Polarized routing: an efficient and versatile algorithm for large direct networks

postprint version

C. Camarero, C. Martínez and R. Beivide, "Polarized routing: an efficient and versatile algorithm for large direct networks,"

2021 IEEE Symposium on High-Performance Interconnects (HOTI), 2021, pp. 52-59.

https://doi.org/10.1109/HOTI52880.2021.00021

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

December 17, 2021

Abstract

Supercomputer and datacenter networks can comprise hundreds of thousands of severs. Focusing on direct networks, different topologies have been proposed to attain such a high scalability, from Flattened Butterfly and Dragonfly to the most disruptive approach represented by Jellyfish, which is based on a random interconnection pattern. The routing problem on such networks remains a challenge that can be tackled as a topology aware solution, or with an agnostic approach. The case of random networks is a very special one because of the lack of an acceptable routing algorithm for them 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 for direct networks that can be used in a number of topologies, including Jellyfish. Polarized routing was conceived following two design criteria: a source–destination symmetry in the routes to enable load-balancing and to avoid undoing previously taken hops. A thorough experimentation shows Polarized routing constitutes an efficient and versatile solution, attaining the highest performance both in benign scenarios under uniform traffic patterns and in adverse ones on the tested networks. Interestingly, this algorithm provides important performance gains of more than 30% in the Jellyfish topology, for different traffic patterns, when compared to the state of the art solutions.

1 Introduction

Supercomputers and cloud datacenters are the largest systems deployed. The scale of supercomputers depends on the computing power of the servers to be interconnected. For example, the IBM Summit has close to 5,000 "fat" server nodes, each one performing at a little above 40 TFLOP/S but the Fujitsu Fugaku deploys close to 160,000 servers at 3.3 TFLOP/S. With these references, around 25,000 (Summit) or 300,000 (Fugaku) servers, which would have to be interconnected to achieve the desired EXAFLOP/S. Although sometimes based on cheaper technology, current cloud datacenters present the same problems of scale and interconnection.

A good system network must supply very large amounts of bandwidth and exhibit very low message latencies. Although interconnection networks have been widely studied over the last 50 years, there is no single, clearly-adopted solution to achieve these basic goals at an affordable cost, revealing that their complexity is not fully understood yet. Two critical and fundamental features of a network are its topology and routing, the two being highly independent as good topology is totally useless without proper routing.

With respect to topology, indirect networks such as the Folded Clos (also called Fat-tree) have dominated the datacenter market and a very significant part of the supercomputing industry. This includes, for example, the network used by the IBM Summit. It is a highly tested topology, inherited from telephony systems, which is relatively easy to partition and which admits very simple, deadlock-free UP/DOWN routing. Nevertheless, its cost is higher than some direct topologies, especially at large scales. It has been shown that, using high-radix routers, low-diameter direct topologies employ the bandwidth better, minimizing cost. Examples of low-diameter topologies include: Flattened

Butterfly [18], Dragonfly [19], HyperX [1], SlimFly [3], Projective networks [7], Dragonfly+ [26] or Megafly [13]. It should be noted that Flattened Butterfly and HyperX networks, are names describing quite similar topologies based on the Cartesian graph product of complete graphs. Thus, we prefer to denote them as Hamming graphs thorough the paper. Most of these networks are based on approximations to Moore graphs, which are those regular topologies that attain the highest number of vertices when fixing the degree and diameter; remember that the complete graph is a trivial Moore graph. All of them 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) [4] can be as good as or even better than the previous structured Moore-based graphs when used as interconnection networks.

With respect to routing, low-diameter direct topologies present interesting challenges. Typically, these networks have exploited a global adaptive routing algorithm (UGAL) [27], often with topology specific modifications such as Clos-AD [18] for Flattened Butterfly and progressive adaptive routing [17] for Dragonfly. Such algorithms pursue the use of the shortest routes when the traffic is benign but when it is adversarial, they rely on longer non-minimal paths. Most of the employed routing algorithms have been implemented so far using a source-based mechanism that selects the packet route at the injection node, which entails a couple of important issues. First, they are unable to adapt in-transit packets to the different regional network conditions. Second, they make global routing decisions at injection time, relying on solely local information and/or stale remote congestion notifications.

