Bubble Flow Control in High-Radix Hierarchical Networks

Marina García[1],Enrique Vallejo[1],Ramon Beivide[1],Miguel Odriozola[1],Cristóbal Camarero [1], Mateo Valero[2], Germán Rodríguez[3], Jesús Labarta[2], and Cyriel Minkenberg[3]

Resumen— **Dragonfly networks have been recently proposed for the interconnection network of forthcoming exascale supercomputers.**

In this paper we introduce a novel routing/flow-control scheme that decouples the routing and the deadlock avoidance mechanisms in that family of interconnection networks. Our model does not impose any dependencies between virtual channels, allowing for on-the-fly (in-transit) adaptive routing of packets. To prevent deadlock we employ a deadlock-free escape subnetwork based on injection restriction.

I. Introduction

Interconnection networks constitute a key subsystem in the architecture of supercomputers. Direct network topologies are those that distribute routers among nodes. Most currently used direct networks are based on torus topologies [1]. Nevertheless, technology trends suggest the use of higher-degree routers to exploit the available pin bandwidth [8]. To this extent, two-layered hierarchical networks, denoted as Dragonflies in [9], have been proposed. This paper focuses on such networks.

Dragonflies are organized as groups of routers. Links between routers can be either *local* or *global*. Routers within a group are interconnected by means of a complete graph using *local* electrical wires. Groups are also interconnected by means of a complete graph, using one *global* optical link between any pair of groups.

The main Dragonfly topological parameters, as defined in [9], are the number of routers per group a, the number of processing nodes per router p and the number of global links per router h. For a well-balanced network with no oversubscription, the equations $a = 2 \times p = 2 \times h$ must hold, [9].

The diameter of the Dragonfly topology is 3, so any minimal path between two routers will employ at most 3 hops. With this minimal routing, a packet typically first traverses a local (l) link at the source group, then a global (g) one to reach the destination group, and finally another local link at the destination group. Since there is only one global link between any pair of groups, adversarial traffic patterns contending for global links can be common. In such demanding conditions, nonminimal routing can be used by randomly selecting an intermediate group to which the packet is sent before heading to its destination, [10], [9]. Under such nonminimal routing, a packet traverses at most 3 local links and 2 global ones.

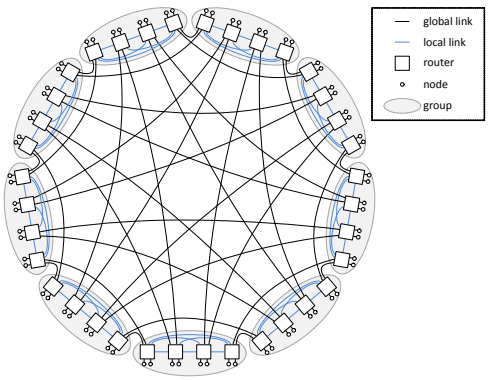


Fig. 1: Sample Dragonfly topology with h=2 (p=2, a=4), 36 routers and 72 compute nodes.

This traffic randomization balances the use of global links reducing contention, but doubles their average utilization halving throughput and increasing latency. Adaptive routing mechanisms select between minimal or nonminimal for each packet sent, depending on the conditions of the network.

The Dragonfly topology induces the appearance of cyclic routing dependencies, thus necessitating a deadlock avoidance strategy. Former proposals for routing in Dragonflies, [9], [2], rely on a set of virtual channels (VCs) that must be visited in a predefined ascending order. It is enough to implement 3 VCs in the inputs of local links and 2 VCs in global links.

In this paper we introduce **OFAR**: an **O**n the **F**ly **A**daptive **R**outing for Dragonfly networks. OFAR is a new flow-control/routing mechanism that decouples the way in which virtual channels are visited from the deadlock avoidance mechanism.

II. Related Work

Significant details about the Dragonfly architecture and routing can be found in [9], [6], [2]. As there is a single minimum path between any pair of groups, a load-balancing mechanism that distributes traffic along nonminimal routes is required when managing adversarial traffic.

Virtual channels regulated under a dateline policy, as proposed in [4], have been widely used for breaking cyclic dependencies in different ring-based networks. In Dragonflies, the mechanism used until now is also based on a restrictive use of VCs per each link.

