A. Artifact Appendix

A.1 Abstract

EDGE is based on gem5-gpu `https://gem5-gpu.cs.wisc.edu/`. It doesn't have any specific sw and hw requirements, but we recommend using ubuntu14.04 as an OS as well as a cluster of X machines as a framework for running experiments. We provide detailed instructions of how to compile EDGE, as well as scripts for running the experiments and collecting the data. We mainly target figures 11 and 12, as they contain main results of our work. Also, figure 1 is heavily based on Figure 11 data, averaged across all background tasks and event kernels we used in our evaluation. In addition to the setup instructions we also provide a VirtualBox VM with all dependencies installed, all code compiled and ready for running the experiments.

A.2 Artifact check-list (meta-information)

- **Program:** Mixed, provided.
- **Compilation:** gcc4.7 + gcc4.4 + cuda3.1
- **Run-time environment:** Ubuntu 14.04 + gem5-gpu dependencies (provided)
- **Metrics:** Latency (min, max, average)
- **Output:** Provided scripts.
- **Experiments:** Provided scripts.
- **How much disk space required (approximately)?:** The main directory requires 5G. Extra memory is needed also for installing dependencies. The size of provided VM is 10G.
- **How much time is needed to prepare workflow (approximately)?:** Less than 1 hour.
- **How much time is needed to complete experiments (approximately)?:** Less than 1 day on a cluster. We never tried running ALL set of experiments on a single machine, but we estimate it as about a week.
- **Publicly available?:** Yes
- **Code licenses (if publicly available)?:** licenses of gpgpusim and gem5-gpu
- **Data licenses (if publicly available)?:** Copyright (c) 2003, Danga Interactive for memcached, Apache License for ip forwarding + Rodinia license
- **Archived (provide DOI)?:** `https://zenodo.org/badge/doi/10.5281/zenodo.3346076.svg`

A.3 Description

A.3.1 How delivered

We created a repository on github, as well as ready-to-use VirtualBox VM with everything installed and compiled. The username for VM is ”maria” and the password is ”asdfqwer”. VM files can be downloaded here: `https://www.dropbox.com/sh/qqp3jy1oy5eezcu/AADCImo6WhofeviGewhOOY3Ua?dl=0`.

A.3.2 Hardware dependencies

In general, there are no strict hw dependencies. But, since gem5-gpu is a heavy tool, every experiment may take up to an hour and our paper data required running more than hundreds of experiments. For that reason, we used a cluster of X machines to run the experiments. We provide scripts both for single machine and for cluster (using PBS job scheduler).

A.3.3 Software dependencies

Our artifact has been tested on Ubuntu 14.04. It doesn't require root access, but it has some dependencies that need to be installed (listed below). These dependencies are essentially a union of gem5 and gpgpusim dependencies. Also, gem5-gpu requires gcc4.7 for the simulator source code portion and gcc4.4 for the benchmark portion.

- Cuda 3.2
- gcc 4.4 for benchmark/edge directory
- gcc 4.7 for gem5 and benchmark/libcuda directories
- libnuma
- python 2.7
- SCons any recent version.
- zlib any recent version. Need the ”zlib-dev” or ”zlib1g-dev” package to get the zlib.h header file as well as the library itself.
- swig
- makedepend
- bison
- flex

A.3.4 Data sets

All used benchmarks are included. Some of them are taken from Rodinia, some prepared by us and some are taken from open source projects.

