Throughput Unfairness in Dragonfly Networks under Realistic Traffic Patterns

Pablo Fuentes, Enrique Vallejo,
Cristóbal Camarero, Ramón Beivide
University of Cantabria, Spain
{*pablo.fuentes, enrique.vallejo,*
cristobal.camarero, ramon.beivide}*@unican.es*

Mateo Valero
Universitat Politecnica de Catalunya (UPC)
and Barcelona Supercomputing Center (BSC), Spain
mateo@bsc.es

***Abstract*—Dragonfly networks have a two-level hierarchical arrangement of the network routers, and allow for a competitive cost-performance solution in large systems. Non-minimal adaptive routing is employed to fully exploit the path diversity and increase the performance under adversarial traffic patterns. Throughput unfairness prevents a balanced use of the resources across the network nodes and degrades severely the performance of any application running on an affected node. Previous works have demonstrated the presence of throughput unfairness in Dragonflies under certain adversarial traffic patterns, and proposed different alternatives to effectively combat such effect.**

In this paper we introduce a new traffic pattern denoted *adversarial consecutive (ADVc)*, which portrays a real use case, and evaluate its impact on network performance and throughput fairness. This traffic pattern is the most adversarial in terms of network fairness. Our evaluations, both with or without transit-over-injection priority, show that global misrouting policies do not properly alleviate this problem. Therefore, explicit fairness mechanisms are required for these networks.

I. Introduction

Dragonfly networks are considered as one of the most promising network topologies for upcoming Exascale systems, and have been employed in the PERCS [1] and Cascade [2] system networks. Unfortunately, these networks easily suffer congestion under certain adversarial traffic patterns. To overcome bandwidth limitations and fully exploit path diversity, non-minimal adaptive routing mechanisms are required. These routing mechanisms employ an intermediate random node to divert the traffic before sending minimally towards the destination, improving the utilization of the inter-group (*global*) links in the event of saturation in a link on the minimal path.

Throughput unfairness was identified in [3] when employing an adversarial traffic pattern (ADV) that heavily congests one router in every group. A new global misrouting policy named *Mixed-mode (MM)* was proposed for the selection of the intermediate group in the non-minimal path for in-transit adaptive routing mechanisms. The *MM* global misrouting policy provides competitive throughput and latency, while avoiding unfairness in the bottleneck router of the group.

Previous evaluations focused on random traffic based on Uniform (UN) and Adversarial (ADV) traffic patterns. UN represents a best-case which is useful to evaluate the topological properties of the network and is considered as a good approximation for the average behavior of several applications, such as data-intensive. *ADV* represents a corner case that could occur when an application is spread over two (or more) different groups of the Dragonfly network. While these two traffic patterns cover the extreme cases in respect to routing, they do not fully represent the complete spectrum of traffic patterns in a Dragonfly network. In this work, we identify a new traffic pattern designated as *Adversarial consecutive (ADVc)*, and justify its potential occurrence in a real system. Under *ADVc*, traffic is sent to several destination groups, with their minimal paths meeting in a single router. This pattern is less adversarial than *ADV* in terms of throughput, but generates the maximum unfairness under both source and in-transit adaptive routing mechanisms.

This *ADVc* traffic pattern threats the benefits of the *MM* policy, as the minimal and non-minimal paths in the bottleneck router overlap; this will be detailed later in Section III. In this work, we demonstrate that none of the previous routing mechanisms or global misrouting policies prevent throughput unfairness under such traffic pattern. We additionally evaluate the impact of prioritizing transit over injection traffic at the router allocator, noticing that it achieves a slightly higher throughput at the cost of lower fairness.

In short summary, our main contributions are:

- We highlight the pitfall of optimizing routing mechanisms exclusively for corner cases, not for the general case. In particular, we identify a new adversarial traffic pattern, *Adversarial consecutive(ADVc)*, and rationalize its correspondence to a use case in an actual system and how it differs from both UN and ADV.
- We quantify the impact of the routing mechanism and the use of transit-over-injection priority on throughput, latency and unfairness, under different traffic patterns including *ADVc*.
- We demonstrate the inability of previous global misrouting policies to prevent throughput unfairness un-

This is an earlier accepted version; a final version of this work can be found in the proceedings of the 2015 IEEE International Conference on Cluster Computing (CLUSTER) under DOI 10.1109/CLUSTER.2015.136. Abstract can be read here.

der *ADVc* traffic pattern, concluding that despite their simplicity, explicit fairness mechanisms are required in these networks.

II. Background and related work

In this section we introduce a description of the Dragonfly network, the global misrouting policies employed to increase throughput and reduce unfairness, and the different routing mechanisms proposed.

A. Dragonfly networks

The Dragonfly [4] is a low-diameter network based on high radix routers. Routers in a Dragonfly network are deployed in a two-level hierarchical layout, with fully-connected groups of routers conforming a virtual high-radix router. Such groups are connected on a second-level interconnection pattern. In this work, we focus on Dragonfly networks with complete graphs in both levels, denoted as *canonical dragonflies* in [5].

A Dragonfly network with complete graphs in both hierarchical levels can be described using three parameters [4]:

- p is the number of nodes linked to every router.
- a is the number of routers per group in the first hierarchical level.
- h is the number of inter-group (*global*) links in each router, connecting with a router in a different group.

