Performance implications of remote-only load balancing under adversarial traffic in Dragonflies

Bogdan Prisacari
IBM Research – Zurich
Saumerstrasse 4
8803 Ruschlikon, Switzerland
bpr@zurich.ibm.com

German Rodriguez
IBM Research – Zurich
Saumerstrasse 4
8803 Ruschlikon, Switzerland
rod@zurich.ibm.com

Marina Garcia
IBM Research – Zurich
Saumerstrasse 4
8803 Ruschlikon, Switzerland
mgg@zurich.ibm.com

Enrique Vallejo
University of Cantabria
Avenida de los Castros, s/n
39005, Santander, Spain
enrique.vallejo@unican.es

Ramon Beivide
University of Cantabria
Avenida de los Castros, s/n
39005, Santander, Spain
beividej@unican.es

Cyriel Minkenberg
IBM Research – Zurich
Saumerstrasse 4
8803 Ruschlikon, Switzerland
sil@zurich.ibm.com

ABSTRACT

Dragonfly topologies are recent network designs that are considered one of the most promising interconnect options for Exascale systems. They offer a low diameter and low network cost, but do so at the expense of path diversity, which makes them vulnerable to certain adversarial traffic patterns. Indirect routing approaches can alleviate the performance degradation that these workloads experience. However, there are limits to the improvements that can be achieved using the indirect routing approach that is popular today, limits that are inherent to the Dragonfly topological structure. In this work, we explore these limits by providing a theoretical justification to why adversarial traffic patterns routed indirectly with an algorithm that perfectly distributes load across inter-Dragonfly-group links can still induce significant bottlenecks in the intra-group links. We equally provide estimations of the performance impact of these imbalances, as well as present a set of simulation based benchmarks that confirm the theoretical predictions for practical Dragonfly systems.

Categories and Subject Descriptors

C.2.1 [**Computer-Communication Networks**]: Network Architecture and Design; C.4 [**Performance of Systems**]

General Terms

Theory, Performance, Experimentation

Keywords

Dragonfly networks, Network throughput, Adversarial traffic, Indirect routing, Load balance

INA-OCMC '14, January 22 2014, Vienna, Austria

1. INTRODUCTION

Dragonflies [9] are being considered today as one of the most promising topologies to build the highly-scalable high-bandwidth low-latency interconnects needed for Exascale computing [3]. Dragonfly topologies simultaneously offer low diameter, high performance for uniform random traffic, a hierarchical design that allows for a very advantageous performance/cost ratio, scalability as well as many other advantages. Several systems today are based on variants of dragonfly topologies, such as IBM's PERCS High-Performance Interconnect [2], or Cray's Cascade Interconnect [6].

Despite these advantages, several traffic patterns characteristic of relevant HPC workloads have been identified that pose significant challenges to these interconnects [4, 7, 8, 1]. Several approaches to alleviate the performance impact of these workloads have been proposed, based on either adapting the routing strategy [9, 7], the topology [7] or the task placement [4]. All these proposal randomize traffic in the network in an attempt to break down the regularity of the adversarial traffic patterns and consequently induce a behavior, from the network perspective, similar to that of uniform random traffic, which in Dragonflies exhibits peak performance. However, some the most popular of these approaches fail to reach this objective.

In this work we will present an analysis of the non-trivial bottlenecks that lead to this sub-optimal performance and provide a model enabling more accurate predictions. Such bottlenecks had been identified in previous work [7] for (adversarial) linear shift patterns and an adaptive-routing-based solution proposed, but a thorough analysis of the source of this bottleneck had not been presented. In this work we will provide a performance model for these bottlenecks as well as exemplify them with a different traffic pattern, bit complement, that, as the linear shift patterns, is also severely affected by them. We will experimentally validate this model by showing how the predictions it generates compare against simulation results of practical network configurations.

