Chapter 8. Shared Memory Multiprocessors

Problem 8.1

a.

$$\begin{aligned}
\lambda &= 2/\text{sec} \\
\rho_A &= \lambda_a T_{sa} = \lambda p T_{sa} = 2/\text{sec} \times .1(\text{ sec}) \times p = .2p \\
\rho_B &= \lambda_b T_{sb} = \lambda(1-p) T_{sb} = .4(1-p) \\
T_{wa} &= \frac{\rho_A}{1-\rho_A} T_s = \frac{.2p}{1-.2p} \times .1(\text{sec})(FromM/M/1model) \\
T_{wb} &= \frac{.08(1-p)}{1-.4(1-p)} \\
T_{\text{A-total}} &= T_{wa} + T_{sa} = \frac{.02p}{1-.2p} + .1 \\
T_{\text{B-total}} &= T_{wb} + T_{sb} = \frac{.08(1-p)}{1-.4(1-p)} + .2 \\
T_{\text{total}} &= pT_{\text{A-total}} + (1-p)T_{\text{B-total}}
\end{aligned}$$

We have to minimize T_{total} because we need to minimize average response time.

$$\begin{aligned}
T_{\text{total}} &= T(p) \\
&= p(\frac{.02p}{1-.2p} + .1) + (1-p)(\frac{.08(1-p)}{1-.4(1-p)} + .2) \\
&= \frac{p}{10-2p} + \frac{1-p}{3+2p}
\end{aligned}$$

Within the range of $0 \le p \le 1$, $T'(p) < 0$. This means that $T(p)$ is a decreasing function. So, we get the minimum at $p = 1$:

$$T(p=1) = \frac{1}{10-2} - \frac{1-1}{3+2} = .125 \text{ (sec)}.$$

b. $\lambda = 6$ per sec

$T(p) = \frac{p}{10-6p} + \frac{1-p}{6p-1}$

$T'(p) = \frac{10}{(10-6p)^2} - \frac{5}{(6p-1)^2}$

With $p = .788$, $T(p)$ has its minimum value.

So, $T(p = .788) = .2063$ sec.

Problem 8.3

a. With no cache miss

1 instruction per cycle with no cache miss

b. With cache miss

CPI loss due to cache miss = (1.5 refs/I) × (.04 miss/refs) × (8 cycles/miss)
= .48 CPI

Total CPI for all four processors= 4 CPI_{base} + .48 $\text{CPI}_{\text{penalty}}$
= 4.48 CPI

CPI for four-processor ensemble = 1.12 CPI

Problem 8.4

a. Single pipelined processor performance

Base CPI = 1

CPI loss due to branch = $.2 \times 2 = .4$

CPI loss due to cache miss = $.01 \times 8 \times 1.5 = .12$

CPI loss due to run-on delay = $.1 \times 3 = .3$

$CPI_{\text{Total}} = 1 + .4 + .12 + .3 = 1.82$

b. How effective in the use of parallelism must be?

CPI for SRMP from problem 8.3 is 1.12

Speedup of SRMP = $\frac{1.82}{1.12} = 1.65$

The speedup of 1.8 can be achieved only when we can find 4 independent instructions that can be concurrently executed on SRMP. So, to provide overall speedup over single processor, we should at least be able to find $4/1.65 = 2.42$ independent instructions on average. In another words, $2.42/4 = 60.6$ % utilization of SRMP.

Problem 8.5

Note: we assume the processor halts on cache miss and bus is untenured.

m processors

Time to transfer a line = 8 cycles

$w = .5$

Miss rate = 2%

1.5 refs/inst

a. Bus occupancy, ρ

Bus Transaction time (per 100 inst)

= 100 (inst) $\times$ .02 $\times$ 1.5 (refs/inst) $\times$ (1 +.5) $\times$ 8 cycles = 36 cycles

Processor time (per 100 inst) = 100 cycles since CPI = 1

$\rho = \frac{36}{100+36} \approx .265$ for a single processor

b. Number of processors allowed before bus saturates

$n = \frac{100}{36} = 2.78$

Only two processors can be placed before saturation occurs.

c. To find ρ_a, solve the following two equations by iteration:

$B(n) = n\rho_a = 1 - (1-a)^n$

$a = \frac{\rho}{\rho + \frac{\rho_a}{\rho}(1-\rho)}$

$\rho_a = .243$ by iteration

$B(n=2) = 2 \times \rho_a = 2 \times .243 = .486$

$T_w = \frac{n\rho - B(n)}{B(n)} T_s = \frac{2 \times .265 - .486}{.486} T_s = .09$ (bus cycles)

Problem 8.6

Note: assume processor blocks on a cache miss and bus is untenured.