Additionally, the global link arrangement specifies the distribution of global links among the routers of each group; in this work we employ the *palmtree* arrangement [5], but the study and results for other arrangements are similar.

Performance in Dragonflies is tightly connected to the pattern of communications and the routing mechanism. For random traffic patterns that stress uniformly the links in the network, the usage of the shortest path between source and destination nodes provides sufficient performance in terms of throughput and latency. However, performance is severely affected under other traffic patterns with higher contention in the inter-group links, due to a poor use of the path diversity. For these cases, non-minimal routing mechanisms are required to achieve good performance.

B. Global misrouting policies

A remote group can be directly or indirectly connected to a given router in the Dragonfly network. When a group is directly linked to the current router, only one global link needs to be traversed to reach such group. Arriving to an indirectly linked group implies traversing another router in the current group, requiring two hops: one local link from the current router to the neighbor router which is connected to the destination group, and one global link between the two groups (*lg*).

The *global misrouting policy* defines the intermediate group in non-minimal paths, depending whether it is a directly or indirectly connected group from the current router. Different global misrouting policies were introduced in [6] for source-based routing:

- ***Random-router Global, (RRG)***: the intermediate group is selected randomly across the network, regardless of its distance from the current router.
- ***Current-router Global, (CRG)***: only those groups that are directly linked to the router are considered valid for the non-minimal path. In this case, there is always a 1 hop distance towards the intermediate group.
- ***Neighbor-router Global, (NRG)***: in non-minimal paths, traffic is diverted to a group connected to a different router in the source group. Packets traverse 2 links (1 local + 1 global) before reaching the intermediate group.

RRG balances evenly the non-minimal traffic load between all the global links in the network, whereas *CRG* reduces the non-minimal path length. *NRG* has the longer average non-minimal path and reduces the performance under a uniform pattern of communications. However, an adversarial traffic pattern can stress more heavily one or more global links in the group, making the *RRG* and *CRG* policies less desirable as they will not alleviate the unbalance of the minimal traffic load. Under any pattern that stresses non-uniformly the global links, a combination of *NRG* and *CRG* global misrouting policies can mitigate this effect and improve the performance on those routers connected to the more congested global links.

As defined in [6], we consider an additional misrouting policy named ***Mixed-mode (MM)*** for in-transit adaptive routing mechanisms:

- ***MM*** employs a *CRG* misrouting policy when attempting misrouting at the source router, and a *NRG* policy for traffic which is in-transit.

This *MM* policy balances the traffic evenly across all the global links in the network, while reducing the impact of non-minimal traffic at those global links that are heavily congested due to the traffic routed minimally under *ADV*.

C. Routing mechanisms

Several routing mechanisms have been proposed for the Dragonfly network [4], [7], [8], [3], [5]. In this work we classify them in three categories: oblivious, source-based adaptive, and in-transit adaptive routing. Oblivious routing selects a path at injection which is independent of the current status of the network, whereas adaptive routing mechanisms react to congestion modifying paths to improve network performance. Source-based adaptive routing selects between multiple paths at injection, depending on a decision which is typically based on a direct or indirect measure of the network congestion. By contrast, in-transit adaptive routing can switch between minimal and non-minimal paths at injection and along the route, what avoids the need for indirect congestion measures.

In particular, the routing mechanisms which have been selected to model these three classes are the following:

- **Oblivious routing:** Several oblivious routing mechanisms are employed as a reference, depending on the traffic pattern. Minimal routing (***MIN***) is the reference for random uniform traffic. It delivers traffic through the shortest path, employing up to three hops (one *local* and one *global* link to reach the destination group, and one *local* link to arrive to the destination node, *lgl*). For adversarial traffic patterns (ADV when all traffic from a source group is sent to the same destination group, and ADVc as introduced in Section III), nonminimal routing is required to avoid the congested links. In this case, Valiant routing (*VAL*, [9]) can be used to send traffic non-minimally. It selects a random intermediate node between the source and the destination to divert the traffic through longer routes. These long paths will be less congested than the minimal path under adversarial traffic patterns. Valiant requires up to six hops to complete the network traversal, three to the intermediate node (*lgl*-) and three from the intermediate node to the destination (-*lgl*). In the original definition of Valiant, the intermediate node is selected randomly between all nodes in the network at packet generation time. In our case, we have implemented two related nonminimal oblivious routing variants according to the global misrouting policies introduced in Section II-B: ***Oblivious-RRG*** is similar to Valiant, since it selects the intermediate destination completely randomly. By contrast, ***Oblivious-CRG*** modifies the initial selection of the random intermediate node, restricting it to nodes in groups directly connected to the source router. This saves the (frequent) first local hop, but restricts the amount of random intermediate nodes.
- **Source-based adaptive routing:** We employ Piggy-Back (PB, [7]) as a source adaptive routing mechanism. It estimates the congestion of the network and selects between *VAL* and *MIN* routing when injecting a new packet depending on the saturation status of the minimal link. A link is considered as saturated when its associated credit count exceeds a given threshold, relative to the other nodes. The saturation status information is shared across the routers in the same group, in a sort of Explicit Congestion Notification (ECN). As in the previous case, we have implemented two variants of source-based adaptive routing, depending on the use of Oblivious-CRG or Oblivious-RRG for the selection of the nonminimal path. We denote these two variants as ***Source-based-CRG*** and ***Source-based-RRG*** respectively.
- **In-transit adaptive routing:** Our implementation applies in-transit global and local misrouting. Global misrouting (sending traffic to a non-minimal group) can be selected at injection or after a first hop in the source group as in *PAR* [7]. The selection relies on the number of credits of the output ports in the current router. In the intermediate or destination groups, local misrouting can be used if the links from the minimal path are considered saturated. This avoids pathological performance issues identified in [8], [10]. Deadlock avoidance implements Opportunistic Local Misrouting *OLM*, [3] to reduce the cost of the implementation. We have implemented the three variants of global misrouting policy introduced in Section II-B, and denoted them ***in-transit-CRG***, ***in-transit-RRG*** and ***in-transit-MM*** respectively.

