Traffic Pattern-based Adaptive Routing for Intra-group Communication in Dragonfly Networks

Peyman Faizian, Md Shafayat Rahman, Md Atiqul Mollah, Xin Yuan
Department of Computer Science
Florida State University
Tallahassee, FL
{faizian,rahman,mollah,xyuan}@cs.fsu.edu

Scott Pakin, Mike Lang
Computer, Computational, and Stat. Sci. Div.
Los Alamos National Laboratory
Los Alamos, NM
{pakin,mlang}@lanl.gov

***Abstract*—The Cray Cascade architecture uses Dragonfly as its interconnect topology and employs a globally adaptive routing scheme called UGAL. UGAL directs traffic based on link loads but may make inappropriate adaptive routing decisions in various situations, which degrades its performance. In this work, we propose to improve UGAL by incorporating a traffic pattern-based adaptation mechanism for intra-group communication in Dragonfly. The idea is to explicitly use the link usage statistics that are collected in performance counters to infer the traffic pattern, and to take the inferred traffic pattern plus link loads into consideration when making adaptive routing decisions. Our performance evaluation results on a diverse set of traffic conditions indicate that by incorporating the traffic pattern-based adaptation mechanism, our scheme is more effective in making adaptive routing decisions and achieves lower latency under low load and higher throughput under high load than the existing UGAL in many situations.**

I. Introduction

The Dragonfly topology features a cost-effective interconnect design and provides high aggregate bandwidth for a diverse set of traffic patterns [1]. It has been deployed in the Cray Cascade architecture and is the interconnect topology in a number of current and near-term supercomputers.

One unique characteristic of the Dragonfly network is that the routing performance is very sensitive to traffic pattern. To achieve high performance, different routing schemes must be used for different traffic patterns [1]. For example, minimal routing (MIN) should be used for uniform traffic while Valiant Load Balance routing (VLB) should be used for other traffic patterns. To unify the two routing algorithms in one system, Universal Globally Adaptive Load-balanced routing (UGAL) [1] adapts the routing decision between MIN and VLB paths based on the link load information derived from queue length. The theoretical UGAL with perfect global link state information (UGAL-G), which cannot be implemented in practice, performs similarly as MIN for uniform traffic and as VLB for adversarial traffic. Various schemes that approximate the theoretical UGAL-G have been developed [2].

An adaptive routing scheme that makes routing decisions based on link loads fundamentally optimizes for network load balancing by distributing the traffic such that link loads on different links are similar. However, load balancing alone is insufficient for achieving high performance in Dragonfly. VLB can achieve load balancing for almost any traffic pattern, yet its performance for random uniform traffic is significantly lower than MIN. An adaptive routing scheme such as UGAL that makes routing decisions based solely on link/path loads has inherent limitations as the traffic pattern is not directly considered. Recent studies have shown that UGAL makes inappropriate routing decisions in various situations, which degrades its performance and that the problem is more severe with imprecise global link state information [3], [4].

In this work, we propose to enhance UGAL by explicitly incorporating a traffic pattern-based adaptation mechanism. This work focuses on intra-group communication; but the techniques can be extended for inter-group communication. The proposed scheme is motivated by several observations. First, Dragonfly is developed for HPC systems; and HPC applications often have repetitive communication patterns that can easily be identified at different levels [5], [6]. Second, modern routers for HPC systems maintain an extensive number of performance counters. The statistics collected in these counters can be used to infer traffic pattern. Finally, UGAL works well under many traffic conditions, but not all [3], [4]. Incorporating the traffic-pattern information in making routing decisions can alleviate the issues with UGAL for the traffic conditions in which it does not perform well.

Our adaptive routing scheme maintains counters for local traffic in each router, which gives additional information about the current traffic pattern. Based on the inferred local and global traffic condition from the counters, our routing scheme identifies nine different situations in which different biases toward MIN and VLB paths are appropriate. In this way, the proposed scheme adapts traffic based on the current traffic pattern in addition to link loads. We performed extensive simulation with a diverse set of traffic conditions. The simulation results indicate that (1) under low load, the proposed scheme performs better than the theoretical UGAL with perfect global link state information (UGAL-G) in terms of packet latency; and (2) under high load, the proposed scheme performs similar to UGAL-G and better than a practical UGAL implementation in terms of aggregate bandwidth. Thus, the proposed scheme is a robust and highly effective adaptive routing scheme.

The rest of the paper is structured as follows. Section II describes the Dragonfly variation adopted in the Cray Cascade architecture and its routing schemes. Section III presents our

proposed routing algorithm. Section IV reports the results of the performance study. The related work is presented in Section V. Finally, Section VI concludes the paper.

II. Background

A. Cray Cascade Dragonfly topology

The Cray Cascade architecture employs Dragonfly as its interconnect topology. The architecture has a fixed structure for each group, but allows variable numbers of groups to form a system. Each group is fully connected with all others using optical cables. The intra-group topology is well-defined, but the total number of groups and the inter-group bandwidth (number of links between groups) are installation-specific.

As this paper is concerned only with intra-group communication in the Cascade architecture, we will describe the Cascade intra-group topology. Details about the Dragonfly and Cascade architectures are provided elsewhere [1], [7]. Every group in Cascade is formed by a pair of cabinets. Each cabinet houses three chassis. Each chassis contains 16 blades. Each blade connects a single router and four processing elements. So in total, each group contains 6 chassis, each have 16 blades with a total of 96 routers.

