Topology-Custom UGAL Routing on Dragonfly - Rahman19

Hassan Mango, Rosie Vuong

INTRODUCTION

- **Universal Globally Adaptive Load-balance routing (UGAL)**: modern routing scheme for Dragonfly.
- In order to achieve high performance, different topologies must use different routing schemes (minimal routing (MIN), valiant load-balanced routing (VLB)).
- **Problem**: conventional does not use different routing schemes for different topologies. Method to choose MIN and VLB are the same in all topologies.
- **Propose**: topology-custom UGAL routing (T-UGAL), having same routing mechanism, same set of MIN paths, but different sets of VLB customized for each topology.
- **Result**: T-UGAL >> UGAL in latency and throughput for different topologies.

DRAGONFLY TOPOLOGY

- 2 layer structure.
- p: number of links per switch connecting to

local compute nodes

- a: number of switches in each group
- h: number of global links per switch connecting

to switches in other groups

- g: number of groups
- Intra-group topology: fully connected graph

where all switches are connected to each other.

- Load-balanced dragonfly system:

  a=2p=2h

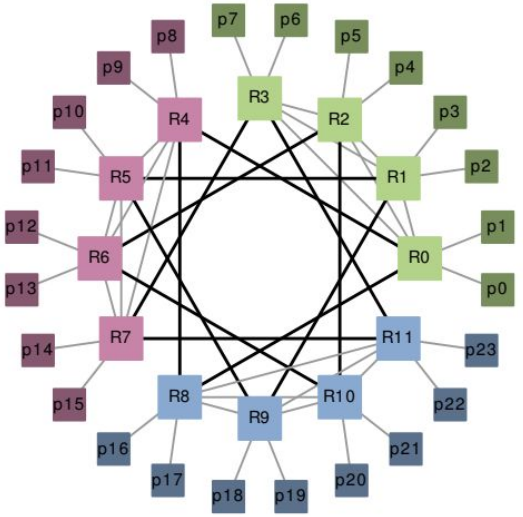


Figure 2: An example Dragonfly topology with absolute global link arrangement ($dfly(p = 2, a = 4, h = 2, g = 3)$

UGAL

- Packets: routed from source compute node to destination compute node.
- Source switch: switch that source compute node connects.
- Destination switch: switch that destination compute node connects.
- Source group: group that source compute node is in.
- Destination group: group that destination compute node is in.
- s ➜ d: represent link from node s to node d.

UGAL, MIN path

- Minimal path (MIN): path from source compute node to destination compute node that has at most 1 global link.
- MIN pros: shortest path, less resource usage, works well in evenly distributed traffic patterns.
- MIN cons: in patterns that many nodes in 1 group must connect to many nodes in other group ➔ poor performance bc it must use small links between 2 groups (adversarial).

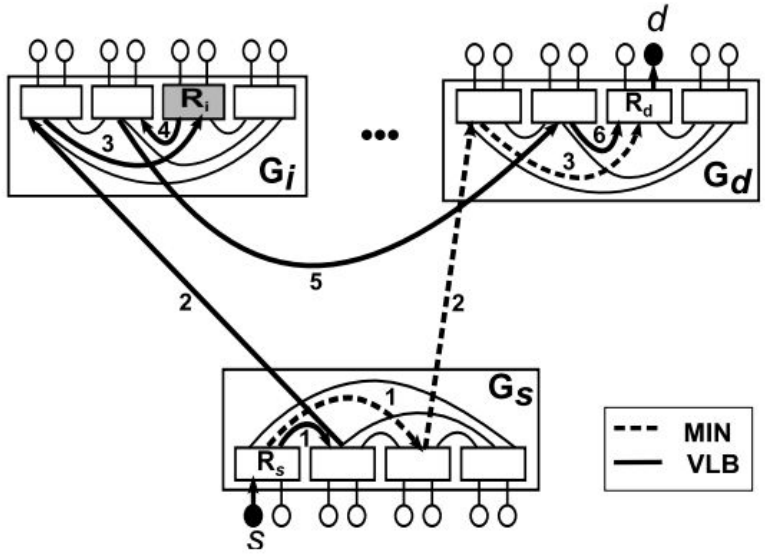


Figure 3: MIN and VLB routing on Dragonfly

UGAL, VLB path

- Avoid cons of MIN: use VLB to spread non-uniform traffic evenly over set of available links.
- UGAL selects MIN and VLB paths based on traffic condition.

$$TQ_{MIN} \leq TQ_{VLB} + T$$

- Traffic condition: inferred from occupancy of packet queues of the network.

