SYSC5704 – ELEMENTS OF COMPUTER SYSTEMS

ASSIGNMENT-5 REPORT

Submitted by

Aarthi Thirumavalavan

Student ID. 100958800

Master of Engineering. Electrical and Computer Engineering

Carleton University

5.1 In this exercise we look at memory locality properties of matrix computation. The following code is written in C, where elements within the same row are stored contiguously. Assume each word is a 32-bit integer.

```
for (I=0; I<8; I++)
for (j=O; J<8000; J++)
A[I][J] = B[I][0]+A[J][I];
```

1. [5] How many 32-bit integers can be stored in a 16-byte cache block?

   SOLUTION: The lines can store four integers
   = 16 x 8 = 128
   = 128 / 32 bit integer
   = 4

2. References to which variables exhibit temporal locality?
   SOLUTION: I and J variables exhibit temporal locality

3. References to which variables exhibit spatial locality?
   SOLUTION:

   A[i][j] exhibit spatial locality.
   Since I and J are located near each other A[I][J] exhibit spatial locality.

   Locality is affected by both the reference order and data layout. The same computation can be written below in Matlab, which differs from from C by storing matrix elements from the same column contiguously in memory.

```
for I=I:8
for J=1:8000
A(I,J)=B(I,0)+A(J,I)
end
end
```

4. How many 16-byte cache blocks are needed to store all 32-bit matrix elements being referenced?
   SOLUTION:

   Total number of elements in 2D-array = 8000 x 8 = 64000 elements for each 32 bits
   = [8 x 800 / (4 x 2)] – [8 x 8 /4] + [8000 / 4]
   = 3596

5. References to which variables exhibit temporal locality?

   SOLUTION:
   The variables I and J are accessed after every iteration of loop.
   I and J exhibits temporal locality

6 References to which variables exhibit spatial locality?

SOLUTION:

A(I , J) exhibits spatial locality, since loop is the same.

5.5

Media applications that play audio or video files are part of a class of workloads called "streaming" workloads; i.e., they bring in large amounts of data but do not reuse much of it. Consider a video streaming workload that accesses a 512 KiB working set sequentially with the following address stream:

0, 2, 4, 6, 8, 10, 12, 14, 16, ...

1. [5] Assume a 64 KiB direct-mapped cache with a 32-byte What is the miss rate for the address stream above? How is this miss rate sensitive to the size of the cache or the working set? How would you categorize the misses this workload is experiencing, based on the 3C model?

   SOLUTION:

   The miss rate does not change with the cache size or the working set, these are the cold misses. **Therefore the miss rate is 12.5%**

2 Re-compute the miss rate when the cache block size is 16 bytes, 64 bytes, and 128 bytes.

What kind of locality is this workload exploiting?

SOLUTION:

For 16 byte the miss rate is 25%

For 64 byte the miss rate is 6.25%

For 128 bytes the miss rate is 3.125%

3 "Prefetching" is a technique that leverages predictable address patterns to specttlatively bring in additional cache blocks when a particular cache block is accessed. One example of prefetching is a stream buffer that prefetches sequentially adjacent cache blocks into a separate buffer when a particular cache block is brought in. If the data is found in the prefetch buffer, it is considered as a hit and moved into the cache and the next cache block is prefetched. Assume a two-entry stream buffer and assume that the cache latency is such that a cache block can be loaded before the computation on the previous cache block is completed. What is the miss rate for the address stream above?

SOLUTION:

The missed rate for the given address stream with the next line prefetching will be around 0%

Cache block size (B) can affect both miss rate and miss latency. Assuming a 1-CPI machine with an average of 1.35 references (both instruction and data) per instruction, help find the optimal block size given the following miss rates for various block sizes.