Incremental adaptive routing algorithms, in which the routing mechanism can make many small adaptive decisions for a packet instead of one large adaptive decision at the source, can yield important performance gains. A particular case for HyperX can be seen in [22]. However, no incremental (or hop-by-hop) adaptive routing algorithm agnostic of the topology is known for direct low-diameter networks. The case of RRGs is paradigmatic for a couple of reasons. First, they model an arbitrary regular topology in some way representing an average network case from a very big graph universe. Second, none of the previously proposed and tested source-based routing algorithms are sufficiently efficient, preventing the adoption of RRGs as interconnection networks.

This paper introduces Polarized routing, an efficient incremental adaptive non-minimal routing algorithm able to run over any practical large direct topology. We have selected and highlighted the case of RRGs since they constitute a difficult challenge because of both their universality and the absence of a good routing for them. Nevertheless, it will also be shown that Polarized routing can be efficiently applied to a plethora of interesting topologies. All these results will be tested and proved by means of an extensive simulation process.

The remainder of the paper is organized as follows. In Section 2 the necessary background and related work is presented. Section 3 introduces Polarized routing and the guidelines for its practical implementation. In Section 4 the router model and the simulation test-bed are described, together with the results of the evaluation over RRGs. The analysis of Polarized routing over other topologies follows in Section 5 and, finally, Section 6 concludes the paper.

2 Background and Related Work

2.1 Random Regular Graphs

Although Polarized routing will be tested in a number of structured low-diameter topologies, such as those introduced above, the first main targets are RRGs. Routing in RRGs is a difficult problem since no a priori topological clues can be exploited. Random graphs almost certainly have the same general aspect, but locally they can have many peculiarities that make an *ad hoc* algorithm fail in different aspects. Hence, in a way, the RRG is the topology most stresses a generic routing algorithm. We will see that Polarized routing is robust enough to work perfectly well in random networks, providing much more performance than previous proposals.

Random graphs are widely known in the mathematical literature [4]. The direct interconnection networks proposed in [21] and [28] select a randomly chosen graph among all the regular graphs with the same number of nodes and degree. Similarly, it was proposed in [20] to augment a cycle with random links, which has little difference compared with a pure random graph. An example of a RRG can be seen in Figure 1; the notation of the paper is introduced in Table 1. As can be observed, the network of Figure 1 connects $N = 27$ routers using degree $\delta = 4$. Every router has radix $R = 6$ and $\delta_0 = 2$ computing servers attached for a total of $S = 54$ servers in the system. The diameter of this network is $D = 3$. An algorithm to generate RRGs was proposed by Steger and Wormald [29]. Moreover, it is known that a δ-regular random graph with N vertices of diameter D can be obtained easily if $\delta^D \approx 2N \ln N$, [4]; thus, $\delta \approx (2N \ln N)^{1/D}$.

2.2 Routing

Depending on the length of the paths, routing algorithms can be classified as minimal or non-minimal [11]. Minimal schemes only employ one of the shortest paths among those that connect any pair of routers. The minimum network load to service some fixed communications is provided by the use of these shortest routes, that is, through the use of minimal routing. Using shortest routes works very well in many topologies when the traffic is uniform, obtaining high performance since the load is not unnecessarily increased. Nevertheless, when the traffic is not uniform, shortest

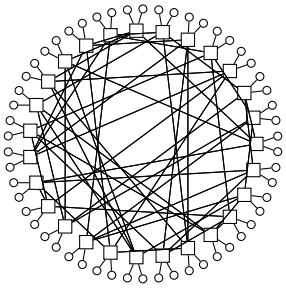

Figure 1: A direct radix-6 Random Regular Network 27 routers and 54 servers.

