Polarized Routing for Large Interconnection Networks

postprint version

C. Camarero, C. Martínez and R. Beivide, ”Polarized Routing for Large Interconnection Networks,” in IEEE Micro, vol. 42, no. 2, pp. 61-67, 1 March-April 2022, doi: 10.1109/MM.2021.3139392.

https://doi.org/10.1109/MM.2021.3139392

Cristóbal Camarero, Carmen Martínez, and Ramón Beivide
Computer and Electronics Department. University of Cantabria. Spain

Abstract

Supercomputers and datacenters comprise hundreds of thousands of servers. Different network topologies have been proposed to attain such a high scalability, from Flattened Butterfly and Dragonfly to the most disruptive Jellyfish, which is based on a random graph. The routing problem on such networks remains a challenge that can be tackled either as a topology aware solution or with an agnostic approach. The case of random networks is a very special one since no a priori topological clues can be exploited. In this paper, we introduce the Polarized Routing algorithm, an adaptive non-minimal hop-by-hop mechanism that can be used in most of topologies, including Jellyfish. Polarized routing follows two design criteria: a source–destination symmetry in the routes and avoiding backtracking. Experimental evaluation proves that Polarized not only outperforms other routings in random graphs but also attains the best performance provided by *ad hoc* solutions for specific outstanding low-diameter interconnection networks.

Nowadays, the computing industry has moved to the cloud. Modern cloud datacenters and their forerunners, the fastest supercomputers, are the largest systems currently deployed. They comprise millions of processing cores linked by an interconnection hierarchy composed of many networks on chip (NoCs), one per computing chip, and a single system network. A good system network for such huge systems must supply very large amounts of bandwidth and exhibit very low message latencies. Two critical features that comprise the performance of a network are its topology and routing, the two being highly independent as a good topology is totally useless without a proper routing. This paper introduces Polarized routing, an efficient incremental adaptive non-minimal routing algorithm able to run over any practical large direct topology. Polarized exhibits higher network performance than previous routing mechanism of comparable cost.

Network Topology

Networks are based on switches, which are the devices that guide the flow of data. Typically, all the switches in a network have the same number of ports or *radix*. This is the case of the topologies depicted in Figure 1, in which switches are represented by squares and servers by circles. In a *direct* network, used in high-end supercomputers, all the switches attach servers that inject and receive data, while in an *indirect* one, used in datacenters and HPC (High

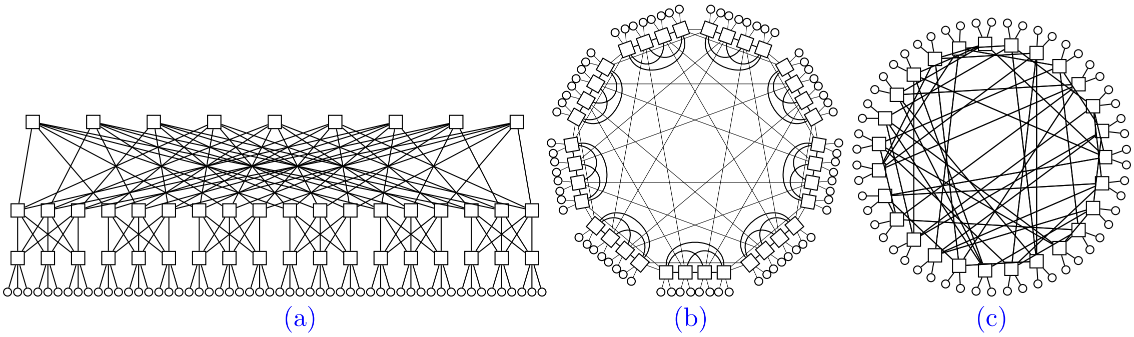

Figure 1: (a) 54-server, 6-radix, 3-level Fat-tree. (b) 72-server, 7-radix Dragonfly. (c) 54-server, 6-radix random regular graph.

