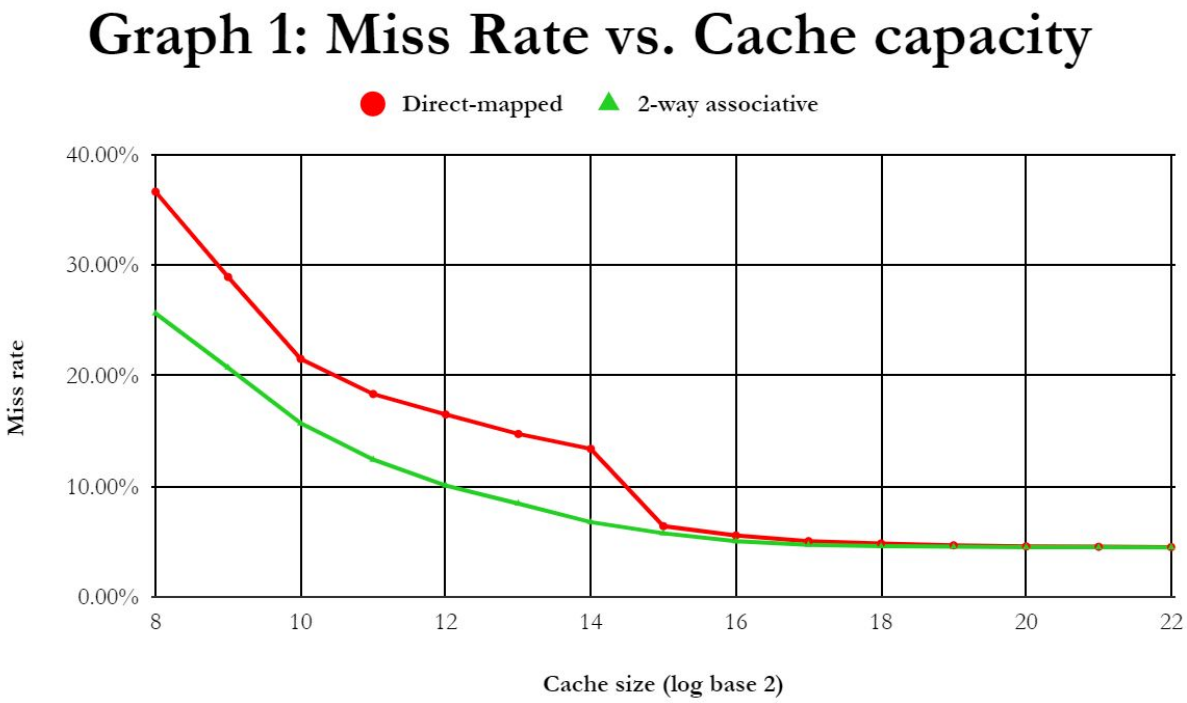
P5: Build-a-Cache



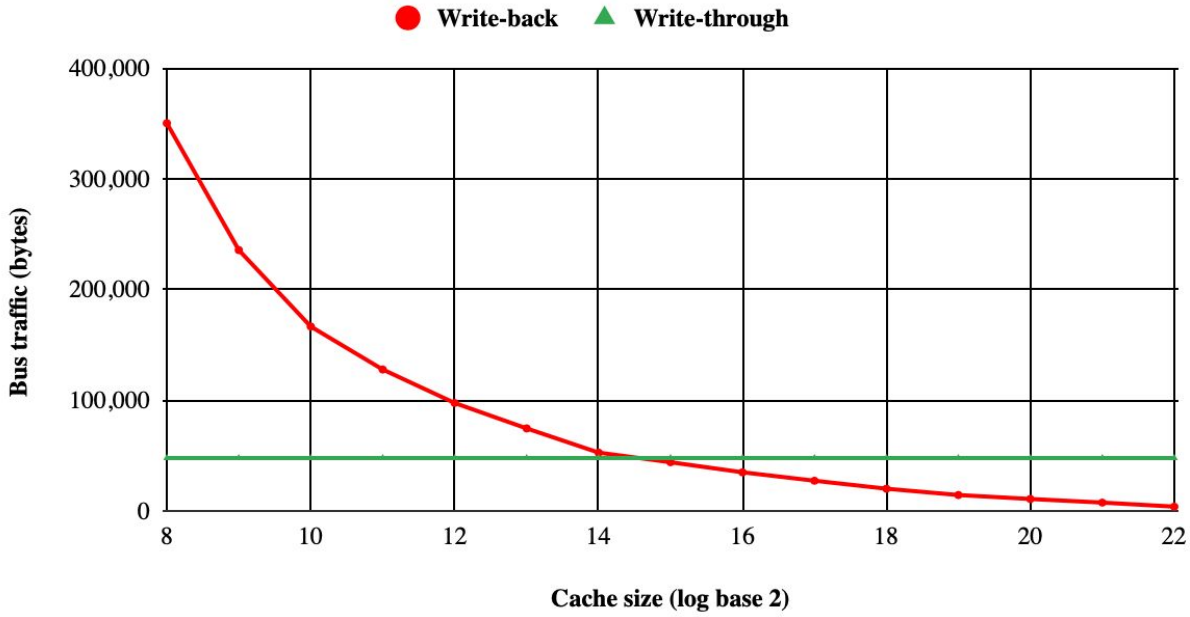
Task 5:

1. 32KB
2. 256KB
3. 2.09 = 13.38/6.39
4. 1.15 = 6.39/5.55
5. Based on the data we collected, 32KB looks like the ideal cache size. We believe this is the ideal cache size because after the 32KB cache size, the improvement in miss rate drops off significantly.

Task 6:

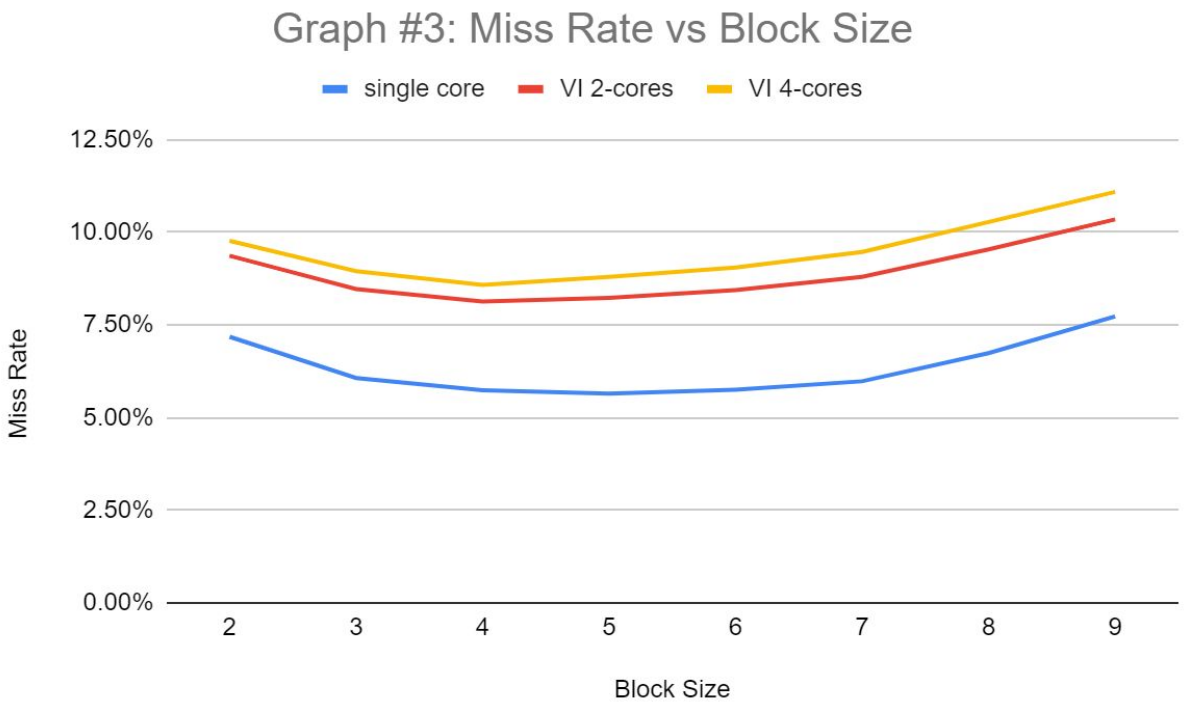
6. 8KB - is the smallest cache size that exceeds the performance of 1KB 2-way assoc
   4 KB - is the largest cache size that doesn't exceed the performance of 1KB 2-way assoc
7. 128KB
8. 8KB

Graph #2: Bus Writes vs Cache Size



Task 7 - Experiment 2:

9. 32KB
10. At small cache sizes, the number of write-through writes increases as there are many more cache evictions. Also, the number of write-back writes stays the same since it only depends on the number of stores. Thus, the difference between the number of the write-through writes and the write-back writes diverges at small cache sizes.
11. At larger cache sizes, the number of write-through writes decreases because there are less cache evictions, while the number of write-back writes doesn't change. Thus, the difference between the number of the write-through writes and the write-back writes diverges at large cache sizes.

