Introduction:

This project is a simulation of caches, which store data temporarily for quick access. This simulation more specifically handels how the different types of sets, block sizes, cache sizes, and replacement strategies all contribute to the hit rate of the cache. For this simulation, I ran 168 different simulated caches in order to best gauge the data and draw the correct conclusions. These caches were simulated in python, as that is the language I am most familiar with.

Description of Tests:

For this project, I tested how hit rates were affected by four different factors. The first being different levels of associativity. I used a direct map cache, a fully associative cache, a 2-way set cache, and a 4-way set cache. This helped me gauge how associativity contributes to hit rate. The next factor I tested was the replacement strategies, which I only changed for the three associative caches, as direct mapped has no different replacement strategies available. I used LRU, or last recently used, where the last recently used item in the cache is ejected, and FIFO, or first in first out, which works like a queue, where the item that has been in the longest is ejected. I also tested how cache size affects the hit rate, where I used the sizes of 128, 256, 512, 1024, 2048, 4096, 8192, and 16384 bytes. The final test I performed was how differing block sizes affect the hit rate, where I used the block sizes of 4, 8, and 16. All of these parameters made logical sense to me as the scope of this project, and I wanted to be thorough in my investigation and use as many test statistics as possible, while not creating a program that takes hours to run. These test statistics are where I found the middle ground.

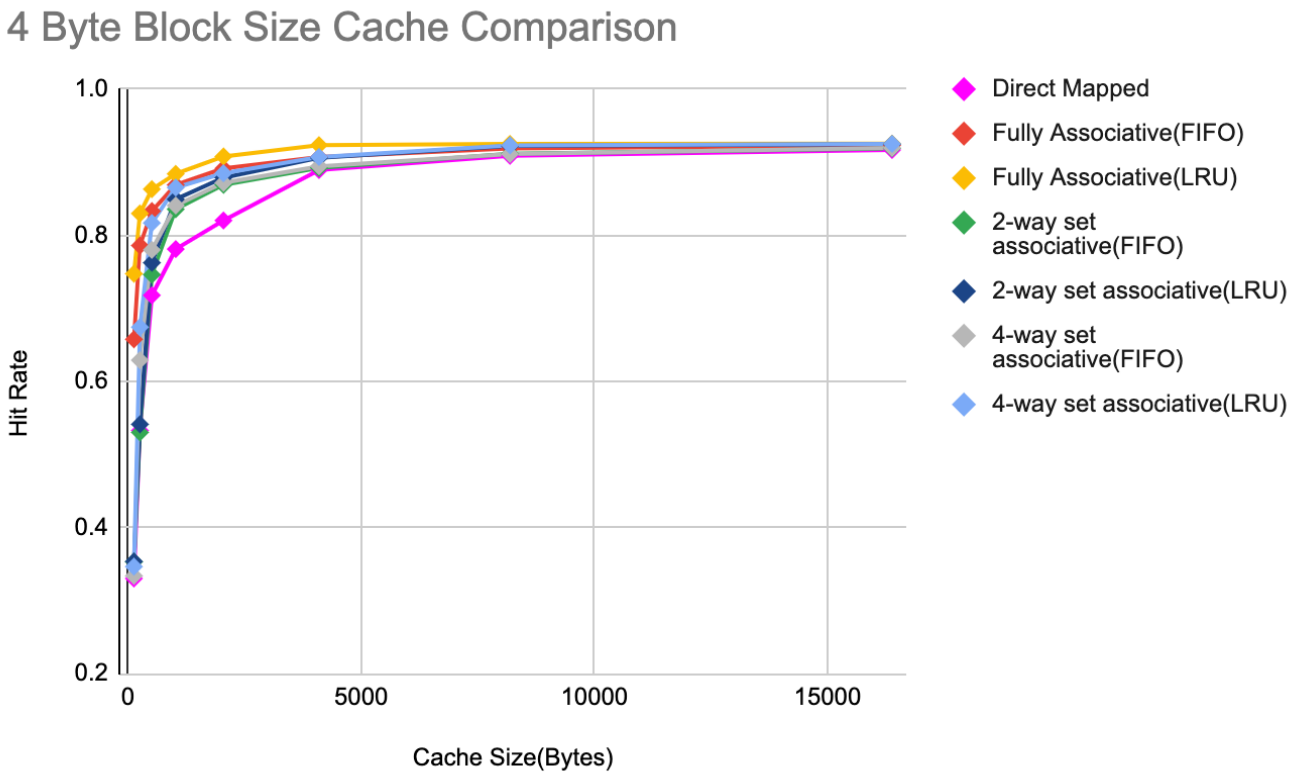
Analysis Results:

Here I have tables followed by the graphs of the data. I have created 3 graphs, each with varying block sizes, and a corresponding table, with everything not in bold being the hit rate, and each graph includes each type of cache I simulated, with their hit rates compared to their cache size.

4 Byte Block Size Comparison:

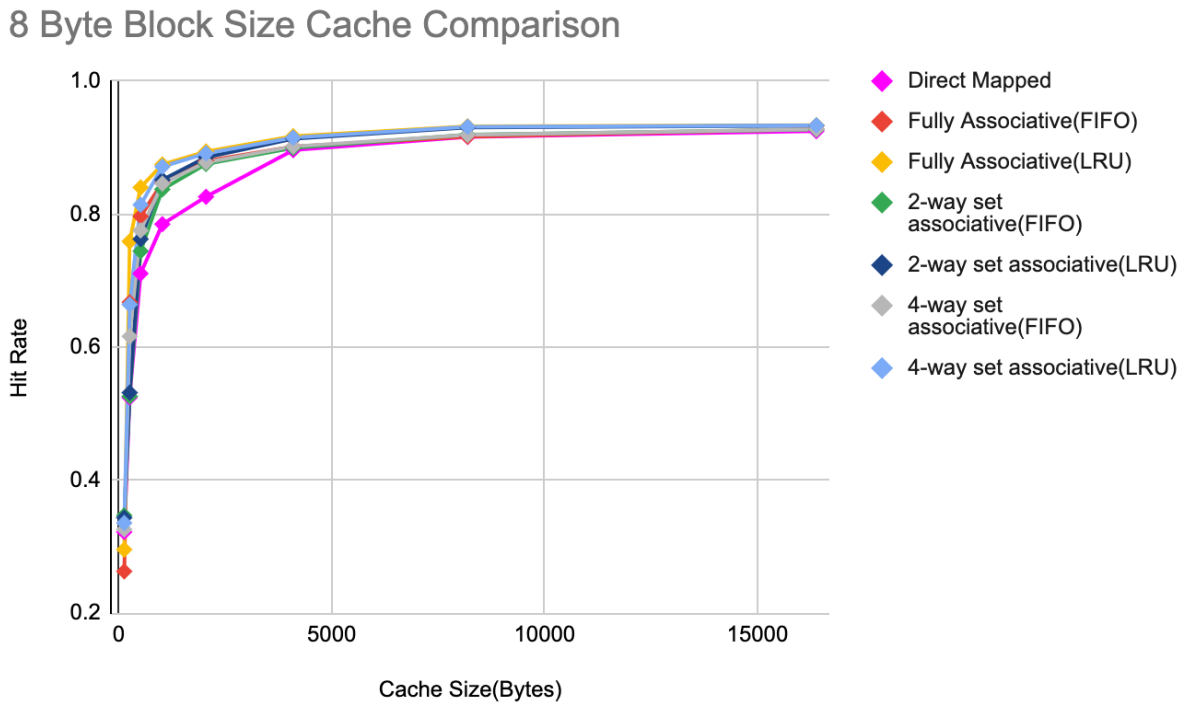
Cache Size	**128 Bytes**	**256 Bytes**	**512 Bytes**	**1024 Bytes**	**2048 Bytes**	**4096 Bytes**	**8192 Bytes**	**16384 Bytes**
Direct Mapped	0.33021	0.532733	0.717843	0.781014 07	0.82023	0.889288 3	0.908728	0.916723
Fully Associative(FIFO)	0.657673 4	0.785915	0.833132 6	0.868942	0.891452	0.907201	0.918840 5	0.923049 01
Fully Associative(LRU)	0.747358 943	0.829784 9884	0.863139 9801	0.884040 8585	0.907860 6696	0.923273 2946	0.925008 1631	0.925436 9329