Parameter	**Definition**
N	Number of routers (vertices).
S	Number of severs.
δ	Degree of the graph.
δ_0	Number of severs per router.
$R = \delta + \delta_0$	Router radix (number of ports).
$D(s,t)$	Graph distance between vertices s and t.
D	Graph diameter.

Table 1: Network Parameters.

routes can be saturated quickly, ruining performance while lots of network links are not used. These adversarial traffic patterns would require longer non-minimal routes to distribute the traffic over all the network's links, although this would provoke a higher network load.

There are many proposals aiming to solve this routing problem, but let us focus on the classical one: Valiant's routing scheme [30]. This load balancing mechanism distributes traffic uniformly, but usually provides paths too long for certain traffic patterns, including uniform, in which latency doubles compared to minimal routing. In Valiant's scheme, to route from a source router s to a destination (or target) one t, an intermediate router r (different from s and t) is selected randomly. Then, minimal routing is used for segments sr and rt. Valiant routing doubles both network traffic and average latency and halves maximum throughput. Source-based adaptive routing algorithms such as UGAL [27] decides between using a minimal or Valiant route according to traffic adverseness, when injecting a new packet.

Static routing always uses the same path to communicate the same pair of source-destination routers while dynamic algorithms employ different paths. Adaptive algorithms are dynamic ones that select among different paths according to traffic and network conditions. Depending on the location of the routers in which the selected path is determined, routing algorithms can be classified as source-based or hop-by-hop (also denoted as incremental in [22]). In the former, route selection is performed at the injection router while in the latter this selection is performed on the fly at every router 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. When using the latter, as in this work, given a source router s and a destination router t, in each current router c along the path it is decided which is the next link in the route towards t. This can be done using only minimal routes or allowing certain non-minimal paths.

For most of the structured low-diameter networks, a source-based mechanism has been assumed for non-minimal adaptive routing with the use of UGAL [27] which would be theoretically able to achieve 100% throughput for benign traffic and 50% throughput for worst case traffic. Nevertheless, when RRGs have been considered for interconnection networks, the k-Shortest Paths (KSP) source routing algorithm has been employed. With KSP, exactly k shortest paths can be selected for every pair of routers (s,t); k stands for a small integer. Usually, those k shortest paths are computed using Yen's algorithm [31]. The change of the pool of paths, the way of selecting them or the length they might have gives place to different routing implementations as rEDKSP-adaptive in [2], KSP-UGAL in [25] and LLSK in [32]. Note that the previous routing algorithms can use non-minimal routes.

However, source adaptive routing can cause severe inefficiencies as it has the constructive deficiency of making global decisions based only on local information. In certain traffics, this policy is unable to adapt to congestion not seen at the source where the adaptive decision is made and in others, a slight amount of congestion close to the source will cause global load-balancing, doubling bandwidth consumption and packet latency. In addition, once it is decided that load-balancing (route non-minimally according to Valiant) is needed, they employ non-minimal routes from that point forward, regardless of the network conditions of the remaining of the path. For the Dragonfly topology, for example, attempting to load-balance the global links from two router hops away produces suboptimal performance [19].

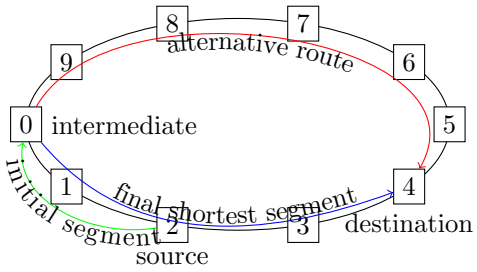

Figure 2: Valiant's scheme provides routes that undo hops when selecting router 0 as middle router to send from router 2 to router 4. There is an alternative route not used by Valiant without that problem.

Thus, for improving network performance, this paper introduces Polarized routing, which is a high-performance, non-minimal, adaptive, hop-by-hop strategy that can be used on all the low-diameter interconnection networks of interest.

