Architecture of Parallel Computers - CSC 506

Machine Problem 2

Abhinav Sarkar - asarkar4
Korak Aich - kaich

Question 1: For MESI, how does the number of cache misses and invalidations change as you increase the cache block size? What are the reasons behind this?

Answer:
Cache size= 1MB
Protocol= MESI
Associativity=8 way

Cache Block Size->	**64**	**128**	**256**
P0 #cache miss	5791	5379	5062
P1 #cache miss	5822	5426	5110
P2 #cache miss	5794	5383	5046
P3 #cache miss	5829	5423	5123
Total # cache miss	**23236**	**21611**	**20341**
P0 #invalidations	2014	2066	2132
P1 #invalidations	2034	2089	2157
P2 #invalidations	2008	2053	2107
P3 #invalidations	2020	2068	2148
Total #invalidations	**8076**	**8276**	**8544**

As we increase the cache block size, we can see that the number of cache misses decrease and the number of invalidations increase. The decrease in cache misses can be attributed to the fact that now more data is being stored in the cache block at a time. This is improve the spatial locality and hence reduce the number of future cache misses.

The number of invalidations increase because of increase in false sharing. Since the block size increases, this increases the number of addresses mapped to the block and hence any modifications in any of the addresses mapped to the block by other processors will cause invalidations.

Question 2: An invalidation is not necessarily harmful if it is not followed by a cache miss to the block that was invalidated. Hence, the number of coherence misses is more representative to performance compared to the number of invalidations. Collect the number of coherence misses, and observe how they are affected as the cache size increases. Then, explain possible reasons for your observations.

coherence misses

	32KB	64KB	128KB	256KB	512KB
Processor 0	67	92	126	144	167
Processor 1	63	101	133	151	177
Processor 2	67	95	128	145	163
Processor 3	69	100	134	153	177
Total	266	388	521	593	684

As we can see from the above table, the number of coherence misses increase with increase in cache size. The possible reason for this is that with a larger cache, the number of invalidations increase. But since the cache is large, the invalidated blocks are not evicted. When those caches are being accessed again, the coherence miss count increases.

Question 3: Compare MSI vs. MESI in terms of the total number of bus transactions. The Exclusive state allows MESI to have fewer bus transactions. However, does the reduction vary with cache block size and cache size?

MSI vs MESI: varying cache size
Cache block size: 64
associativity: 8
#bus transactions

Proc 0

Cache size	MSI	MESI
32	7077	7074
64	6760	6757
128	6558	6555
256	6491	6488
512	6473	6470

Proc 1

Cache size	MSI	MESI
32	7091	7082
64	6763	6754

128	6565	6556
256	6505	6496
512	6492	6483

Proc 2

Cache size	MSI	MESI
32	7107	7098
64	6769	6760
128	6562	6553
256	6496	6487
512	6481	6472

Proc 3

Cache size	MSI	MESI
32	7118	7105
64	6784	6771
128	6593	6580
256	6527	6514
512	6510	6497

Total (for 4 processors)

Cache size	MSI	MESI	Diff
32	28393	28359	34
64	27076	27042	34
128	26278	26244	34
256	26019	25985	34
512	25956	25922	34

So we can see that the difference in number of bus transactions remains the same. As per our observations, we see that the number of bus transactions difference deps on going to the exclusive state. This happens only if a processor requests a exclusive read on the address int the block. Since this is determined on the order of requests made by each processor, we cannot

jump to a conclusion.

MSI vs MESI
Cache size =1MB

Proc0

Cache block size	MSI	MESI
64	6468	6465
128	6069	6067
256	5774	5772

Proc1

Cache block size	MSI	MESI
64	6481	6472
128	6097	6089
256	5805	5797

Proc2

Cache block size	MSI	MESI
64	6477	6468
128	6086	6078
256	5784	5777

Proc3

Cache block size	MSI	MESI
64	6504	6491
128	6117	6105
256	5827	5815

Total

Cache block size	MSI	MESI	Diff
64	25930	25896	34

128	24369	24339	30
256	23190	23161	29

The metric is dependent on the sequence of requests received format the CPU. But in this case we can see that the difference decreases, and hence the performance of MESI is degrading to that of MSI. This might be because, the probability of the case where the exclusive state is useful decreases as the block size increases.