8:4%	16:3%	32:2%	64:1.5%	128:1%

4 What is the optimal block size for a miss latency of 20 × B cycles?

SOLUTION:

Minimize the average latency for memory access
= Miss Rate x Miss Penalty
For Block size 8:
Average latency equation = 4 x 20 x 8
Average Latency = **640**

For Block size 16:
Average latency equation = 3 x 20 x16
Average Latency = **960**

For Block size 32:
Average latency equation = 2 x 20 x 32
Average Latency = **1280**

For Block size 64:
Average latency equation = 1.5 x 20 x 64
Average Latency = **1920**

For Block size 128:
Average latency equation = 1 x 20 x 128
Average Latency = **2560**

The optimal block size for miss latency of 20 x b cycles is 16 bytes

5 What is the optimal block size for a miss latency of 24 + B cydes?

SOLUTION:

For Block size 8:
Average latency equation = 4 x (24 + 8)
Average Latency = **128**

For Block size 16:
Average latency equation = 3 x (24 + 16)
Average Latency = **120**

For Block size 32:
Average latency equation = 2 x (24 + 32)
Average Latency = **112**

For Block size 64:
Average latency equation = 1.5 x (24 + 64)
Average Latency = **132**

For Block size 128:
Average latency equation = 1 x (24 + 128)
Average Latency = **152**

The optimal miss latency for 24 + B cycle is 32 bytes

6 For constant miss latency, what is the optimal block size?

SOLUTION:

For Block size 8:
Average latency equation = 4 x C
Average Latency = **4C**

For Block size 16:
Average latency equation = 3 x C
Average Latency = **3C**

For Block size 32:
Average latency equation = 2 x C
Average Latency = **2C**

For Block size 64:
Average latency equation = 1 x C
Average Latency = **1C**
The optimal block size for constant miss latency will be 64 bytes

---

5.11) As described in Section 5.7, 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 4 KiB 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 Hint: the largest physical page number in the TLB and Page table shown below is 12, so the next page number will be 13. Be sure to track the access time under the "valid" column so that you can perform LRU. If more than one valid entry has NO last access time (like in the example), choose the first one first

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
0	Disk

1 Given the address stream shown, and the initial TLB and page table states provided above, show the final state of the system. Also list for each reference if it is a hit in

the TLB, a hit in the page table (PT), or a page fault (PF).To get you started, here is the first row of the table.

		TLB			
Address	Virtual Page	Hit/Miss	Valid	Tag	Physical Page
4669	1	TLB Miss PT Miss PF	1 1 1 1 (last access 0)	11 7 3 1	12 4 6 13

SOLUTION:

Virtual Page Number: 12 bits Address
0 = Miss in TLB hit on page
7 = Hit in TLB
3 = Miss in TLB hit on page
3 = Hit in TLB
1 = Page Fault
1 = Hit in TLB
2 = Page Fault

TLB:

Valid	Tag	Physical Page Number
1	3	6
1	7	4
1	1	13
1	2	14

Page Table:

Valid	Physical Page or in Disk
1	5
1	13
1	14
1	6
1	9
1	11

0	Disk
1	4
0	Disk
0	Disk
1	12

2 Repeat 5.11.1, but this time use 16 KiB pages instead of 4 KiB pages. What would be some of the advantages of having a larger page size? What are some of the disadvantages?

SOLUTION:

Virtual Page Number: 16 bits
0 = Miss in TLB hit in Page
3 = Hit in TLB
1 = Page Fault
1 = Hit in TLB
0 = Hit in TLB
0 = Hit in TLB
1 = Hit in TLB

TLB:

Valid	Tag	Physical Page Number
1	1	13
1	7	4
1	3	6
1	0	5

Page Table:

Valid	Physical Page or in Disk
1	5
1	13
0	Disk
1	6

1	9
1	Disk
0	4
1	Disk
0	Disk
0	3
1	12

Address can be stored in single page for large page sizes by decreasing the amount of pages that must be bought from disk and increasing the coverage of TLB. If program uses address sparsely then there will penalty for transferring pages.

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? To get you started, here is the first row of the table for the two-way set associative and direct mapped TLB.