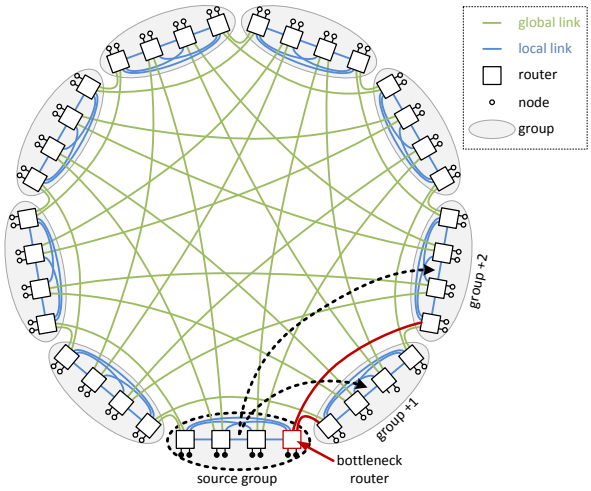


Figure 1: Adversarial-consecutive (*ADVc*) traffic pattern in a Dragonfly with $h = 2$. Traffic from the bottom down group targets the next $h = 2$ consecutive groups $(+1, +2)$. The highlighted router connects to the minimal global links towards those two destination groups.

III. Adversarial-consecutive traffic pattern

In this section we introduce a new traffic distribution which is particularly adversarial in terms of throughput fairness. In the *Adversarial consecutive (ADVc)* traffic pattern, messages are sent randomly to h destination groups. In particular, we select the h consecutive groups $(+1, +2, ..., +h)$ after the source group, which are all connected to the same (bottleneck) router of the source group[1]. Figure 1 illustrates this traffic pattern with a minimal example for a Dragonfly network with 9 groups and 72 nodes ($h = 2$).

Using *MIN* routing, throughput is limited to h/ap phits/node/cycle. This limitation is less severe than under *ADV* (which is $1/ap$) and is avoided by using nonminimal

[1]These destination groups apply when the *palmtree* global link arrangement [5] is used, as depicted in Figure 1. For other arrangements, an equivalent traffic pattern can be derived by selecting h destination groups directly connected to one router in the source group, which will become the bottleneck.

routing. However, *ADVc* traffic constitutes a challenge for throughput fairness, since the bottleneck router of the group is likely to get its minimal global output links congested due to the traffic routed minimally from other neighbours in the group. Furthermore, a *CRG* global misrouting policy will aggravate this effect: from the bottleneck router only, permitted global links for non-minimal paths coincide with minimal global links for flows from other routers, and they are probably congested.

This distribution of communications can occur easily when an application employs several $(h+1)$ groups, not the whole network. This is the common case for HPC applications in large systems. A consecutive allocation of groups is the simplest approach for the job scheduler. In such case, even uniform traffic between the application processes would translate into *ADVc* traffic in the network (at least in one of the Dragonfly groups). Since the execution of applications that employ a complete HPC system is very unfrequent, we believe that this *ADVc* traffic pattern (or small variants) can occur very frequently in large Dragonfly networks. Alternative allocation schemes which avoid consecutive group allocation can also inadvertently generate this traffic pattern, with a different bottleneck router in one of the groups of the system, especially for large h or for different global link arrangements.

IV. Evaluation methodology

In this section we introduce the environment that we have employed for our evaluations, detailing the simulation tool and the parameters we have selected. Then we describe the performance metrics that will be reproduced in Section V.

A. Simulation infrastructure

We employ the in-house designed FOGSim network simulator [11] for our evaluations. We model a Dragonfly network with $h = 6$, 5256 nodes and 876 input-output-buffered routers. Each router employs multiple virtual channels as a deadlock avoidance mechanism, which also mitigate Head-of-Line (HoL) blocking. A fine-grain model of a high-radix router as described in [12] cannot be implemented for a network of this size. Thus, we employ a simpler model of a router with a 5-cycle pipeline and an iterative separable batch allocator. Routers commute traffic at $2\times$ the link speed to reduce the performance limitations from HoL blocking and suboptimal allocator decisions. We also evaluate the impact of prioritizing in-transit traffic from injection traffic, similar to Blue Gene systems [**?**]. Table I reflects the parameters employed in our simulations.

The link latency of 10 and 100 cycles for the local and global links models the use of 2 and 20 meters wires delivering data at a 10GB/s pace, with routers transmitting 10 bytes per cycle and operating at 1 GHz. A more detailed justification for these selection can be found in [7].