3 Polarized Routing

As minimal routing is optimal for uniform traffic while Valiant's scheme is optimal for the worst case traffic, evolved routing algorithms attempt to combine the benefits of both by alternating them with decisions taken on the fly. This can be done in several ways but, to our knowledge, all the mechanisms focus on the target router, without considering from which router they come. This usually breaks the symmetry in the route between source and destination routers and, in some situations, causes hops that are counterproductive as they are redundant after some previous hops.

Both minimal and Valiant routings use the source router and target router in a symmetric way. This means that, if the set of paths from a source s to a target t provided by the routing algorithm is considered, then the reverse of these paths is exactly the set of paths from t to s. This property helps to distribute the load correctly, although it is not sufficient. There is also a subtle—but very important—behavior that should be avoided, which we denote as "undoing a hop", which can be explained with the following simple example. Let us consider a ring of length $N = 10$ represented in Figure 2. If source router $s = 2$ has to communicate with target router $t = 4$, using intermediate router $r = 0$, 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. This is a similar effect to the one observed in Slimflies in [33]. Observe in the figure that there is an alternative path avoiding that.

3.1 The Polarized Routing Algorithm

The *Polarized routing algorithm* enforces the symmetry between source and destination routers, the former acting as a repulsive magnetic pole and the latter as an attractive one, which motivates the name of the routing. As shown next, this routing includes shortest routes and most of the routes that Valiant's scheme would provide, plus many additional routes of intermediate length. Moreover, as described next, in the Polarized routing algorithm all hops must improve a weight function, and therefore, they can never "undo" a previous hop.

Polarized routing is introduced regardless of the topology of the target direct network, which we model by a graph $G(V, E)$, where the set of vertices V represent the N routers of the network and an edge in E represent whether there is a link connecting two routers. Now, let us consider router s source, t target and c current and define the following weight function:

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

where $D(s, t)$ denotes the graph distance from vertex s to t. Thus, μ will be minimum at source and maximum at destination, with values $-D(s, t)$ and $D(s, t)$ respectively.

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 router. That is, no hop decrements the weight function μ. Equivalently, when considering the difference operator $\Delta\mu = \mu_{s,t}(c_{i+1}) - \mu_{s,t}(c_i)$, a polarized route fulfills $\Delta\mu \geq 0$ everywhere. Polarized routing can be illustrated by means of Table 2. It lists the nine possible value variations $(\Delta s, \Delta t) = \big(D(c_{i+1}, s) - D(c_i, s), D(c_{i+1}, t) - D(c_i, t)\big)$ of the distances pair $(D(c, s), D(c, t))$ after a hop, in which each distance can decrease by 1, remain the same, or increase by 1; thus, these nine $(\Delta s, \Delta t)$ pairs take values $\{-1, 0, 1\} \times \{-1, 0, 1\}$. The weight variation, $\Delta\mu$, associated to each pair is indicated as a super-index. Observe that $\Delta\mu = \Delta s - \Delta t$, taking values in the set $\{2, 1, 0, -1, -2\}$.

Polarized routing allows, under conditions, all the routes that approaches to t (first column) and all the routes that get away from s (first row). Options with $\Delta\mu = 2$ or $\Delta\mu = 1$ will always be considered by Polarized, as increasing the value of μ can be interpreted as advancing from s to t. Entry $(+1, -1)^2$ represents the ideal situation, which is

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

becoming nearer to target and farther from source as with minimal routing, thus increasing μ by 2. Some cases that do not change μ must be considered in order to have enough path diversity, although some rules are needed to avoid livelock issues. The option $(+1,+1)^0$ will only be a candidate when the packet is closer to the source than to the target, $D(c,s) < D(c,t)$, and $(-1,-1)^0$ is only a candidate in the opposite case, when $D(c,s) \geq D(c,t)$. Those options that are underlined are not allowed by Polarized since either $\Delta\mu < 0$ or it would allow livelocks to occur. Note that the 5 beneficial options provided by Table 2 are possible in a generic graph, but there are specific topologies in which this is not true; for example, in a ring, variations $(+1,0)$ and $(0,-1)$ cannot take place.