This strict policy of ordering virtual channels does

[1]Universidad de Cantabria, Santander, Spain. First author email: garciamar@unican.es

[2]Universitat Politècnica de Catalunya and BSC Barcelona, Spain

[3]IBM Research GmbH Zurich Research Laboratory Rüschlikon, Switzerland

not allow misrouting an in-transit packet. The PAR mechanism proposed in [6] addresses this limitation by requiring an additional VC to prevent deadlock. Multiple mechanisms have been studied to dynamically misroute or not at injection time, [9], [6], [2].

Our proposal relies on a regulated-injection subnetwork to avoid packet deadlocks, [5].

III. Study of the saturation of local links

Until now, global links have been assumed as the only potential network bottleneck which would limit performance in a Dragonfly network [9], [2]. We will detail it next, to be able to compare this problem with the one in local links.

A Dragonfly network is dimensioned with as many processing nodes as outgoing global links in each group, $h = p$. As the transfer limit of each link is 1 phit/cycle and one global hop is required for each packet, this allows for maximum performance under uniform traffic and minimal routing. However, in a worst case the $2h^2$ nodes in one group could send traffic to the same destination group, competing for the bandwidth of a single global link. This would limit the maximum bandwidth to $1/(2h^2)$ when minimal routing is applied. With Valiant routing, each packet will be sent to a random intermediate group, and then minimally to its destination. Since this implies two global jumps, on average, global links will limit the maximum throughput to 1/2 phits/(node·cycle). We can observe that the initial problem is not the scarcity of global links (since $p = h$), but their unbalanced use under minimal routing, which is caused by the traffic pattern.

Analogously, local links also saturate when all the h compute nodes attached to a router send traffic to the nodes in a neighbor router of the same group. The single local link between these routers can transmit 1 phit/cycle, so the maximum traffic under minimal routing would be $1/h$ in this case. Valiant routing might rise this value to 50%, but it unnecessarily increases the use of global channels by sending the traffic to an intermediate group back and forth.

This effect should be, arguably, more frequent than the saturation of global links, since the applications typically try to exploit the locality between neighbor processes, and those neighbor processes are typically allocated sequentially in the same group.

Even with a nonminimal routing mechanism, there can be second-order congestion effects derived from the saturation of local links. As we will show next, there is a variable severity among different adversarial traffic patterns that, in some cases, provokes saturation on local links.

We consider patterns in which every source node in group i selects a destination node in group $i + N$, denoted as *ADV+N*, with N lower than the number of groups. We assume Valiant routing, in the general case with misrouting applied to an intermediate group different from the source and destination groups. As argued before, Valiant routing will limit the throughput to 0.5 phits/(node·cycle). However, for certain adversarial traffic patterns, the local link l_2 will saturate, even when this leaves global links partially idle.

We consider now the adversarial traffic pattern *ADV+h*. Figure 2a shows two routers R_i and R_o of a given group G_i. Lets consider the traffic misrouted towards group G_i which is received through the h global links entering to R_i. All these packets will have to be forwarded through the h subsequent global links. As global wiring is typically consecutive (observe the topology in Figure 1), all these links happen to be in the next router, R_o. Then, all misrouted traffic received in R_i has to be forwarded to R_o through the single local link connecting both routers. This link can only convey 1 phit/cycle, so even in absence of any other throughput limit in the network, the localized saturation of certain local links will limit throughput to $1/h$ phits/(node·cycle). This throughput limitation will grow with the network size h, which is important as the presented problem could pass unnoticed with small-size simulations.

The maximum throughput for nonminimal traffic will depend on the specific offset value between the source and the destination groups. Figure 2b shows how throughput notably varies depending on this offset between groups, even in a small Dragonfly with $h = 6$ under Valiant routing.

The simplest approach to avoid this saturation would be to allow for *local misrouting* of traffic, this is, diverting packets to a neighbor router to avoid a saturated local link. However, this should be allowed in any group which a packet traverses, leading to very long paths: $l-l-g-l-l-g-l-l$, or even longer if multiple nonminimal hops are allowed per group. Using the deadlock avoidance mechanisms in previous proposals, this would require 6 VCs in the local channels.