Parameter	**Value**
Router size	23 ports (h=6 global, p=6 injection, 11 local)
Router latency	5 cycles
Frequency speedup	$2\times$
Group size	12 routers, 72 computing nodes
System size	73 groups, 5,256 computing nodes
Global link arrangement	Palmtree [5]
Link latency	10 (local), 100 (global) cycles
Virtual Channels	2 (global ports), 3 (local and injection ports), 4 (local ports in oblivious and source-adaptive mechanisms)
Switching	Virtual Cut-Through
Buffer size (phits)	32 (output buffer, local input buffer per VC), 256 (global input buffer per VC)
Packet size	8 phits
Congestion thresholds	43% (Adaptive in-transit), $T = 5$ (PB, local links), $T = 3$ (PB, global links)

Table I: Simulation parameters.

All our evaluations have been conducted employing three different types of synthetic traffic: Uniform Random (*UN*), Adversarial (*ADV+1*) and Adversarial consecutive (*ADVc*). *UN* traffic selects a random destination node across all the network for every packet injected. In *ADV+1* traffic all the nodes in a given group address their traffic towards the nodes in the next group $(+1)$; results for other destination groups are similar. Under *ADVc* traffic the nodes send their packets randomly to the nodes in the $h = 6$ next immediately consecutive groups, as detailed in Section III. Nodes generate the packets following a Bernoulli process with an adjustable injection probability expressed in phits/(node·cycle).

In all our experiments we first warm-up the network for an adequate amount of time before tracking the average latency and throughput statistics during 15,000 cycles of execution. Curves in Section V present the average of 3 different simulations. Results comprise the different metrics explained below.

B. Performance and throughput metrics

We have measured performance and fairness results. Performance metrics measure the capacity of the network to absorb properly a traffic load for a given traffic pattern, whereas the fairness metrics give a quantitative measure of the unbalance in the allocation of network resources between computing nodes.

We consider two performance metrics:

- Throughput: the average amount of traffic (in phits/(node·cycle)) that can be delivered to the destinations during a simulation cycle.
- Latency: the average delay between the moment a flit is inserted into the injection queue at the source router and the time it is delivered at the destination, measured in cycles. This value can be break down into its different components, namely the waiting time at the injection, local transit and global transit queues, the delay associated to the traversal of the links in the

minimal path, and the traversal of the links in the non-minimal path.

As for the fairness metrics, multiple indicators are frequently used to quantify the presence of throughput unfairness:

- Number of injected packets: we compute the number of injected packets at each router of a given group. This allows to determine the difference in network resources allocation to the nodes at each different router, and detect the existence of a router whose nodes suffer starvation.
- Minimal number of injections (*Min inj*): the lowest number of packets injected per router in the network. This allows to detect a case of unfairness across the whole network. However, it fails to determine if it is an isolated anomaly or a common behavior for multiple routers in the network. For this reason, we contemplate the next two metrics.
- Max-to-min ratio (*Max/Min*): quotient between the highest and lowest number of injections per router in the network. This highlights both the case in which a router receives an excessively high or low amount of resources compared with the rest of the network.
- Coefficient of variation (*CoV*): the quotient between the variance and the average number of injections per router:

$$COV = \frac{\sigma}{\mu}$$

  With this metric we are able to discriminate between a case in which one router has an isolated case of starvation and another router is given an abnormally high number of resources, and a case in which half of the routers starve and the the other half benefit from an unfairly high number of allocated resources. Obviously, from the point of view of the applications both situations are undesirable, but it can be argued that the latter would have a more negative impact on the application performance.

V. Results

This section presents the results obtained from our experiments with the different routing mechanism and global misrouting policy combinations, under the three aforementioned traffic patterns. These results are divided into performance (throughput and latency) and fairness results. First we present the outcome employing transit-over-injection priority at the router allocators. Next we repeat the evaluation removing such priority, to determine its impact on throughput fairness.

