ECEC-623

Project Two: Boosting Performance to Tensorflow Workloads with Advanced Memory Controller Design

Instructor: Dr. Anup Das
TA: Shihao Song (ss3695@drexel.edu)

1 Introduction

While the application design and optimization with Tensorflow are well studied, the lower level characteristics, especially the memory access pattern of Tensorflow workloads are less known. In this project, you will explore how advanced memory controller design affects the performance and fairness of Tensorflow workloads.

2 Required Readings

1. BLISS: Balancing Performance, Fairness and Complexity in Memory Access Scheduling
2. Parallelism-Aware Batch Scheduling: Enhancing both Performance and Fairness of Shared DRAM Systems

3 Workloads Reviews

We generated five multi-programmed Tensorflow workloads on an in-house cycle-level simulator. The simulator is configured to simulate an eight-core Skylake processor.

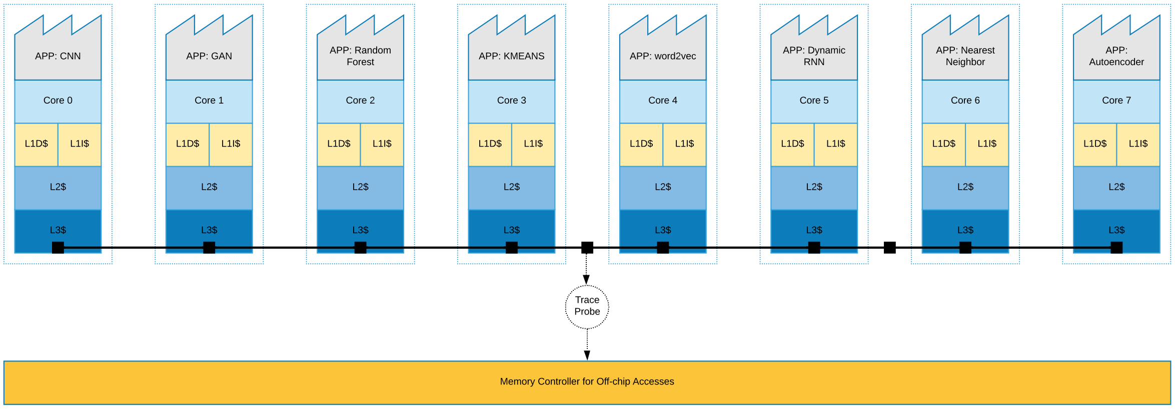

Figure 1: Sample Workload Generation

Figure 1 shows the process of workload generation. We map one Tensorflow application to one core; for example, Random Forest is assigned to core 2. The Trace Probe unit monitors all the off-chip accesses and generates one trace file. The following is the portion of one trace file:

```
0 125513360704 R
1 52028733376 R
2 67874772992 R
3 10500073216 R
4 56717512832 R
5 123034647616 R
6 11052054400 R
7 110490404800 R
0 125513360768 R
1 77506385920 R
```

Each entry is composed of three components:

- Core-ID/Application ID: indicates the core/application sends the memory request.
- Memory Address.
- Request Type.

You can find all the **compressed** trace files here:
`https://www.dropbox.com/sh/rzfggph1xi8abn0/AADbuRm2SSCcyDKdSNTG_gM8a?dl=0`
The name of each trace file indicates the simulated Tensorflow applications. We have also open-sourced the original Tensorflow apps for your reference:
`https://github.com/Shihao-Song/Tensorflow_Simulation`

4 Assignments

1. *git pull* and *cd* into the *Advanced_Memory_Controller* directory.
2. Read through the papers and answer: how fairness is calculated?
3. Implement a BLISS memory controller with the following modification: section 4.2, 1) Non-blacklisted application's requests; 2) Requests hit to a free bank (the same as your project one); 3) Older requests.
4. Evaluate your BLISS memory controller with all the workloads and answer: How the BLISS memory controller improves the fairness compared to the FR-FCFS you implemented in project one.
5. Report the variations of (4) when you set the number of banks to 16 and 8.
6. Report your experiments, zip it with the source codes, and submit through Bblearn.