Then, Polarized will allow forwarding packets from the current router c to a neighbor router c_p through port p when one of the following conditions holds.

C1 $\Delta\mu > 0$: the weight is increased.

C2 $\Delta\mu = 0$ and $D(c,s) < D(c_p,s)$ and $D(c,s) < D(c,t)$: get away from source when closer to it.

C3 $\Delta\mu = 0$ and $D(c,t) > D(c_p,t)$ and $D(c,s) \geq D(c,t)$: approach to destination when closer to it.

3.2 Polarized Routing Implementation

A cheap way to build these Polarized routes adaptively is by selecting the links in a greedy way. This is, by allowing at every router in the path to select any link satisfying any of the Conditions C1–C3. In most topologies of interest, as it will be discussed later, this is guaranteed to complete every route. Polarized routers may employ node-table routing [11, Section 11.1.2], in which each router has a table similar to the one used for minimal routing. In minimal routing, the table is indexed by the destination label, t, and each entry is an R bit vector. Tables are initialized by running a BFS in such a way that ports whose neighbor routers are closer to destination correspond to elements set to 1 and 0 otherwise. Similarly, one implementation of Polarized consists also in running a BFS but filling in the tables recording them with values in the set $\{-1,0,1\}$, representing whether the port *approaches*, *revolves* or *departs* destination, the same as Δt.

Obtaining $\Delta\mu$ for every port at the current router is as simple as reading the table twice, one indexed by source s and another by destination t, and performing the substraction $\Delta s - \Delta t$. This process is illustrated in Figure 3. Ports with $\Delta\mu > 0$ will be always considered as candidates for routing. The boolean $D(c,s) < D(c,t)$ to decide on the cases with $\Delta\mu = 0$ can be managed by including a field $\big(D(c,s), D(c,t)\big)$ in the packet header and update it with each move or by storing the distances in the table, which is trivial when running the BFS. This avoids the $2\log_2 D$ bits in the packet and their management, but multiplies the table size for a $\log_2 D$ factor, which is very small in low-diameter networks and may be a sensible approach. In addition, instead of just $\{1,0\}$ of minimal, Polarized needs one more state to code each port, requiring two bits per port. Finally, supporting two table accesses per cycle can be addressed by using a double-port memory or by duplicating the table.

On every router in a path, Polarized selects a port with maximum (lower is better) priority among all the candidates. The priority of a port is determined by the sum $w+q$ of its weight difference $\Delta\mu$, conveniently scaled, plus its occupancy measured in credits. Similarly to UGAL, in which queue occupation is used to define thresholds for selecting between minimal and Valiant paths, we relate the weight difference, $\Delta\mu$, with occupancy by defining $w_2 = 0$, $w_1 = 64$, and $w_0 = 80$, using w_2 for the candidates with the greatest increase of μ. Note that sometimes there is no candidate with an increase of μ by 2, being the greatest possible 1 or 0. These w_i values have been empirically chosen, and may depend on some other aspects such as the FIFO queue length. A description of the Polarized routing mechanism is shown as Algorithm 1.

Section 5 shows that Algorithm 1 can be applied over all low-diameter topologies of interest. For diameter $D \geq 2$, it can be proved that $4D-3$ is an upper bound on the length of the routes obtained from Algorithm 1 by considering three facts: the sequence of $\big(D(c,s), D(c,t)\big)$ can include (D,D) as the middle point; the variations up to that point can alternate between $(+1,+1)$ and $(+0,-1)$, whose sum is $(+1,+0)$; about $2D$ hops towards (D,D) and again about $2D$ towards destination are needed. Thus, $4D-3$ is obtained as general bound, after considering a few frontier cases. In practice this bound is not attained; for example, it is easy to see that the bound is just $2D$, as with Valiant's routing, in cycles and complete graphs. Thus, their cartesian products, Torus and Hamming graphs, do not surpass the same $2D$ bound. In any case, deadlock is considered to be avoided by using virtual channels in increasing order [15]. In