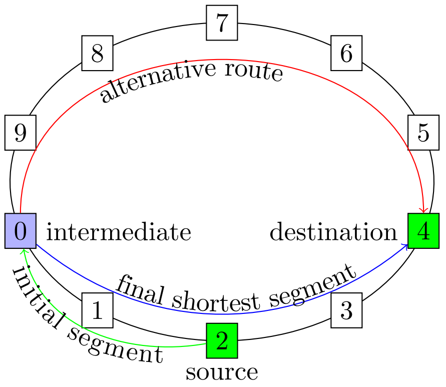

Figure 2: Backtracking in a ring using Valiant, when source $s = 2$, $r = 0$ and $t = 4$ in a ring.

Performance Computing) clusters, there are transit switches that only connect other switches. Figure 1(a) shows an indirect Fat-tree network and Figures 1(b) and 1(c) depict two direct networks: Dragonfly and Jellyfish.

Any direct network can be modelled by a graph, $G = (N, E)$, in which the set of vertices N represents switches and the set of edges E models bidirectional links that connect switches to each other. G is a δ-regular graph if every vertex connects to other δ vertices. The distance between two vertices s and t, $D(s, t)$, is the length, measured in traversed links, of a shortest path between s and t. The diameter, D, is the maximum distance between any pair of vertices.

At large scales, when using high-radix switches, it has been shown that direct low-diameter topologies, such as Dragonfly [6], HyperX [1] and SlimFly [2], maximize performance at a given cost. These are structured topologies typically organized into either a hierarchy or a multi-dimensional space. Nevertheless, it has been known for some time that a random regular graph (RRG) [3] can be as good as or even better than such structured graphs. Jellyfish, the direct network proposed in [7] selects a randomly chosen graph among all the regular graphs with the same number of nodes and degree. The RRG represented in Figure 1(c) connects $N = 27$ switches, with diameter $D = 3$, using degree $\delta = 4$. Every switch has radix $R = 6$ and $\delta_0 = 2$ servers attached for a total of $S = 54$ servers. It is known that a δ-regular random graph with N vertices of diameter D can be obtained easily if $\delta^D \approx 2N \ln N$ [3]; thus, $\delta \approx (2N \ln N)^{1/D}$.

Polarized Routing

Valiant routing [4], and therefore UGAL [3], suffer from a subtle but important miss behavior that we denote as “undoing a hop”. An example of this can be seen in the ring of length $N = 10$ represented in Figure 2. If source switch $s = 2$ has to communicate with target switch $t = 4$, using intermediate switch $r = 0$, as in Valiant, this implies sending from $s = 2$ to $r = 0$ using the 2-length minimum path. However, from $r = 0$ to $t = 4$, the 4-length minimum path consists of “undoing” the two previous hops plus the minimum path from s to t. Notice that there is an alternative path avoiding that in Figure 2.

In Polarized routing, all hops must improve a weight function, and therefore, they can never “undo” a previous hop. To do that, consider a path between source switch s and target switch t when passing through switch c, and define the following weight function:

$$\mu_{s,t}(c) = D(c, s) - D(c, t), \tag{1}$$

A route $s, c_1, c_2, \ldots, c_n, t$ is said to be a *Polarized route* if $\mu_{s,t}(c_i) \leq \mu_{s,t}(c_{i+1})$ for every intermediate switch. That is, no hop decrements the weight function μ, or equivalently, the difference operator $\Delta\mu = \mu_{s,t}(c_{i+1}) - \mu_{s,t}(c_i)$ fulfills $\Delta\mu \geq 0$ everywhere. Notice that μ will be minimum $(-D(s, t))$ at source and maximum $(+D(s, t))$ at destination. A Polarized route example can be seen in Figure 3(a) in which arcs are labeled by their $\Delta\mu$ values.

