Computer Systems Architecture - Cache Simulator Results Report

Name: Alkinoos Sarioglou, ID Number: 10136315

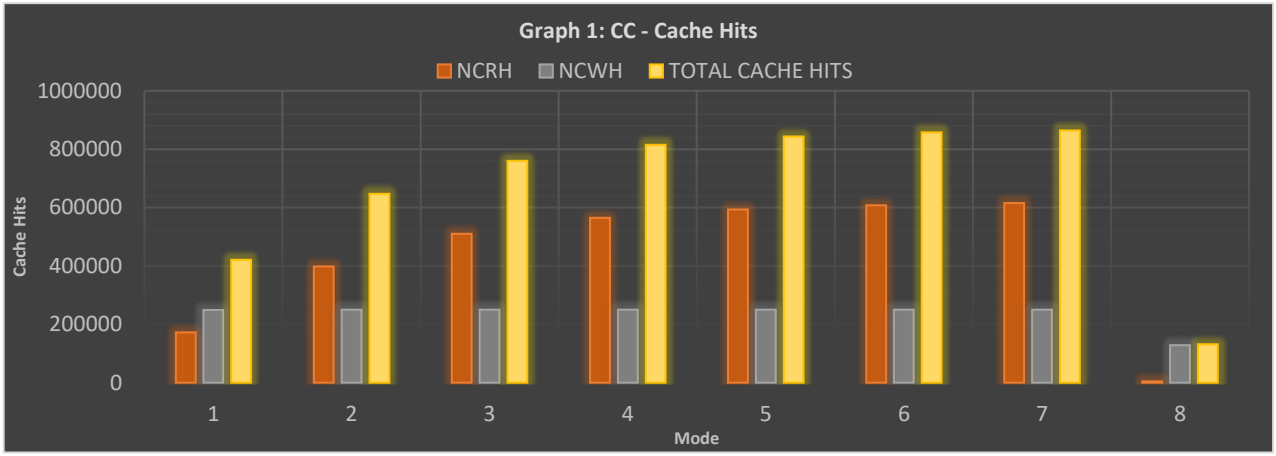
Abbreviations

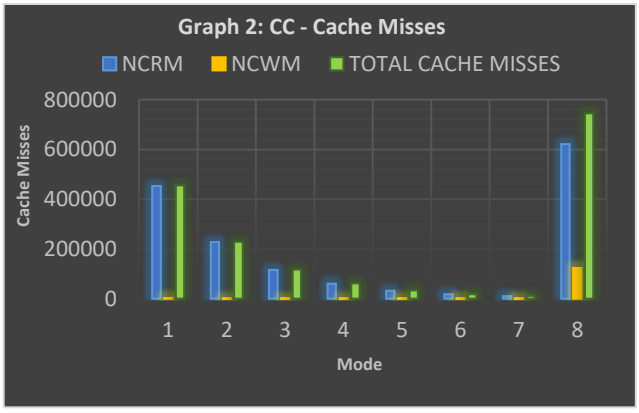
CH – Cache Hits, CM – Cache Misses, MA – Memory Accesses, NC – No Cache

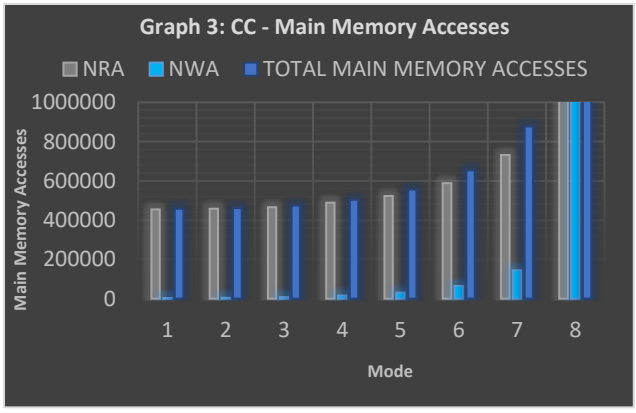
Cross Correlation (CC) Algorithm

The simulation of the cache operation for the CC Algorithm outputs:

Mode	NCRH	NCWH	CH	NCRM	NCWM	CM	NRA	NWA	MA
1	171958	249268	421226	453292	1731	455023	455023	2259	457282
2	397976	249676	647652	227274	1323	228597	457194	4044	461238
3	509910	249873	759783	115340	1126	116466	465864	7680	473544
4	565169	249960	815129	60081	1039	61120	488960	15384	504344
5	593511	249993	843504	31739	1006	32745	523920	31472	555392
6	607877	249997	857874	17373	1002	18375	588000	65760	653760
7	614892	249916	864808	10358	1083	11441	732224	144448	876672
8	4024	127893	131917	621226	123106	744332	95274496	31757440	127031936







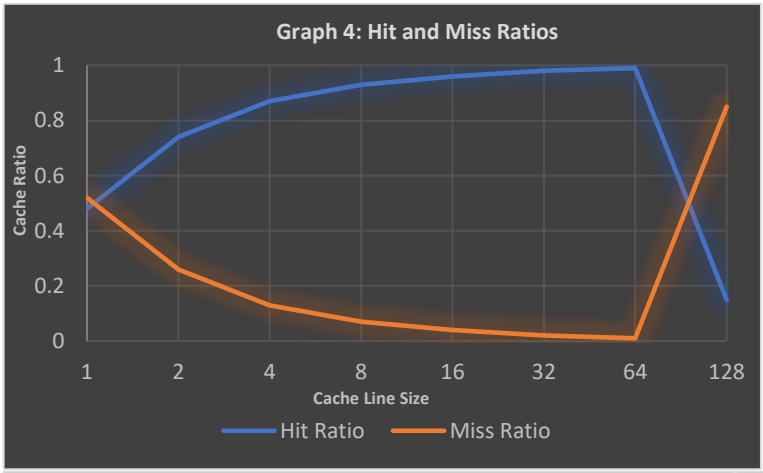
The correct functionality of the cache simulator was verified with custom-made trace files and observing if the results were the expected. The pattern of memory accesses for the CC algorithm starts with the processor accessing the first location of the signal 1 array and the last location of the signal 2 array and outputting the results starting from the first location of the correlation array. Then, for each element of the correlation array, this is initialised to 0 **(1 write access)**. Then, the aligned elements of signal 1 and signal 2 are read **(2 read accesses)** before the element of the correlation array is read for use in the addition **(1 read access)**. Next, the result of the operation is written back to the element of the correlation array **(1 write access)**. The process is repeated for all the aligned elements of signal 1 and signal 2 (except initialisation). Hence, the processor accesses three different parts of memory (arrays). For the cache results it can be noted that:

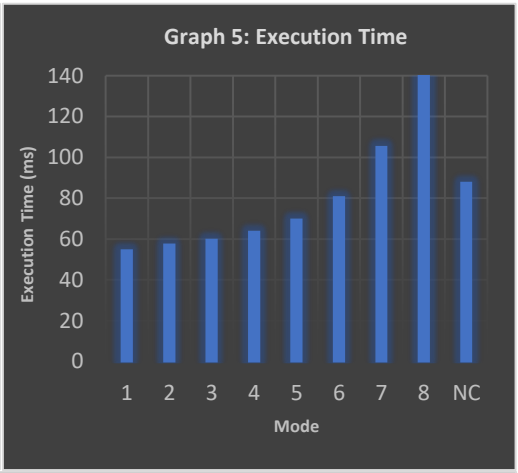
- Apart from Mode 8, the cache hits increase as the block size increases because the accesses in the individual arrays progress in adjacent memory locations (spatial locality).
- In Mode 7 the cache write hits show a slight decrease, because of the reduction of the available cache lines. It is not possible to store more elements of each array in the cache at the same time, therefore there are less cache write hits for access to the correlation array.
- In Mode 8 there is a decrease in cache hits because the number of cache lines (2) becomes less than the arrays accessed by the algorithm (3).
- The read misses decrease as the block size becomes larger because of spatial locality. The only exception is Mode 8 for reasons explained in the previous point.
- Write misses are limited because data is being written only in one of the three arrays, the correlation array. Also because of temporal locality in the accesses of the correlation array.
- Main memory read accesses increase as the block size increases because for every cache miss more memory addresses need to be transferred to the cache at once.
- Main memory write accesses are fewer because the only array that needs to be written back to main memory when replaced is the correlation array.