$n = 4$

Bus time = 8 bus cycles = 8 processor cycles

$B(n=4) = 4\rho_a = 1 - (1 - \frac{\rho}{\rho + \frac{\rho_a}{\rho}(1-\rho)})^4$

$\rho_a = .198$

$B(n=4) = 4 \times .198 = .792$

$T_w = \frac{n\rho - B(n)}{B(n)} T_s = \frac{4 \times .265 - .792}{.792} \times 8$ bus cycles $= 2.707$ bus cycles

For every 100 instructions, $100 \times .02 \times 1.5 \times 1.5 = 4.5$ bus transactions occur. That is, 22.22 instructions are executed between two bus transactions. This is processor execution time.

a. Find λ.

λ_a= 1/(Processor execution time + bus time + T_w)
= $\frac{1}{22.22+8+2.707}$
=.0304/bus cycle

b. Performance
$\frac{\lambda_a}{\lambda} = \frac{22.22+8}{22.22+8+2.707} = .918$
Achieved performance = .918× original performance

Problem 8.8

a. With resubmission: (This is already done in 8.5 c.)
$B(n=2) = .486$, $\frac{T_w}{T_s} = .09$

b. Without resubmission:
$\rho = .265$
$B(n=2) = 1 - (1-\rho)^n = 1 - (1 - .265)^2 \approx .460$
$\rho_a = \frac{B(n=2)}{2} = .23$
$\frac{T_w}{T_s} = \frac{n\rho - B}{B} = \frac{2 \times .265 - .460}{.460} = .15$

Problem 8.9

Baseline network: 4×4 switch, $k = 4$
$N = 1024$
$l = 200$ bits (both requests and reply)
$w = 16$ bits
$h = 1$

Assume we ignore service time:

a. $n = \lceil \log_k N \rceil = \lceil \log_4 1024 \rceil = 5$

b. Without network contention:
$T_{\text{dynamic}} = T_c = n + \frac{l}{w} + h = 5 + \frac{200}{16} + 1 = 18.5$
$T_{\text{total(request+reply)}} = 2 \times 18.5 = 37$ cycles

c. $m = .015$, $\rho = m\frac{l}{w} = .015 \times \frac{200}{16} = .188$
$T_w = \frac{\rho \frac{l}{w}(1-\frac{1}{k})}{2(1-\rho)} = \frac{.188 \times \frac{200}{16} \times (1-\frac{1}{4})}{2(1-.188)} = 1.085$ cycles
$T_{\text{dynamic}} = 18.5 + 1.085 = 19.59$ cycles
$T_{\text{total (request reply)}} = 2 \times 19.59 = 39.17$ cycles

Problem 8.11

Direct static network

Nodes $= 32 \times 32 -$ 2D Torus

$l = 200$

$w = 32$

$h = 1$

a. Expected distance
$nk_d = n \times \frac{k}{4} = 2 \times \frac{32}{4} = 16$

b. $T_{\text{static}} = T_c = h \times n \times k_d + \frac{l}{w} = 16 + \frac{200}{32} = 22.25$ cycles
$T_{\text{total(request+reply)}} = 22.25 \times 2 = 44.5$ cycles

c. $m = .015$

$\rho = \frac{mk_d}{2}\frac{l}{w} = \frac{.015 \times 8}{2} \times \frac{200}{32} = \frac{.75}{2} = .375$

$T_w = \frac{\rho}{1-\rho}\frac{l}{k_d \times w}(1 + \frac{1}{n}) = \frac{.375}{1-.375}\frac{200}{8 \times 32}(1 + .5) = .703$

d. $T_{\text{total}} = 2 \times (T_c + T_w) = 2 \times (22.25 + .703) = 45.9$ cycles

Problem 8.14

Note: In this problem, we ignore the assumption on low occupancy although this might simplify the problem. So, the graph is valid through a relatively large occupancy (or m). Another important point is that a hypercube is likely to have more network requests from neighboring nodes than a grid, since more nodes are connected to a node. So, for fair comparison, the network latency in a grid should be compared with a 16 times larger value of m in a hypercube.