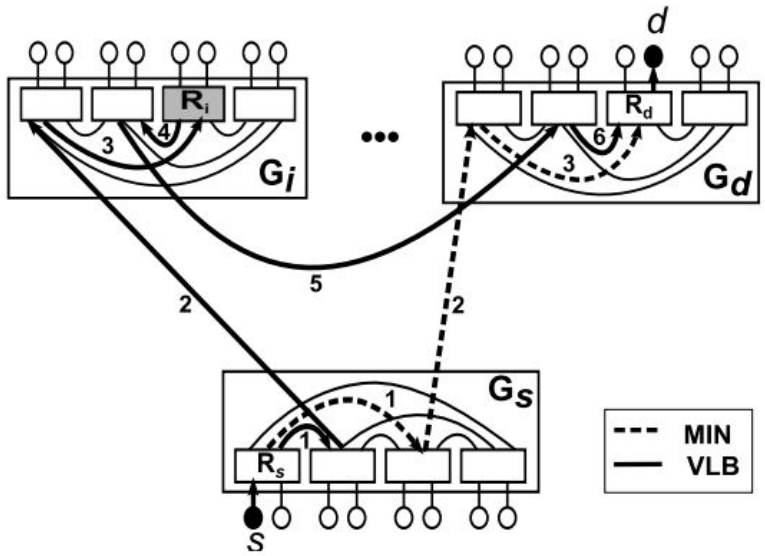


Figure 3: MIN and VLB routing on Dragonfly

MOTIVATION

- Good performance routing scheme has 2 properties:
  - Minimal amount of network resources to deliver each packet (use MIN).
  - Routing should be able to distribute traffic evenly over the network (use VLB).
- UGAL has 2 properties by using both MIN for short and VLB for path diversity and load balancing.
- Question: method to choose candidates for VLB and MIN is not customized in different topologies ➜ whether all VLB paths are necessary for path diversity on all dragonfly topologies?
- Objective: find set of shorter VLB paths with sufficient path diversity to improve performance.

T-UGAL Properties

Our scheme selects paths in T-VLB based on the topology in such a way that T-UGAL has the following properties:

(1) **T-UGAL achieves higher or similar performance in comparison to UGAL for the most demanding adversarial traffic patterns.** The most demanding adversarial traffic patterns are the ones that require most path diversity for performance. The idea is that if T-UGAL can have higher or similar performance for such patterns, it should be able to achieve higher or similar performance for any other traffic patterns, which are less demanding.

(2) **The average path length of T-VLB is as small as possible.** The advantage of T-UGAL over the conventional UGAL is using shorter paths for communications, which results in low packet latency as well as less average network resources to deliver a packet that can yield higher throughput at high load.

(3) **T-UGAL has a similar load-balancing property as UGAL.** The load balancing property is essential for any routing scheme to achieve high performance.

Computing T-VLB - Dragonfly patterns

The main idea of **T-UGAL** is to find **T-VLB** with the smallest average path length while being able to provide sufficient path diversity even for the most demanding traffic patterns for the network.

2 shift patterns that help us decipher the path. (TYPE_1_SET and TYPE_2_SET)

The TYPE_1_SET contains the traffic patterns where each group shifts to any other group. Additionally, nodes from each switch also shift to every other switches. This set contains (g−1)a patterns.

The second type of patterns can be specified by first having a random permutation at the group level, and then having a random permutation at the switch level for each source and destination group pair in the group permutation.

Computing T-VLB - Coarse Grain Estimation of T-VLB

- After figuring out the most demanding traffic patterns, the next step is to find the subset VLB paths for all pairs of source-destination switches.

- Using a coarse grain estimation, a performance model was created to evaluate many potential data points (different subsets of VLBs) and find a small number of candidate configurations.

- In the second step, after some fine-tuning is performed, the final T-VLB is decided through simulation (if there is a working system, T-VLB for the system can be decided by experimentation).

Note:

Although the model is accurate for UGAL with all VLB paths, the accuracy drops when a small percentage of 5-hop or 6-hop paths are used.

To overcome this problem, the data rate allocated for a longer VLB path for a source-destination pair is no more than the data rate allocated for a shorter VLB path for the same source-destination pair.

This is because UGAL has an inherent tendency to prefer shorter paths over longer ones when such paths are available.

Table 1 lists the data points (or configurations) that are probed in the Step 1 coarse grain estimation.