The Cascade system uses Aries routers, which are an application specific integrated circuit (ASIC) developed by Cray. Each chassis backplane provides all-to-all connections among sixteen Aries routers. Each router is also connected to five other routers in the remaining five chassis within the same group using electrical cables. The inter-chassis connections are made with corresponding slots. For example, a router in slot 1 in one chassis will have direct links to the five slot 1 routers in the other five chassis within the same group. Each inter-chassis link is equivalent to three intra-chassis links in terms of bandwidth. An Aries router has a total of 48 ports: 8 ports for local processing nodes, 15 ports connecting to 15 routers in the same chassis, 15 ports to 5 routers in the same slot but different chassis, and 10 ports to other groups. For intra-group communication, each Aries router can have 18 traffic end-points (8 for the local processing nodes and 10 for the ports connecting to other groups). Figure 1 shows the intra-group topology of a Cascade group. Logically, a Cascade group consists of a 6×16 mesh with fully connected x and y dimensions. Each pair in the same row is connected by one link while each pair in the same column is connected by three links.

B. Cascade routing

In Cascade, packets are routed along either a minimal or a non-minimal path. Within a group, the minimal routing (MIN) is the shortest path routing: a minimal path either has 1 hop when the source and the destination are in the same column or row, or 2 hops otherwise. In Figure 1, the minimal path from router S to router D has two hops. The non-minimal routing is the Valiant's Load Balance routing (VLB), which spreads non-uniform traffic evenly over the set of available links in the system. Non-minimal routes (or VLB routes) in a group can be considered as using MIN to find a path from

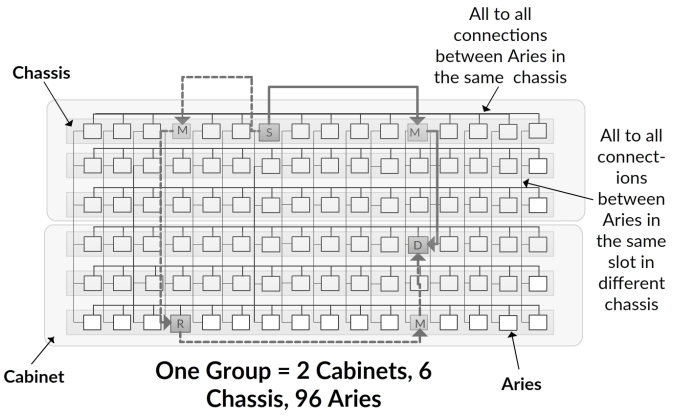

Fig. 1. Cray Cascade intra-group topology

the source to a randomly selected intermediate router and then from the intermediate router to the destination. For example, in Figure 1, a VLB path from S to D can go through intermediate router R, resulting in a 4-hop path.

Cascade supports packet-by-packet adaptive routing with the Universal Globally Adaptive Load-balanced (UGAL) routing scheme. The routing pipeline selects up to four possible routes at random, two minimal and two non-minimal routes and decides the path for the packet based on estimated link loads which are computed using a combination of downstream link load, estimated far-end, and near-end link load. UGAL selects the path with the lightest load. Some details about Cascade UGAL are undisclosed, such as how frequently the downstream load information is propagated and how exactly the loads are estimated and computed. In this work, we consider two versions of UGAL: UGAL with perfect global information (UGAL-G) and UGAL with local information (UGAL-L) [1]. UGAL-G is not a practical routing scheme since it assumes that the precise load on each path in the entire network is known, but it represents the performance upper bound that any UGAL-type routing scheme can achieve. UGAL-L uses the queue length on the local router to approximate the load on the path, and can be implemented in practice.

UGAL-L and UGAL-G choose between MIN and VAL routes for each packet, based on the path load information available at the source router. UGAL-G selects the path with the smallest accumulated queue length on all links along the path. UGAL-L considers the queue length (Q) in the local router and the hop counts (H) of the paths and evaluates the following inequality:

$$Q_{min} \times H_{min} \leq Q_{vlb} \times H_{vlb} + T \quad (1)$$

in which Q_{min} is the queue length on the local channel for the MIN path; H_{min} is the hop count for the MIN path; Q_{vlb} is the queue length of the non-minimal path; H_{vlb} is the hop count of the non-minimal path; and T is an offset constant that can be tuned to decide how much the path selection should be biased toward MIN paths (with a large value of T giving preference to MIN paths). If the inequality holds, UGAL decides that the

minimal path is less congested than the non-minimal path and should be selected to route the packet.

These link load based adaptive routing mechanisms cannot reach the observed performance of MIN under uniform random traffic and VLB under worst-case adversarial traffic because they try to adapt to the network congestion only based on available link load information. As revealed in recent studies [3], [4], these adaptive routing schemes suffer from the following limitations.

- Fluctuations in queue lengths due to temporary load-imbalance can result in suboptimal routing decisions made by the adaptive routing scheme, even when the overall traffic pattern remains unchanged.
- Longer queue lengths can occur for multiple reasons, including high load or the adversarial nature of the traffic. But these routing schemes treat all these situations alike by choosing the least loaded path, which is not always an optimal choice.
- The performance of UGAL (UGAL-G or UGAL-L) depends on the congestion offset (the T in Equation 1), which needs to be tuned empirically based on the traffic pattern. Fixing its value represents a trade-off: the routing will favor either minimal paths, which would degrade the routing performance for adversarial traffic, or non-minimal paths, which would result in low performance for uniform traffic. There does not exist a single value for T that can achieve high performance under all traffic conditions.