A. Performance results with transit-over-injection priority

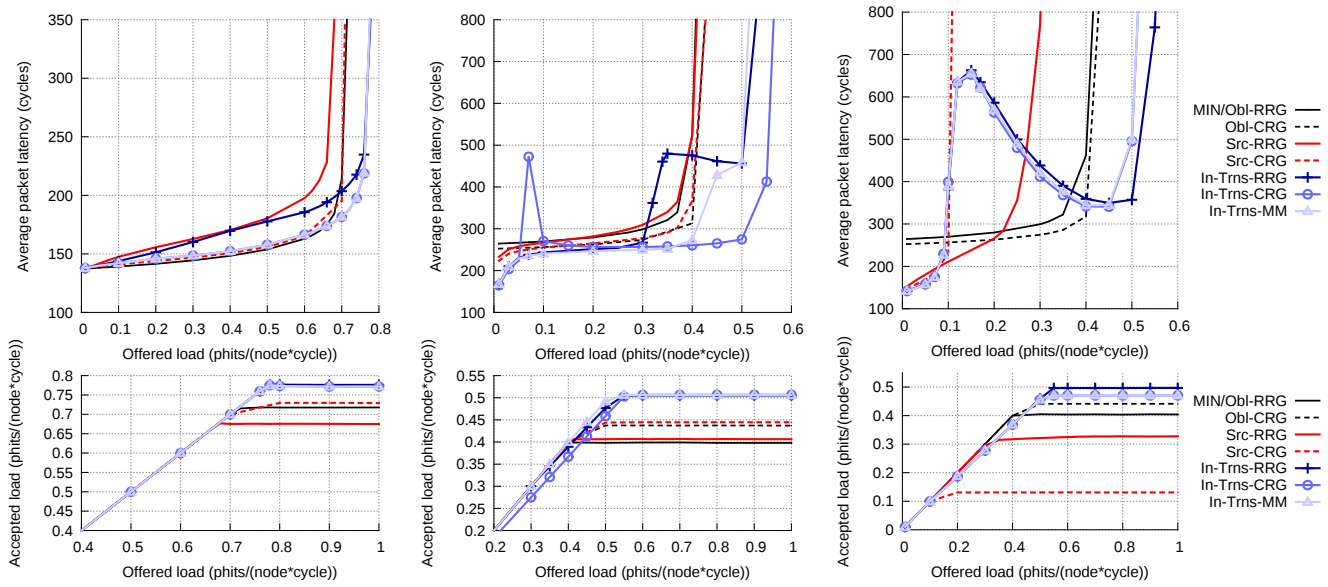
Figure 2 shows average throughput and latency for the described oblivious, source adaptive and in-transit adaptive routing mechanisms under *UN*, *ADV+1* and *ADVc* traffic patterns, using transit-over-injection priority. Performance under *UN* traffic in Figure 2a is good for all the routing mechanisms evaluated. *CRG* and *MM* global misrouting policies (which employ global links for the misrouting at the source router) achieve a latency close to the minimal marked by the *MIN* routing. Since *MIN* routing does not employ non-minimal paths, in this case we do not evaluate the impact of the global misrouting policy in oblivious routing. In this case, the usage of *RRG* is detrimental compared to the other global misrouting policies, as it increases the latency and has a negligible to negative effect in the throughput (the latter being the case with source routing). Nevertheless, it can still be considered competitive.

The impact of the global misrouting policy gains interest under adversarial traffic patterns like *ADV+1* and *ADVc* (Figures 2b and 2c). In these cases, the reference black lines represent nonminimal oblivious routing. Under *ADV+1* traffic, *CRG* again performs better (higher throughput and lower latency) than *RRG* for all the routing mechanisms; the spike in average latency for *in-transit-CRG* is discussed later. *RRG* employs in average longer paths than *CRG* (because of the extra local hop in the source group) what increases latency and reduces throughput. Best performance is achieved by the in-transit adaptive routing with the *MM* global misrouting policy. This advantage was previously described in [6] and is a consequence of combining the most beneficial selection at injection (*CRG*) and during network traversal (*RRG*).

The effect of unfairness with in-transit adaptive routing under *ADV+1* is obvious in Figure 2b. Average latency presents a peak when the bottleneck router starts to suffer starvation. With *CRG*, this occurs at an extremely low load. After this point, the accepted load of this starved router remains still. Its high latency is hidden when averaging with the remaining routers in the group, which are not saturated and inject a higher load. *CRG* and *RRG* experiment this behavior at a higher traffic load and the reduction of the average latency never occurs. Instead, there is a flat region where the general increase in latency is compensated by a lower presence of high-latency packets from the starving routers.

The repercussions in throughput are more subtle, reflecting on accepted load below the offered load before reaching the saturation point. The most prominent case is the in-transit adaptive routing with the *CRG* global misrouting policy.

Under *ADVc* traffic in Figure 2c, all the routing mechanisms fail to perform well in both metrics. The oblivious and source adaptive routing mechanisms have lower latency and do not present peaks due to throughput unfairness below the saturation point, but their throughput is relatively low. In the case of source-adaptive routing, the Piggyback implementation we employ fails to properly identify global links as saturated (because all the links in the bottleneck router are saturated, with the same average load) and a



(a) UN. (b) ADV+1. (c) ADVc.

Figure 2: Latency and throughput under uniform (UN) and adversarial traffic (ADV+1, ADVc), prioritizing transit over injection.

large amount of traffic is sent minimally, especially with *CRG*. In-transit adaptive routing performs best in throughput but clearly suffers from throughput unfairness. This can be appreciated in the throughput curves before saturation which are below those of oblivious routing, and in the peak and subsequent fall in latency at 0.15 phits/(node·cycle).

It is remarkable that *CRG* is the most suitable global misrouting policy for oblivious nonminimal routing under *ADV+1* and *ADVc* traffic, whereas the source adaptive routing benefits from the *RRG* policy under the *ADVc* traffic pattern. This conduct arises because the granularity for the congestion threshold in the local queues is much lower than for the global queues, forcing an excessive amount of minimally-routed traffic through the bottleneck router.

Figure 3 displays a latency breakdown for the in-transit adaptive routing with *MM* global misrouting policy under *ADVc* traffic. Five different components are considered: link traversal through the minimal and non-minimal paths, waiting time in local and global link queues, and waiting time at injection. Misrouting latency (due to the traversal of the non-minimal links) increases with the injection rate until the saturation point, at 0.5 phits/(node·cycle). Congestion (both in local and global links) has a relatively low impact on the total latency under all traffic loads. The average waiting time at injection queues shows a remarkable behavior: it grows before reaching a peak at 0.15 phits/(node·cycle) and then steadily diminishes until reaching saturation. Again, this behavior reflects an unfairness effect in which the bottleneck router saturates at low loads and suffers high latency, but its impact is hidden as more packets from other routers are averaged when the offered load increases.