For any **dfly**(*p, a,h,g*), the number of hops for VLB paths is between **2 and 6**. Each data point is applied to all source destination switches in a synchronized manner.

For example, the point "**4-hop paths**" means that all pairs of switches will use all VLB paths that are 4 hops or less in length.

Notation	explanation
3-hop paths	all paths 3-hop or less
10% 4-hop paths	all paths 3-hop or less plus 10% 4-hop paths
20% 4-hop paths	all paths 3-hop or less plus 20% 4-hop paths
...	...
90% 4-hop paths	all paths 3-hop or less plus 90% 4-hop paths
4-hop paths	all paths 4-hop or less
10% 5-hop paths	all paths 4-hop or less plus 10% 5-hop paths
...	...
90% 5-hop paths	all paths 4-hop or less plus 90% 5-hop paths
5-hop paths	all paths 5-hop or less
10% 6-hop paths	all paths 5-hop or less plus 10% 6-hop paths
...	...
90% 6-hop paths	all paths 6-hop or less plus 90% 6-hop paths
6-hop paths	all VLB paths

Table 1: The data points probed in coarse-grain Step 1

Deciding T-VLB: Step 2 – Finalizing T-VLB

After obtaining candidate configurations from step 1, we first examine the candidate sets to possibly depthen the collection of paths by including some deterministic strategic choices.

For example, 50% 5-hop paths can strategically be obtained either by having all 2-hop MIN paths followed by 3-hop MIN paths or by having all 3-hop MIN paths followed by 2-hop MIN paths.

Calculating Imbalance

T-VLB uses a subset of VLB paths and can potentially result in an imbalanced use of links. The imbalance may happen in two levels: **locally** for each pair of switches when some links are significantly more likely to be used to carry the traffic of this pair of switches than other links, and **globally** for all pairs of switches when some links are significantly more likely to be used to carry traffic than others.

In theory, imbalance can also be removed by replacing paths that use highly loaded links by paths that do not use highly loaded links.

E.g. Global imbalance is calculated by computing the probability of link usage of all links under the assumption that a packet between any pair of switches is equally likely. When such imbalances are detected, we perform simple load balance adjustments by removing paths that cause high link usage probability either at the local level (per pair of switches) or at the global level (all pairs of switches).

Methodology

Putting it all together

The procedure takes a dragonfly topology **dfly(p,a,h,g)** as input and outputs T-VLB.

The procedure first computes the adversarial patterns TYPE_1_SET and TYPE_2_SET (Line 3).

Then lines 4 to 7, that sort the VLB paths based on the path length and randomize the order of VLB paths of the same length.

Then lines 8 to 12 are the coarse-grain estimation of the number of VLB paths needed. Lines 13 to 21 is finalizing T-VLB.

Finally, Line 22 outputs the results.

1 **Input**: A Dragonfly topology, $dfly(p, a, h, g)$
2 **Output**: The sets of T-VLB paths for all pairs of switches
3 Generate *TYPE_1_SET* and *TYPE_2_SET* for the topology
4 **for** *each pair of switches* **do**
5 | Compute all VLB paths and randomize the order
6 | Sort all VLB paths based on the path length
7 **end**
8 /* Step 1: coarse-grain */
9 Compute the modeled throughput for all patterns and all VLB subsets described in Table 1
10 Find the point with the largest average modeled throughput
11 Decide candidate set in the vicinity of the largest average throughput
12 /* end of Step 1 and begin of Step 2 */
13 Expand the candidate set if necessary
14 **for** *each candidate configuration* **do**
15 | Compute per pair link usage probability
16 | Do local load balance adjustment if necessary
17 | Compute all-to-all link usage probability
18 | Do global load balance adjustment if necessary
19 | Simulate a set of traffic patterns and record the results
20 **end**
21 The data point with the highest average simulated throughput is selected
22 Output the associated VLB paths, which is T-VLB

Algorithm 1: Algorithm to determine T-VLB for any $dfly(p, a, h, g)$

Topology

Resulting in these: **dfly(p = 4, a = 8,h = 4, g = 33), dfly(p = 4, a = 8, h = 4, g = 17),** and **dfly(p = 4, a = 8, h = 4, g = 9).**

These topologies are built with 15-port switches and represent a range of Dragonfly topologies with different connectivity characteristics like, having the same intra-group connectivity, but different numbers of groups as well as different numbers of links connecting each pair of groups. A

larger topology was also reported to demonstrate that T-UGAL also works for larger topologies. **dfly(p = 13, a = 26,h = 13, g = 27)**.