III. Traffic Pattern-based Adaptive Routing

Our scheme is built upon the idea that assuming a UGAL-like routing scheme that selects between MIN and VLB paths, each router can observe the traffic that passes through the router and infer useful traffic pattern information to make better routing decisions. With a UGAL-like routing scheme, even when a router is not on the minimal path of a communication, it can still observe the communication, as the router can be selected as an intermediate router when a non-minimal path is used. Note that the proposed adaptive routing algorithm is much like UGAL except that it uses traffic-pattern information to make better routing decisions than conventional UGAL.

To obtain the traffic-pattern information, our scheme maintains a number of counters and infers the local and global pattern information from the counters. The Cray Aries provides an extensive set of performance counters so maintaining the counters needed by our scheme is not an issue on that platform. By explicitly inferring and using the traffic pattern information, custom adaptive routing mechanisms can be tailored for the observed traffic patterns. Our approach improves UGAL as described in Section II-B by using local and global traffic-pattern information to differentiate nine cases for setting the offset value (T) in UGAL or to use MIN or VLB where needed. In other words, the adaptive routing mechanisms used in our scheme range from pure MIN routing to UGAL-L with different biases to MIN paths to pure VLB routing. Each routing mechanism is selected based on inferred-traffic pattern information. In the following text, we first introduce the counters used in our routing scheme. We then discuss how the traffic pattern is inferred. Finally, we present our traffic pattern-based adaptive-routing algorithm.

A. Traffic counters

In each router, our scheme maintains the following counters, which count the number of packets during a window of time (e.g., the last 50 cycles):

- $DestC_i$: These counters record the number of packets sent to the router i from local compute nodes during the window period.
- $Port\_thr_i$: For each port i in the router, $Port\_thr_i$ records the number of through packets (packets originated from other routers) that use that port.

Using these counters, useful local and global traffic-pattern characteristics can be derived. To support intra-group communication in Cascade, each router must have 96 $DestC_i$ counters and 30 $Port\_thr_i$, which is not significant relative to the number of performance counters provided by the Aries router. If the technique is to be extended to inter-group routing, some aggregation scheme must be used to reduce hardware overhead. One natural option is to maintain a counter for each destination group instead of for each destination router.

B. Quantifying local traffic pattern

Local traffic consists of packets generated from the endpoints attached to the router. From earlier studies of Dragonfly, it is known that the traffic is benign when it spreads evenly among all possible destination routers (uniform traffic), and adversarial when the traffic concentrates on a small number of destination routers. To make effective routing decisions, it is important to know how adversarial or how uniform the local traffic pattern is. Such information can be obtained with the $DestC_i$ counter. The basic idea is that if the local traffic is uniform, the values in $DestC_i$ are similar and relatively small. If the local traffic is adversarial, the values in $DestC_i$ will exhibit a small number of spikes (implying traffic is concentrated on a small number of destination routers) whose values can be very large, depending on the injection rate. In our scheme, we quantify local traffic to each destination switch as benign or adversarial by examining $DestC_i$.

We first consider an experiment with the Booksim simulation infrastructure [8]. In this experiment we simulate Cascade's 96 intra-group routers with each router connecting to 18 traffic endpoints. The routing algorithm is UGAL-L. The observation window size is 50 cycles. The traffic pattern is either uniform random (UR) or an adversarial shift pattern (ADV) in which all traffic from endpoints on router i is sent to endpoints on router $i+1$. We observe the counter values ($DestC_1$) in router 0. The results are shown in Table I. The injection rate is the number of packets generated per traffic endpoint per cycle. As can be seen from the table, for uniform traffic, the number of packets sent to each destination is much smaller than the number of packets sent to the same destination under adversarial traffic and same injection rate. Each Cascade

group has 96 routers. With uniform traffic, each router should receive 1/95 of the total local traffic. On the other hand, for the worst case adversarial traffic, all of the local traffic are sent to one destination router. As such, differentiating these two types of traffic to each destination can be done by examining $\frac{DestC_i}{h}$, where h is the window size. We call this value *localimpact*.

TABLE I
THE COUNTER VALUES FOR UNIFORM AND ADVERSARIAL TRAFFIC ($h = 50$)

Injection rate	Pattern	$DestC_1$	*localimpact*
0.1	UR	1	0.02
	ADV	90	1.80
0.44	UR	4	0.08
	ADV	396	7.92
0.9	UR	8	0.16
	ADV	∞	∞

Assuming that at most one packet can be generated from a processing node per cycle, the value for *localimpact* can range from 0 to p, where p is the number of processing nodes attached to each router. As can be seen from Table I, the values for *localimpact* differ significantly for the benign traffic (UR) and the adversarial traffic (ADV). For UR traffic with 0.9 injection rate (very heavy load), the *localimpact* value is only 0.16 while for ADV traffic with 0.1 injection rate (not heavy load), the *localimpact* value is 1.80. Hence, *localimpact* is a good indicator for the local traffic pattern. In our scheme, we classify the local traffic to each destination router into three types, benign, mixed, and adversarial using two threshold values low_l and $high_l$ as follows. The local traffic to destination router i is deemed

- **benign** when $localimpact = \frac{DestC_i}{h} < low_l$,
- **adversarial** when $localimpact > high_l$, and
- **mixed** when $low_l \leq localimpact \leq high_l$.

