EDA284

Solutions for exercise set 1

Problem 1.2

We solve this problem assuming that N≥1024 and the number of available processors P in the machine goes from 1 to 1024.

a. $$Speedup = \frac{T_1}{T_p} = \frac{NT_c}{\frac{N}{P}T_c + NT_b} = \frac{PR}{R+P}$$

b. For a given R, the speedup increases monotonically as P increases. The maximum speedup is thus achieved when P is 1024 and the maximum speedup is

$$Speedup_{max} = \frac{1024R}{1024+R} \quad :$$

c. $P > \frac{R}{R-1}$

d. The execution time stays constant at NxT_c for all P's >1 and given N. As P increases from 1 to 1024, the workload size (N_1) increases so that:

$$NT_c = \frac{N_1}{P}T_c + N_1T_b$$

and:

$$N_1 = \frac{NPR}{R+P}$$

As P grows N_1 tends to an asymptote equal to NxR.

e. Reconsidering a-c above in the context of growing workload size.
In this part, for given N, the workload (N_1) grows according to d) above, so that the execution time remains constant. In this context T_1 is equal to N_1T_c and T_P remains fixed at NT_c

$$Speedup = \frac{N_1T_c}{NT_c} = \frac{PR}{R+P}$$

Surprisingly the speedup with a growing workload is the same as the speedup in part a with constant size workload, and therefore the maximum speedup and minimum P are also the same as in part a. The reason is that the serial part (the bus accesses) grows with P, contrary to Amdahl's or Gustafson's laws where the serial part remains a constant.

Problem 5.2

a) The average memory access time is calculated as follows:

Average access time = Hit time + Miss rate x Miss penalty

Private cache organization:
The hit time in the private cache is a single cycle and the miss penalty is 100 cycles according to the assumptions. Since the private cache only can host eight blocks and 16 different blocks are accessed twice in a row, all accesses will result in a cache miss (the miss rate is 100%). Therefore, the average memory access time is:

Average = 1 + 100 cycles = 101 cycles

Shared cache organization:
Since there are four processors there are four banks where each bank contains eight blocks. The shared cache can thus host 4x8 = 32 blocks. As a result, the first 16 accesses will miss and the subsequent 16 accesses will hit (the miss rate is 50%). The average memory access time is

Average = 2 + 0.50x100 cycles = 52 cycles

The shared cache is almost twice as fast as the private cache.

b)Private cache organization:
Since each processor accesses 16 different blocks twice in a row, all accesses will miss and the average memory access time will be the same as in a) for the private cache.

Shared cache organization:
One of the processors (say processor 1) will miss on each of the first 16 accesses but will bring the blocks into the shared cache. All other processors (say processor 2, 3, and 4) will hit on each access. Hence:

Average memory access time for processor 1: (16 x (2 + 100) + 16 x 2)/32 cycles = 52 cycles
Average memory access time for processors 2-4: 2 cycles
Average memory access time = (52 + 3 x 2)/4 = 14.5 cycles

The shared cache is more than 7 times faster.

c) From the assumptions we note that each processor accesses the following blocks:
Processor 1: Blocks 0 through 7 twice
Processor 2: Blocks 8 through 15 twice

Processor 3: Blocks 16 through 23 twice
Processor 4: Blocks 24 through 31 twice

Private cache organization:
Since there are eight blocks in the private cache, the first eight accesses will miss whereas the next eight accesses will hit. Hence:

The average memory access time = (1+100 + 1)/2 cycles = 51 cycles

Shared cache organization:
The exact same miss behavior shows up for the shared cache; the first eight accesses from each processor will miss whereas the next eight accesses will hit. However, the cache hit time is two cycles.

The average memory access time = (2 + 100 + 2)/2 cycles = 52 cycles

d) Private cache organization:
According to the assumptions the eight blocks are already fetched into cache. The eight processor writes result in eight invalidations where each invalidation takes 10 cycles. The subsequent eight processor reads result in coherence misses where each coherence miss takes 10 cycles.

Average memory access time = 1 + 10 cycles = 11 cycles

Shared cache organization:
An advantage of a shared cache is that no cache coherence actions are needed. Each access will take two cycles:

Average memory access time = 2 cycles

e) There was not a single case where the private cache organization outperforms the shared cache. Yet, the disadvantage of the shared cache is its longer cache hit time. Obviously the tradeoff is between the miss rate and hit time. Let's assume that Shared_Hit_Time =KxPrivate_Hit_Time. The private cache organization has an advantage when

Private HT + Private MR x MP < K x Private HT + Shared MR x MP

Private MR x MP/Private HT - Shared MR x MP/Private HT < K - 1

MP(Private MR - Shared MR)/Private HT < K - 1

Assuming that the hit time for the shared cache (Shared HT) is K times that of a private cache and the miss penalty (MP) is 100 times that of the private hit time (Private HT). Then

Private MR - Shared MR < (K-1)/100
K>(Private_MR-Shared_MR)+1
Providing a fast access to a shared cache is not trivial. Also, contention effects which we have not taken into account will make the hit time for the shared cache longer. That's why private first-level caches are the preferred solution whereas having a shared secondary or tertiary cache can provide the benefits shown in this exercise.