Table 2 lists the major parameters of topologies.

Topology	No. of PEs	No. of switches	No. of groups	links per group pair
$dfly(4, 8, 4, 33)$	1056	264	33	1
$dfly(4, 8, 4, 17)$	544	135	17	2
$dfly(4, 8, 4, 9)$	288	72	9	4
$dfly(13, 26, 13, 27)$	9126	702	27	13

Table 2: Topologies used in the experiments

Routing Variations and Simulator Settings

The study also makes use of Booksim 2.0, a cycle-accurate interconnection network simulator. The Dragonfly topology code that Booksim comes with always creates a network where g = a * h + 1.

To study the performance of other UGAL variations, we added PAR and UGAL-G and incorporated T-UGAL with the three variations: UGAL-L, UGAL-G and PAR.

Noting that UGAL-L and PAR are practical and can be deployed while UGAL-G is an ideal-case scheme.

Using BookSim in the Study

Booksim provides a feature to increase the speed of the router's internal pipeline. With a speedup of 2, the router pipeline runs at twice the speed of the network channels.

Booksim uses credit-based flow control for buffer management among adjacent routers. Credits are sent back to the opposite direction when a packet reaches its destination. In order to accommodate this round-trip delay, we set each virtual channel buffer size to 32.

Parameter	value
# of virtual channels	4 for UGAL-L and UGAL-G 5 for PAR
buffer size	32
link latency	10 cycles (local) 15 cycles (global)
switch speed-up	2

Table 3: Default network parameters in the simulations

In Booksim, injection rate (offered load) is specified as packets per cycle (per node). So an injection rate (offered load) of 0.1 means a node can generate 1 packet on average in 10 cycles.

Throughput is also measured in unit of packets, per cycle per node. For each synthetic traffic pattern, we simulate with a sufficient number of injection rates to infer the latency curve.

The study makes use of the notation TMIXED(% of uniform random traffic, and % of adversarial traffic) to represent such a traffic pattern. In TMIXED(UR%, ADV%), each packet from every node has an UR% probability to have a uniform random destination and an ADV% probability to have an adversarial destination.

Figure 4: dfly(4, 8, 4, 9) with different configurations

Figure 4 shows the average modeled throughput for **dfly(4, 8, 4, 9)**.

The error bar in the figure is the standard error of the mean.

The best throughput of 0.58 for this topology is achieved at 60% 5-hop: all VLB paths that are 4-hop or less and 60% of 5-hop VLB paths. The throughput of 0.58 means that each node can communicate at 58% of its link speed when the network saturates.

With 4 global links between each pair of groups, sufficient path diversity is provided by short VLB paths; and not all VLB paths are needed to achieve the best performance for the most demanding adversarial traffic patterns.

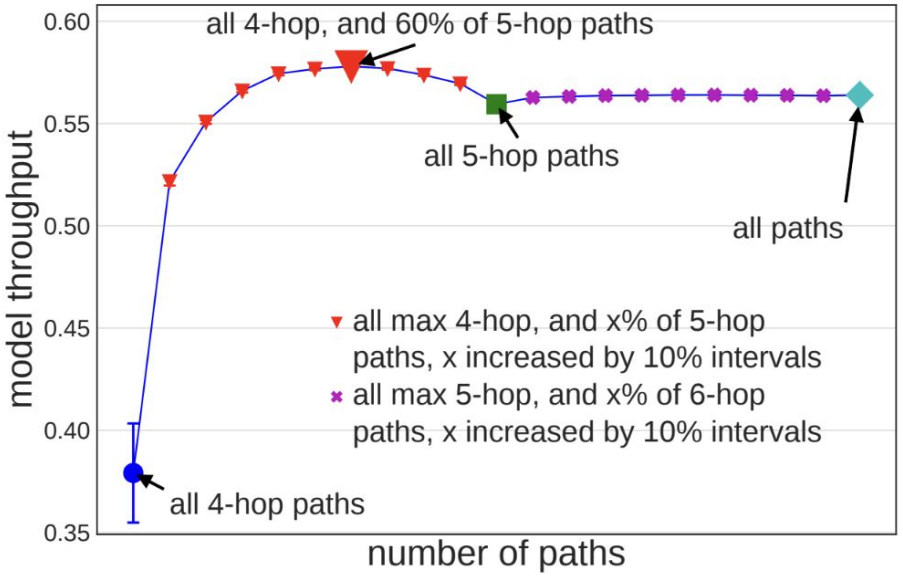


Figure 4: Average modeled throughput in Step 1 calculation for $dfly(4, 8, 4, 9)$

Figure 5: Avg. Modeled Throughput - dfly(4,8,4,33)

- The best performance for this topology is achieved when all VLB paths are used.
- All VLB paths are necessary to achieve high performance for the adversarial patterns.
- Maximum-sized Dragonfly topologies were used, like **dfly**(4, 8, 4, 33) with 1 link per pair of groups have been used.

