The University of Alabama in Huntsville
ECE Department
Homework Assignments
CPE 431/531 01/91/92
Fall 2015
Homework #7 Solution

5.8.1(5), 5.8.2(5), 5.8.3(5), 5.8.4(5), 5.9.1(5), 5.9.3(10), 5.10.1(10), 5.10.4(10), 5.11.1(10), 5.11.3(15), 5.12.2(10), 5.15.2(10)

5.8 Mean Time Between Failures (MTBF), MEAN Time To Replacement (MTTR),a nd Mean Time to Failure (MTTF) are useful metrics for evaluating the reliability and avilablility of a storage resource. Explore these basic concepts by answering the questions about devices with the following metrics.

MTTF	MTTR
3 Years	1 Day

5.8.1 Calculate the MTBF for the device given.

MTBF = MTTF + MTTR = 3*365 + 1 = 1096 days

5.8.2 Calculate the availability for the device given.

Availability = MTTF/(MTTF + MTTR) = 1095/(1096) = 99.9%

5.8.3 What happens to availability as the MTTR approaches 0? Is this a realistic situation?

As MTTR $\rightarrow$ 0, availability approaches 1. With the emergence of inexpensive drives, having a nearly 0 replacement time *for hardware* is quite feasible. However, replacing fi le systems and other data can take signifi cant time. Although a drive manufacturer will not include this time in their statistics, it is certainly a part of replacing a disk.

5.8.4 What happens to availability as the MTTR gets very high, i.e., a device is difficult to repair? Does this imply the device has low availability?

As MTTR $\rightarrow\infty$, availability $\rightarrow$ 0.
MTTR becomes the dominant factor

5.9 This Exercise examines the single error correcting, doble error detecting (SEC/DED) Hamming code.

5.9.1 What is the minimum number of parity bits required to protect a 128-bit word using the SEC/DED code?

p is the number of parity bits needed
d is the number of data bits
For single error correction, the expression that must hold is $2^p \geq p + d + 1$

p has to be at least 7 because $\log_2 128 = 7$
Try p =8, $2^8 \geq 128 + 8 + 1$, 256 > 137, that works
1 more bit is needed for double error detection, bringing the total to 9.

5.9.3 Consider a SEC code that protects 8 bit words with 4 parity bits. If we read the value 0x375, is there an error? If so, correct the error.

Extract the correct data value from the even parity SEC string 0001010011010

```
1234 5678 9111
            012
0011 0111 0101
```

C_1 = XOR(1, 3, 5, 7, 9, 11) = XOR(0, 1, 0, 1, 0, 0) = 0
C_2 = XOR(2, 3, 6, 7, 10, 11) = XOR(0, 1, 1, 1, 1, 0) = 0
C_4 = XOR(4, 5, 6, 7, 12) = XOR(1, 0, 1, 1, 1) = 0
C_8 = XOR(8, 9, 10, 11, 12) = XOR(1, 0, 1, 0, 1) = 1

C = 1000, it's a single error in bit 8, Corrected value = 0011 0110 0101

5.10 For a high-performance system such as a B-tree index for a database, the page size is determined mainly by the data size and disk performance. Assume that on average a B-tree index page is 70% full with fix-sized entries. The utility of a page is its B-tree depth, calculated as $\log_2$(entries). The following table show that for 16-byte entries, and a 10-year-old disk with a 10 ms latency and 10 MB/s transfer rate, the optimal page size is 16K.

Page Size (KiB)	Page Utility or B-tree Depth (Number of Disk Accesses Saved)	Index Page Access Cost (ms)	Utility/Cost
2	6.49 (or $\log_2$(2048)/16 x 0.7))	10.2	0.64
4	7.49	10.4	0.72
8	8.49	10.8	0.79
16	9.49	11.6	0.82
32	10.49	13.2	0.79
64	11.49	16.4	0.70
128	12.49	22.8	0.55
256	13.49	35.6	0.38

5.10.1 What is the best page size if entries now become 128 bytes?

Page Size (KiB)	**Page Utility or B-tree Depth (Number of Disk Accesses Saved)**	**Index Page Access Cost (ms)**	**Utility/Cost**
2	**3.48 (or $\log_2$(2048/128 x 0.7))**	**10.2**	**0.34**
4	**4.48**	**10.4**	**0.43**
8	**5.48**	**10.8**	**0.51**
16	**6.48**	**11.6**	**0.56**

32	**7.48**	**13.2**	**0.57**
64	**8.48**	**16.4**	**0.52**
128	**9.48**	**22.8**	**0.42**
256	**10.48**	**35.6**	**0.29**

The best is now 32 KiB.