Question 4: Compare MESI vs. MOESI in terms of the total number of flushes to the main memory. In general, MOESI should reduce the total number of flushes because it allows dirty sharing. However, how does the reduction affected by the cache size? Make observation and explain possible reasons for your observation.

Cache block size: 64
MESI vs MOESI

Processor 0

Cache size(KB)	MESI	MOESI
32	647	550
64	573	471
128	420	315
256	254	141
512	190	74

Processor 1

Cache size(KB)	MESI	MOESI
32	635	548
64	555	466
128	434	343
256	235	143
512	170	78

Processor 2

Cache size(KB)	MESI	MOESI
32	655	549
64	587	475
128	446	331
256	278	158
512	205	84

Processor 3

Cache size(KB)	MESI	MOESI
32	646	561
64	570	485
128	414	326
256	234	146
512	171	83

Total

Cache size(KB)	MESI	MOESI	Difference
32	2583	2208	375
64	2285	1897	388
128	1714	1315	399
256	1001	588	413
512	736	319	417

We notice that number of flushes to main memory in MOESI is considerably less than in MESI. This is expected in MOESI due to the introduction of the Owner state.

We can see that increasing the cache size decreases the flushes to main memory for both MESI and MOESI protocol. This is because the number of conflict cache misses decreases and lesser blocks needs to be evicted to main memory.

Part C

For a 256KB cache with 64B block size, which protocol (MSI, MESI, and MOESI) produces the lowest B*M? Explain why

Assuming 8-way associativity and 4 processors,
We calculate B =8*(command bus transfers) =64*(data bus transfers)
for the sake of simplicity we have taken command bus transfers = data bus transfers.
data bus transfers= write backs+ flushes to main memory + total cache misses -cache to cache transfers.

MSI:
M1 = 5814 B1 = 436896
M2 = 5846 B2 = 437832
M3 = 5813 B3 = 438552
M4 = 5852 B4 = 438192
M = 23325 B = 1751472
B*M=4.085*10^10

MESI:
M1 = 5814 B1 = 119736
M2 = 5846 B2 = 118080
M3 = 5813 B3 = 121320
M4 = 5852 B4 = 120600
M=23325 B=479736
B*M=1.1189*10^10

MOESI:
M1 = 5814 B1 = 113112
M2 = 5846 B2 = 313776
M3 = 5813 B3 = 415296
M4 = 5852 B4 = 414288
M = 23325 B = 1256472
B*M = 23325*1256472 = 2.9307*10^10