In our experiments, we set $low_l = 0.8$ and $high_l = 4$. These thresholds should be tuned in an operational system.

C. Quantifying global traffic pattern

Global traffic consists of packets generated by endpoints connected to other routers. For the global traffic pattern, the most important information for making routing decisions is how the global traffic would affect the local router. The impact of global traffic on a router can be quantified by the amount of through traffic on each port of the current router, which is the $Port\_thr_i$ counter. Table II shows the average through traffic on each port of a router under uniform random (UR) and adversarial traffic (ADV) with different loads. The simulation setting is the same as that used in Table I. As can be seen from the table, there are distinguishable differences between the through traffic counts for uniform and adversarial traffic. In particular, very large $Port\_thr_i$ (e.g. > 30) is only observed for adversarial traffic with heavy loads.

Our scheme uses $\frac{Port\_thr_i}{h}$ to quantify the global traffic on a particular port (port i) of a router and we call this value *globalimpact*. The scheme also classifies the global traffic on the port to be benign, mixed, and adversarial using two

TABLE II
THE AVERAGE $Port\_thr_i$ VALUES FOR UNIFORM AND ADVERSARIAL TRAFFIC ($h = 50$)

Injection rate	Pattern	$Port\_thr_i$	*globalimpact*
0.1	UR	2.24	0.04
	ADV	5.45	0.11
0.44	UR	9.86	0.20
	ADV	33.7	0.67
0.9	UR	20.5	0.41
	ADV	∞	∞

threshold values low_g and $high_g$. The global traffic on port i is deemed

- **benign** when $globalimpact = \frac{Port\_thr_i}{h} < low_g$,
- **adversarial** when $globalimpact > high_g$, and
- **mixed** when $low_g \leq globalimpact \leq high_g$.

In our experiments, we set $low_g = 0.33$ and $high_l = 0.6$. These parameters can be tuned for a particular system. As discussed earlier, an adversarial global traffic pattern as defined implies a mostly adversarial global traffic pattern with heavy load. On the other hand, benign or mixed global-traffic patterns can be interpreted as either uniform (with a higher load) or adversarial global traffic patterns (with a lower load). In the next sub-section, we will discuss how the local traffic pattern information can be used along this global traffic information to improve routing effectiveness.

D. The routing algorithm

Let p be the packet to be routed. We will denote $src(p)$ as the source router for the packet, $dst(p)$ as the destination router, $MINP(p)$ as the local output port in the source router for the MIN path. Let $loc(p)$ be the *localimpact* factor at $src(p)$ for p: $loc(p) = \frac{DestC_{dst(p)}}{h}$. Let $glo(p)$ be the *globalimpact* factor at $src(p)$ for p, which is the *globalimpact* for the local output port for the MIN path: $glo(p) = \frac{Port\_thr_{MINP(p)}}{h}$. The routing decision is made at $src(p)$ that has all of the needed information.

For each packet p, the source router will decide the path based on the current traffic pattern and the link loads. The current traffic condition is represented by $loc(p)$ and $glo(p)$. Based on values of $loc(p)$ and $glo(p)$, the routing algorithm operates in nine operating regions, each with a different routing mechanism. The routing mechanisms used in our algorithm range from the pure MIN routing, different UGAL-L routing schemes with different offset values T, and the pure VLB routing. UGAL-L is an effective adaptive routing algorithm whose performance is largely influenced by the offset value. With a large offset value, the routing favors minimal paths [1]. By using a range of offset values, progressively decreasing (from $+64$ to -64), UGAL-L progressively biases towards the VLB paths. We will use the notion *UGAL(T)* to denote UGAL-L with offset T, where T may be positive or negative. By using the traffic pattern information derived from the counters, the proposed routing scheme can identify different operating regions and tune the offset value accordingly to help UGAL-L perform better. The VLB routing used in our scheme, which

we call path-length oblivious VLB, is slightly different from the conventional VLB. As discussed in Section II-B, in the conventional UGAL-L routing, to select a non-minimal path, load estimation multiplied by the hop count of the path is used as the metric to decide the path load. As such, the traditional UGAL-L routing favors shorter paths since shorter paths have smaller hop counts. In our scheme, VLB is only used for the extreme adversarial condition. In such a condition, preferring shorter non-minimal paths does not perform as well as path-length oblivious VLB where only load estimation is used (without multiplying hop count) to decide the path load.

In making a routing decision, the routing algorithm considers up to four paths: two MIN paths and two VLB paths. It determines the less loaded MIN path and the less loaded VLB path, then selects between the two paths according to the traffic condition, as shown in Table III. Consider for example, the third row and third column where both $loc(p)$ and $glo(p)$ are benign: this identifies largely uniform traffic or very low load adversarial traffic when MIN achieves high performance. In this case, our algorithm either selects MIN or use UGAL-L with a very large offset (64) that strongly favors the MIN path. The mechanism to differentiate these two routing schemes in the region is to check whether the combined local and global traffic exceeds a threshold. Specifically, when $loc(p) + glo(p) < thr$, MIN is used; otherwise $UGAL(64)$ is used. Similar mechanism is also used for the case when $loc(p)$ is benign and $glo(p)$ is mixed. When both local traffic and global traffic are adversarial, pure VLB routing is used. From the table, the following can also be observed. First, for the same local traffic ($log(p)$), as the global traffic becomes more adversarial, the algorithm biases more towards a non-minimal path with a decreasing offset value for UGAL-L. Second, for the same global traffic ($glo(p)$), as the local traffic becomes more adversarial, the algorithm biases more towards a non-minimal path.