Keeping "frequently used" (or "hot") pages in DRAM can save disk accesses, but how do we determine the exact meaning of "frequently used" for a given system? Data engineers use the cost ratio between DRAM and disk access to quantify the reuse time threshold for hot pages. The cost of a disk access is $Disk/accesses_per_sec, while the cost to keep a page in DRAM is $DRAM_MiB/page_size. The typical DRAM and disk costs and typical database page sizes at several time points are listed below:

Year	DRAM Cost ($/MiB)	Page Size (KiB)	Disk Cost ($/disk)	Disk Access Rate (access/sec)
1987	5000	1	15,000	15
1997	15	8	2000	64
2007	0.05	64	80	83

5.10.4 What are the reuse time thresholds for these three technology generations?

The reuse threshold occurs when the cost of keeping a page in memory is the same as the cost per page for the disk system.
Cost of page in DRAM = (Price of 1 MB DRAM/# of pages in 1 MB)
Cost of page in disk system= (Cost of disk/# of pages)
In disk system, # of pages = accesses_per_sec x # of seconds
Let x be the number of seconds

$$\frac{\$/MB}{\#\,pages/MB} = \frac{\$Disk}{accesses/s \times x}$$

$$x \times \$/MB \times accesses/s = \$Disk \times \#\,pages/MB$$

$$x = \frac{\$Disk \times \#\,pages/MB}{\$/MB \times accesses/s}$$

$$x_{1987} = \frac{\$15000 \times 1024}{\$5000 \times 15/s} = 205s \qquad x_{1997} = \frac{\$2000 \times 128}{\$15 \times 64/s} = 267s$$

$$x_{2007} = \frac{\$80 \times 16}{\$0.05 \times 83/s} = 308s$$

5.11 As described in Section 5.4, virtual memory uses a page table to track the mapping of virtual addresses to physical addresses. This exercise shows how this table must be updated as addresses are accessed. The following data constitutes a stream of virtual addresses as seen on a system. Assume 4KiB pages, a 4-entry fully associative TLB, and true LRU replacement. If pages must be brought in from disk, increment the next largest page number.

4669, 2227, 13916, 34587, 48870, 12608, 49225

TLB

Valid	Tag	Physical Page Number
1	11	12
1	7	4
1	3	6
0	4	9

Page table

Valid	Physical page or in disk
1	5
0	Disk
0	Disk
1	6
1	9
1	11
0	Disk
1	4
0	Disk
0	Disk
1	3
1	12

5.11.1 Given the address stream, and the shown initial state of the TLB and page table, show the final state of the system. Also list for each reference if it is a hit in the page table, or a page fault.
These are byte addresses. The page offset is $\log_2 4K = 12$. The virtual page number can be obtained by dividing the byte address by 4KB (4096) and taking the integer part of the result.

	VPN	TLB?	PT?	PF?
4669	1	M	M	Y
2227	0	M	H	N
13916	3	H	H	N
34587	8	H	M	Y
48870	11	M	H	N
12608	3	H	H	N
49225	12	M	M	Y

TLB

Valid	Tag	PPN
1	~~11~~, 8	~~12~~, 14
1	~~7,~~ 0	4, 5
1	3	6
0, 1	~~4,~~ ~~1~~, 12	~~9,~~ ~~13~~, 15

Page Table

VPN	Valid	PPN or in disk
0	1	5
1	0	~~Disk~~, 13
2	0	Disk
3	1	6
4	1	9
5	1	11
6	0	Disk
7	1	4
8	0	~~Disk~~, 14
9	0	Disk
10	1	3
11	1	12
12	~~0~~, 1	~~Disk~~, 15

