CARLETON UNIVERSITY

Department of Systems and Computer Engineering

SYSC 5704 **Elements of Computer Systems** **Assignment 5**

Due date: Tuesday, November 24th, 18:00.

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=0; J<8000; J++)
        A[I][J] = B[I][0]+A[J][I];
```

1. [5] <§5.1> How many 32-bit integers can be stored in a 16-byte cache block?
2. [5] <§5.1> References to which variables exhibit temporal locality?
3. [5] <§5.1> References to which variables 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. [10] <§5.1> How many 16-byte cache blocks are needed to store all 32-bit matrix elements being referenced?
5. [10] <§5.1> References to which variables exhibit temporal locality?
6. [10] <§5.1> References to which variables exhibit spatial locality?

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] <§§5.4,5.8> 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?
2. [5] <§§5.1, 5.8> 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?
3. [10] <§5.13> "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?

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. [10] <§5.3> What is the optimal block size for a miss latency of $20 \times B$ cycles?
5. [10] <§5.3> What is the optimal block size for a miss latency of $24 + B$ cydes?
6. [10] <§5.3> For constant miss latency, what is the optimal block size?

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	1

1. [10] <§5.7> 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.

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

2. [15] <§5.7> 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?
3. [15] <§§5.4, 5.7> 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

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

Direct Mapped

Address	Virtual Page	Tag	Index	TLB				
				Hit/Miss	Valid	Tag	Physical Page	Index
4669	1	0	1	TLB Miss PT hit PF	1	11	12	0
					1	0	13	1
					1	3	6	2
					0	4	9	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	8 KiB	4 bytes

4. [5] <§5.7> 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.

5. [10] <§5.7> 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.

6. [10] <§5.7> 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?

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. [5] <§5.15> 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?
2. [10] <§5.15> How can you reorganize the data structure to improve cache utilization and access locality? Show your structure definition code.
3. [10] <§5.15> 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?
4. [– skip –] <§5.15> – skip –
5. [– skip –] <§5.15> – skip –
6. [– skip –] <§5.15> – skip –