TABLE III
LOCAL AND GLOBAL TRAFFIC PATTERNS AND THE CORRESPONDING ROUTING MECHANISM FOR PACKET p

		loc(p)		
		benign	mixed	adv.
glo(p)	benign	MIN/UGAL(64)	UGAL(-4)	UGAL(-48)
	mixed	MIN/UGAL(64)	UGAL(-20)	UGAL(-64)
	adv.	UGAL(48)	UGAL(-40)	VLB

The effectiveness of the proposed scheme can be further refined by using smaller categories of local and global traffic and tailoring the offset value for each category. However, as will be shown in the next section, our algorithm is already more effective, sometimes significantly, than the traditional UGAL-L scheme. Note that like UGAL-L, our scheme only uses local information to make routing decisions. This demonstrates that using traffic pattern to filter situations and facilitate different adaptive mechanisms is an effective approach for improving routing performance.

IV. PERFORMANCE STUDY

A. Simulation methodology

The proposed scheme has been implemented in *BookSim* [8], an interconnection network simulator, which is used to evaluate the routing algorithms in this study. The experiments were designed to investigate the intra-group performance of different routing schemes on a single group of the Cray Cascade architecture [7]. The network includes 96 routers, each having 18 traffic end points as described in Section II.

The simulation configurations are similar to Jiang et al.'s [2]. We assumed single-flit packets and a 2x speedup for router crossbar over network links. The latency of each network link is set to 10 cycles. To avoid deadlocks, we used 4 VCs with a 32-entry buffer size. For each data point, the network was warmed up for 10,000 cycles, and network statistics were collected for 50,000 cycles.

B. Routing schemes

We compare the performance of 5 routing schemes under various traffic patterns. This includes MIN, VLB, UGAL-L (UGAL with local information), UGAL-G (the theoretical UGAL with global information), and TPR (our proposed traffic pattern-based routing). In TPR the windows size for the counters is 128 cycles. All adaptive routing decisions are made at the source router after a packet is injected into the network.

C. Traffic patterns

Four different families of traffic patterns are used in the evaluation. To examine the extreme case performance, we used a uniform-random traffic pattern (UR) and an adversarial shift traffic pattern (ADV). With UR, the probability of sending a packet to each destination is equal, whereas with ADV, each node connected to a given router i sends all of its traffic to nodes connected to router $i+1$. In addition, two families of mixed traffic patterns are considered by combining UR and ADV traffic patterns at processing node or router levels. In node-level combined traffic patterns, each packet is sent as either UR traffic or ADV traffic with a certain probability. Thus, the traffic from each router contains both UR and ADV components. We will use the notation NLC_URADV(UR%, ADV%) to represent the node-level combined traffic patterns, where UR% is the percentage of UR traffic and ADV% is the percentage of ADV traffic. For example, in NLC_URADV(20,80), each node sends UR traffic with 20% probability and ADV traffic with 80% probability. The other mixed traffic pattern is the router-level combined traffic pattern. In this case, all processing nodes connected to a specific router have the same traffic pattern (either UR or ADV). However, nodes at different routers may have different traffic patterns. We will use the notation RLC_URADV(UR%, ADV%) to represent the router-level combined traffic patterns. For example, in RLC_URADV(20,80), 20% of the routers generate UR traffic and 80% of the routers generate ADV traffic.

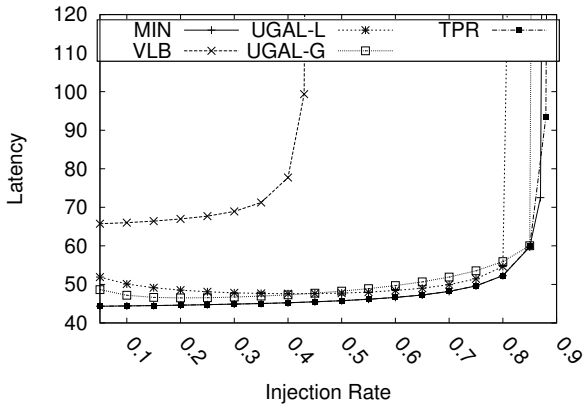

Fig. 2. Uniform Random traffic (UR)

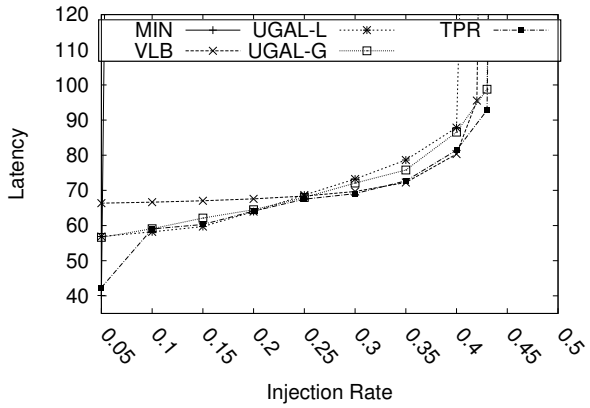

Fig. 3. Adversarial shift traffic (ADV)