Routing symmetry is a desirable property since it helps to balance traffic. Symmetry implies that, if the routing provides a set of paths from a source s to a target t, then the reverse of these paths is exactly the set of paths from t to s. In *Polarized routing* symmetry is guarantied because $\mu_{s,t}(c) = -\mu_{t,s}(c)$. Source and destination switches act, respectively, as repulsive and attractive magnetic poles, which motivates the name of the mechanism.

Implementing Polarized routing is straightforward. In table-based routing implementations used in current networks, there is a routing table per switch which is indexed by the destination label, t, recorded in the packet header [5]. Each table has N entries, one per destination, of R bits, one per port. Tables are initialized by running a *Breadth-first search* (BFS) in such a way that ports whose neighbor switches are closer to destination are set to 1 and 0 otherwise. Polarized tables are also initialized by running a BFS but coding ports as $\{-1, 0, 1\}$ representing the possible distance variation after a hop through such port in the path towards a given destination t, denoted as Δt. These $\{-1, 0, 1\}$ values indicate whether the considered port *approaches*, *revolves* or *departs* destination t.

Routing Mechanisms

Depending on the length of the paths, routing can be *minimal*, only employing shortest paths, or *non-minimal*. Using shortest routes works well when the traffic is uniform, but when it is not, shortest paths can be saturated quickly, ruining performance while lots of network links are not used. Adversarial traffic patterns would require longer non-minimal routes to distribute the traffic over all network's links, although this would provoke a higher network load.

Deterministic routing always uses the same path to communicate the same pair of switches. By contrast, *Adaptive* algorithms select different paths according to traffic and network conditions. Depending on the location in which the selected path is determined, routing algorithms can be further classified as *source-based* or *hop-by-hop*. In the former, route selection is performed at the injection server, while in the latter this selection is performed on the fly at every switch on the path. Regardless of whether the routing is minimal or not, adaptive source-based policies fully determine the route at injection time while hop-by-hop strategies perform an adaptive routing decision on every hop. This can be done using only minimal routes or allowing certain non-minimal paths.

Valiant's scheme [4] is the typical solution that uses non-minimal routes. A random intermediate switch is selected and minimal routing is used from source to intermediate, and then to target switch. This distributes traffic uniformly, but doubles both network traffic and average latency and halves maximum throughput. Large direct topologies have exploited a source-based adaptive non-minimal routing algorithm denoted as UGAL (Universal Globally Adaptive Load-balanced) [3]. UGAL decides between using a minimal or Valiant route according to traffic adverseness, when injecting a packet. It pursues the use of the shortest routes when the traffic is uniform but when it is adversarial, it relies on longer non-minimal paths; it would be theoretically able to achieve 100% throughput for benign traffic and 50% throughput for worst case traffic. Nevertheless, these source-based mechanisms make global routing decisions at injection time, relying on solely local information and/or stale remote congestion notifications, being unable to adapt in-transit packets to the different regional network conditions.

Ad hoc solutions for specific topologies are also important. A particular case of hop-by-hop routing for HyperX networks can be seen in [1]. However, Random Regular Graphs (RRGs) are the topologies that most stress a generic routing algorithm as they almost certainly have the same general structure, but locally they can have many peculiarities that make an *ad hoc* routing fail in different aspects. The first solution for RRGs is the k-Shortest Paths (KSP) source-based routing algorithm, in which k shortest paths are selected, not necessarily minimal, for every pair of switches. Usually, those k shortest paths are computed using Yen's algorithm [5]. The pool of paths, the way of selecting them, or the length they might have leads to different routing improvements and implementations as KSP-UGAL in [2].

Routing Readings

[1] N. McDonald *et al.*, "Practical and efficient incremental adaptive routing for HyperX networks," in *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, ser. SC '19. New York, NY, USA: Association for Computing Machinery, 2019.

[2] M. A. Mollah *et al.*, "A comparative study of topology design approaches for HPC interconnects," in *2018 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, 2018, pp. 392–401.

[3] A. Singh, "Load-balanced routing in interconnection networks," Ph.D. dissertation, Stanford University, 2005.