2. BACKGROUND

A Dragonfly is a two-level hierarchical network. At the first level, a certain number of low-radix switches form a group that behaves like a virtual high-radix switch. These

low-radix switches are typically interconnected to form a fully-connected mesh (e.g., in the PERCS interconnect) or some other low-radix topology (e.g. a flattened butterfly, such as in the Cray Cascade interconnect). The virtual high-radix switches are interconnected to form another fully-connected graph of groups [9] at the second level. The ports that the virtual switches use to connect to the other virtual switches are distributed across the low-radix real switches that make up the virtual switch. In the remainder of this work we will focus on dragonflies where the intra-group topology is a fully-connected mesh.

In this case, a Dragonfly network can be described by three parameters: p, the number of nodes connected to each switch, a, the number of switches in each first level group, and h, the number of ports that each switch uses to connect to switches in other groups. For certain parameter values, it can be shown that ideal throughput can be achieved for uniform traffic under minimal routing.

In a dragonfly, minimal or shortest paths between pairs of nodes are unique. The longest possible shortest path is made up of a traversal of a *local* intra-group (L) link in the group of the source node to get to the switch that has the *global* or *remote* (R) link towards the destination group, a traversal of that remote link and a second *local* link traversal in the destination group to get to the switch directly connected to the destination node.

This lack of shortest path diversity can lead to an extreme degradation in performance for certain adversarial traffic patterns. One option to alleviate this degradation is to use Valiant's algorithm [11]. This algorithm routes a packet to a randomly chosen intermediate switch first, before routing it to the actual destination. The expectation is that, by using a different random intermediate switch for each packet, the original nature of the traffic is shifted towards a uniform random traffic at the expense of longer paths and of doubling the load under originally random traffic [5]. In addition to the increased latency they induce, the longer paths in Valiant routing also require the use of additional virtual channels to guarantee deadlock freedom. In particular, the Valiant routing variant for dragonflies as described in [9], which we will call *Valiant [Kim:2008]*, requires 3 virtual channels (instead of 2 in the case of minimal routing) for the L links and 2 virtual channels (instead of 1 for minimal routing) for the R links. *Valiant [Kim:2008]* can be described as follows: when a source s in group S sends a message to destination d in group D, an intermediate misroute group I is chosen. A minimal route (consisting of at most one L and one R hop) is taken to arrive to the first reachable switch in group I. From there, the packet follows the unique minimal route to the destination d (requiring at most two L hops and one R hop). The longest path using this Valiant variant would visit the following link types: *LR-LRL*.

Another variant of Valiant routing could choose any switch (not just the first reachable) in the intermediate group as the intermediate misroute destination, thus potentially incurring an extra L hop within the intermediate group, and requiring one extra virtual channel for the L channels, with the longest paths possible being of the type: *LRL-LRL*. This routing variant might negatively affect some traffic patterns (due to an increased load on the L links) while helping other patterns (providing a route around a congested intermediate group L link). This option has not been thoroughly studied in the literature because of the cost of the extra virtual channel. We will call this routing *Valiant Any*.

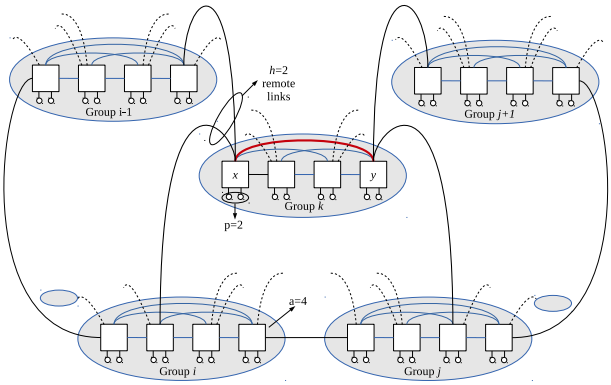

Figure 1: Illustration of the L link bottleneck. When group i sends to group j and group $i-1$ sends to group $j+1$, under indirect *Valiant [Kim:2008]* routing, and due to the typical wiring of a dragonfly, group k experiences a load on an L link that is a factor of h higher than would be expected, because with *Valiant [Kim:2008]*, once the intermediate group is reached, the minimal path will be taken towards groups j and $j-1$, making the traversal of the single L link unavoidable.