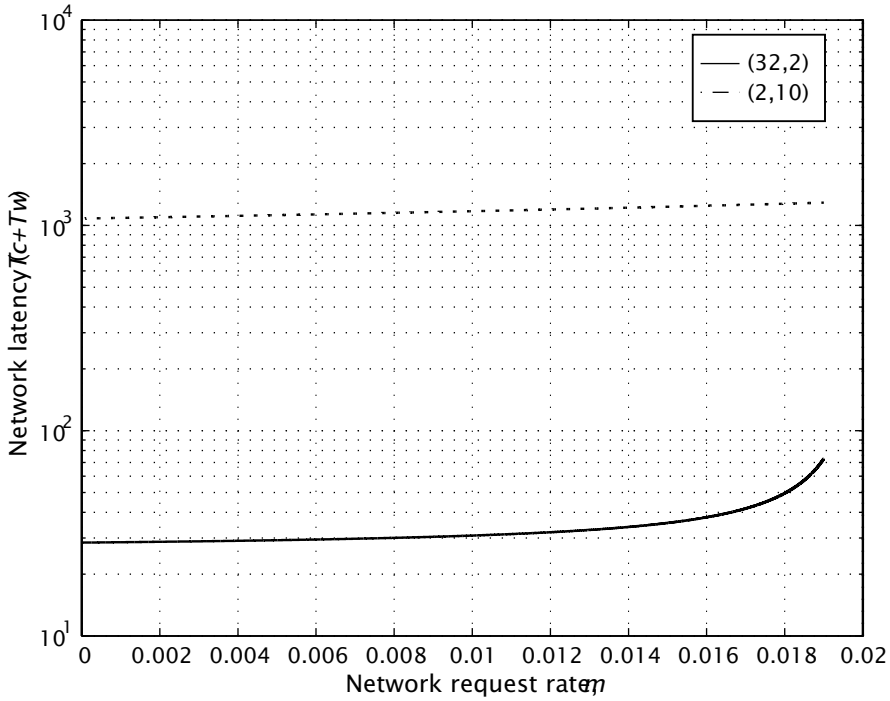

Figure 4: Problem 8-14.

a. $(k, n) = (32, 2)$-grid

$T_{ch} = C_1(32^{\frac{2}{2}-1}) = C_1$ s

$l/w = 200/16 = 12.5$

Fan-in + Fan-out $= 64 = 2nw \rightarrow w = \frac{64}{2\times 2} = 16$

$k_d = \frac{k}{4} = \frac{32}{4} = 8$

$T_c = nk_d + \frac{l}{w} = 2 \times 8 + 12.5 = 28.5$ cycles

$\rho = \frac{mk_d l}{2w} = m\frac{8}{2} \times 12.5 = 50m$

$T_w = \frac{\rho}{1-\rho}\frac{l}{w}\frac{1}{k_d}(1 + \frac{1}{n}) = \frac{50m}{1-50m} \times 12.5 \times \frac{1}{8} \times 1.5 = \frac{117.188m}{1-50m}$ cycles

$\frac{T}{C_1} = \frac{(T_c+T_w)T_{ch}}{C_1} = 28.5 + \frac{117.188m}{1-50m}$

b. (2,10)

$T_{ch} = C_1(2^{\frac{10}{2}-1}) = 16C_1$ s

$l/w = 200/3.2 = 62.5$

Fan-in + Fan-out $= 64 = 2nw \rightarrow w = \frac{64}{2\times 10} = 3.2$

$k_d = \frac{k}{4} = \frac{2}{4} = .5$

$T_c = nk_d + \frac{l}{w} = 10 \times .5 + 62.5 = 67.5$ cycles

$\rho = \frac{16mk_d l}{2w} = 250m$

$T_w = \frac{\rho}{2(1-\rho)}\frac{l}{w} = \frac{250m}{2(1-250m)} \times 62.5 = \frac{7812.5m}{1-250m}$ cycles

$\frac{T}{C_1} = \frac{(T_c+T_w)T_{ch}}{C_1} = 16 \times 67.5 + \frac{7812.5m}{1-250m} = 1080 + \frac{7812.5m}{1-250m}$