[4] L. G. Valiant and G. J. Brebner, "Universal schemes for parallel communication," in *Proceedings of the thirteenth annual ACM symposium on Theory of computing*, ser. STOC '81. New York, NY, USA: ACM, 1981, pp. 263–277.

[5] J. Y. Yen, "An algorithm for finding shortest routes from all source nodes to a given destination in general networks," *Quarterly of Applied Mathematics*, vol. 27, no. 4, pp. 526–530, 1970.

As Polarized not only looks at which destination a packet wants to go, but also the source from which it was generated, it considers as well the distance variation to the source switch after a hop, Δs. This is illustrated in Table 1, which lists the nine possible value variations $(\Delta s, \Delta t)$. Observe that $\Delta\mu$ is precisely $\Delta s - \Delta t$, which takes values in the set $\{2, 1, 0, -1, -2\}$, indicated as a super-index in Table 1. Obtaining $\Delta\mu$ is as simple as reading the routing table twice, one indexed by source s and another by destination t, and performing the substraction $\Delta s - \Delta t$, which is illustrated in Figure 3(b).

Polarized routing allows, under conditions, all the routes that approach to t (first column) and all the routes that get away from s (first row) in Table 1. Green-coloured options, which correspond to ports with $\Delta\mu = 2$ or $\Delta\mu = 1$ will always be considered, as increasing the value of μ can be interpreted as advancing from s to t. Entry $(+1, -1)^2$ represents the ideal situation, becoming nearer to target and farther from source as with minimal routing. Some cases that do not change μ must be considered to have enough path diversity, although a few rules are needed to avoid livelocks. The option $(+1, +1)^0$ will only be a candidate when the packet is closer to the source than to the target,

	Approaches t	**Revolves t**	**Departs t**
Departs s	$(+1,-1)^2$	$(+1,0)^1$	$(+1,+1)^0$
Revolves s	$(0,-1)^1$	$(0,0)^0$	$(0,+1)^{-1}$
Approaches s	$(-1,-1)^0$	$(-1,0)^{-1}$	$(-1,+1)^{-2}$

2 The ideal situation of increasing the value of μ by 2.
1 Increases the value of μ by 1.
0 Leaves μ as it is, just one of the options is allowed by the algorithm, depending on whether $D(c,s)$ is less than $D(c,t)$.

Table 1: Distance variations to target and source, $(\Delta s, \Delta t)$, after a hop.

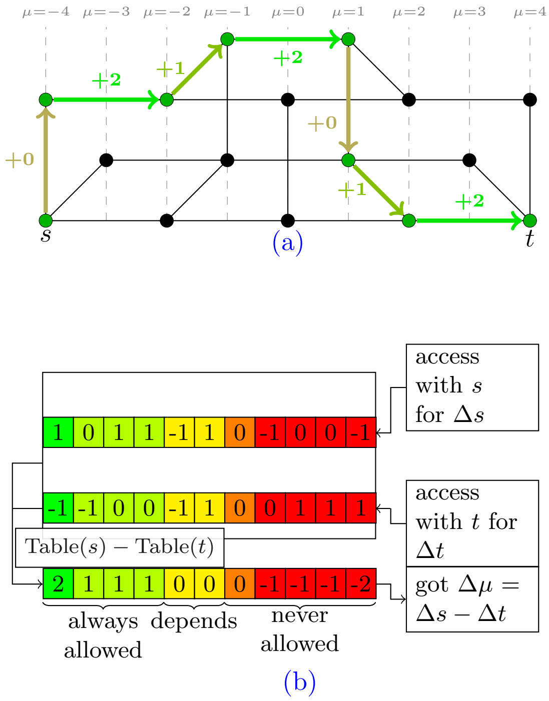

Figure 3: (a) A Polarized route. (b) The two accesses to the routing table and their difference, with the ports sorted by priority.