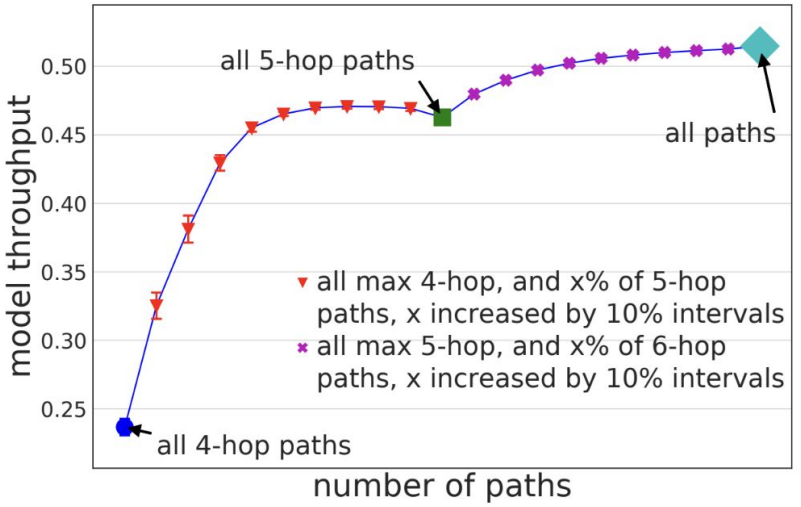


Figure 5: Average modeled throughput in Step 1 calculation for $dfly(4, 8, 4, 33)$

Figure 6:

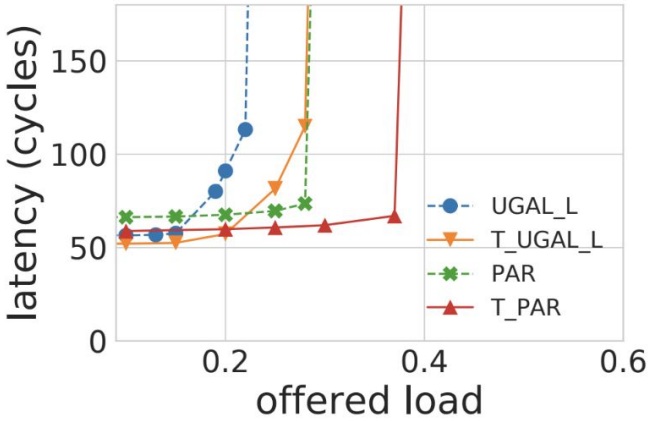


Figure 6: Latency for the adversarial shift(2, 0) pattern for UGAL-L and PAR on $dfly(4, 8, 4, 9)$

Figure 6 shows the latency vs. offered load for UGAL-L, T-UGAL-L, PAR, and T-PAR on d f ly(4, 8, 4, 9).

T-UGAL-L improves over UGAL-L in latency when the network is not saturated and has a much higher saturation throughput.

The results for PAR: when the offered load is 0.2, the average packet latency for T-PAR is 12.9% lower than the 67.6 cycles average packet latency for PAR. The saturation throughput of T-PAR is 31.0% higher than the 0.29 saturation throughput of PAR.

Figure 7:

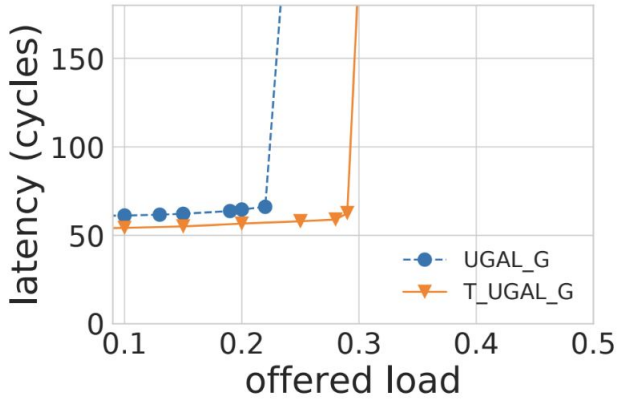


Figure 7: Latency for the adversarial shift(2, 0) pattern for UGAL-G on $dfly(4, 8, 4, 9)$

Figure 7 shows results for UGAL-G. T-UGAL-G improves the latency when the network is not saturated: At 0.1 offered load, the average latency for T-UGAL-G is 12.9% lower than the 61.2 average latency for UGAL-G.