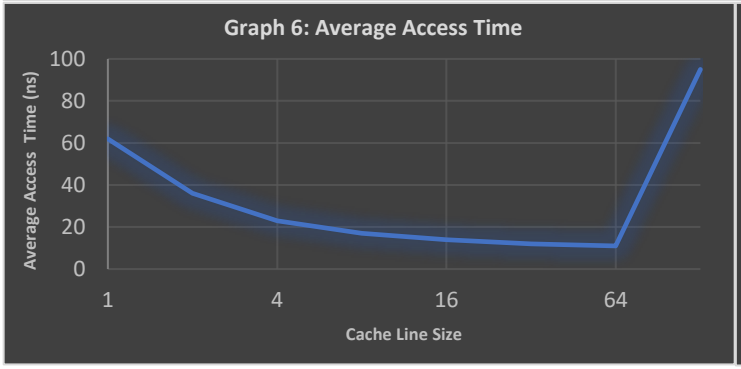
For the assessment of each cache mode, the following performance metrics are calculated:

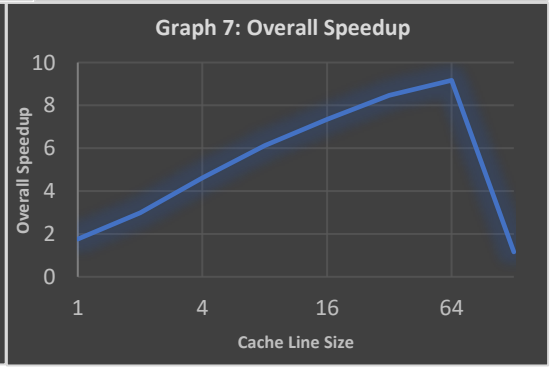
$(Total\ execution\ time = [(110\ ns \times Main\ Memory\ Accesses) + (10\ ns \times Cache\ Hits)] \times 10^{-6}\ \ ms)$

Mode	Line Size	Hit Ratio	Miss Ratio	Average Access Time (ns)	Overall Speedup	Execution Time (ms)
1	1	0.48	0.52	62	1.76	54.51
2	2	0.74	0.26	36	2.99	57.21
3	4	0.87	0.13	23	4.61	59.69
4	8	0.93	0.07	17	6.13	63.63
5	16	0.96	0.04	14	7.35	69.53
6	32	0.98	0.02	12	8.47	80.49
7	64	0.99	0.01	11	9.17	105.08
8	128	0.15	0.85	95	1.16	13974.83
NC				100		87.62







