Cuttlefish: Library for Achieving Energy Efficiency in Multicore Parallel Programs (Artifact)

1 Getting Started

Cuttlefish was evaluated on Intel Xeon Haswell E5-2650 20-core processor with a total of 94GB of RAM. The operating system (OS) was Ubuntu 16.04.7 LTS. We used OpenMP supported by the Clang compiler version 3.8.0 and used the same compiler for compiling HClib benchmarks. We used -O3 flag with the compiler.

1.1 Software and Hardware Dependencies

1. Disabling intel_pstate

   ```
   #go to  /etc/default/grub.d/50-cloudimg-settings.cfg
   #append intel_pstate=disable to GRUB_CMDLINE_LINUX_DEFAULT
   GRUB_CMDLINE_LINUX_DEFAULT="console=tty1 console=ttyS0 intel_pstate=disable"
   sudo update-grub
   sudo reboot
   ```

   Note: If you cannot find /etc/default/grub.d/50-cloudimg-settings.cfg on the system you can use the guidelines provided in the following link to disable intel_pstate,
   https://silvae86.github.io/2020/06/13/switching-to-acpi-power/

2. Likwid

   ```
   git clone https://github.com/RRZE-HPC/likwid.git
   cd likwid
   git checkout v5.0.1
   make
   sudo make install
   ```

3. Clang

   ```
   sudo apt-get install  clang
   sudo apt install  libomp-dev
   ```

4. Numactl

   ```
   sudo apt-get install  numactl
   ```

5. msr-tools

   ```
   sudo apt-get install  msr-tools
   ```

6. cpufrequtils

```
sudo apt-get install cpufrequtils
```

7. MPI

```
wget -c https://www.mpich.org/static/downloads/3.3/mpich-3.3.tar.gz
tar -xvf mpich-3.3.tar.gz
cd mpich-3.3
./configure
make -j 10
sudo make install
```

8. linux-headers

```
sudo apt-get install linux-headers-`uname -r`
```

9. HClib prerequisite

```
sudo apt-get install libtool
sudo apt-get install libxml2-dev
sudo apt-get install m4
sudo apt-get install autoconf
```

10. Python packages (for generating graphs)

```
pip install pandas
pip install scipy
pip install numpy
pip install matplotlib
```

1.2 Artifact Description

Go to the directory **cuttlefish/AD** and see the machine-generated environment data by using the bash script available at https://github.com/SC-Tech-Program/Author-Kit.

1.3 Artifact Evaluation

1. All the software required for evaluating the results presented in our paper can be installed by running setup.sh script. This script will build all the variants of Cuttlefish and the benchmarks.

```
cd cuttlefish/AE
./setup.sh
#Finished in around 10 minutes on our machine
```

2. Add the following in the bashrc file and source it:

```
#Append this to your ~/.bashrc:
export LC_ALL=en_US.UTF-8
export LANG=en_US.UTF-8
#And then run:
source ~/.bashrc
```

2 Instructions on how to reproduce the results

This section describes how to reproduce the results presented in Figures 2, 3, 10, and 11; and Tables 2 and 3 in the paper. We have provided the scripts to a) reproduce the actual graphs presented in the paper along with the original log files generated during our experimental analysis (Section 2.1), and b) reproduce the graphs by launching fresh experiments for each individual benchmark (Section 2.2).

2.1 Generating graphs from original logs

We have provided scripts to generate the results presented in our paper using the original log files. There is a single script that the user has to invoke for creating all the figures and tables as mentioned below.

1. Create all the graphs

   ```
   cd cuttlefish/AE/sc21/artifact/paper_results
   ./create_graphs.sh
   ```

2. All the graphs will be created in cuttlefish/AE/sc21/artifact/paper_results/output

2.2 Reproducibility analysis

Note that generating the results for all the ten benchmarks would take substantial time because the result reported for each benchmark is an average of ten invocations. Hence, to help the user generate the results only for a specific benchmark and for a lesser number of invocations, we have provided command-line arguments (optional) to experimental scripts that would take the benchmark id and number of invocations as input. *For quick verification, we would recommend launching single invocation execution for any benchmark.* The default is to launch the experiment for ten invocations.