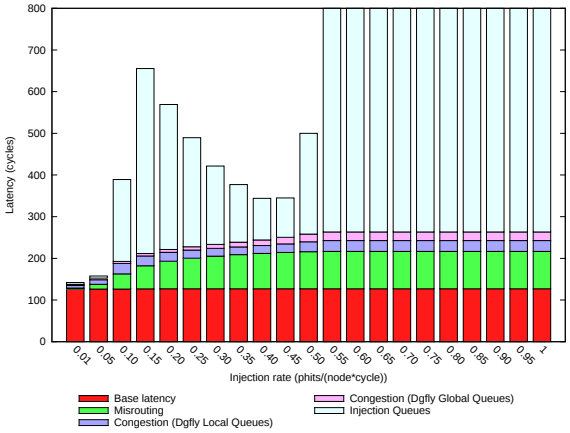


Figure 3: Breakdown of the latency components for the in-transit adaptive routing with *MM* policy under *ADVc* traffic.

B. *Throughput fairness with transit-over-injection priority*

After asserting the presence of throughput unfairness, we quantify it for the different traffic patterns and routing mechanisms. Figure 4 portrays the number of injected packets from every router of one group under *ADVc* traffic with a traffic load of 0.4 phits/(node·cycle), for the different combinations of routing mechanism and global misrouting policy.

Oblivious non-minimal routing does not suffer from

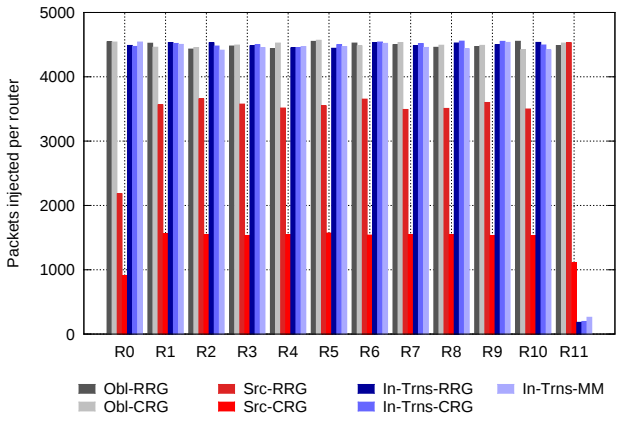


Figure 4: Number of injected packets per router in a group of the Dragonfly network, under *ADVc* traffic with a traffic load of 0.4 phits/(node·cycle). In-transit traffic is given priority over injection.

	Min inj	Max/Min	COV
Obl-RRG	4079	1.149	0.0175
Obl-CRG	4307	1.095	0.0145
Src-RRG	2134	2.196	0.1217
Src-CRG	847	2.735	0.1029
In-Trns-RRG	37	585.69	0.2866
In-Trns-CRG	31.67	185.60	0.2861
In-Trns-MM	69.33	72.576	0.2858

Table II: Fairness metrics for the different routing mechanisms and global misrouting policies, under *ADVc* traffic with a load of 0.4 phits/(node·cycle). Traffic in the transit queues is being prioritized over traffic in the injection queues.

throughput unfairness, presenting a similar amount of injected packets per router in all the routers of the group. This behavior is not significantly affected by the global misrouting policy in use. However, adaptive routing mechanisms present a completely different conduct. Source adaptive routing tends to favor some routers in detriment of others: with a *RRG* global misrouting policy, router R0 injects a significantly lower amount of packets than the rest, whereas router R11 injects a higher amount of traffic. This trend changes with the *CRG* global misrouting policy: both R0 and R11 routers inject a lower amount of traffic than the others. Observe that R11 is the bottleneck router, and R0 is the router that receives the traffic sent minimally from other groups.

However, the difference between routers becomes specially pervasive when in-transit adaptive routing is employed. The amount of injected packets at the bottleneck router is several orders of magnitude lower than in the other routers of the group, and the impact of the global misrouting policy can be considered trivial. This behavior is considerably harmful since the in-transit adaptive routing achieves the highest throughput under all the traffic patterns considered, and the lowest latency under *UN* and *ADV+1* traffic when combined with the *CRG* or the *MM* global misrouting policy.

We quantify the unfairness through the metrics described in Section IV-B. Table II refers the minimum injection, max/min ratio, and coefficient of variation for all the routers in the network for the simulation depicted in Figure 4.

In-transit adaptive routing and source adaptive routing with *CRG* perform worse than oblivious and *Src-RRG*, with a significantly lower minimum number of injected packets per router. The *Min/Max* metric adds further information, with all the routing mechanisms achieving the same order of magnitude with the different global misrouting policies. This implies the injection drop from *Src-RRG* to *Src-CRG* is not constrained to the bottleneck routers but a general trend. This is confirmed by the *COV* which is actually lower for *Src-CRG*, implying lower unfairness. Higher *COV* values for in-transit adaptive routing refer a high variability between routers, implying that the unfairness is not constrained to a few isolated cases.