D. Results

1) Uniform random traffic: Figure 2 shows the latency-throughput results for UR traffic. As expected, MIN achieves the best throughput and the lowest latency among all evaluated routing schemes due to the load-balanced nature of the UR traffic pattern. VLB achieves approximately half of the minimal routing throughput and with a higher latency because with VLB, each packet uses twice the network resources as MIN. UGAL-L achieves 92% of MIN throughput under UR, albeit at a higher latency, which stems from the fact that it routes 10–38% of the traffic non-minimally (38% under low load). This is due to the fluctuation of the queue length even with the UR traffic, which Won et al. also observed [4]. Even with the perfect global information, in comparison to MIN, UGAL-G still has noticeable higher latency at low load and achieves 98% of MIN throughput. By incorporating traffic-pattern based adaptation mechanism, TPR minimizes the inappropriate decisions to route non-minimally by distinguishing UR traffic pattern early on. For all traffic injection rates, TPR routes no more than 1% of the traffic through VLB paths and achieves almost identical throughput and latency as MIN.

2) Adversarial traffic: Results for ADV traffic are shown in Figure 3. MIN fails to balance the load for this traffic, and, as a result, the network saturates at a very low injection rate. In contrast, VLB distributes the traffic evenly among all intermediate nodes and achieves 90% of the theoretically achievable throughput. UGAL-L starts by sending 28% of the traffic through minimal paths under low load and gradually decreases this amount to as low as 1% at the saturation point. UGAL-G reaches roughly 10% higher throughput than UGAL-L by exploiting the global information and keeps sending about 12% of the traffic minimally even at high loads. TPR uses minimal routing under very low load which explains very low latency at 0.05 injection rate. Then as the load increases, *localimpact* and *globalimpact* increase as well. Therefore, TPR uses smaller offsets while using UGAL-L to balance the adversarial load and eventually switches to our path-oblivious VLB at very high load. This results in a lower latency for TPR at high load in compare to UGAL-L and UGAL-G, as well as a higher throughput than UGAL-L and traditional VLB, which is mainly due to the adoption of path oblivious VLB.

3) Node-level combined traffic: Figures 4, 5, and 6 show the results for three types of node-level combined traffic: NLC_URADV(50,50), NLC_URADV(20,80), and NLC_URADV(80,20), respectively. Note that NLC_URADV(20,80) means that the traffic is 20% uniform and 80% adversarial. These simulation studies are designed to test the performance of the proposed routing scheme under mixed and more complex traffic patterns. As can be seen from the figures, in all cases, TPR outperforms UGAL-L in terms of throughput and latency. Also TPR achieves up to 25% decrease in latency in compare to UGAL-G under low loads in all different cases. In Figure 6, under low load (injection rate < 0.2), TPR routes at least 99% of the traffic using minimal paths, in contrast to UGAL-L, which sends less than 71% using minimal paths. This explains the lower latency. As the load increases and minimal paths get congested, TPR swiftly shifts towards using UGAL-L with smaller offset values. This results in a 4% increase in throughput and 9% improvement in latency in comparison to UGAL-L. Figures 5 and 6 also verify that TPR correctly distinguishes the local and global traffic patterns and chooses the appropriate paths to route the traffic accordingly. This creates a noticeable gain in terms of latency, even relative to UGAL-G (up to 25% improvement). TPR also achieves about 98% of the throughput performance of UGAL-G.

4) Router-level combined traffic: For all traffic patterns presented so far, every router in the network has the same behavior. With router-level combined traffic (RLC), traffic from different routers may have different characteristics. The network is logically partitioned into two interleaving parts with each part having a different traffic pattern. This creates an imbalance between locally generated traffic and through traffic observed at each router, which helps us to understand if our routing scheme can handle cases where different parts of the network behave differently.

Figures 7, 8, and 9 show the results for three types of router-level combined traffic: RLC_URADV(50,50),

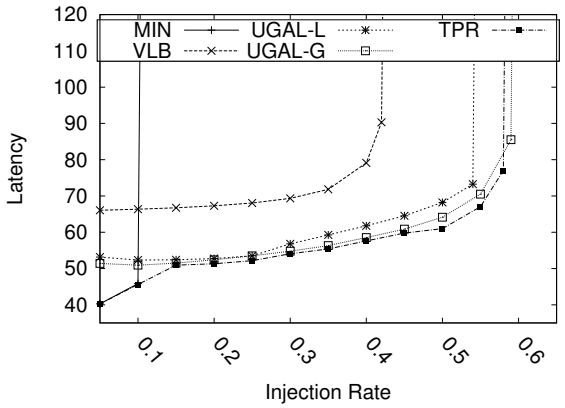

Fig. 4. NLC_URADV(50, 50)

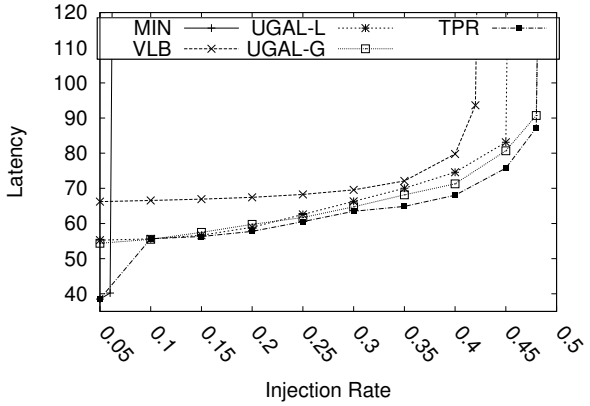

Fig. 5. NLC_URADV(20, 80)

