CS622 Assignment#3

In this assignment, you will develop a simulator to model a multi-core cache hierarchy supporting directory-based cache coherence. An example model is shown in Figure 1.

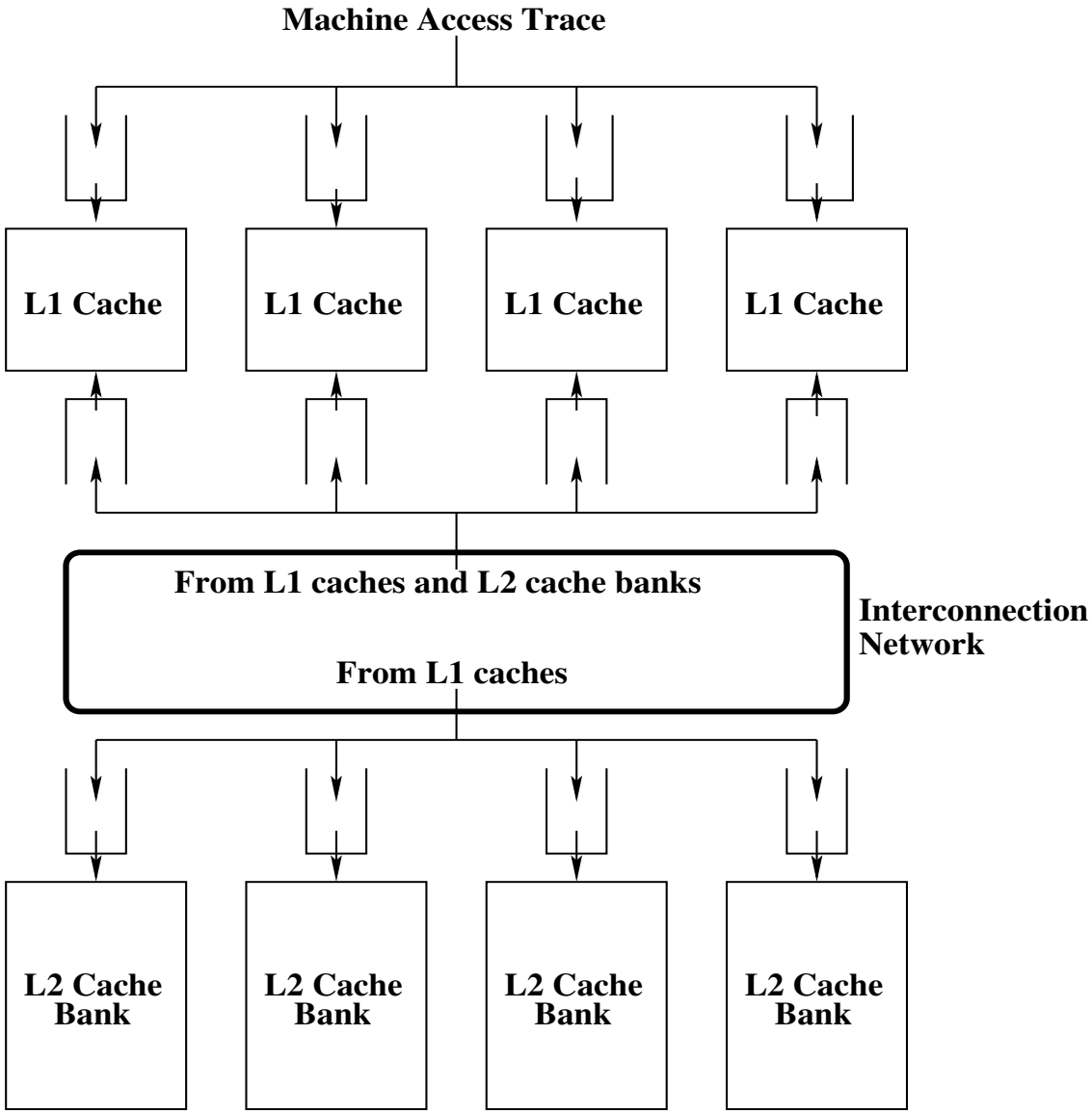

Figure 1. Example model.

