Introduction

Heterogeneous Architecture Configurations Generator for Multi2Sim simulator (HeteroArchGen4M2S)

`HeteroArchGen4M2S`: An automatic generator tool for configuring and running heterogeneous CPU-GPU architectures on Multi2Sim (M2S) simulator. This tool runs on top of M2S simulator, it allows us to configure the various heterogeneous CPU-GPU architectures (e.g., number of CPU cores, GPU cores, L1, $L2$, memory (size and latency (via `CACTI 6.5`)), network topologies (currently support 2D-Mesh, customized 2D-Mesh, and Torus networks)...). The output files include the results of network throughput and latency, caches/memory access time, and and dynamic power of the cores (can be collected after running `McPAT`).

`HeteroArchGen4M2S` is free software, which is freely to be redistributed and modified it under the terms of the GNU General Public License as published by the Free Software Foundation (For more details `http://www.gnu.org/licenses`).

`HeteroArchGen4M2S` is written to help you configure M2S easily, but non-warranty and non-mechantability.

A pdf version of this manual is also available in `HeteroArchGen4M2S.pdf`.

Setup Requirements

1. Currently HeteroArchGen4M2S has been tested on 64-bit platforms:
   - Ubuntu 14.04 (final)
2. Required tools to build and run with HeteroArchGen4M2S:
   - Python 2.7
3. Download and install `multi2sim-5.0` from:
   - `https://github.com/Multi2Sim/multi2sim`.
4. Download `McPAT` (current version-1.3) from:
   - `https://code.google.com/archive/p/mcpat/`.
   - Unzip it, and copy all files from mcpat folder to under multi2sim-5.0 directory and install it following the README file.
5. Download and install `CACTI6.5` from:
   - `http://www.hpl.hp.com/research/cacti/cacti65.tgz`.
   - CACTI 6.5 is used to obtain the cache and memory latency.

6. Required benchmarks to run:

   - Download benchmarks from `https://github.com/Multi2Sim`.
   - Then unzip the benchmarks' files under the installed multi2sim directory
   - Compile the benchmarks following the README file.

     Note: In case you want to run CUDA benchmarks, you can download other benchmarks for CPU-GPU systems such as Rodinia, Parboil, etc. Your desktop should have a NVIDIA graphic card (e.g., NVIDIA Quadro 4000), and you need to install the graphic card driver for running the simulation. (When compiling benchmarks, use `-m32` flag after `gcc` to make compatible with `multi2sim` 32-bit (no support 64-bit at this time)).

Download HeteroArchGen4M2S

```
git clone https://github.com/ttungl/HeteroArchGen4M2S.git
```

Build configuration files with HeteroArchGen4M2S

Let's assume you are in the home directory (`$multi2sim-5.0/HeteroArchGen4M2S`)

Where are the configuration files?

- Run `/multi2sim-5.0/HeteroArchGen4M2S$ python create_sim_configs_files.py`.
- The output files will be saved in the `configs` directory.
- `cd configs` >>> the `configs` folder contains four files, including `memconfig`, `netconfig`, `x86_cpuconfig`, and `si_gpuconfig`.

How to run the simulation?

- Previous steps show how to generate the configuration files. By running `create_sim_configs_files.py`, it also generated a shell script file inside `run_simulation_files` folder. The bash file (shell script) has been `chmod 777` for running.
- Go back under `multi2sim-5.0` directory.
- Run `./HeteroArchGen4M2S/run_simulation_files/run-bash-sim.sh`.
- This will create the output files which are the results of the simulation.

Where are the output files after simulation?

- `cd results` >>> Note that, `results` folder contains two files at this point, including `pipeline.out`, `mem.out`.
- With `net_report.out` file, it is generated under the `multi2sim-5.0` directory (outside of `HeteroArchGen4M2S` folder), you need to copy this file to `HeteroArchGen4M2S/results`.
- Now, there are three files should be in `results` folder, including `pipeline.out`, `mem.out`, and `net_report.out`(just copied).

Demonstration:

How to run multi2sim-5.0 with HeteroArchGen4M2S ?

Let's use the `blacksholes` example with 16 cores CPUs (`8` x86 CPUs), 16 cores GPUs (`4` Southern Islands GPUs), 4 Memory Controllers, in a 2D-Mesh for demonstration.

