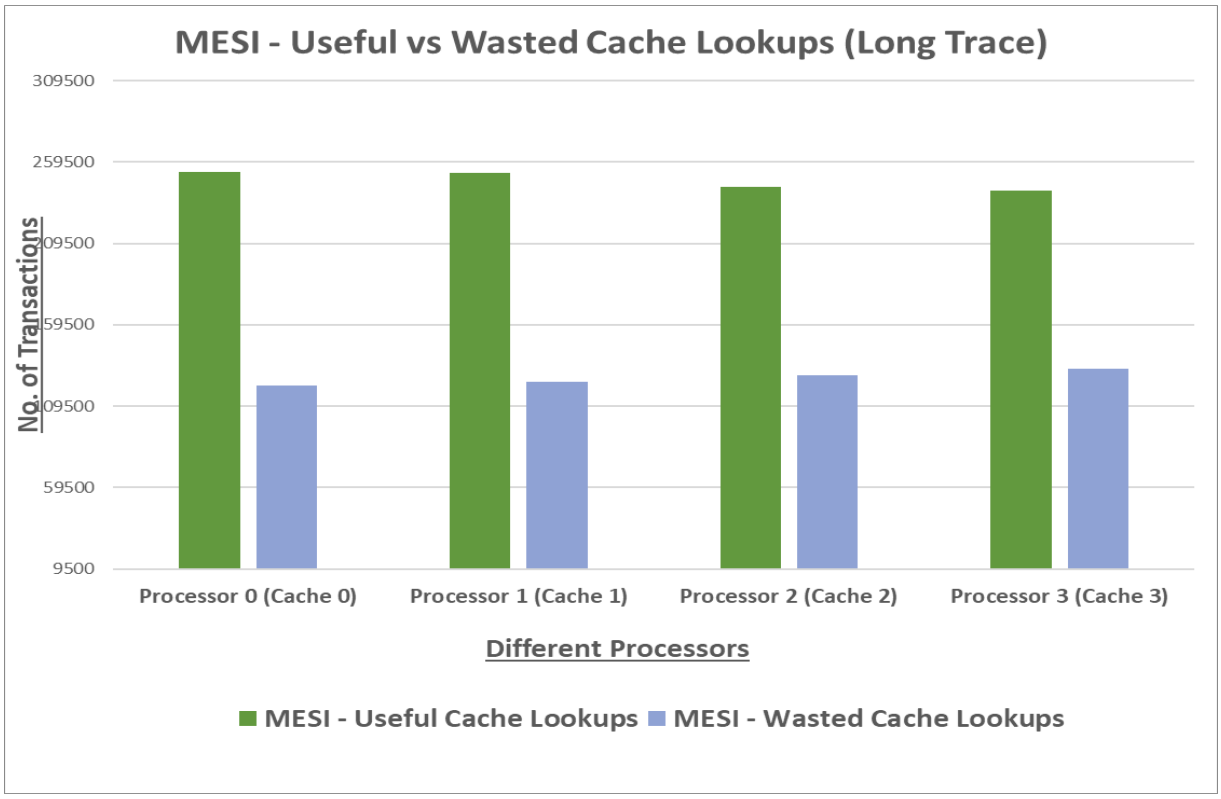
In MESI, when a cache block is in an **Invalid** state sends **BusRdx** or **BusRd** command if any of the other processors (in its cache) has the data in a shared/exclusive state, it does **Cache-to-Cache transfer** instead of a **Memory transaction**.

We can also observe that the sum of **Cache-to-Cache transfers** and **Memory transactions** is **equal** for MESI and MSI BusUpgr for a given trace.

Since a memory transaction is more time-consuming than a Cache-to-Cache transfer, MESI has a better performance than MSI, MSI BusUpgr.