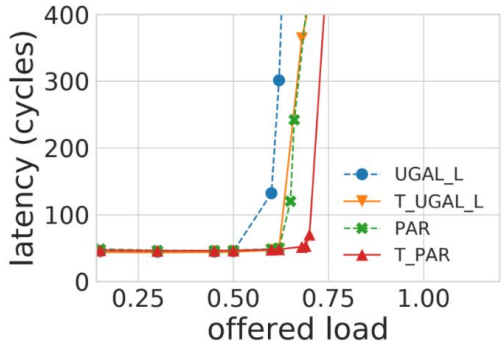


Figure 8: Latency for a random permutation pattern for UGAL-L and PAR on $dfly(4, 8, 4, 9)$

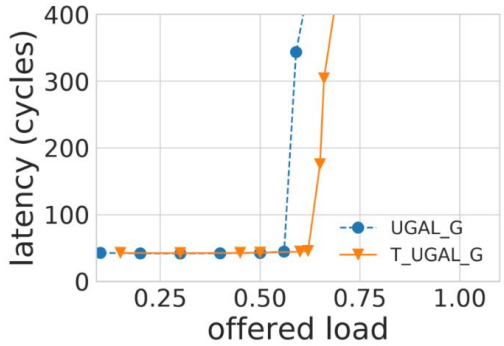


Figure 9: Latency for a random permutation pattern for UGAL-G on $dfly(4, 8, 4, 9)$

Figure 8 shows the latency vs. offered load in UGAL-L, T-UGAL-L, PAR, and T-PAR on **dfly(4, 8, 4, 9)** for a random permutation pattern.

Figure 9 shows results for UGAL-G. In this case, T-UGAL-G has similar average packet latency when the network is under low load. However, the saturation throughput for T-UGAL-G, 0.66, is 11.9% higher that the 0.59 saturation throughput for UGAL-G. This is due to the use of shorter paths that reduces the overall network load and improves the saturation throughput.

Figure 10 and Figure 11 show the results for MIXED(75, 25) and MIXED(25, 75) on **dfly(4, 8, 4, 17)**. T-UGAL only optimizes VLB paths.

As such, its advantage can only be observed when more traffic are routed using VLB paths.

As the traffic becomes more adversarial (MIXED(25,75)), the saturation throughput decreases for all schemes, but the advantage of T-UGAL-L and T-PAR becomes larger.

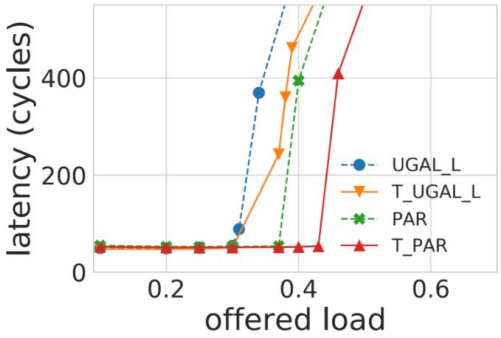


Figure 10: Mixed traffic: MIXED(75, 25) with UGAL-L and PAR on $dfly(4, 8, 4, 17)$