$D(c,s) < D(c,t)$, and $(-1,-1)^0$ is only a candidate when $D(c,s) \geq D(c,t)$. The boolean $D(c,s) < D(c,t)$ can be managed by including a field in the packet header and updating it with each move or by storing the distances in the routing table. In the polarized route of Figure 3(a), vertical links do not change the μ function. The first one is upwards, as it gets away of both source and destination, and it is allowed since it is closer to the source. The second vertical link is downwards, as it gets closer to both of them, and it is allowed since it is closer to the destination.

Orange-colored option is never permitted since there is no symmetric counterpart neither red-coloured ones since either $\Delta\mu < 0$ or it would allow livelocks to occur.

On every switch in a path, Polarized selects a port with maximum (lower is better) priority among all the candidates. The priority of a port is the sum $q + p$ of its occupancy q, measured in credits, plus a penalty p associated to its $\Delta\mu$. Candidates with the greatest $\Delta\mu$ have no penalization ($p = 0$) but those with lower $\Delta\mu$ are penalized. For example, for the case in which there are candidates with $\Delta\mu$ equal to 2, 1 and 0, we have used $p_2 = 0$, $p_1 = 64$, and $p_0 = 80$, respectively. These values have been empirically chosen, and depend on the FIFO (First In First Out) queue length.

Empirical Evaluation

First, we present results for a RRG with diameter 4 connecting a total of 720 switches and 5040 servers. This is done using 24-radix switches, where 17 links are used to connect switches and 7 to servers. The number of servers per switch has been chosen to ensure a bit of over-subscription when enduring uniform traffic. Although out of the scope of this paper, it can be formally proved that if the degree δ of a RRG fulfills the relation $\delta > \frac{2}{\ln 2} \ln N (\sim 2.885 \ln N)$, then Polarized routing can be successfully applied. The number of switches in current and forthcoming datacenter and supercomputer deployments, and their radix, easily meets this condition.

We simulate a typical switch model, with FIFO buffers at both input and output ports. It contains a credit counter at the output ports with an estimation of the available space in the input buffer of the neighbour switch. The queue occupancy is estimated as the occupation in the output buffers of the corresponding VC (Virtual Channel) plus the credit counter. Packet deadlock is avoided by using virtual channels in increasing order. More evaluation details can

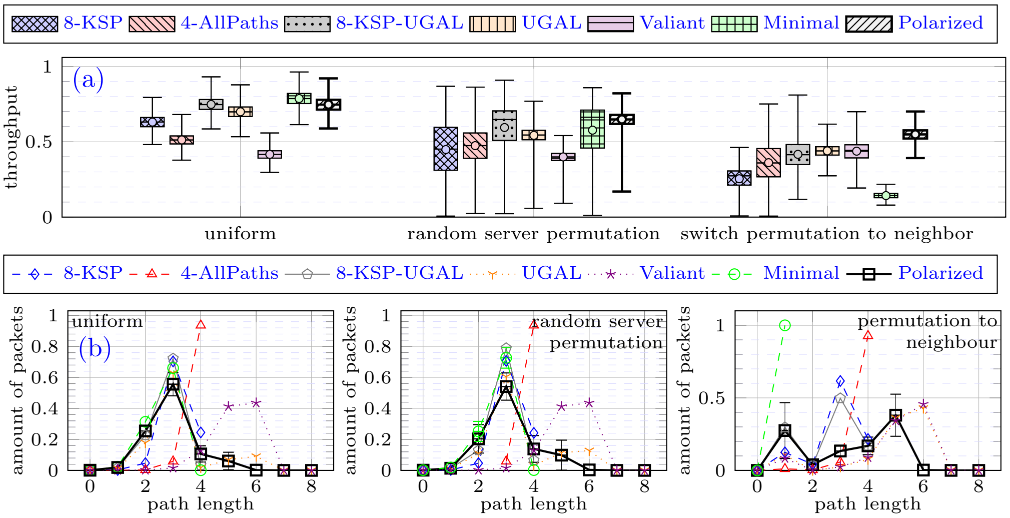

Figure 4: (a) Box Plot of throughput relative to consumed load in each server: box lines are quartiles and median values, whiskers are the loads of the worst and best server, circles are averages. (b) Histogram of packets by length of their hop count.

be found in [4].