You will model an array of L1 private caches and an array of shared L2 cache banks. The number of L1 caches should be equal to the number of L2 cache banks. The L2 cache bank id is derived by taking the least significant few bits of the L2 cache set index. For example, if a 2 MB 16-way L2 cache with 64-byte blocks has eight banks, the bank id is obtained from the three bits right after the block offset.

The modeled L2 cache is inclusive and each block's tag is extended to have a directory entry. Each L1 cache has two unbounded input queues. One queue accepts memory operations from a machine access trace. The other queue accepts messages from the L2 cache banks and other L1 caches. Both queues are unbounded to avoid deadlocks arising from shortage of queue space and therefore, can be implemented using a linked list. In fact, the queue that inputs memory operations from an access trace can be thought of as a trace file. Each L2 cache bank has one input queue accepting messages from L1 caches. Notice that neither the L1 caches nor the L2 cache banks have any outgoing queues. This is to simplify the simulator. Whenever an L1 cache needs to send a message to an L2 cache bank, it computes the bank id and directly enqueues the message at the tail of the input queue of the destination L2

cache bank. Similarly, if an L1 cache needs to send a message to another L1 cache, it directly enqueues the message at the tail of the input queue of the destination L1 cache. Also, if an L2 cache bank needs to send a message to an L1 cache, it directly enqueues the message at the tail of the input queue of the destination L1 cache. The simulator should correctly model the behavior of a directory-based cache coherence protocol. You do not have to model the main memory. An L2 cache miss would just run the L2 cache replacement algorithm and allocate the missing block. The replaced block can just be dropped after ensuring inclusion.

To model some notion of time, you will maintain a global variable that counts cycles. Each cycle, you will visit all the input queues of all the L1 caches and L2 cache banks and process the message at the head of each queue. Specifically, the actions in each cycle are divided into two parts. In the first part, you dequeue the head message from each non-empty queue and copy it in a temporary place. In the second part, you process these dequeued messages in some order. The exact order is irrelevant from correctness viewpoint because any order is correct. Then increment the cycle counter.

You will start with the solution for the second assignment and generate the thread-wise machine access trace files from the four programs you used in the second assignment. The additional information your trace will now have is the distinction between a load and a store operation. You will also have to model a modified state in the L1 cache blocks and L2 cache blocks. A dirty block replaced from an L1 cache must be written back to the L2 cache. Additionally, for a MESI cache coherence protocol, the L1 cache blocks need to model the E state. Note that the directory cannot distinguish between M and E states. Feed the trace input queue of an L1 cache from the corresponding thread's trace file. The simulation stops when all input queues in the system are empty.

You need to collect results for the following system specification.

```
Number of cores: 8
Cache coherence protocol: directory-based MESI.
L1 cache: 32 KB, 8-way, 64-byte block size, LRU
          S state blocks can be replaced silently from L1 caches.
          8 in number
L2 cache: 4 MB, 16-way, 64-byte block size, LRU, 8 banks (each bank is 512 KB), inclusive
```

Run each of the four programs (prog1, prog2, prog3, prog4) with eight threads using the PIN tool to generate the machine access traces. For each program report the following.

- Number of simulated cycles
- Number of L1 cache accesses and misses
- Number of L2 cache misses
- Prepare a table showing the names and counts of all messages received by the L1 caches.
- Prepare a table showing the names and counts of all messages received by the L2 cache banks.

Prepare a list explaining the message names used in the aforementioned tables.

Prepare a zip ball of your submission (PIN tool, other codes, and the report; no binary or trace please) and mail it to cs622autumn2020@gmail.com. The report should contain the required results backed by adequate explanations and comments (e.g., why certain message types are seen frequently in certain programs, etc.). I will grade only those submissions with PDF reports. Make sure to name the zip ball groupX.zip where X is your group number. Please use only the 'zip' utility for preparing your submission. Avoid using any other compression utilities due to possible portability issues.