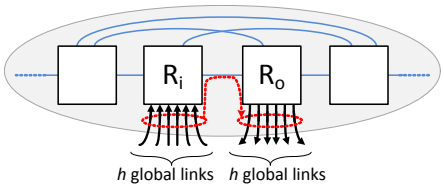
IV. *OFAR*: A New Flow-Control/Routing for Dragonflies

In this section we introduce our mechanism *OFAR*.

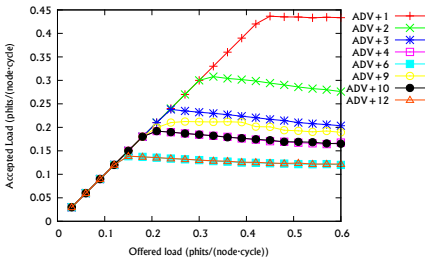
A. Dynamic misrouting in OFAR

In *OFAR* traffic can be sent non-minimally in each router to avoid network congestion. This misrouting can use local or global links of the current router. We restrict the number of times that this misrouting can be applied to prevent livelock: at most, one global non-minimal hop can be applied per packet, and one non-minimal local hop can be applied per group. We include two flags in the packet header to limit this misrouting.

When traffic is internal to a group, only local misroute is allowed; when traffic is external to the group, both local and global misroutes are allowed. Global misroute always occurs when the packet is still in its source group. Each packet in each input buffer always has a 'minimal output' according to its minimal output to the destination node. Depending on the credits of the minimal output and the header flags,



(a) Two routers in a group, joined with a single link.



(b) Accepted load vs. offered load in a $h = 6$ dragonfly with Valiant routing and different Adversarial Traffic Patterns

Fig. 2: Study of the throughput of adversarial traffic patterns

the control logic of the input unit of the router can try to misroute the packet to an output with more credits. In a group not being the source, only local misrouting is allowed.

We use the following policy to determine which port to use for misrouting in the source group. When the packet is still in the source group, the misroute type (local or global) will depend on the type of the input buffer that contains the packet. Those packets in injection queues (still in their source router)are misrouted by global channels. This saves the first local hop when using Valiant routing. By contrast, packets in local queues are first misrouted locally, and then globally. Although not obvious, this prevents starvation issues when the traffic pattern is adversarial.

Finally, in the evaluations of the next section we have included two *OFAR* models. The base *OFAR* model is the one described above. By contrast, the *OFAR-L* model does not allow for local misroute, same as the previously proposed routing mechanisms for the Dragonfly. This is used to dissect the specific benefits of using local misrouting in the routing mechanism.

B. *Contention-aware misrouting in* OFAR

Previous proposals select a random intermediate destination *before* determining if the packet should be sent minimally or not. By contrast, *OFAR* relies on the contention observed in the minimal (or Valiant) path to allow for non-minimal indirect routing. We assume an input-buffered router with a separable allocator. When a packet is in the header of an input queue, the routing subsystem will report which is its corresponding minimal (or Valiant) path, along with the allowed non-minimal paths (e.g., using local or global links, since all of them are isomorphic in the topology). Depending on the measured network congestion, the allocator input unit can request the minimal path or one of the non-minimal ones.

To decide if misrouting is applied, *OFAR* observes the occupancy, Q_{min}, of the queue in the minimal path (*minimal queue*), and the occupancy, $Q_{non-min}$, in any non-minimal output (*non-minimal queue*). As these queues have different sizes for local and global links, we consider the percentage of buffer occupancy rather than the actual occupancy in phits. To determine when misrouting is allowed, we use two thresholds: Th_{min} and $Th_{non-min}$. Specifically, misrouting is allowed only when $Q_{min} >= Th_{min}$ *and* the minimal port is not available (it is already assigned to another input or $Q_{min} = 100\%$). When misrouting is allowed, each input unit will request a random output port among those non-minimal ports that fulfil the occupancy condition $Q_{non-min} <= Th_{non-min}$.