Problem 8.17

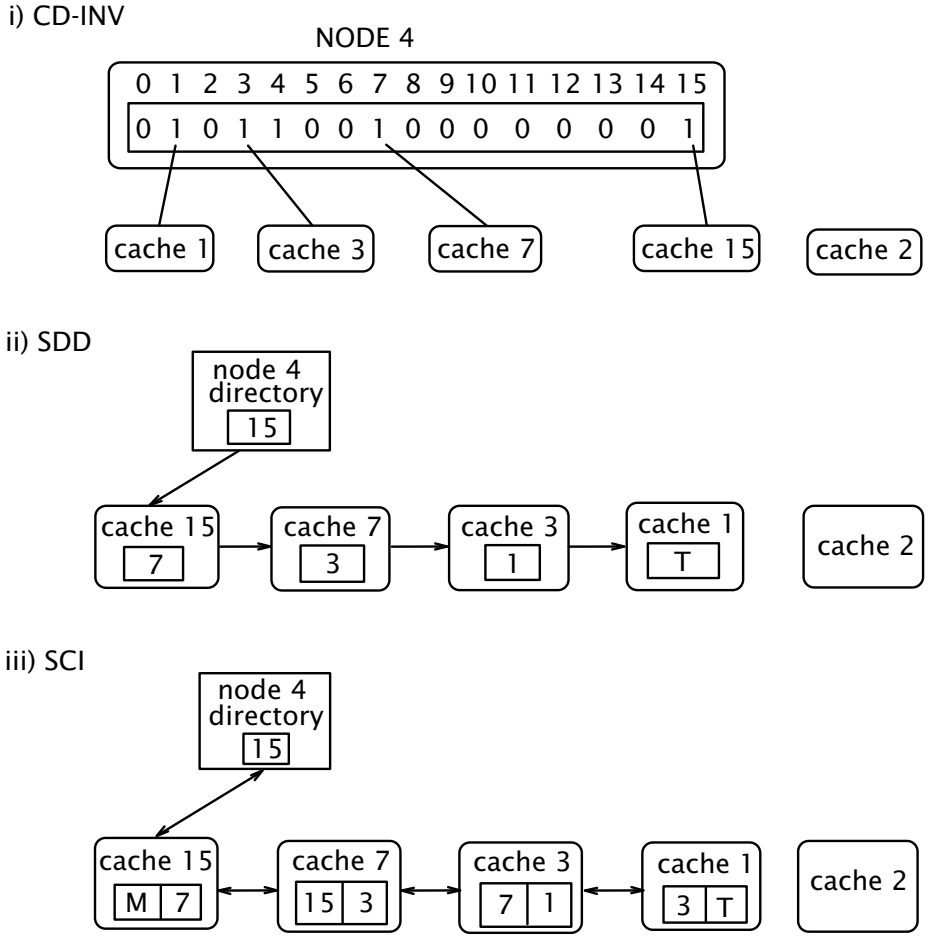

Figure 5: Problem 8-17(a): Immediately before the write takes place.

Problem 8.18

Line transmission takes 9 cycles

Invalidation or acknowledgement takes 1 cycle per hop

$15 \leftrightarrow 0 \leftrightarrow 1 \leftrightarrow 2 \leftrightarrow 3 \leftrightarrow 4 \leftrightarrow 5 \leftrightarrow 6 \leftrightarrow 7$.

a. CD-INV

Step 1: Node 2 sends write miss to node 4 (2 cycles)

Step 2: Node 4 replies to node 2 with invalidation count and data (9 + 1 cycles). Assume step 3 begins immediately after node 4 sends.

Step 3: We assume invalidation or acknowledgement sends out in both directions.

Cycle	1	2	3	4	5	6	7	8	9
	4→7 (I) 4→15 (I)			7→2 (A)		15→2 (A) done			done done
		4→1 (I)			1→2 (A) done				
			4→3 (I)	3→2 (A)					