Two Way Set Associative								
				TLB				
Address	Virtual Page	Tag	Index	Hit/Miss	Valid	Tag	Physical Page	Index
4669	1	0	1	TLB Miss PT Miss PF	1 1 1 1(last access 0)	11 7 3 0	12 4 6 13	0 1 0 1

Direct Mapped								
				TLB				
Address	Virtual Page	Tag	Index	Hit/Miss	Valid	Tag	Physical Page	Index
4669	1	0	1	TLB Miss PT Miss PF	1 1 1 1	11 0 3 4	12 13 6 9	0 1 2 3

There are several parameters that impact the overall size of the page table. Listed below are key page table parameters.

Virtual Address Size	Page Size	Page Table Entry Size

32 bits	8KiB	4 bytes

SOLUTION:

2-Way Associative:
TLB:

Valid	**Tag**	**PPN**	**Valid**	**Tag**	**PPN**
1	0	5	1	01	14
1	0	13	1	01	6

Direct Mapped:

Valid	**Tag**	**Physical Page Number**
1	0	5
1	0	13
1	0	14
1	0	6

TLB avoids high access time to memory in order to translate virtual address to physical address. Without TLB the page table will be referenced by using virtual addresses causing slowdown.

4 Given the parameters shown above, calculate the total page table size for a system running 5 applications that utilize half of the memory available. The assumption: "half the memory available" means half of the 32-bit virtual address space for each running application.

SOLUTION:

4KB = 12 Offset bits
20 Page Number Bits
2^{20} = 1M page table entries
= 1M entries x 4bytes/entry = 4MB page table per application
For 5 applications = 2^{22} x 5
For 5 applications = 20.92 MB

5 Given the parameters shown above, calculate the total page table size for a system running 5 applications that utilize half of the memory available, given a two level

page table approach with 256 entries. Assume each entry of the main page table is 6 bytes. Calculate the minimum and maximum amount of memory required.

SOLUTION:

4KB = 12 Offset bits
20 Page Number Bits
256 entries = Minimum of 128 entries per application
128 x 4096 entries for second level = 2^{19} entries
1M x 4byres/entry = 4MB second level page table per application
256 entries x 6bytes/entry = 1536 bytes for first level page table per application
For 5 application = 20.98MB

6 A cache designer wants to increase the size of a 4 KiB virtually indexed, physically tagged cache. Given the page size shown above, is it possible to make a 16 KiB direct-mapped cache, assuming 2 words per block? How would the designer increase the data size of the cache?

SOLUTION:

Given,
16KB Direct mapped cache
2 words per block = 8bytes/ block
16KB/8 bytes per block = 2K sets = 11 bits for indexing

16 bit Direct-Mapped cache needs the lower 14 bits to remain the same between VA to PA translation. So it's not possible to build this cache. If the cache's data size is to be increased, a higher associativity must be used. If the cache had 2 words per block, then only 9 bits are available for indexing. Therefore, to make a 16 KB cache, 4-way associativity must be used.

5.19) In this exercise we show the definition of a web server log and examine code optimizations to Improve log processing speed. The data structure for the log is defined

```
struct entry {
int srcIP; // remote IP address
char URL[128]; // request URL (e.g.. "GET index.html")
long long refTime; // reference time
int status; // connection status
char browser[64]; // client browser name
```

} log [NUM_ENTRIES];

Assume the following processing function for the log:

topK_sourceIP (int hour);

This function finds the top 1000 IP addresses during the period specified.

1 Which fields in a log entry will be accessed for the given log processing function? Assuming 64-byte cache blocks and no prefetching, how many cache misses per entry does the given function incur on average?

SOLUTION:
srcIP and refTime fields. 2 misses per entry.

2 How can you reorganize the data structure to improve cache utilization and access locality? Show your structure definition code.

SOLUTION: Group the srcIP and refTime fields into a separate array.

3 Another example of a log processing function finds the peak hours that connection request has the given status, e.g.:
peak_hour (int status); // peak hours of a given status
If both functions are important, how would you reorganize the data structure to improve the overall performance?

SOLUTION:
Group srcIP, refTime, and status together..