C. *Deadlock-free subnetwork in* OFAR

The proposed *OFAR* routing can generate cyclic dependencies that block the network. Our proposal relies on the existence of a deadlock-free subnetwork added to the original network, [5]. We use a Hamiltonian ring with bubble flow control [3]. Packets are freely allowed to circulate in the escape ring, as long as there is space in the next buffer for the whole packet. However, when a packet is deflected to the escape ring from the canonical Dragonfly network, an extra free space for another packet is required (a bubble). In highly congested scenarios, some packets will enter this escape ring, partially increasing the length of their paths to destination. This escape subnetwork can be added either physically or virtually to the base topology. If we consider a physically added escape ring, it requires two additional ports per router and N additional wires on a N-router Dragonfly. Alternatively, a virtual embedded Hamiltonian ring maintains the same topology but requiring only an extra virtual channel in the corresponding links.

A packet could be inserted in the ring to prevent deadlock, and then return to the previous router using a minimal path. This is avoided by limiting the number of times that a packet can abandon the escape ring.

V. Methodology

We have implemented the different routing proposals on an in-house developed single-cycle simulator. We model an input FIFO buffered Virtual Cut-through (VCT) router, [7].

The routing decision for a packet is taken when it reaches the head of an input buffer. This selects be-

tween the preferred minimal output or one of those non-minimal outputs allowed by the misrouting policy and the misroute thresholds. We use 2 VCs per global link and 3 per local link and injection queues. These are the values required by previous mechanisms to avoid deadlock. VCs are not required to prevent deadlock in *OFAR* but we use them to reduce HOL blocking. We employ the same number of VCs for the escape ring for regularity, although the injection restriction already guarantees deadlock freedom.

We modeled a maximum size Dragonfly with $h = 6$. We use packets of 8 phits. The default network latencies are 10 cycles for local links and 100 cycles for global ones. Each local FIFO can store 32 phits, and 256 phits in the case of global FIFOs, enough for the flow control requirements dictated by round-trip latencies. The *OFAR* models employ a variable misroute threshold, $Th_{min} = 0\%$ and $Th_{non-min} = 0.9 \times Q_{min}$. Misrouting is allowed at any time if the minimal queue is not available, but only by those queues that have less than 0.9 times the occupancy of the minimal queue. The selection of this policy was empirical.

We employ synthetic traffic to evaluate performance. The destination node is selected depending on the traffic model: Uniform (*UN*) and Adversarial+*N* (*ADV+N*).

We have implemented the following routing mechanisms: Minimal (*MIN*), Valiant (*VAL*), Piggybacking (*PB*, The injection router selects between minimal and non-minimal paths based on remote congestion information broadcasted among all the routers of each group, [6]), *OFAR* and *OFAR-L*.

VI. Performance results

We measured the network performance in three different scenarios: steady state, transient variations and traffic bursts. We detail each of these cases next.

A. Steady state

On these tests we measure the average latency and throughput over a long period, after a sufficient network warm-up. Each point in the plots shows the measured value for a given offered load in phits/(node·cycle).

Figure 3a shows the average latency under uniform random traffic (*UN*). Using *MIN* as a reference, we observe that *OFAR* models provide a competitive latency under low loads, but they saturate significantly later. The latency of the adaptive mechanism *PB*, by contrast, is significantly larger, due to a higher number of misrouted packets. Figure 3b reports throughput. The *OFAR* models improve over *MIN* and *PB*, but in either case, the use of local misrouting does not make a significant difference.

Figure 4 shows results under adversarial traffic *ADV+2*. In this traffic pattern, the reference is *VAL*, which always misroutes traffic, instead of *MIN*, which suffers from strong congestion caused by the saturated global links.

We can observe that the *OFAR* model shows very competitive latency values. Regarding throughput, Figure 4b shows that the *OFAR* saturates at 0.45, when comparatively, *PB* saturates around 0.38. The difference comes mainly from the better performance of in-transit adaptive routing decisions with larger path diversity in *OFAR*, rather than the use of delayed information about congestion in global queues and the evaluation of a single nonminimal alternative in *PB*. Under this traffic pattern, we can observe how the complete *OFAR* model achieves better performance (especially in terms of throughput) than the *OFAR-L* model, but the difference is very low.