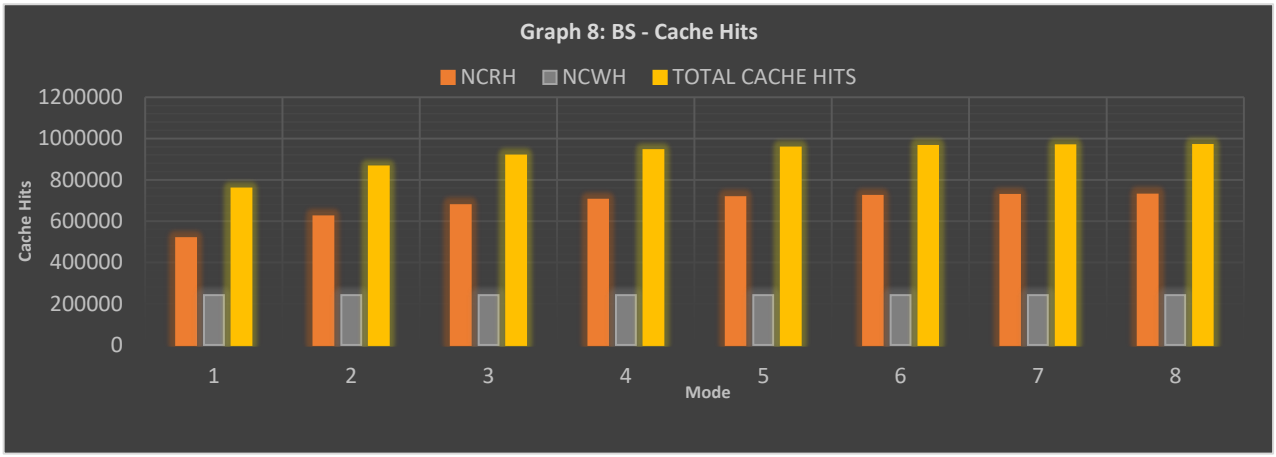


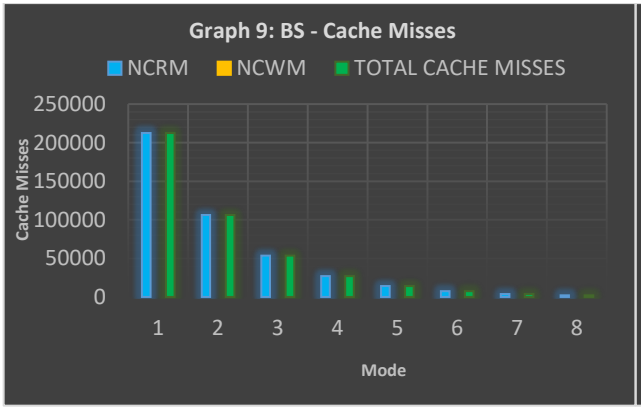
From the above it can be observed that all modes apart from 7 and 8 offer benefits in execution time, which is the most important factor. Therefore, for this specific input signal length of 500 samples, Cache Mode 1 would be the most suitable. However, it is worth noting that Cache Mode 7 demonstrates the best performance metrics. If we had to choose a Cache Mode for this algorithm but for varying input signal length, then the choice would be Cache Mode 7 as it would have better chances of giving faster execution for a wide range of input signal lengths. Overall, it is proved here that better performance metrics do not result in faster execution of an algorithm.

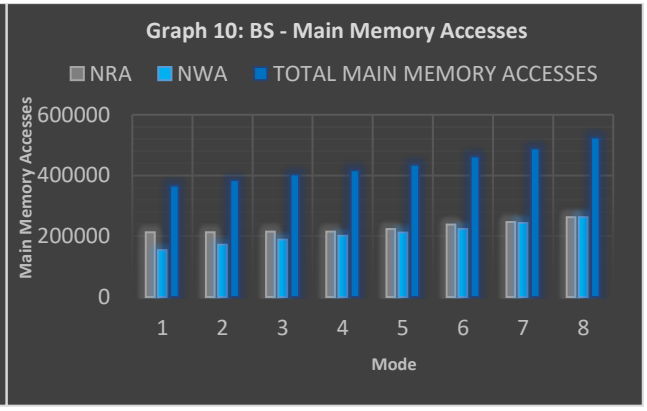
Bubble Sort (BS) Algorithm

The simulation of the cache operation for the BS Algorithm outputs:

Mode	NCRH	NCWH	CH	NCRM	NCWM	CM	NRA	NWA	MA
1	517258	240668	757926	212710	0	212710	212710	153819	366529
2	623502	240668	864170	106466	0	106466	212932	171616	384548
3	676272	240668	916940	53696	0	53696	214784	188996	403780
4	703008	240668	943676	26960	0	26960	215680	201776	417456
5	716018	240668	956686	13950	0	13950	223200	212240	435440
6	722510	240668	963178	7458	0	7458	238656	223936	462592
7	726116	240668	966784	3852	0	3852	246528	243264	489792
8	727914	240668	968582	2054	0	2054	262912	262144	525056