A.4 Installation

Install dependencies.

```
#Clone git repo
sudo apt-get install git
git clone https://github.com/marialubeznov/
    event_driven_gpu_execution.git
#Install gcc versions
sudo apt-get install build-essential
sudo cat "deb␣http://dk.archive.ubuntu.com/
    ubuntu/␣trusty␣main␣universe" >> /etc/apt/
    sources.list
sudo apt-get update
sudo apt-get install g++-4.4
sudo apt-get install g++-4.7
sudo update-alternatives --remove-all gcc
sudo update-alternatives --install /usr/bin/gcc
     gcc /usr/bin/gcc-4.4 10
sudo update-alternatives --install /usr/bin/gcc
     gcc /usr/bin/gcc-4.7 20
sudo update-alternatives --install /usr/bin/g++
     g++ /usr/bin/g++-4.4 10
sudo update-alternatives --install /usr/bin/g++
     g++ /usr/bin/g++-4.7 20
#Update gcc version
```

```
sudo update-alternatives --config g++
sudo update-alternatives --config g++
#Install dependencies:
sudo apt-get install python-dev
sudo apt-get install scons
sudo apt-get install zlib1g-dev
sudo apt-get install swig
sudo apt-get install xutils-dev
sudo apt-get install flex bison
sudo apt-get install libnuma-dev
#Install cuda 3.2
wget http://developer.download.nvidia.com/
    compute/cuda/3_2_prod/toolkit/cudatoolkit_3
    .2.16_linux_64_ubuntu10.04.run
wget http://developer.download.nvidia.com/
    compute/cuda/3_2_prod/sdk/gpucomputingsdk_3
    .2.16_linux.run
chmod +x cudatoolkit_3.2.16_linux_64_ubuntu10
    .04.run gpucomputingsdk_3.2.16_linux.run
    ./cudatoolkit_3.2.16_linux_64_ubuntu10.04.
    run
./gpucomputingsdk_3.2.16_linux.run
cd /NVIDIA_GPU_Computing_SDK/C/common
make
cd ../../../
cd event_driven_gpu_execution/benchmarks/edge/
ln -s ../common
```

Update `$LOCAL_GEM5_PATH/set_env` with relevant paths:

- `LOCAL_GEM5_PATH` path of git clone directory in local system.
- `ZLIB_PATH` path to directory that contains libz.so (if not default)
- `CUDAHOME, CUDA_HOME`
- `NVIDIA_CUDA_SDK, NVIDIA_CUDA_SDK_LOCATION`

Compile gem5-gpu

```
cd $LOCAL_GEM5_PATH
source set_env
cd benchmarks/libcuda
#set gcc version to 4.7
make
cd $LOCAL_GEM5_PATH/gem5
#set gcc version to 4.7
./build_command
cd $LOCAL_GEM5_PATH/benchmarks/edge/<name>/
#set gcc version to 4.4
make
```

A.5 Evaluation and expected result

Running experiments required for Figures 11 and 12 can be done using the following set of scripts (present in all 4 benchmark directories). Single machine:

```
run_no_overlap.sh
run_low_util.sh
run_high_util.sh
```

Cluster:

```
run_<benchmark_name>_no_overlap.pbs
run_<benchmark_name>_low_util.pbs
run_<benchmark_name>_high_util.pbs
run_cluster_no_overlap.sh
run_cluster_low_util.sh
run_cluster_high_util.sh
```

0	1	2	3
Convolution	Matrix multiply	Backprop (rodinia)	BFS (rodinia)

Table 1.

The results for Figure 11 won't be an exact match to ones reported in the paper because for each combination of event and background kernel, we launch an event at random times over three runs of the background kernel. But the trend should be similar to the one in the paper. Figure 12 results can be replicated.

After completion of all the experiments above, data can be collected using the scripts below:

```
#Figure 12
$LOCAL_GEM5_PATH/benchmarks/common/GetDataFig12
    .sh <background task> <type>  > <
    background_task>_<type>
bash $LOCAL_GEM5_PATH/benchmarks/common/
    process_data_fig12.sh
#Figure 11
$LOCAL_GEM5_PATH/benchmarks/common/GetDataFig11
    .sh <background task>  > <background task>
    _no_overlap
bash $LOCAL_GEM5_PATH/benchmarks/common/
    process_data_fig11.sh
```

`type = {low_util, high_util}` when these keywords represent low and high event kernel rates respectively. Background task is given as an index according to the mapping in Table 1.

Our main results are reported using a latency metric. Since many event kernels are being launched during every experiment, we report 4 types of latency: min, max and average. Number of launched event kernels is reported too.

A.6 Experiment customization

GPU configuration is set in:

```
$LOCAL_GEM5_PATH/gem5gpu/configs/gpu_config/
    gpgpusim.fermi.config.template
```

Experiment launch command format:

```
$LOCAL_GEM5_PATH/gem5/build/X86_VI_hammer_GPU/
    gem5.opt $LOCAL_GEM5_PATH/gem5-gpu/configs/
    se_fusion.py -c $LOCAL_GEM5_PATH/benchmarks
    /edge/<benchmark_name>/<benchmark_exe_name>
     -o '<test_options>'
```

A.7 Methodology

Submission, reviewing and badging methodology:

- `http://cTuning.org/ae/submission-20190109.html`
- `http://cTuning.org/ae/reviewing-20190109.html`
- `https://www.acm.org/publications/policies/artifact-review-ba`