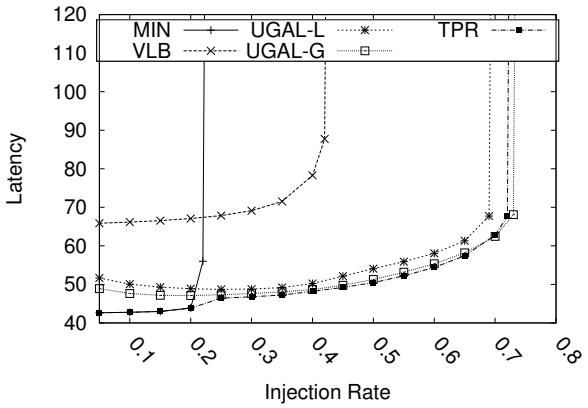

Fig. 6. NLC_URADV(80, 20)

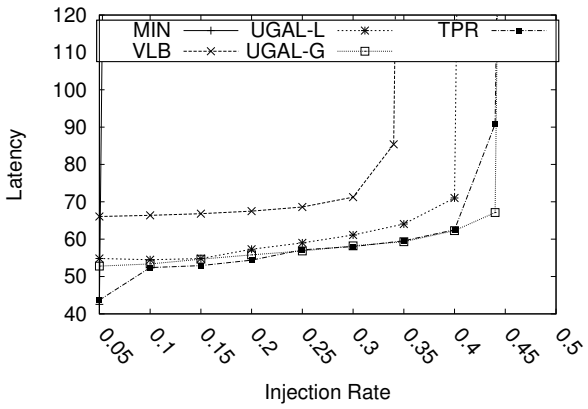

Fig. 7. RLC_URADV(50, 50)

RLC_URADV(20,80), and RLC_URADV(80,20), respectively. For RLC, a router will see the local traffic pattern (e.g. UR) to be different from the global traffic pattern (e.g. ADV). Under such a condition, TPR is able to make appropriate routing decisions by correctly identifying the local and global traffic impacts as shown in all three figures. As shown in Figure 8, except when injection rate = 0.05, global traffic is known to be **mixed** or **adversarial**. Therefore, the routing decision can be taken with regards to the local traffic impact. If the generated traffic is uniform (20% of the routers), a large offset value will be used to prefer minimal paths, whereas for other routers which are generating adversarial traffic, a very small offset value is selected. Regardless of the traffic combination, TPR always performs better than UGAL-L (and very close to UGAL-G) by identifying the traffic pattern at each router and routing traffic accordingly. This demonstrates that traffic pattern-based routing can be effective even when local traffic is very different from global traffic.

V. Related Work

Since the time when the Dragonfly interconnection network was first introduced, it has been clear that a globally adaptive routing scheme is needed to overcome the traffic pattern dependent shortcomings of minimal and non-minimal routing schemes. In the seminal work by Kim et al [1], the authors proposed selecting a random intermediate group to route non-minimally in order to load-balance adversarial traffic patterns over global channels. Jiang proposes several adaptive routing heuristics that approximate UGAL with global link state information [2]. Garcia et al. [9] were the first to address local congestion inside Dragonfly groups and proposed allowing non-minimal routing on both intra- and inter-group communication in their OFAR routing scheme. OFAR-CM [10] proposes throttling packet injection at local node as well as routing through an escaping subnetwork to mitigate congestion on OFAR routing at the cost of additional hops. Opportunistic Local Misrouting (OLM) proposed in [11] allows non-minimal routing on both local and global levels of the Dragonfly hierarchy and the routing decision may be updated at any hop.

Kim et al. [4] analyzed the effect of far-end congestion in Dragonfly network. The authors showed that because of link latency, the load estimation at the current router is inaccurate, which leads to errors in adaptive routing decisions. They proposed a history-window based approach which keeps track of the number of in-flight packets for a given window of time, and uses that information to eliminate the problem. The

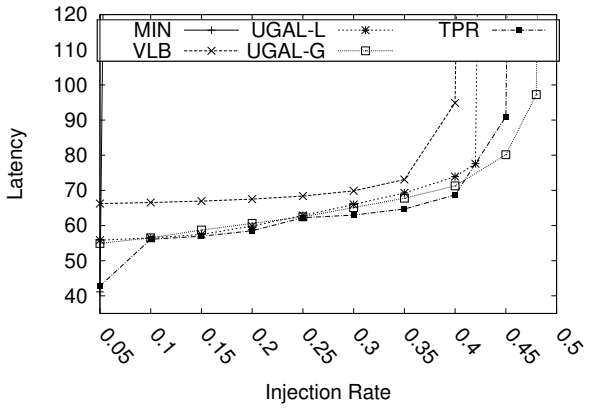

Fig. 8. RLC_URADV(20, 80)