5.11.3 **Show the final contents of the TLB if it is 2-way set-associative. Also show the contents of the TLB if it is direct mapped. Discuss the importance of having a TLB to high performance. How would virtual memory accesses be handled if there were no TLB?**

$$4entries \times \frac{1set}{2entties} = 2sets$$

	VPN	Index	TLB?	PT?	PF?
4669	1	1	M	M	Y
2227	0	0	M	H	N
13916	3	1	M	H	N
34587	8	0	M	M	Y
48870	11	1	M	H	N
12608	3	1	H	H	N
49225	12	0	M	M	Y

Page Table

VPN	Valid	PPN or in disk
0	1	5
1	0	~~Disk~~, 13
2	0	Disk
3	1	6
4	1	9
5	1	11
6	0	Disk
7	1	4
8	0	~~Disk~~, 14
9	~~0~~	~~Disk,~~
10	1	3
11	1	12
12	~~0~~	~~Disk,~~ 15

TLB

Index	Valid	Tag	PPN	Valid	Tag	PPN
0	1	~~2~~, 4	~~9~~, 14	~~0~~, 1	~~5~~, ~~0~~, 6	~~3~~, ~~5~~, 15
1	1	~~5~~, ~~0~~, 5	~~12~~, ~~13~~, 12	1	~~3~~, 1	4, 6

$$4entries \times \frac{1set}{1entry} = 4sets$$

	VPN	Index	TLB?	PT?	PF?
4669	1	1	M	M	Y
2227	0	0	H	H	N
13916	3	3	M	H	N
34587	8	0	M	M	Y
48870	11	3	M	H	N
12608	3	3	M	H	N
49225	12	0	M	M	Y

TLB

Index	Valid	Tag	PPN
0	1	~~0~~, ~~2~~, 3	~~5~~, ~~14~~, 15
1	1	~~1~~, 0	~~11~~, 13
2	1	2	3
3	0	~~2~~, ~~0~~, ~~2~~, 0	~~12~~, ~~6~~, ~~12~~, 6

Page Table

VPN	Valid	PPN or in disk
0	1	5
1	0	~~Disk~~, 13
2	0	Disk
3	1	6
4	1	9
5	1	11
6	0	Disk
7	1	4
8	0	~~Disk~~, 14
9	~~0~~	~~Disk~~
10	1	3
11	1	12
12	~~0~~	~~Disk~~, 15

5.12 In this exercise, we will examine space/time optimizations for pge tables. The following list provides parameters of a virtual memory system.

Virtual Address (bits)	Physical DRAM Installed	Page Size	PTE Size (byte)
43	16 GiB	4 KiB	4

5.12.2 Using a multilevel page table can reduce the physical memory consumption of page tables, by only keeping active PTEs in physical memory. How many levels of page tables will be needed in this case? And how many memory references are needed for address translation if missing in TLB?

You want each level of the page table to fit in one page, so 4 KiB/4B = 1 K entries, using 10 bits each. So, there are 31 bits of virtual page number (43 – the 12 bits of page offset requires by 4

KiB pages). The number of levels is equal to ⌈#bits in virtual page #/#bits in one level of page table⌉. In this case, that is ⌈31/10⌉, or 4.Each address translation will take 4 memory accesses.

5.15 One of the biggest impediments to widespread use of virtual machines is the performance overhead incurred by running a virtual machine. Listed below are various performance parameters and application behavior.

Base CPI	Privileged O/S Accesses per 10,000 Instructions	Performance Impact to Trap to the Guest O/S	Performance Impact to Trap to VMM	I/O Accesses per 10,000 Instructions	I/O Access Time (Includes Time to Trap to Guest O/S)
1.5	120	15 cycles	175 cycles	30	1100 cycles

5.15.2 I/O accesses often have a large impact on overall system performance. Calculate the CPI of a machine using the performance characteristics above, assuming a non-virtualized system. Calculate the CPI again, this time using a virtualized system. How do these CPIS change if the system has half the I/O accesses? Explain why I/O bound applications have a smaller impact from virtualization.

Non virtualized: CPI = 1.5 + 30/10000 × 1100 = 4.8
Virtualized: CPI = 1.5 + 120/10000 × (15 + 175) + 30/10000 × (1100 + 175) =7.60
Virtualized with Half I/O: CPI = 1.5 + 120/10000 × (15 + 175) + 15/10000 × (1100 + 175) =5.69