Interconnection Networks for High-Performance Systems

ECE 6115 / CS 8803 – ICN

Spring 2022

Lab 2: Topology Comparison – Questions (20 points)

This is an individual assignment.
Please do not post the solutions on Piazza.
You are also not allowed to discuss the solutions with each other.
Answer in-place.

Student Details:

Name: Varun Saxena
GT id: 903562211

Part A: Equal Link Widths [7 pts credit]

QA.1: Which topology provides the lowest low-load latency across the majority (at least two out of three) of traffic patterns? Can you comment why? [1 pt]

Answer: Torus topology has the lowest low-load latency in most cases because the average hop count of the Torus network is lower than the others.
Torus has an avg hop count of 2
Mesh has an avg hop count of 2.67
HeterogenousMesh has an avg hop count of 2.75

Plugging these values in the "ideal" latency equation:

$$T = (H+1).(t_{router} + t_{stall\_avg}) + (H+2).(t_{wire}) + T_{ser}$$

Shows that latency is lower is hop count is lower.

QA.2: Which topology provides the highest throughput across the majority (at least two out of three) of traffic patterns. Can you comment why? [1 pt]

Answer: Torus topology has the highest throughput in most cases because the bisection bandwidth of Torus for same link width is more than the other two, which leads to lower maximum channel load. Now, maximum throughput is inversely proportional to maximum channel load, so the lower the maximum channel load results in more maximum throughput, which is happening in the case of Torus topology.

QA.3: Consider *Neighbor* traffic, where node X always sends to node X+1 (e.g, 0 → 1, 1 → 2, 2→3 and so on). Do you think all topologies will provide identical low-load latency? If yes, why? If no, why not? [1 pt]

Answer:
Neighbor traffic as per GarnetSyntheticTraffic.cc is that the message stays in the same row as well. So, for a current topology sizes, 3->0 will be the last direction in the same row.
No, all topologies will not provide identical low-load latency. This can be illustrated by a simple hop-count calculation for each topology (as link width is same):

1. Mesh_XY:
    a. 0->1 => 1 hop
    b. 1->2 => 1 hop
    c. 2->3 => 1 hop
    d. 3->0 => 3 hops
2. Torus
    a. 0->1 => 1 hop
    b. 1->2 => 1 hop
    c. 2->3 => 1 hop
    d. 3->0 => 1 hop
3. HeterogeneousMesh: This calculation varies for row #1 and row #4 vs row #2 and row #3
    a. Row #1 and Row #4:
        i. 0->1 => 1 hop
        ii. 1->2 => 3 hops
        iii. 2->3 => 1 hop
        iv. 3->0 => 5 hops
    b. Row #2 and Row #3: Calculations are similar to that of the mesh

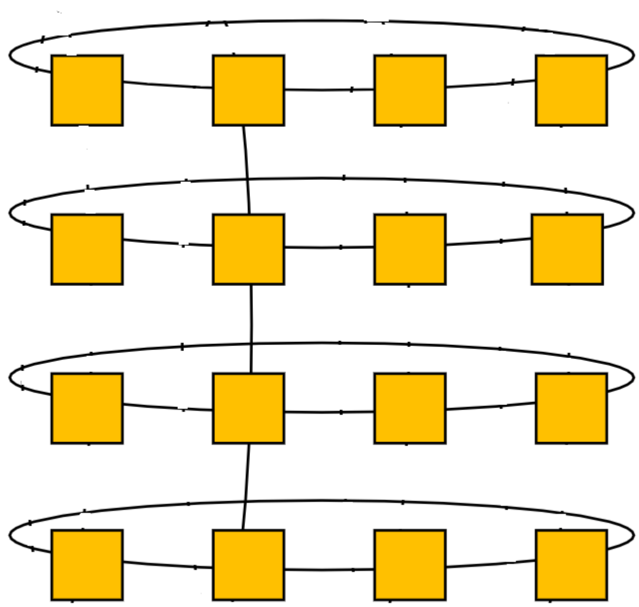
This calculation gives us an idea of the average hop count and shows that the low-load latency Torus > Mesh_XY > HeterogenousMesh