The traffic patterns used are: *uniform*, representing the most benign case, *random server permutation*, a slightly adversarial pattern and *switch permutation to neighbor*, consisting on a permutation of the switches with all destinations at distance 1 from their sources. With minimal routing all servers in the same switch send their traffic through a unique link, thus constituting a severe adverse pattern.

Polarized is compared with another seven routing mechanisms. Minimal and Valiant represent bounds for benign and adversarial patterns. In minimal, each switch selects, hop-by-hop, the next switch belonging to some minimum path towards the destination. UGAL is a common source-based adaptive solution. 4-AllPaths, 8-KSP and 8-KSP-UGAL are optimized source-based mechanisms proposed for RRGs; the first one selects a random route among all the paths up to distance diameter; the second is a typical KSP, which randomly selects one path among 8 candidates, and 8-KSP-UGAL selects a path between minimal and one from the 8-KSP's pool.

Figure 4(a) measures the throughput of Polarized routing when applied to RRGs. Results are presented in an order that reflects the traffic adverseness: benign, medium and severe. The upper and lower limits of the boxes are respectively the loads accepted by the servers in the Q1 and Q3 quartile positions, this is, values such that 25% of the servers accept less load or *vice versa*. Similarly, the upper and lower whiskers show the maximum and lowest values of accepted load. Median value is shown as middle line and average as a circle. For reducing the observed standard deviation, which is more noticeable with adversarial patterns, a proper congestion control mechanism should be added. It should be noted that congestion control is orthogonal to the routing problem considered in this paper.

As it can be seen, with uniform traffic, Polarized is able to achieve almost the same throughput as minimal routing. With this pattern, the distribution of load around their means is almost identical for all routings. Valiant provides around 40% of the maximum load, as expected. AllPaths, UGAL and KSP perform quite modestly, but KSP-UGAL does it quite well. Polarized is better than the other alternatives for adversarial patterns. In the case of random server permutations, Polarized appears to give just a bit better throughput than using minimal routes or KSP-UGAL, but it is clearly superior when dealing with switch permutations. Furthermore, for random server permutations, Polarized routing's quartile box is more narrow than minimal or KSP-UGAL routing, which entails a more fair distribution of the load.

With respect to the selected paths, it can be proved that $4D-3$ is a theoretical upper bound on the length of the polarized routes for diameter $D \geq 2$. In practice, this bound is not attained; for example, the longest paths in the RRG tested in our experiments use just $2D$ links, as with Valiant's routing, as can be seen in Figure 4(b). Observe that the different distributions of the path's lengths lead to different routing performance. However, although the length of the paths in Polarized is longer than in minimal routing, this hardly affects the average latency after saturation.

Polarized routing can be applied to many other topologies such as cycles, complete graphs, n-dimensional tori, Hamming graphs (Flattened Butterflies and HyperX), Slimflies and most common topological configurations of Dragonflies of diameter 3, which encompasses all the representative direct topologies for HPC and datacenters systems. Next, we provide partial evaluations for a subset of these notable low-diameter topologies, including a Dragonfly, a 3D Hamming graph (Flattened Butterfly and HyperX), and a Slimfly.