3. ADVERSARIAL LOAD DISTRIBUTION

Dragonfly networks have been shown to be especially well suited for efficiently accommodating workloads with a uniform traffic matrix. Other types of workloads however can experience significant performance degradation due to there being a single link connecting pairs of groups. Such adversarial traffic patterns include the bit complement pattern and certain shift permutation patterns [5]. In general, a workload will be adversarial for the Dragonfly under direct routing if the traffic matrix is such that the traffic issued by all sources in each group is predominantly destined to destinations that all belong to the same other group. Indeed, such a pattern will induce a large amount of traffic on the link connecting the source-destination group pairs causing it to become a bottleneck and limit overall throughput. The solutions proposed in literature to improve performance in this case usually rely on spreading the focused remote link load throughout the network by using either indirect routing [9, 7] or randomized task placement [4]. We will show that for a large subcategory of these adversarial patterns, including the examples we gave earlier, the indirect routing approach that is currently the most popular in practice, *Valiant [Kim:2008]*, is unable to increase performance significantly. By selecting an intermediate group uniformly at random to which to misroute the traffic, *Valiant [Kim:2008]* is indeed able to avert the bottleneck described earlier, by distributing the load previously experienced by a single remote link across all remote links . However, only the shortest path to the intermediate group is considered, causing the load-balancing to be done only for the remote links. Such a remote-only approach is unable to prevent the subsequent formation of bottlenecks in the intra-group links in the pres-

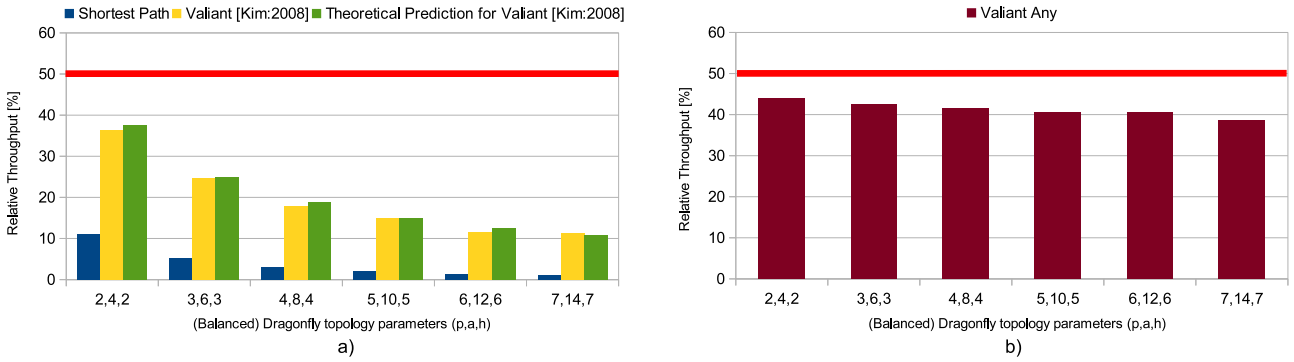

Figure 2: Measured and predicted throughput for a bit complement workload running on balanced dragonflies of increasing size. Fig. a). shows performance under *Valiant [Kim:2008]* indirect routing, along with the local imbalance aware estimation we introduced in this work while Fig. b) shows the performance under the locally balanced *Valiant Any* indirect routing. The load-imbalance-oblivious theoretical maximum is shown as a red line.

ence of adversarial traffic patterns.

In practice, full scale Dragonfly networks are interconnected as follows. Given a Dragonfly network characterized by the (p, a, h) parameters [9], there is a numbering of remote links on every switch from 0 to $h-1$, a numbering of switches in each group from 0 to $a-1$ and a numbering of groups from 0 to $a*h$ such that switch number x in group i connects through its m-th remote link to a switch in group $j = (i + x \cdot h + m) \bmod (ah + 1)$. This leads to consecutive remote links interconnecting a given group to groups that are also consecutive. We will show that, if the adversarial traffic pattern is such that the pairs of communicating groups have the property that consecutive source groups are paired with consecutive destination groups, as it is the case for both bit complement and shift permutation patterns, then intra-group bottlenecks are unavoidable under *Valiant [Kim:2008]* indirect routing.