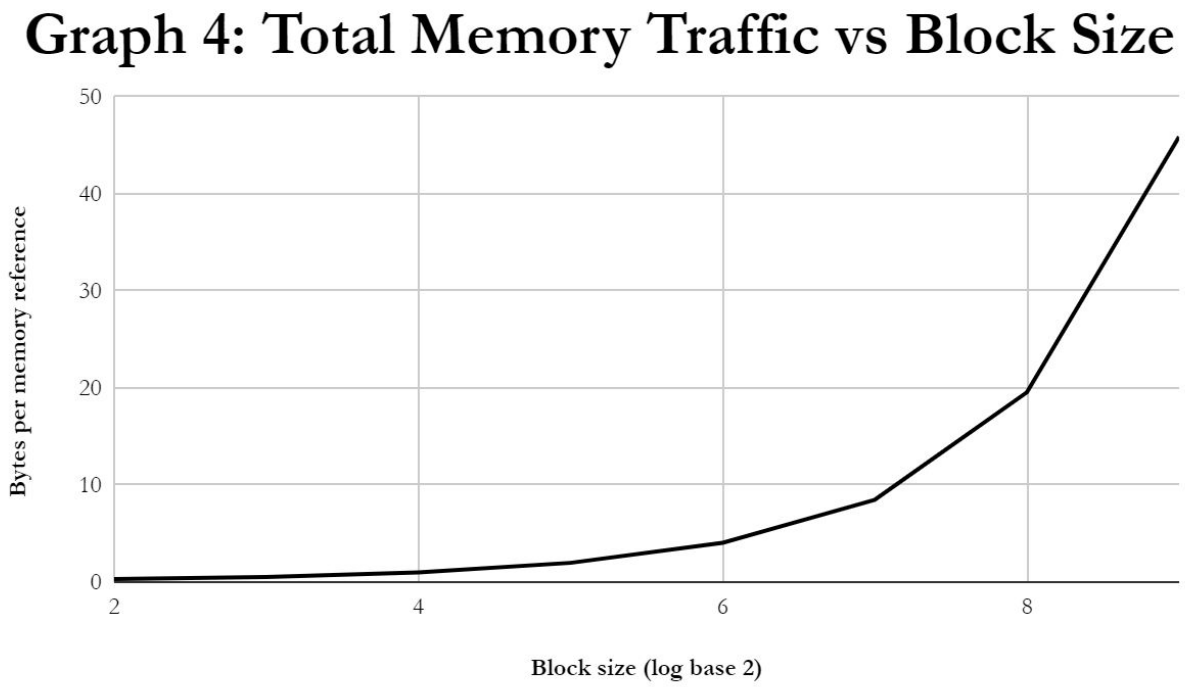


Task 7 - experiment 3 - graph 3:

12. The smaller block sizes are unable to take advantage of the principal of spatial locality in memory accesses. In other words, for each memory reference, the cache is not able to store enough bytes from memory to provide the most benefit.
13. With the larger block sizes, given that the cache size stays constant, the number of sets in the cache decreases to the point where the cache is able to take advantage of spatial locality, but the cache doesn't have enough sets to keep these entries in the cache for long enough. In other words, higher cache turnover is the driver behind the higher miss rate.
14. 32B