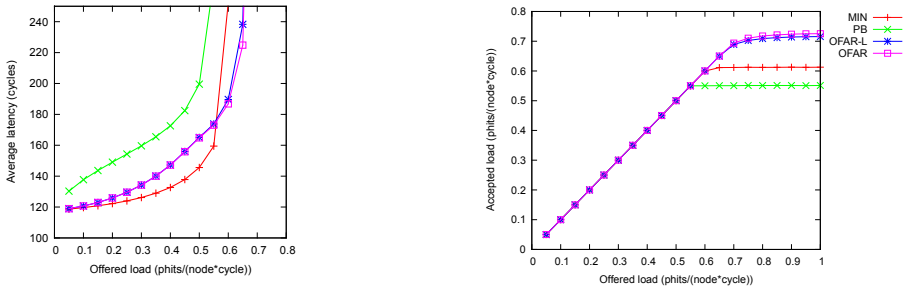
Finally, we evaluated network performance under the worst traffic pattern, *ADV+6*. It is presented in Figure 5. In this case, misrouted traffic can generate the largest congestion in local links as described in Subsection III. If this is not avoided, throughput would be limited to $1/h = 1/6 = 0,166$ phits/(node·cycle). Figure 5a shows that this occurs for *VAL, PB* and *OFAR-L*. *OFAR* obtains, by far, the best result. Throughput, reported in Figure 5b, confirms the significant performance difference between the base and *"-L"* models. The troughput of *OFAR* is limited to 0.36, closer to the theoretical limit of 0.5 imposed by global channels, than the 0,166 which would be imposed by the local ones without local misroute.

B. Transient traffic

These measures explore the response time when the traffic pattern changes. We warm-up the network with a given traffic pattern. Once it reaches the steady state, we change the traffic pattern, and observe how each mechanism adapts to the change. We measure the average latency of the packets that are *sent* each cycle. This is, when a packet is received, we account for the latency in the cycle that it was sent. We used *OFAR*, *OFAR-L* and *PB*, in three different transient cases: UN to ADV+2; ADV+2 to UN and ADV+2 to ADV+6 (ADV+h). We apply a load of 0.14 phits/(node·cycle), except for the last case (ADV2 to ADV6) which would saturate the network using *PB*; we use 0.12 in that case. Figure 6 shows that for the transition of ADV+2 to UN , all the mechanisms converge very fast, since they suddenly find the required links un-congested. By contrast, in the other two cases *OFAR* makes the transition almost instantaneous, while *PB* suffers from an adaptation period.

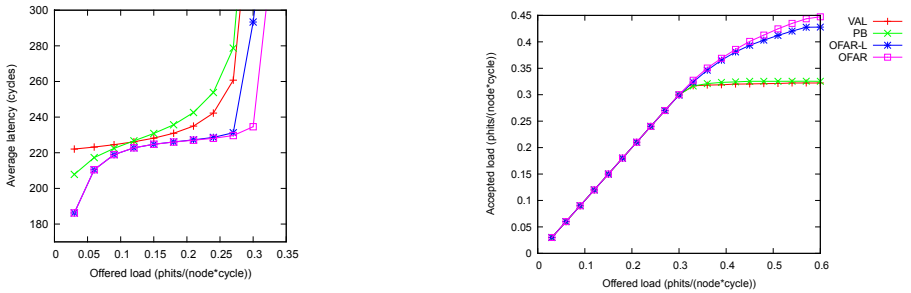
C. Traffic bursts

In parallel programs, communication and computation phases are typically synchronized, so traffic bursts after barriers are common. We simulate this using packet bursts. Each node injects a fixed amount of packets (2.000) as fast as possible, with a mixture of different traffic patterns. With $h = 6$, this figure corresponds to around a million packets received. We measure the time to consume all the packets in the network. The destination of each



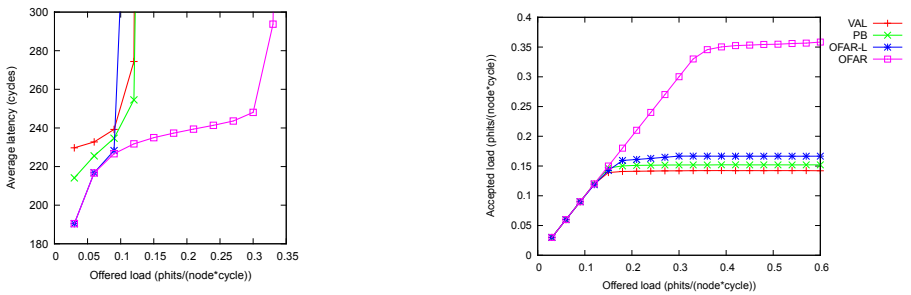
(a) Latency. (b) Throughput.