Let us consider an arbitrary switch x of an intermediate group k and also consider that we are using *Valiant [Kim:2008]* indirect routing. The h remote links of x will then connect it to groups for which traffic originating there that uses k as an intermediate group will be routed minimally to x and from there minimally to the respective destinations (as explained in Sec. 2). Due to the interconnection pattern described above, these h groups will be consecutive, and, due to the assumptions we've made on the traffic pattern, the destination groups of traffic originating in them will also be consecutive. This implies that the h destination groups will be connected to group k via consecutive links, i.e., by links m to $h-1$ on some switch y and links 0 to $m-1$ on switch $z = (y+1) \bmod a$, for some values of m and y that are specific to the traffic pattern. This further implies that traffic incoming on the h remote links of x will be distributed across at most only 2 intra-group links, i.e. the links connecting x to y and x to z. All other intra-group links originating on x will be completely unused by indirect traffic, leading to a significant intra-group imbalance. Fig. 1 illustrates this effect for the bit complement pattern.

This imbalance has an important performance impact. Indeed, the traffic incoming over m of x's remote links and the traffic incoming over the remainder $h-m$ of x's remote links will each effectively be limited to the throughput of a single intra-group link, leading to a throughput limitation of each remote link that is between $1/h$ and $2/h$ of the total link bandwidth, depending on the value of m. As mentioned above, for every intermediate switch x, m takes a value that is pattern specific, and given a traffic matrix, it can be computed explicitly. The cumulative effect across all switches x is generally well approximated by an average R link load limitation equal to the average of these extreme values, i.e., $3/(2h)$.

Given that the path taken by every message under indirect routing traverses two remote links, it can be shown fairly straightforward that the maximum aggregate injection throughput that the network can sustain is upper limited by half the aggregate throughput of all remote links. It follows that the relative performance of the network is given by:

$$T = \frac{(ah+1) \cdot a \cdot h}{(ah+1) \cdot a \cdot p} \cdot \frac{3}{2h} \cdot \frac{1}{2} = \frac{3}{4p}, \qquad (1)$$

where T is the relative (to total injection bandwidth) throughput the network can sustain.

4. EXPERIMENTAL RESULTS

In this section we will experimentally measure (via simulation) the performance of bit complement traffic on a set of dragonfly topologies and compare these results to the theoretical estimates derived in Sec. 3.

4.1 Framework, parameters and metrics

The results presented in this section were obtained using a simulation framework [10] that is able to accurately model and measure generic and custom networks, including dragonflies, at a flit level. The switch architecture chosen was that of an input-output-buffered switch with 4 Kbytes of buffer space per port per direction per virtual channel. The links had a bandwidth of 40 Gbit/second. Credit based flow control was used and the exchanged messages consisted of a single 64 byte flit. The routing algorithms used were minimal, *Valiant [Kim:2008]* and *Valiant Any* routing with virtual channel based deadlock avoidance.

We benchmarked bit complement traffic, where each task t (where $0 \leq t < T$, T being the total number of tasks) selects

as the destination of every message task $T-t-1$. Each dragonfly node was assigned a single task and every such task sent messages at maximum rate (equal to the bandwidth of the link connecting the node to its corresponding switch). The assignment of tasks to nodes was performed in a contiguous fashion, i.e., every task was assigned to the node that has a node index equal to the task's index, where the nodes are indexed topologically: the nodes are numbered consecutively (starting with 0) one group at a time, and within a given group, one switch at a time.

For every experiment, the system was simulated for a fixed amount of time (10 milliseconds) that was chosen such that, in all cases, the throughput was estimated within a confidence interval of 1%.

4.2 Results

Figure 2 shows the relative throughput achieved by the bit complement traffic under both variants of Valiant routing. We can see that for the *Valiant [Kim:2008]* approach, indirect routing is unable to randomize traffic such that it exhibits uniform random like behavior (shown as the red horizontal line in the figure). Indeed, we can see that instead we obtain the performance predicted by the analysis presented in Sec. 3, due to the local bottlenecks shifting to the L links and routing being unable to distribute the load within the intermediate group (as explained in detail in the same section).