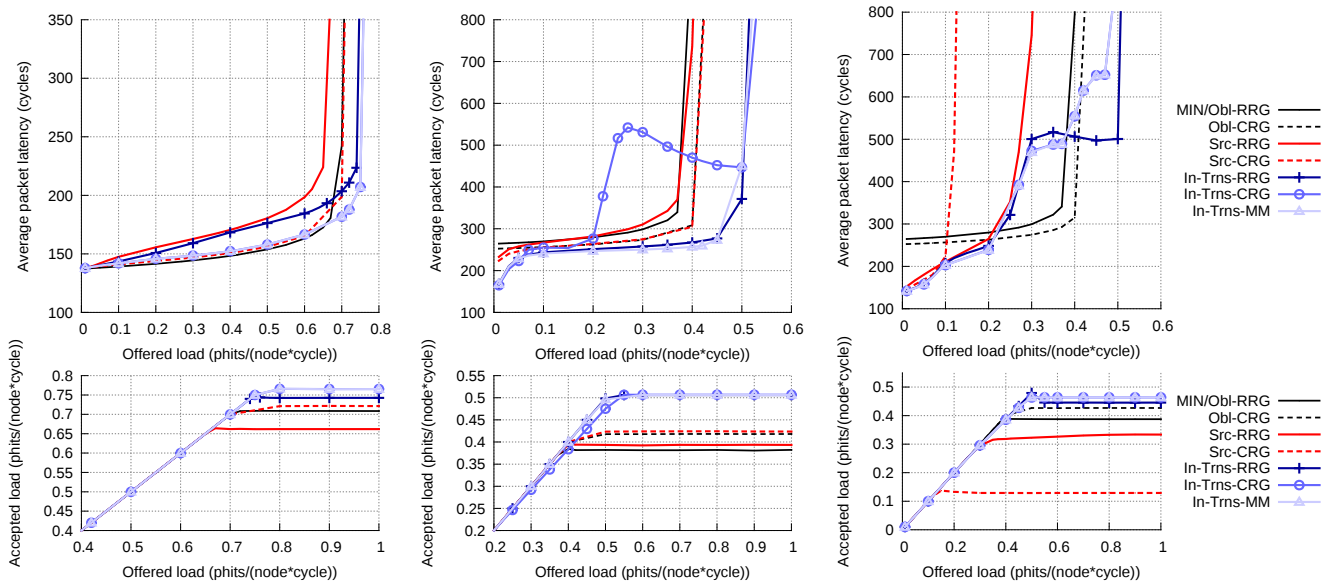
The unfairness problem for the in-transit adaptive routing is partially originated from the use of a transit-over-injection priority. Under the *ADVc* traffic pattern and using the *CRG* or *MM* policies, minimal and non-minimal global links fully overlap at one router of every group, as it was explained in Section III. Prioritizing the in-transit traffic over new injection is benign for the overall network performance, but prevents any router from injecting when such overlap exists. Consequently, we have evaluated the performance and throughput fairness when such priority is removed.

C. Performance and throughput fairness without transit-over-injection priority

Performance results without transit-over-injection priority are presented in Figure 5. Removing this priority increments the congestion level in the network, leading to slightly lower throughput results; under *UN* traffic, throughput for *MIN* decreases around a 1.2%. It also significantly affects latency under adversarial traffic. Under *ADV+1*, in-transit adaptive routing with *CRG* or *MM* global misrouting policies do not show any peak caused by starvation. With *RRG* global misrouting, the peak appears but at a much higher load.

Nevertheless, *ADVc* traffic still exhibits a latency behavior that can be undoubtedly attributed to throughput unfairness. The improvement over the results with the priority in Figure 2c is noteworthy, but unable to effectively eliminate it.

Figure 6 presents the number of injected packets per router in a group of the Dragonfly network under *ADVc* traffic with a load of 0.4 phits/(node·cycle), when the transit-over-injection priority has been removed. Oblivious routing mechanisms maintain the trend observed in Figure 4, without any significant throughput unfairness between the routers. Source adaptive routing displays a difference with the *CRG* global misrouting policy in the bottleneck router



(a) UN. (b) ADV+1. (c) ADVc.

Figure 5: Latency and throughput under uniform (UN) and adversarial traffic (ADV+1, ADVc), without prioritizing transit over injection.

	Min inj	Max/Min	COV
Obl-RRG	3937	1.190	0.0173
Obl-CRG	4314	1.093	0.0144
Src-RRG	2247.33	2.086	0.1194
Src-CRG	690.5	6.673	0.5562
In-Trns-RRG	2553.33	1.850	0.1106
In-Trns-CRG	2549.33	1.852	0.1111
In-Trns-MM	2554.33	1.843	0.1101

Table III: Fairness metrics for the different routing mechanisms and global misrouting policies, under *ADVc* traffic with a load of 0.4 phits/(node·cycle), without transit-over-injection priority.

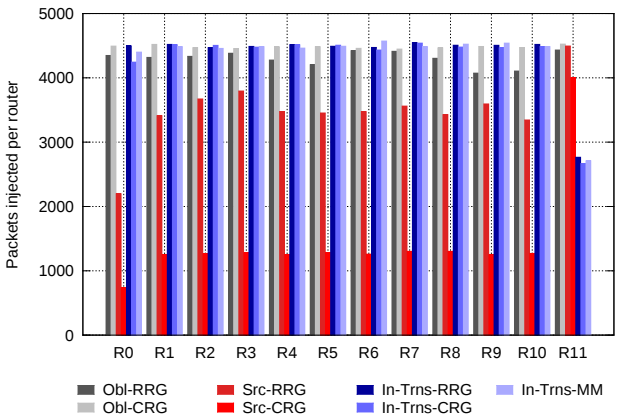


Figure 6: Injected packets per router in a group of the Dragonfly network, under *ADVc* traffic with a traffic load of 0.4 phits/(node·cycle), without transit-over-injection priority.