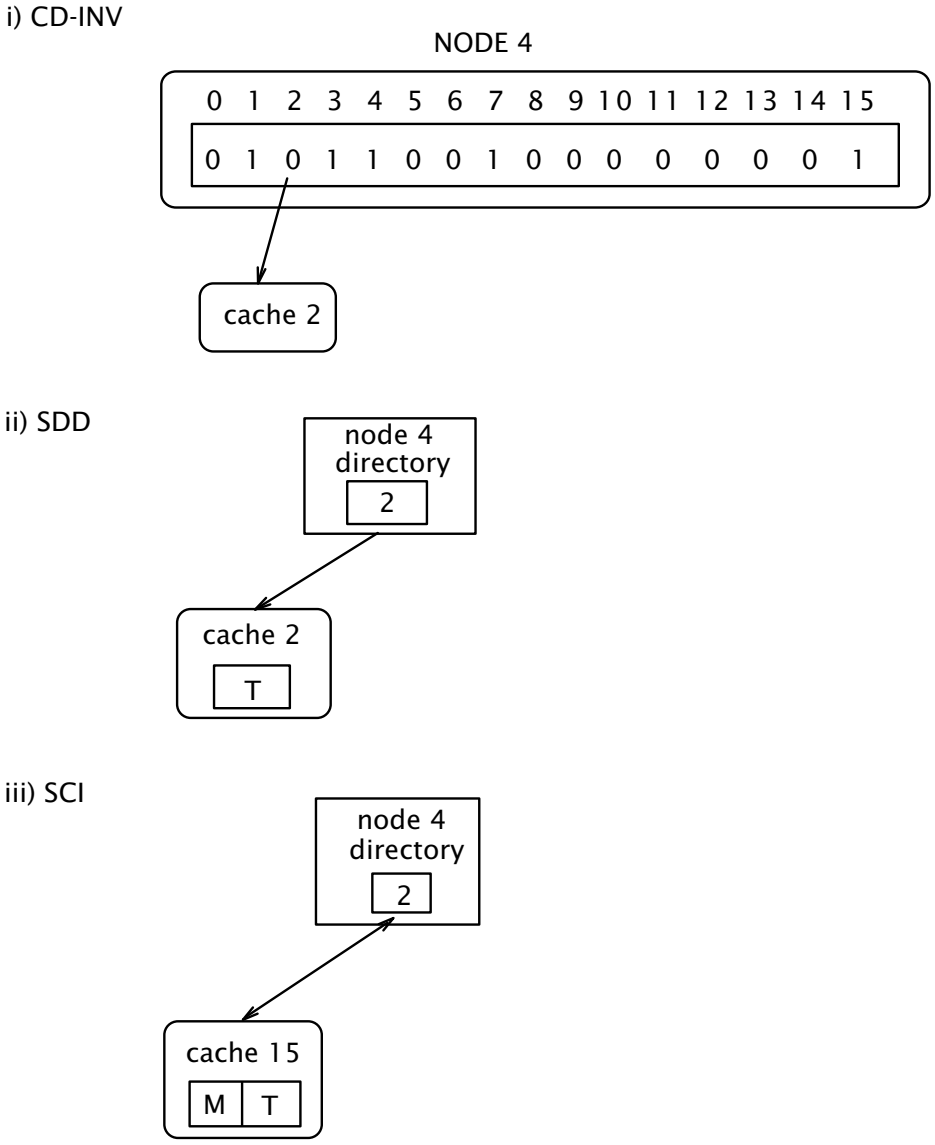

Figure 6: Problem 8-17(b): After the write takes place and is acknowledged.

I means Invalidation, and A means acknowledgement.

Total cycles = 2 + 10 + 9 = 21 cycles.

b. SCI

Step 1: cache 2 sends write miss to node 4 (2 cycles)
Step 2: node 4 sends cache 15 to cache 2 (2 cycles)
Step 3: cache 2 invalidates cache 15 (3 cycles)
Step 4: cache 15 acknowledges to cache 2 with cache 7 and data (3 + 9 cycles)
Step 5: cache 2 invalidates cache 7 (5 cycles)
Step 6: cache 7 acknowledges to cache 2 with cache 3 (5 cycles)
Step 7: cache 2 invalidates cache 3 (1 cycle)
Step 8: cache 3 acknowledges to cache 2 with cache 1 (1 cycle)
Step 9: cache 2 invalidates cache 1 (1 cycle)
Step 10: cache 1 acknowledges to cache 2 with tail (1 cycle)

Total network cycles = 33.

Problem 8.19

a. Immediately before the write takes place

(i) CD-UP Node 4 directory

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	0	1	0	0	0	1	0	0	0	0	0	0	0	1

(ii) DD-UP

node 4		cache 15		cache 7		cache 3		cache 1		cache 2
15	→	7	→	3	→	1	→	T		

b. After write takes place and is acknowledged

(i) CD-UP Node 4 directory

0	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15
0	1	1	1	0	0	0	1	0	0	0	0	0	0	0	1

(ii) DD-UP

node 4		cache 2		cache 15		cache 7		cache 3		cache 1
2	→	15	→	7	→	3	→	1	→	T