QA.4: Can you suggest some changes to the Torus you implemented that will provide the same latency and throughput for *Tornado* traffic, but reduce the total number of links? <u>*Note: you need to come up with a topology with minimum possible links that can provide the same latency and throughput for Tornado traffic as Torus but can still functionally support the other patterns -- Uniform Random, Shuffle and Bit Complement.*</u>
Let us call this TorusLite. Draw this topology. How many links does this updated topology have compared to the regular Torus? [4 pts]

The Tornado traffic as seen in GarnetSyntheticTraffic.cc has the communication pattern of:

For the bottom row: 0 <-> 2, 1 <-> 3. Similar communication pattern exists in the other rows with no communication between rows.
So, to maintain the same latency and throughput, we mainly need to ensure that each row has the same structure.
Now, to support the other traffic patterns “functionally”, we need atleast one link between the rows. To keep the number of links minimum we only add 1 link between each.

Number of links in TorusLite = 19



Part B: Equal Bisection Bandwidth. [3 pts]

QB.1: Which topology provides the lowest load-load latency across the majority (at least two out of three) of traffic patterns? Can you comment on why? [1.5 pts]

Answer: Mesh_XY has the lowest low-load latency in most cases because:

1. With 16b link width of Torus as compared to 32b of Mesh_XY, all the links in the Torus can only carry a lower load than Mesh_XY meaning it would require greater latency to send data from source to destination for data > 16b
2. The average hop count for HeterogeneousMesh is greater than Mesh_XY which can be clearly seen by the increased hops required to communicate between the extreme points in the same rows/columns like 0 <-> 3, 0 <-> 12, etc.

QB.2: Which topology provides the highest throughput across the majority (at least two out of three) of traffic patterns. Can you comment on why? [1.5 pts]

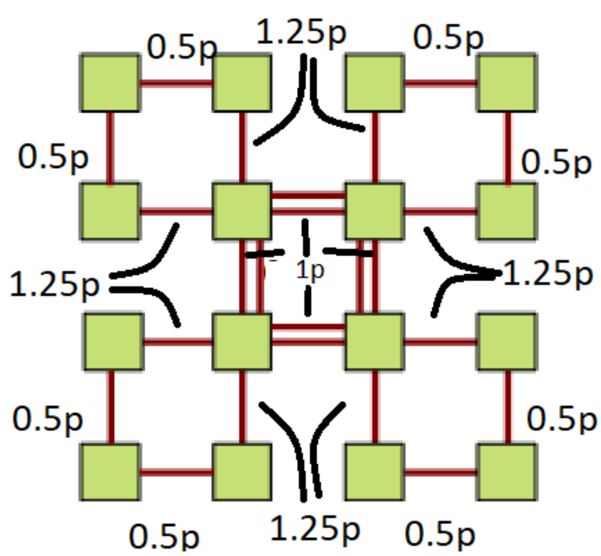
Answer: Mesh_XY has the highest throughput in most cases because:

1. With 16b link width of Torus, the data that can be transferred per cycle gets reduced compared to Mesh_XY
2. For the HeterogeneousMesh, the links in the middle becomes a bottleneck as the 4 outer squares always need to use the 4 middle links to communicate to the other 3 squares which can lead to congestion and reduced throughput.

Part C: Theoretical Runtime Metrics

C.1. Fill out the following table for Uniform Random Traffic. (10 points)
Assumptions

- **Ideal Routing (equal probability of using all shortest paths)**
- **Equal Link widths.**

Topology	**Average Hop Count (3 points)**	**Mark the link(s) with max channel load + write down the channel load on the link(s). (4 points)**	**Theoretically Max Throughput (3 points)**
HeterogeneousMesh	2.75	 Detailed description of channel loads: Say router order is (12,13,14, 15) (8, 9, 10, 11)	0.8 flits/node/cycle

		(4, 5, 6, 7) (0, 1, 2, 3) Links: (0,1), (0,4), (2,3), (3,7), (8,12), (12,13), (14,15), (11,15) have load 8p/16 Links: (1,5), (4,5), (2,6), (6,7), (8,9), (9,13), (10,14), (10,11) have load 20p/16 Links: (5,6), (5,6), (5,9), (5,9), (9,10), (9,10), (6,10), (6,10) have load 16p/16	