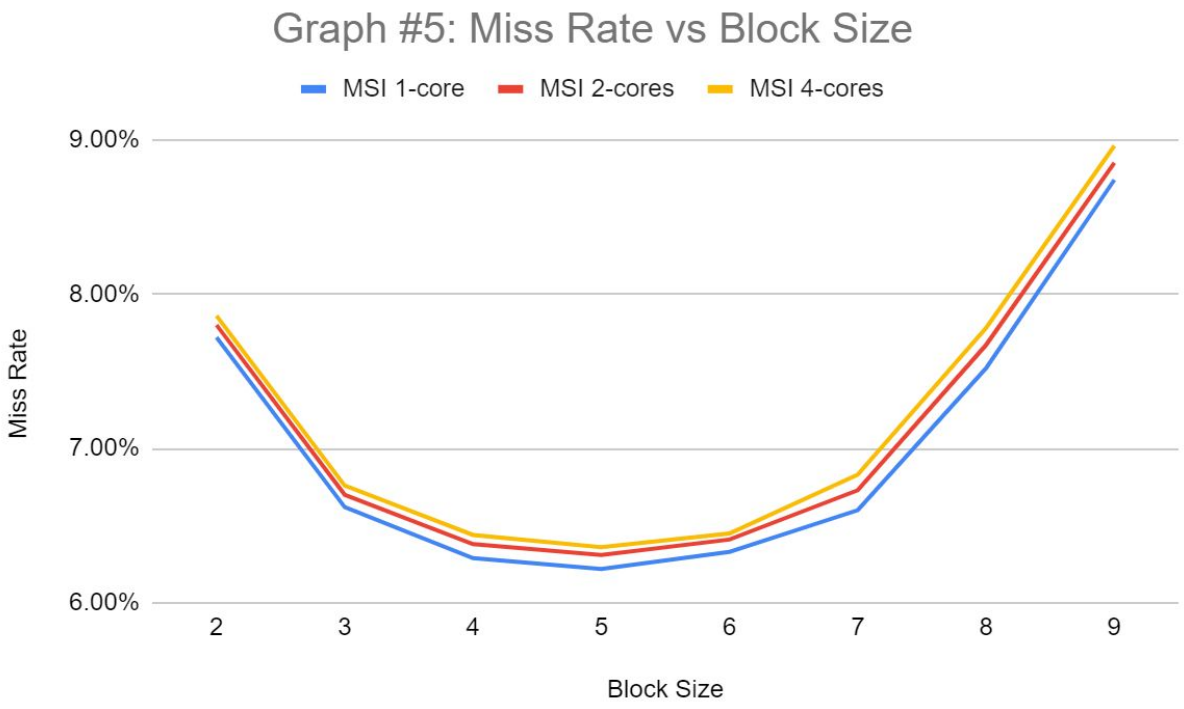
Task 9 - experiment 4 - graph 3:

18. Since only one core can be in a valid state for a particular cache address, as the number of cores increases, the proportion of cores that are in valid state decreases, so the probability for a core to be in a valid state for a particular data becomes smaller and smaller. Thus, the number of miss increases.
19. While the VI protocol doesn't perform well with multiple cores in terms of miss rate, it still solves the problem with coherency that arises with multicore, while using no protocol would not solve this problem.



Task 7 - experiment 3 - graph 4:

15. 4B
16. The two sources that increase byte traffic is the increase in bytes that have to be written back to memory and the increase in the number of dirty evictions that occur. Given that the dirty bit applies to the whole cache line, all of the bytes must be brought back to memory in the event that there is a dirty eviction. Thus, keeping the number of dirty eviction constant, as block size increases, so does byte traffic. To make matters worse, as block size increases, cache sets decrease, meaning that the frequency of dirty evictions increases as well. This is due to the fact that storing just one byte to memory makes an increasingly large amount of bits dirty as block size increases given that one dirty bit is for a whole cache line.
17. Clearly, given that the 64B cache has the second lowest miss rate as compared to other block sizes, and given that the 64B cache has a high byte traffic as compared to other block sizes, modern day cache designs are trying to minimize miss rate rather than traffic.



Task 10 - experiment 5 - graph 5:

20. In general, when upgrade misses are a significant portion of all misses, VI protocol is more efficient. VI protocol would be preferable to the MSI protocol if we iterate through an array and we change each element. Then we would firstly read and then write in the cache for each element, which would cause one cold miss per cache line in the VI protocol and one cold miss + one upgrade miss in the MSI protocol, so we would get more misses in the MSI protocol.
21. total_snoop_hits/total_bus_snoops = 5921/7840 = **75.5 %**
22. total_snoop_hits/total_bus_snoops = 17834/23680 = **75.3**%
23. The workload has a significantly lower number of upgrade misses, which is consistent with the low miss rate and the high snoop hit rate.