Part 4. Snoop Filter

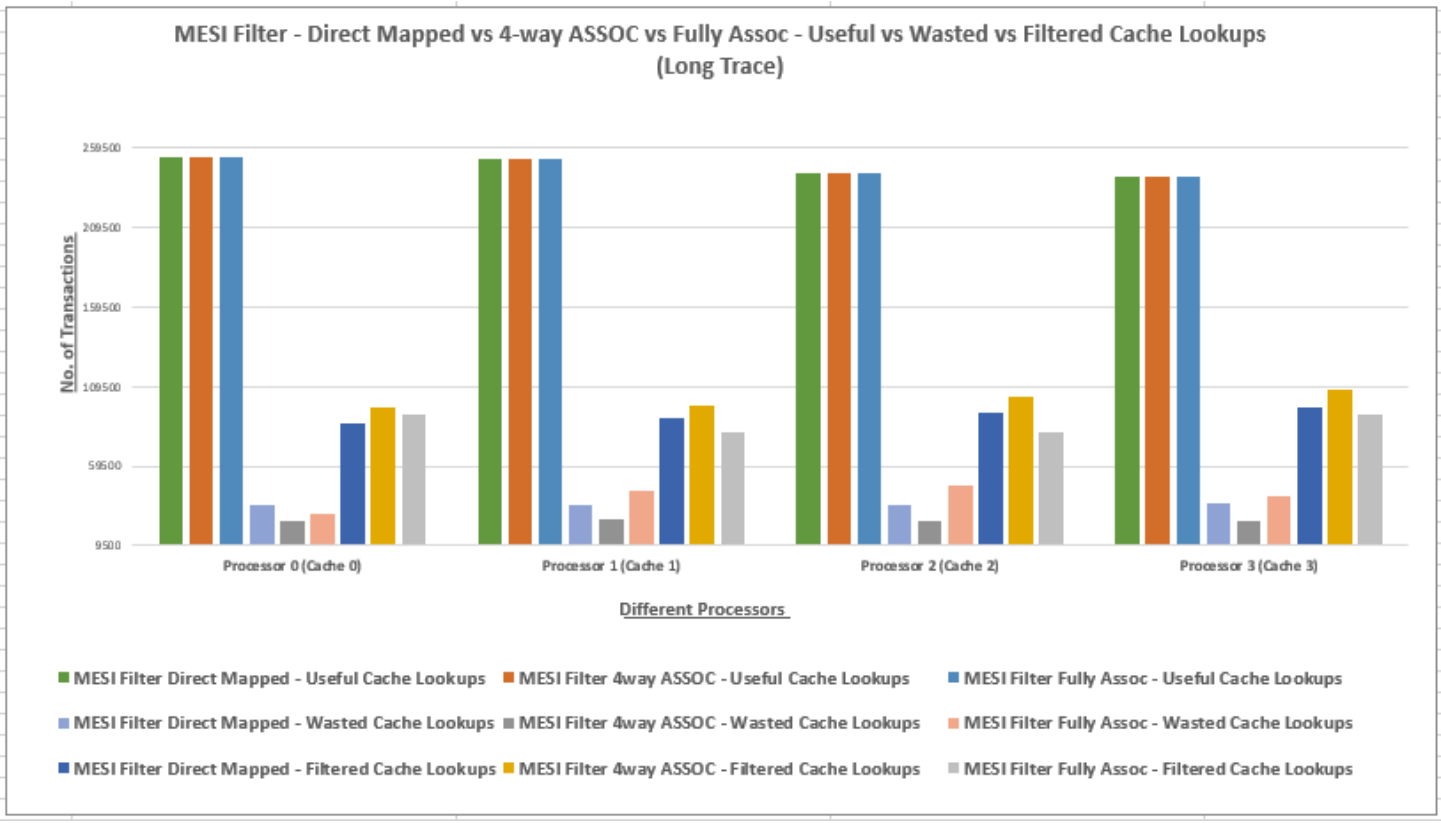
a. Comparison of Useful and Wasted cache lookups while snooping in MESI Protocol