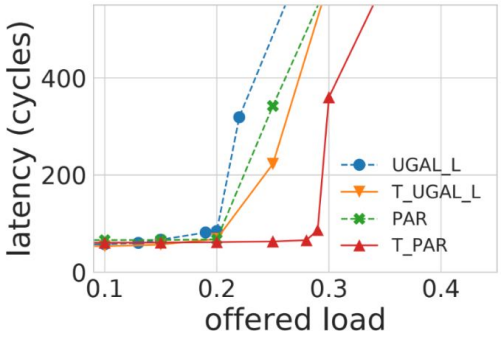
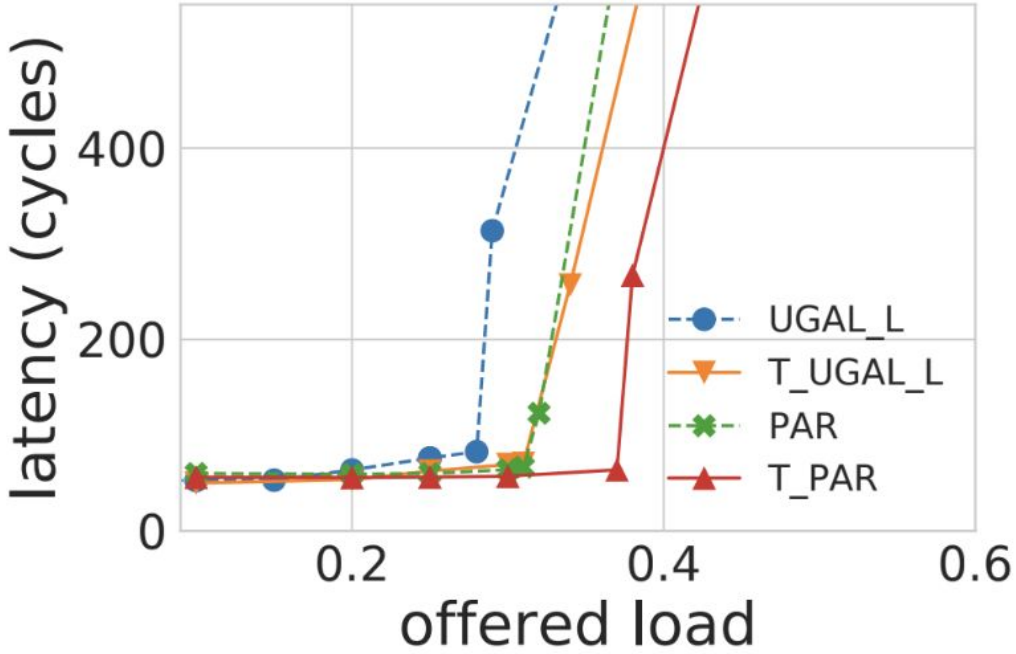


Figure 11: Mixed traffic: MIXED(25, 75) with UGAL-L and PAR on $dfly(4, 8, 4, 17)$

Figure 12: Time based mixed traffic - TMIXED(50,50) with UGAL-L and PAR on dfly(4,8,4,17)



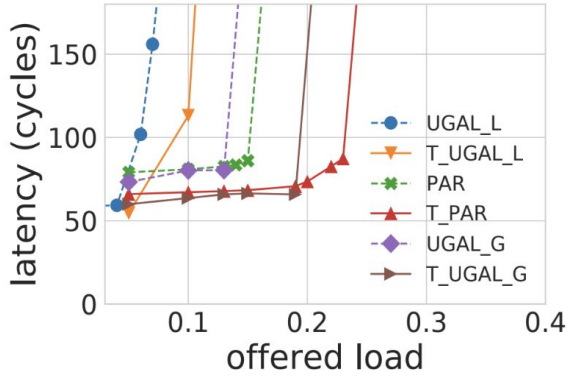


Figure 13: Adversarial traffic (shift(1, 0) pattern) for UGAL-L, PAR, and UGAL-G on $dfly(13, 26, 13, 27)$

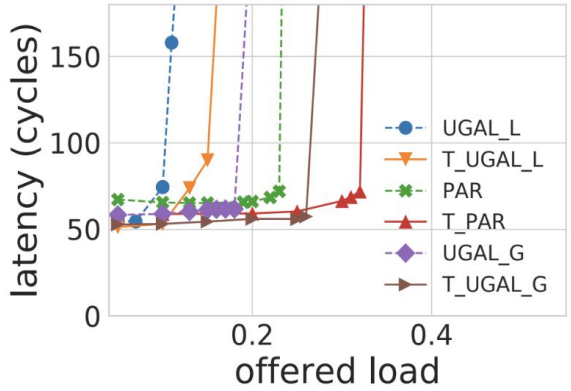


Figure 14: Mixed traffic: MIXED(50, 50) for UGAL-L, PAR, and UGAL-G on $dfly(13, 26, 13, 27)$

Figure 13 shows the latency as the offered load increases for UGAL-L, T-UGAL-L, PAR, T-PAR, UGAL-G, and T-UGAL-G, on a larger Dragonfly topology d f ly(13, 26, 13, 27) for an adversarial traffic pattern (shift(1, 0) pattern). While the specific numbers differ, the trend is very similar to that for the smaller topologies **dfly(4, 8, 4, 9)** and **dfly(4, 8, 4, 17).**

Figure 14 shows the results for a mixed traffic (MIXED(50, 50)). Again, T-UGAL variations have clear advantage over their corresponding UGAL variations: T-UGAL offers advantages over the corresponding UGAL for Dragonfly topologies of different sizes and shapes.

