

Cuttlefish-offline (Artifact)

1 Getting Started

Cuttlefish-offline 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. Python packages (for generating graphs)

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

1.2 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-offline and the benchmarks.

```
cd cuttlefishOffline
./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 9 and 10; and Tables 2 in the paper. a) We have provided the logs of the actual graphs presented in the paper.(Section 2.1), and b) reproduce the logs by launching fresh experiments for each individual benchmark (Section 2.2).

Application Name	Application ID
UTS	0
Heat-irt	1
MiniFE	2
HPCCG	3
SOR-irt	4
AMG	5
Heat-ws	6
Heat-rt	7
SOR-ws	8
SOR-rt	9

Table 1. Application ID corresponding to the applications used to evaluate Cuttlefish-offline

2.1 Instructions to see original logs

All logs are presented in the directory `cuttlefishOffline/artifact/paper_results/results`. Figure 9 and 10 were generated by using these logs.

2.2 Reproducibility analysis

Note that generating the results for all 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.

2.3 Creating Training dataset (Optional)

`cd /scripts` We have already kept the training data in the directory `cuttlefishOffline/artifact/reproducibility/Training_20`.

1. Before launching the experiment, you have to create regressor of each benchmark using training data. Run the following command to generate regressor

```
cd cuttlefishOffline/artifact/reproducibility
bash regressors_OMP.sh (It will create a directory named T_20)
```

2. To reproduce the results, go to the `cuttlefishOffline/script` directory and launch `experiment_policy_offline_OMP.sh`

```
cd cuttlefishOffline/scripts
sudo ./experiment_policy_offline_OMP.sh benchmark_id num_iterations
```

Now check the logs inside `cuttlefishOffline/artifact/reproducibility/results`