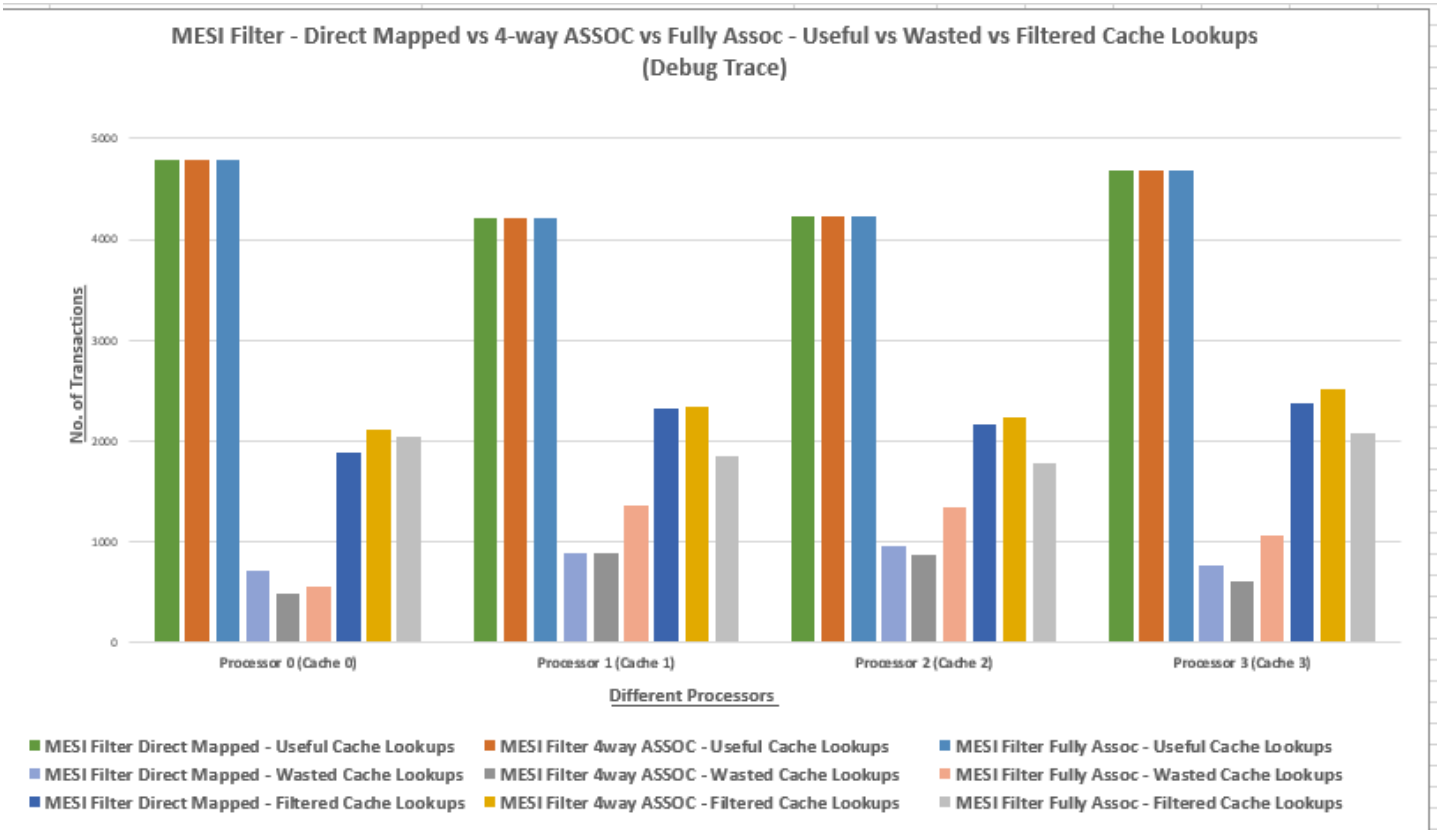


While snooping a request for a cache block, not all the requests are useful. From the graph, it can be inferred that a significant amount of snoop requests is not useful. That is, if a cache block is not there in a particular cache, then the cache lookups for that particular block are not useful as there will be no effect of snoop request.

We can significantly reduce the number of wasted cache lookups by implementing filters that filter out the cases in which we don't have to do cache lookups, thereby saving snooping time.

c. Comparison of Useful, filtered, and wasted cache lookups with different history filter configurations





From both the graphs (long trace and debug trace), the introduction of a snoop filter of any configuration reduces the number of wasted cache lookups.

For the three different history snoops filter configurations (ASSOC = 1,4 Fully ASSOC (16)), we can see that the number of wasted lookups and filtered lookups vary but useful lookups remain the same.

Also, Waste lookups + Filtered Lookups are the same for different history snoop filters.

For the direct-mapped history snoop filter, there may be conflict among cache blocks that have the same indexes for sets as there will only be one block for each index. (Total blocks available = 16)

For the Fully Associative history snoop filter, there will only be one set that has all 16 blocks. Even this increases contention among cache blocks. Also, it can be seen from the graph, that the number of filtered cache lookups is significantly higher than in the other two configurations.

For the 4-way 4-Set history snoop filter, we can see that the number of filtered cache lookups is high compared to the other two configurations. This implies that this configuration has minimum contention among different blocks.

CONCLUSION

From the above results, we can say that there is significant performance improvement in the order of MSI -> MSI with BusUpgr -> MESI -> MESI with Snoop filter. All these improvements are focused on reducing the number of memory transactions which is time-consuming.