2-way set associative(FIFO)	0.353035855	0.5301837443	0.7460330549	0.8356426435	0.8690108281	0.8922138704	0.9116140544	0.9188932462
2-way set associative(LRU)	0.3535635717	0.5412591979	0.7623757804	0.8492412424	0.8787175166	0.9058784339	0.92225744	0.9245629022
4-way set associative(FIFO)	0.3338236701	0.6290085853	0.7799883243	0.8407120217	0.8717714459	0.8944038945	0.9110599519	0.9193681912
4-way set associative(LRU)	0.3465482383	0.6742635879	0.816783369	0.8652772327	0.8848852051	0.9072307078	0.9230556114	0.9250741277



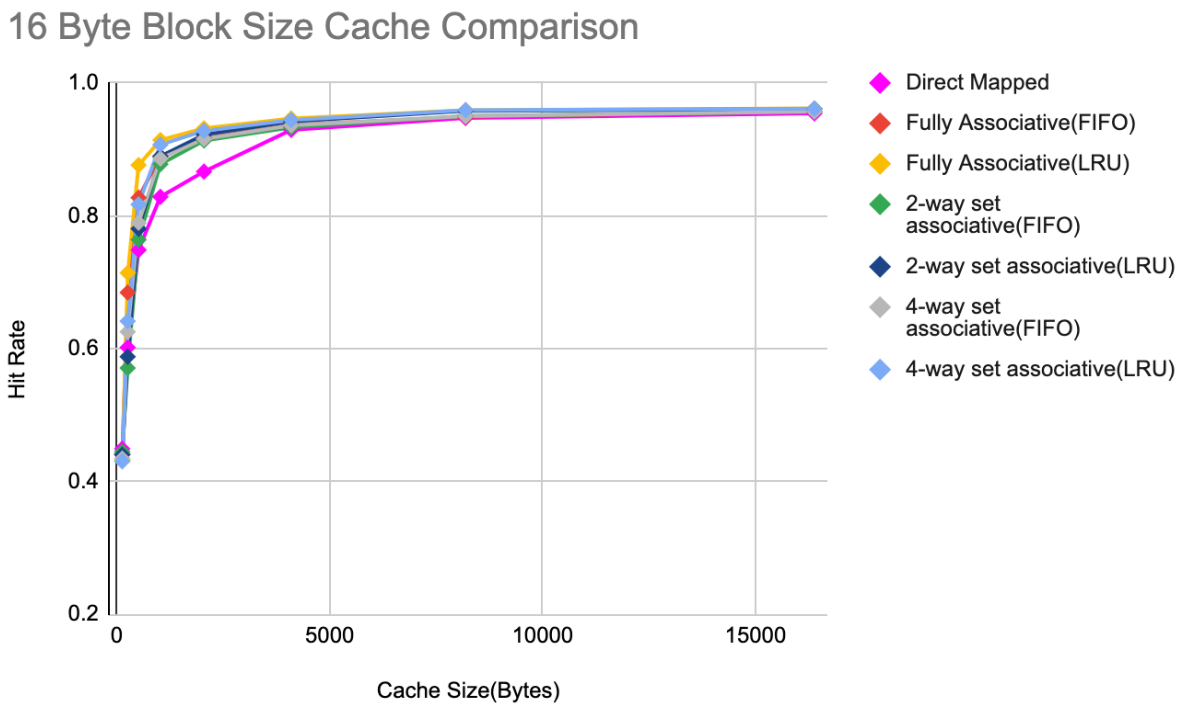
8 Byte Block Size Comparison:

Cache Size	128 Bytes	256 Bytes	512 Bytes	1024 Bytes	2048 Bytes	4096 Bytes	8192 Bytes	16384 Bytes
Direct Mapped	0.3219302556	0.5239665823	0.7105638982	0.7848301247	0.826008516	0.896194833	0.9161425231	0.9245694986
Fully Associative(FIFO)	0.2626775684	0.6674065694	0.7966839604	0.8474107252	0.879947756	0.901462105	0.9164822407	0.9278050615
Fully Associative(LRU)	0.295541124	0.7589390256	0.8400754635	0.8745815372	0.8941070539	0.9171880617	0.9320630753	0.9336891023
2-way set associative(FIFO)	0.3468384824	0.5259521163	0.7443311686	0.8372126006	0.875399498	0.8994963604	0.9195462956	0.9270827493
2-way set associative(LRU)	0.3435270603	0.5315096325	0.7623559911	0.8514708453	0.8852941856	0.9133324318	0.9304106625	0.9328612468
4-way set associative(FIFO)	0.3258980254	0.6162147543	0.7753477158	0.8457550141	0.8776620832	0.9017193669	0.9190284736	0.9276335535
4-way set associative(LRU)	0.3355948191	0.6639104465	0.8140227512	0.8714614124	0.8912540857	0.9149122836	0.9312912897	0.933425244



16 Byte Block Size Comparison:

Cache Size	128 Bytes	256 Bytes	512 Bytes	1024 Bytes	2048 Bytes	4096 Bytes	8192 Bytes	16384 Bytes
Direct Mapped	0.4492056215	0.6016860548	0.7485265161	0.8285778366	0.8665173668	0.9292595805	0.9471788597	0.9540886498
Fully Associative(FIFO)	0.4416032032	0.6844155373	0.8273080183	0.8896511463	0.9168549406	0.9367300696	0.948824676	0.9570603543
Fully Associative(LRU)	0.4320911103	0.7140237407	0.876306511	0.9139492007	0.9316376038	0.9464565475	0.9589568361	0.9615624371
2-way set associative(FIFO)	0.4448057838	0.5708278225	0.764067772	0.8776884691	0.9133555194	0.9331152104	0.9502000376	0.9565986022
2-way set associative(LRU)	0.440016755	0.5877774223	0.7804566728	0.8894268667	0.9221453002	0.9420468151	0.9580300337	0.9608368267
4-way set associative(FIFO)	0.4343767831	0.6252552005	0.7896290482	0.8860329889	0.9157302444	0.936189 16	0.9497943554	0.9569614074
4-way set associative(LRU)	0.4302935754	0.6413868394	0.8171362795	0.9069437619	0.9279303942	0.9442038569	0.9590392918	0.9613117 717



Analysis Conclusions:

Overall, the different parameters affect the cache's hit rate in interesting ways. Firstly, the Cache size overwhelmingly affects the hit rate, as it directly increases in a logarithmic manner approaching 1 as the size increases. The differing levels of associativity also correlate with the hit rate. In general, the fully associative caches have the highest hit rate, with 4-way set associative following, then 2-way set associative, and Direct Mapped is usually ranked last. This makes theoretical sense, as the higher set associative a cache is, the less conflict misses it has. The next parameter tested is the replacement methods, and LRU ranks better than FIFO generally as well. This is due to LRU being a smarter algorithm, because it takes into account what has come before, to try and predict what will come later. The last test I performed was on block size. Block size was the least inconclusive of the 4 tests, as it decreases the amount of lines in the cache, which decreases hit rate, while simultaneously increasing the offset width, which increases hit rate. In general, the conclusion that I came to in my simulation is that it can either decrease or increase the hit rate of a fully associative cache, depending on the size of the cache, while generally increasing every other cache's hit rate, which makes some theoretical sens; however, this is not a certain fact, as I only tested 3 different sizes due to the scope of the project, and different trace files can cause different results, especially in this area.

Generally, the results I gained from this project make theoretical sense, with some outliers, but overall, I feel that this project helped me better understand how the cache system works, and gave me some experience in gaining knowledge, and drawing conclusions from simulations in the computer science industry.