Figures 15, 16, 17, and 18 show the sensitivity of UGAL and T-UGAL to different network parameters.

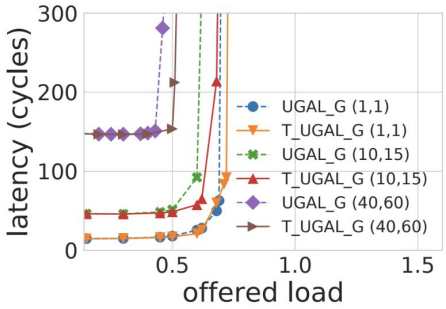


Figure 15: Effect of varying link latency on UGAL-G on $dfly(4, 8, 4, 17)$ for random permutation pattern

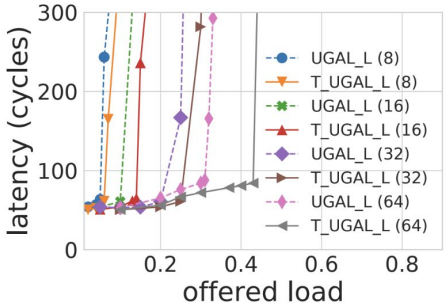


Figure 16: Effect of varying buffer length on UGAL-L on $dfly(4, 8, 4, 17)$ for MIXED(50,50) pattern

Figure 15 shows the sensitivity to the link latency. For example, UGAL_G(40, 60) denotes UGAL_G with local link latency of 40 cycles and global link latency of 60 cycles.

Figure 16 shows the sensitivity to the buffer length. For example, UGAL_L(8) denotes UGAL_L with buffer size of 8 flits.

Figure 17 shows the sensitivity to the switch speedup. The legend format for this figure is routing(speedup). PAR(1) denotes PAR with switch speedup of 1.

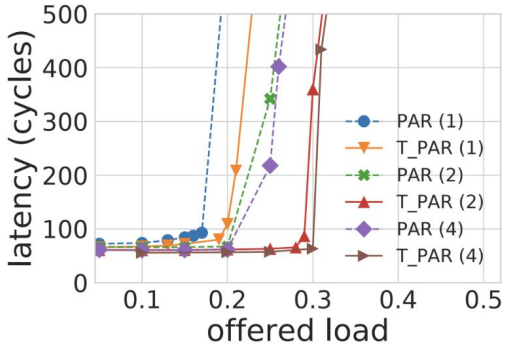


Figure 17: Effect of varying router internal speedup on PAR on $dfly(4, 8, 4, 17)$ for MIXED(25,75) pattern

Figure 18 shows the sensitivity to the virtual channel allocation scheme through displaying the effect of different virtual channel allocation schemes on UGAL-G on **dfly(4,8,4,9) with the (1,0) adversarial shift pattern**.

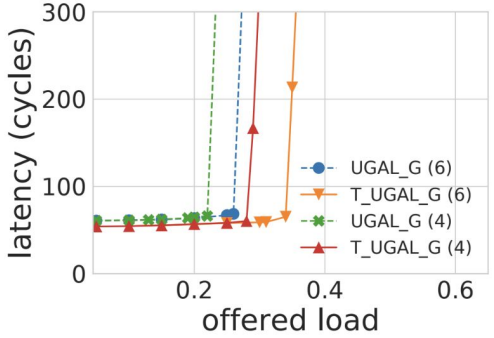


Figure 18: Effect of different virtual channel allocation schemes on UGAL-G on $dfly(4, 8, 4, 9)$ for the adversarial shift(1,0) pattern

T-UGAL has a clear advantage when the adversarial traffic components are present in the network.

By using a subset of shorter VLB paths computed based on the network topology, T-UGAL reduces the packet latency when the network is not saturated while improving the saturation throughput.

Therefore, we can conclude that by using a subset of VLB paths with shorter average path length (than the average path length of all VLB paths), T-UGAL improves over the conventional UGAL routing very significantly on many topologies, especially the ones with a small number of groups and a large number of links between each pair of groups which is common in many practical systems.

Thank you !

Any questions?