Fig. 3: Latency and throughput under random uniform traffic (UN).



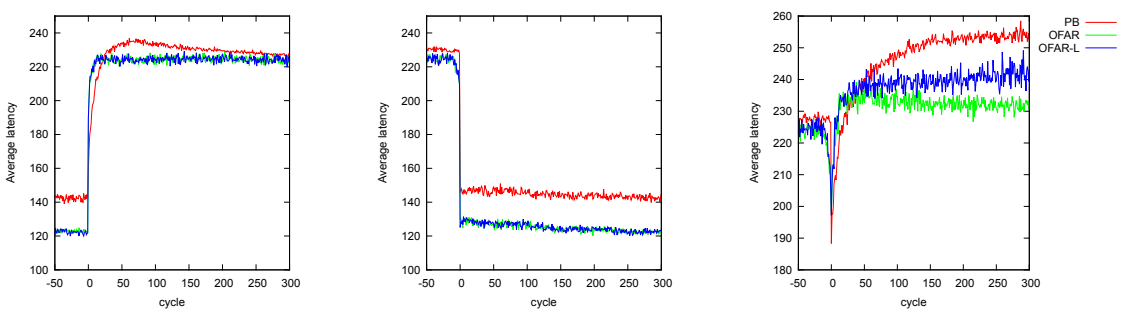
(a) Latency. (b) Throughput.

Fig. 4: Latency and throughput under adversarial +2 traffic (ADV+2).



(a) Latency. (b) Throughput.

Fig. 5: Latency and throughput under adversarial +6 traffic (ADV+6).



(a) UN to ADV+2 traffic (b) ADV+2 to UN traffic (c) ADV+2 to ADV+6 traffic, load=0.12

Fig. 6: Latency evolution under transient traffic.

packet is variable according to a certain distribution. We have simulated *UN*, *ADV+2*, *ADV+6* and three mixes of traffic with different rates of uniform and adversarial: In *MIX1* 80% of the traffic is UN, 10% is *ADV+1* and 10% is *ADV+6*. In *MIX2* the rates are 60-20-20 and in *MIX3* they are 20-40-40.

Figure 7 shows the execution time normalized to the result of *PB* on each case. The *OFAR* mechanisms always finish faster. Compared to *PB*, the execution time of *OFAR* ranges from a 43.1% to a

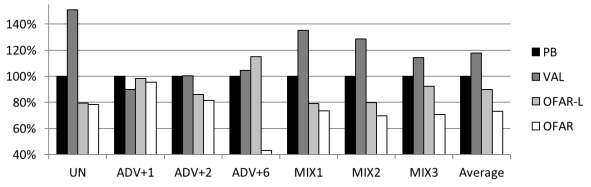


Fig. 7: Burst consumption time, normalized to *PB*. Lower is better.

81.5%. On average, the time to consume traffic for *OFAR* is 0.695 the time for *PB*, which corresponds to a speedup of 43.8%. It is noticeable that the complete *OFAR* model always finishes faster than their -L counterparts.

VII. Discussion

All the previous evaluations have been performed using a Hamiltonian physical ring and the same number of VCs as required by previous mechanisms. In this Section we will address issues related to cost and reliability of *OFAR*.

The additional cost of an *OFAR* implementation based on a physical ring is easy to compute. A rough calculation shows that the proportion of added links is in the order of $2/3h$; with $h = 16$, this means 4% more wires. Nevertheless, the cost of these networks is mainly dominated by long wires. In *OFAR*, $2h^2+1$ are added to the $2h^4+h^2$ original long wires. With $h = 16$, this accounts for only 0,3% more global wires. In addition, two router ports are needed to implement the ring.