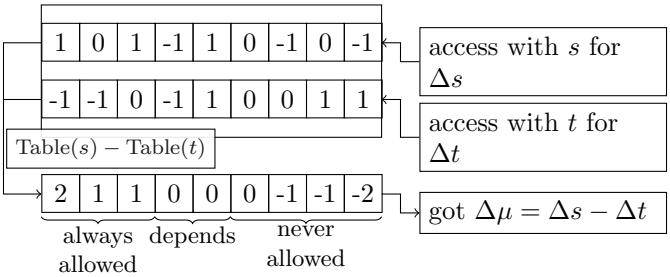

Figure 3: Illustration of the access to the routing table.

Algorithm 1 Hop-by-hop Polarized Routing Algorithm.

Require: source s, target t, current c
Ensure: a next router c_p neighbor of c, or none
  candidates ← empty.
  for p from 1 to Δ **do**
    $c_p \leftarrow p$-th neighbour of c.
    $r \leftarrow \Delta\mu = \mu_{s,t}(c_p) - \mu_{s,t}(c)$ via the router table.
    if c_p satisfies any of Conditions C1–C3 **then**
      add (c_p, r) into candidates.
    end if
  end for
  $r^+ \leftarrow \max\{r \mid \text{for } (c_p, r) \text{ in candidates}\}$.
  for (c_p, r) in candidates **do**
    $w \leftarrow$ access [0,64,80] at index $r^+ - r$.
    $q \leftarrow$ queue occupation towards c_p.
    replace entry in candidates by $(c_p, w + q)$
  end for
  $q^- \leftarrow \min\{q \mid \text{for } (c_p, q) \text{ in candidates}\}$.
  candidates$\leftarrow \{c_p \mid (c_p, q) \text{ in candidates with } q = q^-\}$.
  discard any candidate not satisfying virtual cut-through.
  select randomly from the remaining if not empty.

particular, there are as many virtual channels as the maximum of length of available routes and in i-th hop, the i-th virtual channel is used.

4 Empirical Evaluation

4.1 Router Model and Experimental Setup

We simulate a typical router model, with FIFO buffers at both input and output ports. It contains at the output ports a credit counter with an estimation of the available space in the input buffer of the neighbour router. The queue occupancy used in Algorithm 1 is estimated as the occupation in the output buffers of the corresponding VC plus the credit counter.

This simple router works like one with a single stage, allowing an incoming phit to reach the output port in a single cycle if there is no contention. A *phit* is the basic unit of data in which packets are divided. For this, the router operates every cycle as follows:

- For each input queue, without an output buffer assigned, look at the leading phit. If it is a header, it calls the routing algorithm, resulting in at most one selected output port, to which a request is made.
- The requests are processed in random order. Each output port and virtual channel pair can accept the request from a single input buffer.
- All phits in the headers of input buffers with an assigned output buffer are moved to it.
- For each physical output port, one of its output buffers is selected. The leading phit is extracted from the selected buffer and sent via the link. This creates an event that inserts the phit into the appropriate buffer of the next router or server.

Parameter	Value
Simulated cycles	25,000
Virtual channels	as maximum path length
Input Buffer size	4 packets
Output Buffer size	2 packets
Flow control	Virtual cut-through
Packet length	16 phits
Link latency	1 cycle