On the other hand, we can see in the same figure that the *Valiant Any* approach, by effectively balancing not only remote traffic, but also local traffic in the indirect groups, is able to approximate much better a uniform traffic scenario. Indeed, whereas perfectly uniform traffic is expected to achieve a 50% relative throughput, bit complement traffic routed with *Valiant Any* achieves between $39\% - 42\%$ while the same traffic routed with *Valiant [Kim:2008]* achieves progressively worse performance as the system size grows, dropping below 15% throughput for networks with more than 256 switches.

5. CONCLUSIONS

In this work we have provided a detailed analysis of bottlenecks induced by adversarial communication patterns on the local links of Dragonfly networks under a specific Valiant routing approach. *Valiant [Kim:2008]* requires less resources than other Valiant routings and, in addition, leads to shorter end-to-end indirect paths. However, we have shown that, on-par with observations from other works, the performance obtained for *Valiant [Kim:2008]* routing is not equal to the equivalent performance of indirectly routed uniform random traffic. Indeed, for certain adversarial patterns for dragonflies, it had been observed that *Valiant [Kim:2008]* is prone to creating severe intra-group load imbalances that induce a significant degradation of the expected performance, degradation that increases with the scale of the network. In this work we provide a theoretical performance model that accurately predicts this behavior.

In support of these theoretical results, we equally provided simulation-based experimental measurements that showed our conclusions to hold in practical system configurations. Finally, by means of similar experiments, we also showed that an alternative indirect routing approach, *Valiant Any*, requiring one extra virtual channel on each link and increasing by one the length of indirect paths, is able to better balance load both inter and intra-group, significantly improving the similarity of the resulting traffic to uniform random traffic, and consequently improving performance.

6. REFERENCES

[1] M. Alvanos, G. Tanase, M. Farreras, E. Tiotto, J. N. Amaral, and X. Martorell. Improving performance of all-to-all communication through loop scheduling in pgas environments. In *Proceedings of the 27th international ACM conference on International conference on supercomputing*, ICS '13, pages 457–458, New York, NY, USA, 2013. ACM.

[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. In *Proceedings of the 2010 18th IEEE Symposium on High Performance Interconnects*, HOTI '10, pages 75–82, Washington, DC, USA, 2010. IEEE Computer Society.

[3] K. Bergman and et al. Exascale computing study: Technology challenges in achieving exascale systems, 2008.

[4] A. Bhatele, N. Jain, W. D. Gropp, and L. V. Kale. Avoiding hot-spots on two-level direct networks. In *Proceedings of 2011 International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '11, pages 76:1–76:11, New York, NY, USA, 2011. ACM.

[5] W. Dally and B. Towles. *Principles and Practices of Interconnection Networks*. Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 2003.

[6] G. Faanes and 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*, SC '12, pages 103:1–103:9, Los Alamitos, CA, USA, 2012. IEEE Computer Society Press.

[7] M. García, E. Vallejo, R. Beivide, M. Odriozola, C. Camarero, M. Valero, G. Rodríguez, J. Labarta, and C. Minkenberg. On-the-fly adaptive routing in high-radix hierarchical networks. In *The 41st International Conference on Parallel Processing (ICPP)*, 09 2012.

[8] A. Jokanovic, B. Prisacari, G. Rodriguez, and C. Minkenberg. Randomizing task placement does not randomize traffic (enough). In *Proceedings of the 2013 Interconnection Network Architecture: On-Chip, Multi-Chip*, IMA-OCMC '13, pages 9–12, New York, NY, USA, 2013. ACM.

[9] J. Kim, W. J. Dally, S. Scott, and D. Abts. Technology-driven, highly-scalable dragonfly topology. *SIGARCH Comput. Archit. News*, 36(3):77–88, June 2008.

[10] C. Minkenberg, W. Denzel, G. Rodriguez, and R. Birke. End-to-end modeling and simulation of high-performance computing systems. *Springer Proceedings in Physics: Use Cases of Discrete Event Simulation: Appliance and Research*, page 201, 2012.

[11] L. G. Valiant. A scheme for fast parallel communication. *SIAM J. Comput.*, 11(2):350–361, 1982.