In spite of this low cost, cheaper solutions can be envisaged. For example, instead of adding a physical Hamiltonian ring, it can be virtually embedded on the original topology, connecting consecutive routers. This implementation has no added cost in terms of wires. We just use an additional virtual channel in the links that constitute the Hamiltonian embedded ring. Some simulations have been done to compare the behavior of *OFAR* with an embedded ring and with a physical one. The results show that no significant differences can be reported from the use of the physical or embedded link. This is coherent with the idea that the escape subnetwork is not used to route traffic, but only to resolve potential deadlock situations.

The previous results have shown that, even under high loads, throughput remains constant after saturation even with an embedded ring. However, in our model, the capacity of the escape network (the Hamiltonian ring) is much lower than the capacity of the canonical network (the Dragonfly). This might lead to network congestion if all the buffers of the canonical network were completely full, and only the escape ring was used to deliver packets at destination. To verify if congestion could happen, we simulated *OFAR* with less resources: an embedded ring, and only 2 VCs for local links and 1 for global ones, without any congestion management. We observed that, in some cases, throughput significantly falls as the canonical network gets completely congested. A proper congestion control mechanism should be considered to guarantee that this case never happens in practice for a given configuration of network resources.

With respect to reliability, *OFAR* could block the system with more than a single failure in its Hamiltonian ring. This issue could be addressed by using several disjoint hamiltonian rings.

VIII. Conclusions

This paper has introduced *OFAR*, a flow-control/routing mechanism that addresses some of the main performance limitations of Dragonfly topologies, namely the saturation of local links and the poor efficiency of the misrouting decision process in existing mechanisms. We have presented an efficient alternative that allows for flexible on the fly misrouting of packets.

We have studied a misrouting policy for for *OFAR* which balances the traffic among the global links of the group, and obtains good values for latency, burst consumption and response time for transient traffic.

Ongoing work includes the use of congestion avoidance mechanisms and the design of alternative router architectures and escape subnetworks.

Acknowledgements

This work has been supported by the Spanish Ministerio de Ciencia e Innovación, under project TIN2010-21291-C02-02, The HiPEAC Network of Excelence and the Consolider Project “Supercomputación y e-Ciencia”

Referencias

[1] http://www.top500.org.
[2] B. Arimilli, R. Arimilli, V. Chung, S. Clark, W. Denzel, B. Drerup, T. Hoefler, J. Joyner, J. Lewis, J. Li, N. Ni and R. Rajamony. The PERCS High-Performance Interconnect. IEEE Symposium on High Performance Interconnects. pp. 75-82. 2010.
[3] C. Carrión, R. Beivide, J.A. Gregorio and F. Vallejo. A Flow Control Mechanism to Prevent Message Deadlock in k-ary n-cube Networks. IEEE/ACM International Conference on High Performance Computing (HiPC’97), pp. 322-329. 1997.
[4] W. J. Dally. Virtual-Channel Flow Control. IEEE Transactions on Parallel and Distributed Systems, Vol. 3, No. 2, pp.194-205, 1992.
[5] J. Duato. A New Theory of Deadlock-Free Adaptive Routing in Wormhole Networks. IEEE Trans. Parallel Distrib. Syst., IEEE Press, Vol. 4, pp. 1320-1331, 1993.
[6] N. Jiang, J. Kim, amd W.J. Dally. Indirect Adaptive Routing on Large Scale Interconnection Networks. Intl Symposium on Computer Architecture (ISCA’09), Austin, USA, 2009.
[7] P. Kermani and L. Kleinrock. Virtual Cut-through: A New Computer Communication Switching Technique. Computer Networks, vol. 3, pp. 267-286. 1979.
[8] J. Kim, W. J. Dally, B. Towles, and A. K. Gupta. Microarchitecture of a High-Radix Router. Intl. Symposium on Computer Architecture, (ISCA’05), pp 420-431. 2005.
[9] J. Kim, W. J. Dally, S. Scott, and D. Abts. Technology-driven, highly-scalable dragonfly network. Intl. Symposium on Computer Architecture (ISCA’08), Beijing, China, 2008.
[10] L. G. Valiant, A Scheme for Fast Parallel Communication. SIAM J. on Computing, 1982, Vol. 11, pp. 350-361