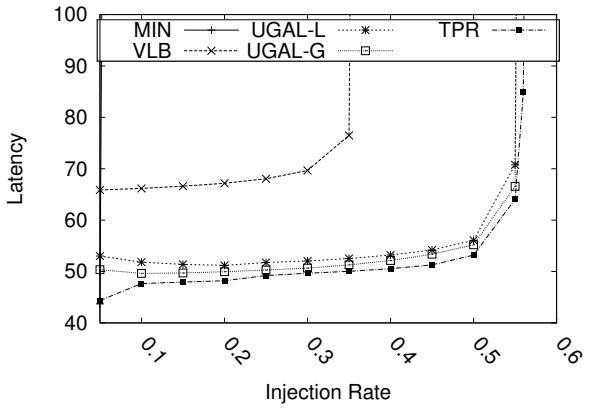

Fig. 9. RLC_URADV(80, 20)

paper also shows that the local queue length at any router can fluctuate very rapidly depending on the network traffic, so the length of the queue of a specific channel at a particular moment may not accurately represent the load. Fuentes et al., [3] observed that routing schemes based on queue length information suffer from a number of limitations, including slow response time and frequent oscillation of choice between minimal and non-minimal paths. As an alternative, the authors propose contention based routing scheme where each output port is equipped with a contention level counter that is used in lieu of queue occupancy to make routing decisions. We believe that not taking traffic pattern into consideration is a fundamental issue with all link load-based adaptive routing schemes, and propose to address the issue by explicitly estimating the current traffic pattern.

VI. Conclusion

Existing adaptive routing schemes for the Dragonfly topology make adaptive decisions based on link loads and do not consider the traffic pattern on the network that plays an important role in effective routing in this topology. In this work, we proposed a traffic-pattern-based adaptive routing scheme and showed that by incorporating the traffic-pattern based adaptive scheme, a more effective adaptive routing scheme for intra-group communication in Dragonfly can be obtained. Since the Dragonfly inter-group connectivity bears significant similarity with its intra-group connectivity, an effective routing scheme for intra-group communication can naturally be extended for inter-group communication. We are developing pattern-based adaptive routing schemes for inter-group communication.

Acknowledgements

This material is based upon work supported by the U.S. Department of Energy, Office of Science, Office of Advanced Scientific Computing Research, under Award Numbers 0000219853 and DE-SC0016039.

References

[1] John Kim, Wiliam J Dally, Steve Scott, and Dennis Abts. Technology-Driven, Highly-Scalable Dragonfly Topology. In *ACM SIGARCH Computer Architecture News*, volume 36, pages 77–88. IEEE Computer Society, 2008.

[2] Nan Jiang, John Kim, and William J. Dally. Indirect adaptive routing on large scale interconnection networks. *SIGARCH Comput. Archit. News*, 37(3):220–231, June 2009.

[3] P. Fuentes, E. Vallejo, M. Garcia, R. Beivide, G. Rodrguez, C. Minkenberg, and M. Valero. Contention-based nonminimal adaptive routing in high-radix networks. In *Parallel and Distributed Processing Symposium (IPDPS), 2015 IEEE International*, pages 103–112, May 2015.

[4] J. Won, G. Kim, J. Kim, T. Jiang, M. Parker, and S. Scott. Overcoming far-end congestion in large-scale networks. In *High Performance Computer Architecture (HPCA), 2015 IEEE 21st International Symposium on*, pages 415–427, Feb 2015.

[5] Gregory Johnson, Darren J. Kerbyson, and Mike Lang. Optimization of InfiniBand for scientific applications. In *Proceedings of the 2008 IEEE International Symposium on Parallel and Distributed Processing*, pages 1–8. IEEE Computer Society Press, 2008.

[6] Xin Yuan, R. Melhem, and R. Gupta. Compiled communication for all-optical TDM networks. In *Supercomputing, 1996. Proceedings of the 1996 ACM/IEEE Conference on*, pages 25–25, 1996.

[7] Greg Faanes, Abdulla Bataineh, Duncan Roweth, Edwin Froese, Bob Alverson, Tim Johnson, Joe Kopnick, Mike Higgins, James Reinhard, et al. Cray Cascade: A Scalable HPC System Based on a Dragonfly network. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, page 103. IEEE Computer Society Press, 2012.

[8] Nan Jiang, Daniel U. Becker, George Michelogiannakis, James D. Balfour, Brian Towles, David E. Shaw, John Kim, and William J. Dally. A detailed and flexible cycle-accurate network-on-chip simulator. In *ISPASS*, pages 86–96. IEEE Computer Society, 2013.

[9] M. Garcia, E. Vallejo, R. Beivide, M. Odriozola, C. Camarero, M. Valero, G. Rodrguez, J. Labarta, and C. Minkenberg. On-the-fly adaptive routing in high-radix hierarchical networks. In *Parallel Processing (ICPP), 2012 41st International Conference on*, pages 279–288, Sept 2012.

[10] M. Garcia, E. Vallejo, R. Beivide, M. Valero, and G. Rodrguez. OFAR-CM: Efficient Dragonfly networks with simple congestion management. In *High-Performance Interconnects (HOTI), 2013 IEEE 21st Annual Symposium on*, pages 55–62, Aug 2013.

[11] M. Garcia, E. Vallejo, R. Beivide, M. Odriozola, and M. Valero. Efficient routing mechanisms for Dragonfly networks. In *Parallel Processing (ICPP), 2013 42nd International Conference on*, pages 582–592, Oct 2013.