Table 3: Simulation parameters

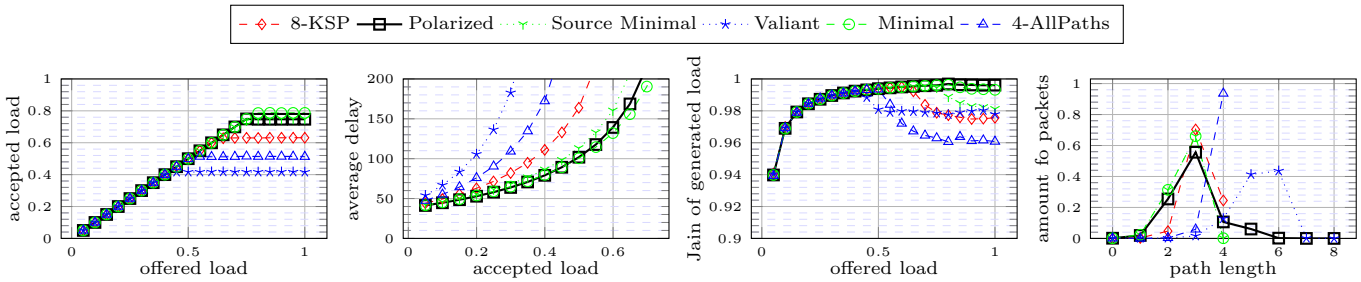

Figure 4: RRG n=720 R=17+7: uniform traffic pattern.

The experimental environment employed in this work is CAMINOS [8], a simulator developed in the recent Rust language. The basic parameters used in the simulations are detailed in Table 3 and the topologies are described in Table 4. The number of servers in the RRG has been chosen to ensure a bit of over-subscription when enduring uniform traffic. Specifically, the number of servers per router has been taken as $\delta_0 = \lceil \frac{\delta}{\bar{D}} \rceil = \lceil 6.4 \rceil$, where $\bar{D} = 2.640$ is the average distance, which makes the average load of servers links to be lower than the average load of links between routers. In this section we will show the results for RRGs, and in Section 5 for the rest of the topologies considered.

Topology	D	Radix	Servers	Routers
RRG of given parameters	4	$17 + 7 = 24$	5040	720
3D-Hamming of side 8	3	$7 \cdot 3 + 8 = 29$	4096	512
dragonfly (palm-tree [14])	3	$4 \cdot 6 - 1 = 23$	5256	876
Slimfly MMS [3]	2	$19 + 9 = 28$	3042	338
projective (Levi graph) [7]	3	$18 + 8 = 26$	4912	614

Table 4: The topologies simulated in the paper.

Polarized routing will be compared to *minimal* (shortest path) routing and generic *Valiant* routing. In the first case, each router selects, on the fly (hop-by-hop), the next router belonging to some minimum path towards the destination. In generic Valiant, for each transmission a random intermediate router is selected. Then, the packet is routed to the intermediate router and from there to the destination, using for both sub-routes the hop-by-hop minimal algorithm. Also, source routing mechanisms KSP and AllPaths will be tested, their routes are randomly selected among the pool of routes.

The traffic patterns considered are:

- Uniform: each source selects a new random target for each new communication.
- Random server permutation: there is a randomly selected permutation π of the servers which is used for the whole simulation. Each time the server x performs a new communication, its target is the server πx.
- Router permutation to neighbor: a randomly selected permutation of the routers among their neighbors.

The first traffic, uniform, represents the most benign case, the second, a slightly adversarial pattern and, the third, a more severe one.

We consider the average measures of *throughput* and *latency* as done in [11, Section 23.1]. For throughput, we indicate in ordinates the accepted load, that is, the average number of phits per cycle consumed per server. For latency we indicate in ordinates the average message delay, from when the packet was created until it was consumed. In the abscissa of latency plots, accepted load is shown, such that strictly below the curve would correspond to strictly better performance. In many cases, there is interest in the flows providing least throughput, or minimal throughput. The differences between average and minimal throughput come from unfairness situations, so we also include the *Jain fairness index* [16] for measuring the fairness in the generation of messages.