Problem 3.27

a. A bank number where a component Xi of a vector is stored is computed by (Stride * i) mod 32. The number of component accesses between two consecutive accesses to the same bank is called the gap denoted K. If the stride is 1, the same bank is accessed after 32 consecutive component accesses of a vector and K is 32.

The value of K for each stride S from 1 to 32 is shown in Table 44.

Table 44: Number of vector components fetched between conflicts(32 banks)

S	K	S	K	S	K	S	K
1	32	9	32	17	32	25	32
2	16	10	16	18	16	26	16
3	32	11	32	19	32	27	32
4	8	12	8	20	16	28	8
5	32	13	32	21	32	29	32
6	16	14	16	22	16	30	16
7	32	15	32	23	32	31	32
8	4*	16	2*	24	4*	32	1*

The results in Table 44 can be obtained informally by considering each stride, from 1 to 32. Since the access time of a bank is 8 clocks, conflicts occur when the gap is less than 8, i.e., 1,2 or 4. The strides causing conflicts are marked by an asterisk in Table 44 (S=8,16,24, and 32).

To dig further, let's start with the first access i=0 to bank 0. Whenever an access i accesses bank 0 it must be that ixS is a multiple of 32 so that ixS mod32=0 and, of course, ixS must be a multiple of S. Thus ixS must be divisible by both S and 32 and the first such instance is for ixS = LCM(S,32) (LCM is the least common multiple).
Thus the gap K is equal to LCM(S,32)/S, which is equal to 32/GCD(S,32). This formula extends the computation of K for all S. Because the bank access time is 8, a conflict occurs if K is less than 8.

Conclusion: A bank conflict occurs if $\frac{32}{GCD(S,32)} = 8$.

Hence, all strides except multiples of 8, such as 8,16,24, and 32, avoid conflicts.

b. The same formula obtained in part a applies here too and K is $\frac{31}{GCD(S,31)}$.

Because 31 is a prime number and the GCDs of all strides except for multiples of 31 are 1, the same bank is accessed every 31 accesses (K=31), except if the stride is 31, in which case the same bank is accessed on every access. Hence, all strides except multiples of 31 avoid bank conflicts.

c. In a memory system with 31 banks the number of strides causing bank conflicts is much less than in a memory system with 32 banks because 31 is a prime number. However, computing an address modulo 31 is much more complex than computing an address modulo 32.

Solution to 2019 Re-exam Problem 3

(a) States E and O:
‘E’ means ‘exclusive’ . Cache has single copy, and memory is clean
‘O’ means ‘owner’. Cache has multiple copies, only one in state O. ‘Owner’ block replies to BusRd requests, enables sharing of dirty data.

(b) Protocol transitions

	MSI		**MESI**		**MOESI**	
Sequence	Cache 1	Cache 2	Cache 1	Cache 2	Cache 1	Cache 2
R1/X	I → S (BusRd)		I → E (BusRd)		I → E (BusRd)	
R2/X		I → S (BusRd)	E → S	I → S (BusRd)	E → O (Flush)	I → S (BusRd)
R2/Y		I → S (BusRd)		I → E (BusRd)		I → E (BusRd)
W1/X	S → M (BusUpgr)	S → I	S → M (BusUpgr)	S → I	O → M (BusUpgr)	S → I
W2/Y		S → M (BusUpgr)		E → M		E → M

(c) Latencies and traffic values

	MSI		**MESI**		**MOESI**	
Sequence	Cache 1	Cache 2	Cache 1	Cache 2	Cache 1	Cache 2
R1/X	5 + 100 / 6+B		5 + 100 / 6+B		5 + 100 / 6+B	
R2/X	5 + 100 / 6+B		5 + 100 / 6+B		5 + 20 / 6+B	
R2/Y	5 + 100 / 6+B		5 + 100 / 6+B		5 + 100 / 6+B	
W1/X	5 + 10 / 10		5 + 10 / 10		5 + 10 / 10	
W2/Y	5 + 10 / 10		1 / 0		1 / 0	
Total	345 / 38 + 3xB (96) 345 cycles / 134 bytes		310 +21 = 331 / 28 + 3xB (96) = 124 bytes 331 cycles / 124 bytes		230 + 26 = 256 / 124 bytes 256 cycles / 124 bytes	

Protocol	**Latency (cycles)**	**Traffic (bytes)**
MSI	345	134

MESI	331	124
MOESI	256	124