Benchmarks	Benchmark id	Approx. time of single invocation for Figure-2 (minutes)	Approx. time of single invocation for Figure-3 (minutes)	Approx. time of single invocation for Figure-10 (minutes)	Approx. time of single invocation for Figure-11 (minutes)	Approx. time of single invocation for Table-3 (minutes)
UTS	0	1.2	7	5	N.A.	7
SOR-irt	4	1.2	7	5	5	7
SOR-rt	9	N.A.	N.A.	5	5	7
SOR-ws	8	N.A.	N.A.	5	5	7
Heat-irt	1	1.5	7	6	6	8
Heat-rt	7	N.A.	N.A.	6	6	8
Heat-ws	6	N.A.	N.A.	6	6	8
MiniFE	2	2.5	13	10	N.A.	19
HPCCG	3	1.5	8	6	N.A.	8
AMG	5	2	11	8	N.A.	16

Table 1. Approximate time for single invocation execution of each benchmark

Table-1 presents the `benchmark id` for each benchmark and the approximate time it took to execute a single invocation for the experiments reported in the paper. For a quick verification of results, we would recommend users run a single invocation for each benchmark. In the steps mentioned below, ignore the below-mentioned error (if any), as it implies the kernel module we are trying to insert already exists.

```
insmod: ERROR: could not insert module
```

2.3 Generating Figure 2

Here we will discuss how to generate Figure 2(a) and Figure 2(b) mentioned in the paper.

1. Launch the script experiment_timeline_haswell.sh

   ```
   cd cuttlefish/AE/scripts
   sudo su
   ./experiment_timeline_haswell.sh <benchmark_id>
   #where benchmark_id is the id for the benchmark mentioned in  Table 1
   ```

2. Create graphs of Figure 2

   ```
   cd cuttlefish/AE/sc21/artifact/reproducibility
   ./create_fig2.sh
   ```

3. Graphs will be created in cuttlefish/AE/sc21/artifact/reproducibility/output by the name figure2-a.png and figure2-b.png

2.4 Generating Figure 3

Here we will discuss how to generate Figure 3(a) and Figure 3(b) mentioned in our paper.

1. Run the script experiment_motivation_haswell.sh

   ```
   cd cuttlefish/AE/scripts
   sudo su
   ./experiment_motivation_haswell.sh <benchmark_id>
   #where benchmark_id is the id for the benchmark mentioned in  Table 1
   ```

2. Create graphs

   ```
   cd cuttlefish/AE/sc21/artifact/reproducibility
   ./create_fig3.sh
   ```

3. Graphs will be created in cuttlefish/AE/sc21/artifact/reproducibility/output by the name figure3-a.png and figure3-b.png

2.5 Generating Figure 10

Here we will discuss how to generate Figure 10(a), Figure 10(b) and Figure10(c) mentioned in our paper.

1. Run the script experiment_policy_haswell_OMP.sh

   ```
   cd cuttlefish/AE/scripts
   sudo su
   ./experiment_policy_haswell_OMP.sh <benchmark_id> <num_iters>
   #where benchmark_id is the id for the benchmark mentioned in Table 1
   #num_iters is the number of invocations of benchmark.
   ```

2. Create graphs

   ```
   cd cuttlefish/AE/sc21/artifact/reproducibility
   ./create_fig10.sh
   ```

3. Graphs will be created in cuttlefish/AE/sc21/artifact/reproducibility/output by the names figure10-a.png, figure10-b.png and figure10-c.png

2.6 Generating Figure 11

Here we will discuss how to generate Figure 11(a), Figure 11(b) and Figure 11(c) mentioned in our paper.

1. Run the script experiment_policy_haswell_HClib.sh

   ```
   cd cuttlefish/AE/scripts
   sudo su
   ./experiment_policy_haswell_HClib.sh <benchmark_id> <num_iter>
   #where benchmark_id is the id for the benchmark mentioned in Table 1
   #num_iters is the number of invocations of benchmark.
   ```

2. Create graphs

   ```
   cd cuttlefish/AE/sc21/artifact/reproducibility
   ./create_fig11.sh
   ```

3. Graphs will be created in cuttlefish/AE/sc21/artifact/reproducibility/output by the names figure11-a.png, figure11-b.png and figure11-c.png.

2.7 Generating Table-2

Here we will discuss how to generate Table 2 mentioned in our paper. The steps mentioned below in this section would work only if you have completed the steps mentioned in Section 2.5 above. Please note that the below steps will not generate the last column (Default CFopt/UFopt) as that detail was generated manually for the paper.

1. Create Table-2

   ```
   cd cuttlefish/AE/sc21/artifact/reproducibility
   ./create_Table_2.sh
   ```

2. Table-2 will be created in cuttlefish/AE/sc21/artifact/reproducibility/output by the name Table-2.txt.

2.8 Generating Table-3

Here we will discuss how to generate Table 3 mentioned in our paper.

1. Run the script experiment_policy_t_inv.sh

   ```
   cd cuttlefish/AE/scripts
   sudo su
   ./experiment_policy_t_inv.sh <benchmark_id> <num_iters>
   #where benchmark_id is the id for the benchmark mentioned in Table 1
   #num_iters is the number of invocations of benchmark.
   ```

2. Create Table 3

   ```
   cd cuttlefish/AE/sc21/artifact/reproducibility
   ./create_Table_3.sh
   ```

3. Table 3 will be created in cuttlefish/AE/sc21/artifact/reproducibility/output
   by the name Table-3.csv