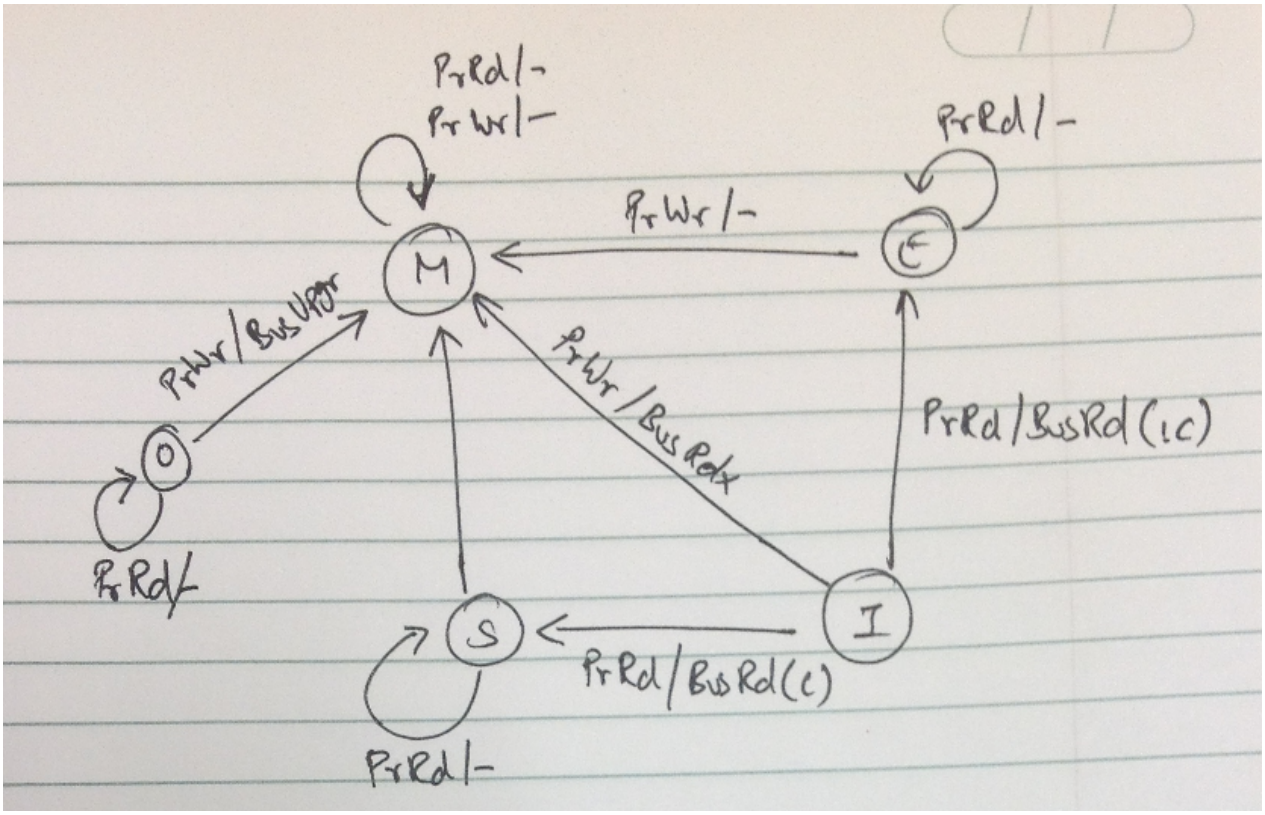
As we can see from the results, the MESI protocol gives the best B*M metric. Number of misses (M) is the same for all three protocols, but the value of B is least in MESI. MESI has the most cache to cache transfers, so the value of B is minimum.

Question 2: Find a cache configuration and modify any of the above protocol to improve B*M. You are restricted to 1MB of cache size and 8-way associativity. You can only modify the cache block size and the coherence protocol. Your answer should not introduce new predictor structure, or use prefetching. But you are allowed to modify the cache coherence protocol such as adding new states, changing the state transition diagrams, introducing new bus transactions, etc. You are allowed to add at most 4 new