R11, showing a significatively higher amount of injected packets. Not only is it higher than itself when the priority was used, but also higher (more than $2\times$) the number of packets injected in other routers in the group. Such variation can be easily explained by the absence of transition-over-injection priority, which was preventing a higher injection at the bottleneck router. Since the selection between minimal and nonminimal paths is based on the saturation of the links, the bottleneck router becomes itself aware of the status of the minimal global links faster than any other network. Hence, it is capable of exploiting the global links as soon as they stop being saturated, and makes an unfairly high use of said resources.

In-transit adaptive routing vastly improves the fairness between routers under all three global misrouting policies (*RRG*, *CRG*, *MM*), with an identical improvement for all of them. Unfortunately, this improvement is not sufficient to consider the use of the global links as fair. Values in Table III quantify the unfairness without the priority, demonstrating that the fairness of the adaptive routing mechanisms is far from the oblivious nonminimal routing. It must be noticed the *COV* for the *Src-CRG* routing, which reflects the impact of the increase of load in the bottleneck router.

VI. Conclusions

In this work we have evaluated the throughput unfairness in a Dragonfly network with different routing mechanisms. For this purpose, we describe a new *Adversarial consecutive*

traffic pattern that we believe more likely to appear in real applications than other synthetic traffic workloads such as *UN* or *ADV+1*. Results indicate that a global misrouting policy is not sufficient to eradicate unfairness under this new traffic. We have also evaluated the impact of the transit-over-injection priority, determining a noticeable but insufficient improvement with in-transit adaptive routing. Furthermore, priority removal has a negative impact on throughput fairness with one source adaptive routing. Explicit fairness mechanisms are required to ensure an effective lack of unfairness with this traffic pattern and adaptive routing mechanisms.

For future work, we plan to evaluate explicit fairness mechanisms, such as Age Arbitration [13]. In particular, the use of such mechanisms in Dragonflies and their potential interaction with nonminimal adaptive routing and global misrouting policies has not been evaluated before. The results in this paper motivate such evaluation.

ACKNOWLEDGMENT

This work has been supported by the Spanish Ministry of Education, FPU grant FPU13/00337, the Spanish Science and Technology Commission (CICYT) under contracts TIN2012-34557 and TIN2013-46957-C2-2-P, the European Union FP7 under Agreement ERC-321253 (RoMoL), the European HiPEAC Network of Excellence, and the JSA no. 2013-119 as part of the IBM/BSC Technology Center for Supercomputing agreement.

REFERENCES

[1] B. Arimilli, R. Arimilli, V. Chung, S. Clark, W. Denzel, B. Drerup, T. Hoefler, J. Joyner, J. Lewis, J. Li *et al.*, "The PERCS high-performance interconnect," in *18th Symposium on High Performance Interconnects*. IEEE, 2010, pp. 75–82.

[2] R. Alverson, "Cray high speed networking," in *IEEE Hot Interconnects*, 2012.

[3] M. García, E. Vallejo, R. Beivide, M. Odriozola, and M. Valero, "Efficient routing mechanisms for dragonfly networks," in *The 42nd International Conference on Parallel Processing (ICPP-42)*, 2013.

[4] J. Kim, W. Dally, S. Scott, and D. Abts, "Technology-driven, highly-scalable dragonfly topology," in *ISCA'08: 35th International Symposium on Computer Architecture*. IEEE Computer Society, 2008, pp. 77–88.

[5] C. Camarero, E. Vallejo, and R. Beivide, "Topological characterization of hamming and dragonfly networks and its implications on routing," *ACM Trans. Archit. Code Optim.*, vol. 11, no. 4, pp. 39:1–39:25, 2014.

[6] M. García, E. Vallejo, R. Beivide, M. Odriozola, C. Camarero, M. Valero, J. Labarta, and G. Rodríguez, "Global misrouting policies in two-level hierarchical networks," in *INA-OCMC: Workshop on Interconnection Network Architecture: On-Chip, Multi-Chip*, 2013, pp. 13–16.

[7] N. Jiang, J. Kim, and W. J. Dally, "Indirect adaptive routing on large scale interconnection networks," in *Intl. Symp. on Computer Architecture (ISCA)*, 2009, pp. 220–231.

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

[9] L. Valiant, "A scheme for fast parallel communication," *SIAM journal on computing*, vol. 11, p. 350, 1982.

[10] 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*, Feb 2015, pp. 415–427.

[11] M. García, P. Fuentes, M. Odriozola, E. Vallejo, and R. Beivide. (2014) FOGSim Interconnection Network Simulator. University of Cantabria. [Online]. Available: https://code.google.com/p/fogsim/

[12] J. Kim, W. Dally, B. Towles, and A. Gupta, "Microarchitecture of a high-radix router," in *ACM SIGARCH Computer Architecture News*, vol. 33, no. 2. IEEE Computer Society, 2005, pp. 420–431.

[13] D. Abts and D. Weisser, "Age-based packet arbitration in large-radix k-ary n-cubes," in *Supercomputing, 2007. SC '07. Proceedings of the 2007 ACM/IEEE Conference on*, Nov 2007, pp. 1–11.