> Important: You need to download `parsec` benchmark from `https://github.com/Multi2Sim/m2s-bench-parsec-3.0`, then unzip it under the `benchmarks` folder in `multi2sim-5.0` directory for demonstration.

1. Suppose that you already got the cache and memory latencies for your proposed architecture by running `CACTI6.5`.
2. Suppose that you are under the `multi2sim-5.0\HeteroArchGen4M2S$` directory:
   - `sudo vim create_sim_configs_files.py` to configure your architecture. This file includes many parameters that need to be configured.
   - For CPU cores, a set of CPU includes two cores. Each core in the set can have its own L1$ (Data&Instr) or it can share the Instruction-L1$ with the other core in that set, by enabling `L1_Inst_shared` flag in the CPU Memory Parameters settings.
   - For GPU cores, a set of GPU includes four compute units. Each two compute units share with one L1, $eachtwoL1$ shares with one L2$.
   - For benchmarks, you need to modify the name of specific benchmark you want to run, and modify the command line of this benchmark and its path in `create_shell_script` file.

- For network topologies, HeteroArchGen4M2S currently supports three types of network, including `2D-Mesh`, `customized 2D-Mesh`, and `2D-Torus`. For `customized 2D-Mesh`, you need to specify the paths for local links and hybrid links in your network, as well as their linkwidths.

3. After modifying `create_sim_configs_files.py`:

   - Run `\HeteroArchGen4M2S$ python create_sim_configs_files.py` to generate the configuration files.
   - Checking the configuration files in the `configs` folder.
   - A shell script (`.sh`) has also been generated in the `run_simulation_files` folder. The shell script looks like as below.

     m2s –x86-sim detailed –x86-report HeteroArchGen4M2S/results/blackscholes_pipeline.out –mem-report HeteroArchGen4M2S/results/blackscholes_mem.out –x86-config ./HeteroArchGen4M2S/configs/x86_cpuconfig –si-sim detailed –si-config ./HeteroArchGen4M2S/configs/si_gpuconfig –mem-config ./HeteroArchGen4M2S/configs/memconfig –net-config ./HeteroArchGen4M2S/configs/netconfig –x86-max-inst 100000000 –net-report blackscholes_net_report.out benchmarks/m2s-bench-parsec-3.0/blackscholes/blackscholes 16 in_4K.txt prives.txt data-small

   - `cd ..` to `multi2sim-5.0` directory.
   - `./HeteroArchGen4M2S/run_simulation_files/run-sim-16-CPU-16-\ SouthernIslands-GPU-benchmark-blacksholes.sh`

4. To check the results, all of them are in the `results` folder, including:

   - `blacksholes_mem.out`
   - `blacksholes_pipeline.out`
   - With `net-l2-mm_blacksholes_net_report.out`, it is saved under the `multi2sim-5.0` directory, so just copy into the `results` folder.

5. To read the results:

   - Make sure you are in `HeteroArchGen4M2S` directory.
   - Run `python read_results.py`, the total cycles and network performance results are saved in `results` folder, under the names: `blacksholes_totalCycles.out` and `blacksholes_network_performance.out`.

     For `blacksholes_totalCycles.out`:

   - Cycles: 306794
   - Time: 3.94

For `blacksholes_network_performance.out`:

- Network Throughput: 32366.64
- Network Latency (cycles): 23.6969

6. To get the dynamic power from McPAT.

   - Make sure you are in `multi2sim-5.0` directory.
   - Run `m2s2xml4mcpat.sh` with the input files `hetero_mcpat.xml` file and `blacksholes_pipeline.out` to get the `mcpat.xml` file.
   - Run `./mcpat -infile mcpat.xml -opt_for_clk 1 -print_level 5 > HeteroArchGen4M2S/results/mcpat_result.txt`
   - `mcpat_result.txt` contains the results of McPAT.

Now you are ready to go. Happy hacking the code!

Claims:

I would like to thank the open source multi2sim community. This work is also inspired by M2StoMcPAT in Matlab, but I implemented completely in Python, with heterogeneous CPU-GPU architectures.

Tung Thanh Le
ttungl at gmail dot com
Version 1.0
Date: 02/18/17