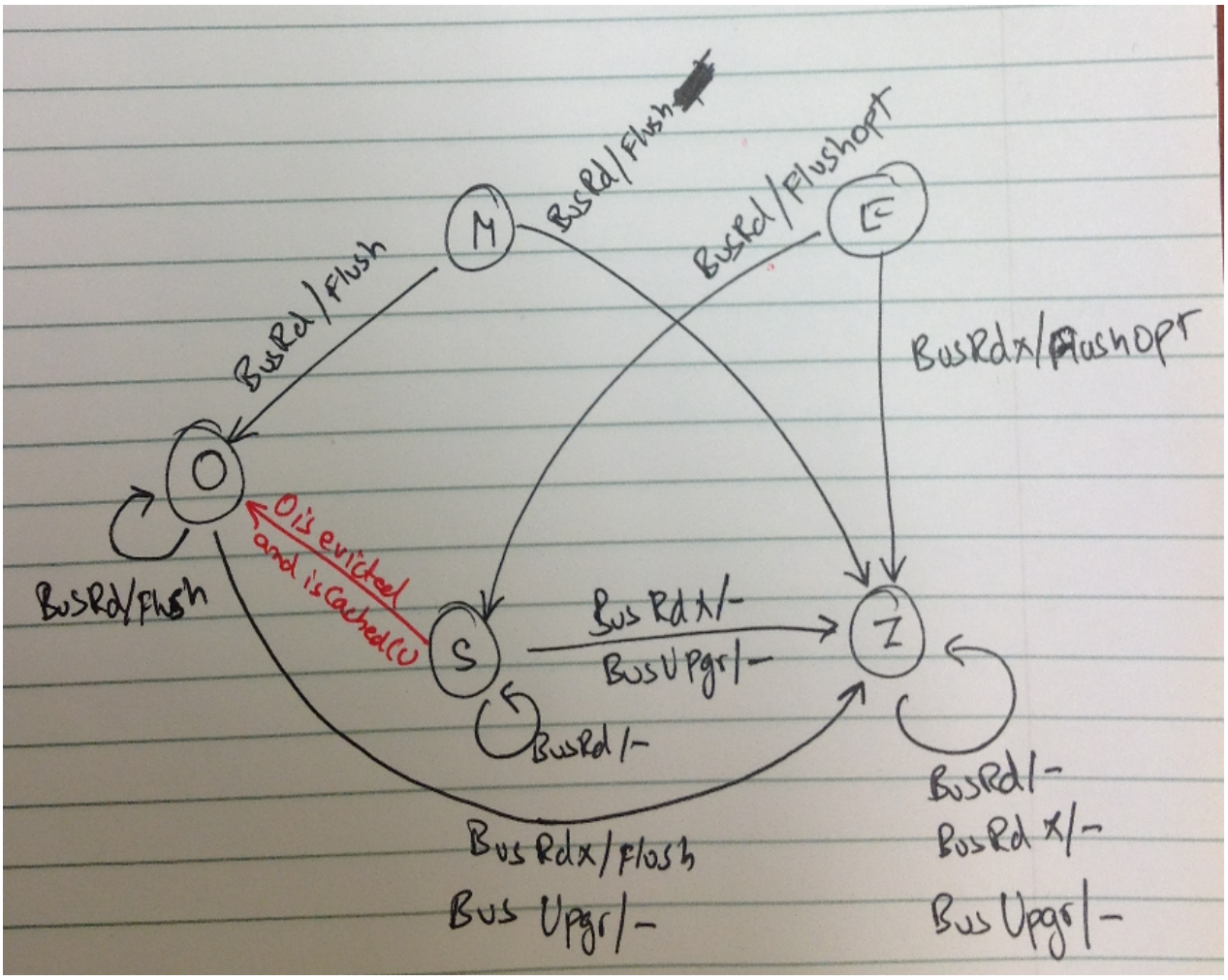
states (if necessary). Explain your approach & readings. Also include the state diagram of the enhanced protocol.

We enhance the MOESI protocol, by trying to further delay the write backs in an attempt to conserve the off-chip bandwidth. We have a new case where in during eviction, we check if the block is in shared with any other processor then, we delay the write back by making one of the shared states to owner. This makes sure that the dirty write-back is performed only when absolutely necessary. The observations we base this on, is the fact that if the block is in owned state, then the other caches can contain it in only shared states and hence are present in the other caches that have not yet been evicted and hence we transfer the ownership and we silently discard the writeback.

The Processor Initiated Diagram:



Bus Initiated Transaction Diagram:



For the configuration, 256KB cache, 64B block size, 8 way associativity and 4 processors, we get:

M1 = 5814 B1 = 110016
M2 = 5846 B2 = 310536
M3 = 5813 B3 = 412200
M4 = 5852 B4 = 411840
M = 23325 B = 1244592
B*M = 2.903*10^10

This is better than the MOESI protocol's performance as presented in the last question.

The best configuration for the above protocol is as follows:

```
===== 506 SMP Simulator Configuration =====
L1_SIZE:                        1048576
L1_ASSOC:                       8
L1_BLOCKSIZE:                           256
NUMBER OF PROCESSORS:                   4
COHERENCE PROTOCOL:             MOESI
TRACE FILE:                     ./Validation_runs/canneal.04t.longTrace
M: 2034100.000000, B: 1026864000.000000, M*B = 20887440624.000000
```