Interconnection Networks for High-Performance Systems

ECE 6115 / CS 8803 – ICN

Spring 2020

Lab 1: Running Synthetic Traffic through a Network

The purpose of this lab is to install gem5, and run two synthetic traffic traces through its NoC simulator Garnet2.0.

Follow the instructions here to setup gem5:
http://tusharkrishna.ece.gatech.edu/teaching/garnet_gt/

Part I: Uniform Random Traffic

You will run **uniform random traffic** at increasing injection rates through a 8x8 Mesh NoC for 10000 cycles, and plot the ***latency-throughput*** curve.

Command to Run:

```
./build/Garnet_standalone/gem5.opt
configs/example/garnet_synth_traffic.py \
--network=garnet2.0 \
--num-cpus=64 \
--num-dirs=64 \
--topology=Mesh_XY \
--mesh-rows=8 \
--sim-cycles=10000 \
--inj-vnet=0 \
--injectionrate=0.02 \
--synthetic=uniform_random
```

The injection rate is in units of packets/node/cycle.

Command to Extract Network Stats:

```
./my_scripts/extract_network_stats.sh
```

This creates a **network_stats.txt** file, which has the following stats (among others):

```
packets_injected = 63945
packets_received = 63862
average_packets_latency = 16.019495
…
average_hops = 3.34
```

Note: The units for average_packet_latency is in cycles.
average_packet_latency is average_packet_queueing_latency + average_packet_network_latency.
The injected and received packets will be slightly off since the simulation stops at 10,000 cycles at which point some packets are still in the network.

Goal:
You need to increase the injection rate at intervals of 0.02, till it reaches 0.5
This will give you a total of 25 data points.
For each data point, add the ***average_packet_latency*** value in a file called **uniform_random.txt** one after the other.

For instance, uniform_random.txt might look like this at the end of 25 runs

```
5.67343
5.78787
5.88190
6.11213
…
…
…
700.4343
```

At the end, plot these values on a graph, with the x-axis representing injection rate going up in intervals of 0.02 up to 0.5, and the y axis representing the *average_packet_latency*.

You will notice that the latency values shoot up a lot after the network saturates, primarily due to the queueing delay. We will discuss this in class.

IMPORTANT: In the graph, cut the y-axis off at 50, otherwise the low latency values will not be visible at all.

Save the graph as **uniform_random_plot.pdf** or **uniform_random_plot.jpg** or **uniform_random_plot.png**

If you do not want to manually run this 25 times, I would recommend writing a script to run this command with changing injection rates, running extract_network_stats.sh, and extracting the latency values you need into another file.

Part II: Shuffle Traffic

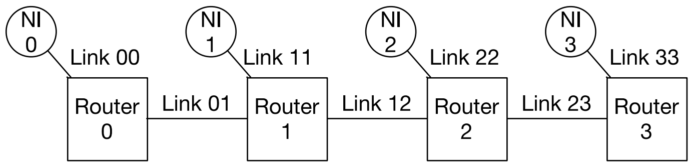
You will run **shuffle traffic** at increasing injection rates through a 8x8 Mesh NoC for 100000 cycles, and plot the latency-throughput curve.

Essentially repeat Part I, but change `--synthetic` **to** **`shuffle`**
Create **shuffle.txt** and **shuffle.pdf/jpg/png** as before.

Part III: Analysis

Create a file called **results.txt** with answers to the following questions:
Q1: Which of the two traffic patterns has a lower low-load latency?
Q2: Which of the two traffic patterns has a higher throughput?
Q3: Which of the two traffic patterns has a lower average hop count?
Q4: What is the **pipeline delay through every router** (in cycles)? The pipeline delay is the constant delay incurred by every router – irrespective of congestion. It is a positive integer.

Hint for Q4: *You can estimate this from the* ***average_hops*** *and* ***average_packet_network_latency*** *fields in the network_stats.*
The following figure shows how hops and network_latency are estimated in Garnet.
The delay of every link in garnet is 1-cycle.
Note: average_queueing_delay is the delay at the source NI before the packet is injected into the network.



Suppose we send a packet from NI 0 to NI 3:

Network Latency = Link00 + Router0 + Link01 + Router1 + Link12 + Router2 + Link23 + Router3 + Link33

Hops = 3 (Router 0 to Router 3)

In **results.txt**, just add 4 lines with the responses.
Do not add the question or the question number, or the grading script will not be able to parse your file.
The following is an example of a valid result file:

```
uniform_random
uniform_random
shuffle
2
```

What to Submit:

You need to submit 5 files in total:
uniform_random.txt
uniform_random_plot.pdf/jpg/png
shuffle.txt
shuffle_plot.pdf/jpg/png
results.txt