Figure 5 plots the latency curves for each topology when managing moderately adversarial traffic. All the plots include, at least, minimal, Valiant, and UGAL routing as references. It can be seen that UGAL is always near the best of either minimal or Valiant, but cannot do much better. Polarized is notably superior in two of the topologies

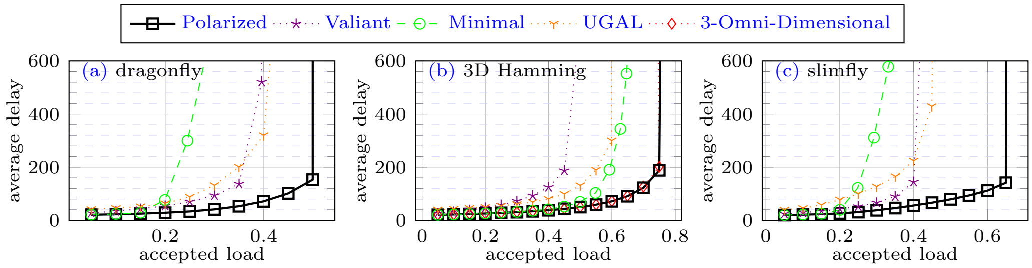

Figure 5: Random server permutation latency for (a) Dragonfly, (b) 3D Hamming and (c) Slimfly.

and is even with the best *ad hoc* mechanism known for Hamming graphs: Omni-Dimensional routing. Polarized and Omni-Dimensional provide almost the same performance. However, a deeper comparison of these routings shows that the paths used by both are different, even when providing similar throughput.

Finally, it is worthwhile to know Polarized routing, as presented in this paper, cannot be applied to certain topologies, which are focused to a different application domain. The simplest example is a path graph (a.k.a. linear graph); it is easy to see that paths that start with a hop in the opposite direction from the destination can never reach it. The same occurs with a mesh, where the routing may initiate a path towards a wrong corner that cannot be completed into a whole route. Simple modifications could be added in order to deploy a Polarized routing able to deal with these topologies.

Conclusions

Polarized routes preserve symmetry and avoid backtracking, which translates into better performance. The Polarized routing algorithm matches or improves the behavior of previous mechanisms that have been conceived as *ad hoc* solutions for specific topologies. Moreover, it constitutes an unprecedented solution for the very interesting but less widely explored case of random networks, achieving noticeable performance gains compared to previous solutions. The versatility of the Polarized routing algorithm has at least two key implications. The first one is that it makes the implementation of practical system networks based on RRGs more feasible and hence, it may be useful for the adoption of any other direct topology coming in the future. The second is that Polarized routing can be understood as a homogeneous solution for almost any network, such as minimal and Valiant routing schemes, but offering the best features of both.

Acknowledgements

This work has been supported by the Spanish Ministry of Science and Innovation under contracts PID2019-105660RB-C22 and FJCI-2017-31643. Simulations were performed in the Altamira supercomputer, a node of the Spanish Supercomputing Network.

References

[1] Jung Ho Ahn, Nathan Binkert, Al Davis, Moray McLaren, and Robert S. Schreiber. HyperX: Topology, routing, and packaging of efficient large-scale networks. In *Proceedings of the Conference on High Performance Computing Networking, Storage and Analysis*, SC '09, pages 1–11, New York, NY, USA, 2009. ACM.

[2] Maciej Besta and Torsten Hoefler. Slim Fly: A cost effective low-diameter network topology. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '14, pages 348–359, Piscataway, NJ, USA, 2014. IEEE Press.

[3] Béla Bollobás. *Random Graphs.* Cambridge studies in advanced mathematics, 2nd edition, 2001.

[4] Cristóbal Camarero, Carmen Martínez, and Ramón Beivide. Polarized routing: an efficient and versatile algorithm for large direct networks. In *2021 IEEE Symposium on High-Performance Interconnects (HOTI)*, pages 52–59, 2021.

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

[6] John Kim, William J. Dally, Steve Scott, and Dennis Abts. Technology-driven, highly-scalable dragonfly topology. In *Proceedings of the 35th Annual International Symposium on Computer Architecture*, pages 77–88. IEEE Computer Society, 2008.

[7] Ankit Singla, Chi-Yao Hong, Lucian Popa, and P. Brighten Godfrey. Jellyfish: Networking data centers randomly. In *Proceedings of the 9th USENIX Conference on Networked Systems Design and Implementation*, NSDI'12, pages 225–238, Berkeley, CA, USA, 2012. USENIX Association.