Show your calculations (need to show calculations to get any points):

Calculations for average hop count:

1. Messages sent from 0 to others:
   a. Total hops in the same loop: 1 + 1 + 2 = 4
   b. Total hops in the adjacent loop: 3 + 4 + 4 + 5 = 16
   c. Total hops in the diagonally opposite loop: 4 + 5 + 5+ 6 = 20
   d. Total hops = 4 + 2*16 + 20 = 56
2. Messages sent from 1 or 4 to others:
   a. Total hops in the same loop: 4
   b. Total hops in the adjacent loop: 2 + 3 + 3 + 4 = 12
   c. Total hops in the diagonally opposite loop: 3 + 4 + 4 + 5 = 16
   d. Total hops = 4 + 2*12 + 16 = 44
3. Messages sent from 5 to others:
   a. Total hops in the same loop: 4
   b. Total hops in the adjacent loop: 1 + 2 + 2 + 3 = 8
   c. Total hops in the diagonally opposite loop: 2 + 3 + 3 + 4 = 12
   d. Total hops = 4 + 2*8 + 12 = 32

Total hops in the network = (56*4 + 44*2*4 + 32*4)/(16*16) = (224 + 352 + 128)/(16*16) = 2.75

Calculation of maximum channel load: (targeting only the channels which are different in their traffic pattern):
Say, message is p and one node sends p/16 to other nodes.

1. Node 0 (link 0 <-> 1)
   a. Incoming messages:
      i. From nodes in other loops which can then use either of 0 <-> 1 or 0 <-> 4 links: (12p/16) *(1/2) = 6p/16
      ii. From adjacent node: p/16
      iii. From diagonally opposite node in the same loop which has 2 link options: (p/16)*(1/2) = 0.5p/16
      iv. Communication between 1 and 4 uses this link: (p/16)*(1/2) = 0.5p/16
      v. Total = **8p/16**
   b. Outgoing messages:

        i. Total outgoing message from node 0: (15p/16). It can use either of links connected to it => effective link load for outgoing message => (15p/16)*(1/2) = 7.5p/16
        ii. Communication between 1 and 4 uses this link: (p/16)*(1/2) = 0.5p/16
        iii. Total **= 8p/16**

2. Node 1 (link 1<-> 5)
    a. Incoming messages:
        i. To Node 1 from other loops: 12p/16
        ii. To Node 1 from Node 5: p/16
        iii. To Node 1 from diagonally opposite Node 4: (p/16)*(1/2) = 0.5p/16
        iv. To Node 0 from other loops through current link: (12p/16)*(1/2) = 6p/16
        v. To Node 0 from Node 5 through current link: (p/16)*(1/2) = 0.5p/16
        vi. Total = **20p/16**
    b. Outgoing messages:
        i. To nodes in other loops: 12p/16
        ii. To node 5: p/16
        iii. To node 4: (p/16)*(1/2) = 0.5p/16
        iv. From Node 0 to other loops: (12p/16)*(1/2) = 6p/16
        v. From Node 0 to Node 5 through current link: (p/16)*(1/2) = 0.5p/16
        vi. Total = **20p/16**
3. Node 5 (link 5 <-> 6)
    a. Incoming messages:
        i. To node 5 from adjacent loop containing 6: 4p/16
        ii. To node 5 from diagonally opposite loop containing 10: (4p/16)*(1/2) = 2p/16
        iii. To node 9 from diagonally opposite loop containing 6: (4p/16)*(1/2) = 2p/16
            1. This will also use the same link
        iv. The signal probability to other 3 nodes in the same loop will be the same and will use the same link but given there are 2 links, the load is divided in 2: Total = 4 * (1/2) * (4p/16 + 2p/16 + 2p/16) = **16p/16**
    b. Outgoing messages: Due to symmetry the same calculations can be done to yield the same result of **16p/16**

Maximum throughput = 1/(maximum channel load) = 1/(20p/16) = 16/20 = 4/5 = 0.8 (taking p = 1)