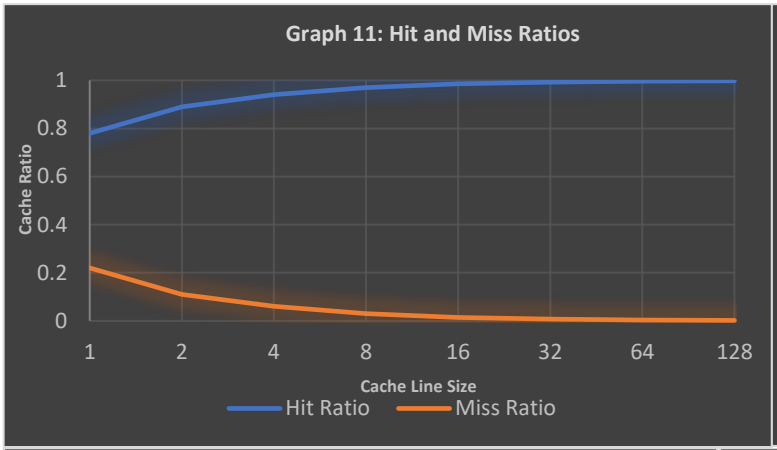


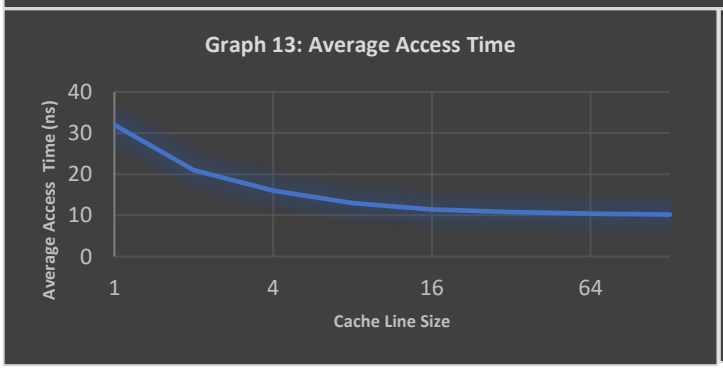
The pattern of memory accesses begins with the processor reading and comparing the elements of the first two memory addresses [j] and [j+1] **(2 read accesses)**. If the one of address [j+1] is smaller then it swaps the elements. This happens by reading and storing the element of memory address [j] in a temporary variable **(1 read access)**. Then, the element of memory address [j+1] is read **(1 read access)** and written to memory address [j] **(1 write access)**. Next, the value of the temporary variable is written to memory address [j+1] **(1 write access)**. However, if the element of memory address [j+1] is bigger, there is no swapping. The algorithm continues with the next two adjacent memory addresses for all of them. For the cache results it can be noted that:

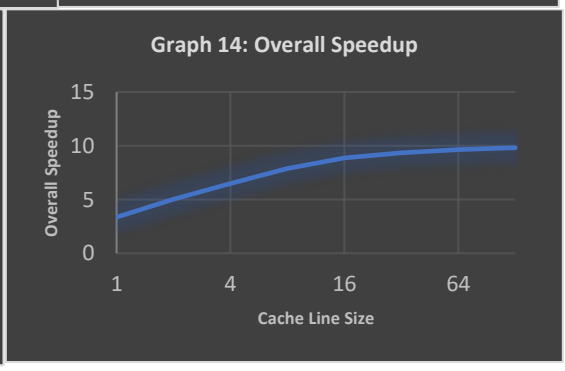
- There are more cache hits and less cache misses as the block size gets larger because the algorithm accesses neighbouring memory addresses (spatial locality).
- There are no cache write misses, because the memory addresses are first read for comparison before they are written to (temporal locality).
- Read main memory accesses increase as the block size increases since for every cache miss more memory addresses are transferred to the cache at once.
- As the block size increases, so does the probability of writing to memory addresses in the cache, because more comparisons take place. As a result, blocks of bigger size need to be written back to main memory more often and thus more write memory accesses are performed.

The same performance metrics are calculated for the BS Algorithm:

Mode	Line Size	Hit Ratio	Miss Ratio	Average Access Time (ns)	Overall Speedup	Execution Time (ms)
1	1	0.78	0.22	32	3.36	47.9
2	2	0.89	0.11	21	5.03	50.94
3	4	0.94	0.06	16	6.49	53.59
4	8	0.97	0.03	13	7.87	55.36
5	16	0.986	0.014	11.4	8.88	57.47
6	32	0.992	0.008	10.8	9.33	60.52
7	64	0.996	0.004	10.4	9.65	63.54
8	128	0.998	0.002	10.2	9.82	67.44
NC				100		97.06







It can be observed that all modes of the cache offer benefits in execution time. The fastest execution occurs for Cache Mode 1 therefore this would be the most suitable. However, in this algorithm as well, the best performance metrics are achieved by a separate Cache Mode, i.e. Cache Mode 8. Therefore, if we had to choose a Cache Mode for this algorithm with varying array size then Cache Mode 8 would be chosen. It is also proved here as well that better performance of a Cache Line Size does not imply faster execution time.