# Interconnection Networks for High-Performance Systems

ECE 8823 A / CS 8803 – ICN

Spring 2019

## Lab 4: Impact of NoC on Real Application Traffic [35 points]

In this lab, you will study the impact of the NoC with real traffic. Inside gem5, you will run benchmarks from the LIGRA graph workload suite over a 16-core RISC-V system, vary the parameters of the Cache sub-system and NoC, and study their impact on overall runtime.

### Step 0:

Clone the following copy of gem5

```
hg clone /nethome/tkrishna3/teaching/simulators/gem5/repo/gem5_riscv
```

On a fresh login to the machines, don't forget to source the environment file:

```
source <path_to_gem5>/my_scripts/set_env.sh
```

Build the RISC-V ISA and the MESI Two Level protocol. **This is a directory-based coherence protocol with a Private L1 + Shared L2 slice per tile.**

```
python `which scons` -j 16 build/RISCV_MESI_Two_Level/gem5.fast
```

### Step 1:

### Example Run Command:

```
./build/RISCV_MESI_Two_Level/gem5.fast \
--outdir my_STATS/RISCV_64c_MESI_BFS_L1-1kB_L2-4kB_VC-4_tr-1 \
configs/example/se.py \
--cpu-type=TimingSimpleCPU \
--num-cpus=16 \
--num-l2caches=16 \
--l1d_size=1kB \
--l1i_size=1kB \
--l1d_assoc=2 \
--l1i_assoc=2 \
--l2_size=4kB \
--l2_assoc=8 \
--num-dirs=4 \
--mem-size=4096MB \
--ruby \
--network=garnet2.0 \
--topology=MeshDirCorners_XY \
--mesh-rows=4 \
--vcs-per-vnet=4 \
--router-latency=1 \
--maxinsts=100000 \
-c my_benchmarks/ligra/bin/riscv/BFS -o '-n 16 my_benchmarks/ligra/input/rMatGraph_J_5_100'
```

The parameters in blue are what you will be varying in this lab.

**What the command does:**

- Launches gem5 in "SE" or System Emulation mode. Here, any system calls from the application are "emulated" within gem5, without requiring an actual OS[1].

[1] *Note: it is also possible to run gem5 in a full-system mode by booting Linux on the CPU cores being modeled within gem5, and running an application on top of that.*
*See http://tusharkrishna.ece.gatech.edu/teaching/garnet_gt/ if you are interested.*

- Creates the specified system configuration (coherence protocol, cache sizes, network topology and configuration).
- Runs the RISC-V binary for the specific LIGRA benchmark for `--maxinsts` number of instructions (if specified) or till completion.

## Benchmarks:

You will run the following 4 benchmarks:
`BFS`
`PageRank`
`BellmanFord`
`Radii`

You can run them by just changing the name in `"-c"`

## Stats File:

`my_STATS/RISCV_64c_MESI_BFS_L1-1kB_L2-4kB_VC-4_tr-1/stats.txt`
**You can give any name to the output stats directory – just make sure it is unique for all your configurations and you are not overwriting other stats!**

Each of your sims will create its own folder in my_STATS, with the appropriate **`stats.txt`** file.
Here are some useful stats:
`sim_ticks` → This is the total application runtime (in cycles)
`system.ruby.network.*` → All network stats

Take a look at `system.ruby.network.packets_injected`
You will see something like:

```
|  697528   46.08%  46.08% |  27741  1.83% 47.91% |  788484   52.09%  100.00%
```

This is a histogram of packets injected per virtual network:
VNet 0 has 697528 packets which is 46.08% of total packets.
VNet 1 has 27741 packets which is 1.83% of total packets.
VNet 2 has 697528 packets, which is 52.09% of total packets

`system.ruby.network.packets_injected::total` is the total packets injected.

Similarly, take a look at `system.ruby.network.flits_injected` to see the corresponding *flits* per vnet. and `system.ruby.network.flits_injected` to see total flits.

***Note: You will see that some vnets have more flits than packets. In these vnets, there is a mix of 5-flit data response packets, and 1-flit control packets.***
***Some vnets have the same packet and flit counts as these are 1-flit control packets.***

`system.ruby.network.average_packet_network_latency` → Avg latency of the network packets

## What Configurations to Run:

- Run the 4 LIGRA benchmarks for 100K instructions.
- Vary L2 Size: 1kB, 16kB.
- Vary Num VCs (per vnet): 1, 8
- Vary Router Latency: 1, 5

## What to Plot:

**A.** Fix the L2 Size to 1kB. Vary VCs (per Vnet) as <1, 8> and Router Delay <5, 1>.

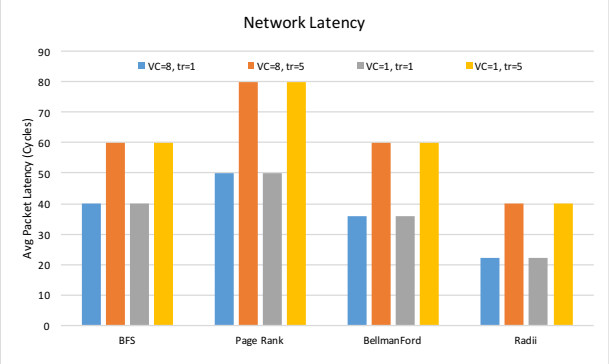
- **a.** Plot the *absolute network packet latency* (in cycles) for each benchmark as a bar graph. Add this to your Report **[4 points]**
- **b.** Plot the *normalized application runtime* for each benchmark as a bar graph. Each benchmark bar should be normalized to its runtime with <VC=8, tr=1> which is the best network configuration. Thus for all benchmarks, the first bar will be 1, and the remaining will be relative to it. Add this to your Report. **[4 points]**

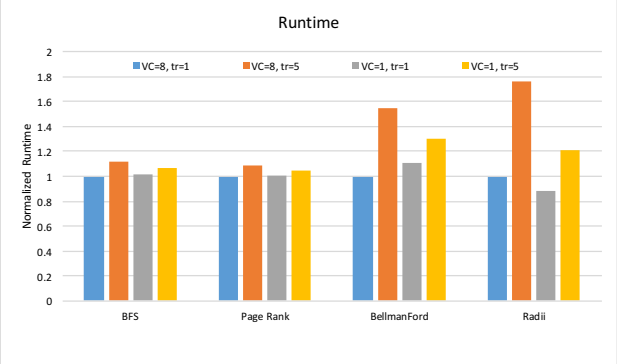
- **c.** Repeat Part A, but with L2 Size = 16kB. **[8 points]**

**<u>Note: If your axes and bars in the graphs are not labelled, you will not get any points.</u>**

**B.** Answer Lab4-Questions.docx **[19 points]**

The following are **<u>examples</u>** of the kind of graphs you should plot:




## What to Submit:

Report.doc/pdf
Lab4-Questions.pdf