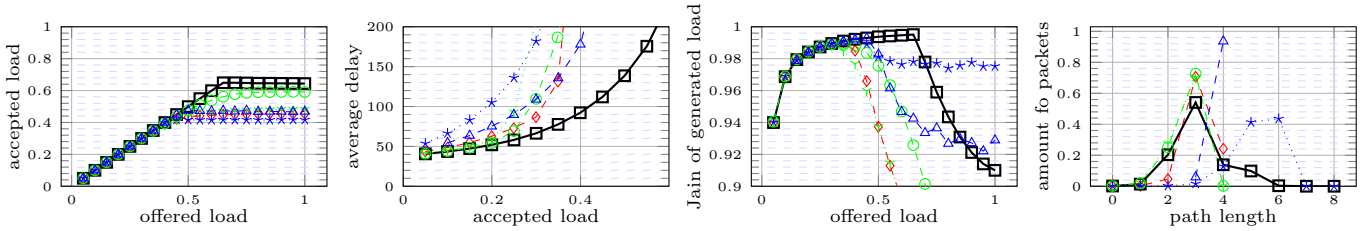

Figure 5: RRG n=720 R=17+7: random server permutation traffic pattern.

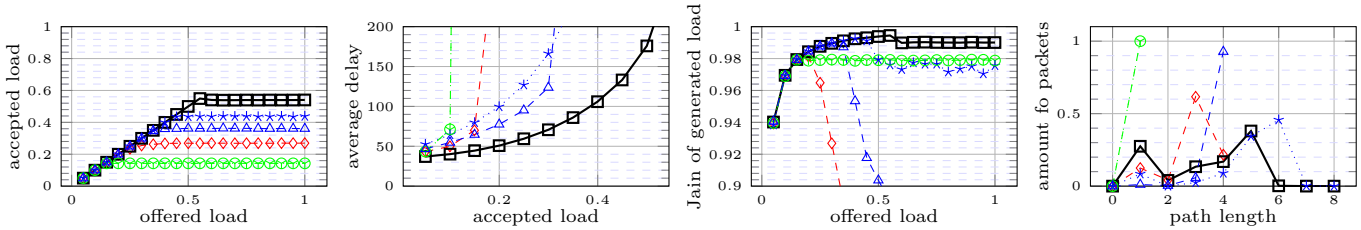

Figure 6: RRG n=720 R=17+7: router permutation to neighbour traffic pattern.

4.2 Results for Random Regular Graphs

Figures 4, 5 and 6 show the performance of the Polarized routing when applied to RRGs running the three different traffic patterns. They are presented in an order that reflects the traffic adverseness: benign, medium and severe. Although many simulations were carried out for different RRG instances, their behaviors were quite similar. Thus, we present the case of a RRG with diameter $D = 3$, as described in Table 4, having 720 routers and 5,040 servers. The Polarized routing is compared with another five routing mechanisms. Four of them are source-based: source minimal and Valiant represent the limits for benign and adversarial patterns while KSP and AllPaths are supposedly optimized mechanisms proposed for RRGs. The fifth one, minimal, is a hop-by-hop minimal routing strategy taken as a reference to compare against source minimal. No UGAL implementation is presented but minimal and Valiant mechanisms can be used as references for that mechanism. For each traffic tested, four plots are presented recording the following metrics: throughput, average latency, fairness and path lengths.

As shown in Figure 4 for uniform traffic, Polarized is able to achieve almost the same throughput, average latency and fairness as the minimal hop-by-hop routing. The length of the paths in Polarized has to be longer than in minimal routings, but as can be seen, it hardly affects the average latency after saturation. Source-based minimal routing behaves similarly to hop-by-hop routing except for some slight degradation after saturation. Valiant, as expected, saturates a little above 40% of the maximum load and exhibits the longest latencies because of the distribution of its longer paths. The different distribution of the Polarized path lengths implies much better latency figures. It is remarkable than the other source-based strategies, KSP and AllPaths, perform quite modestly in all the metrics.

Figures 5 and 6 present results for adversarial traffic patterns, with the latter being more severely adverse. As can be seen, Polarized routing is better than the other alternatives, providing noticeable gains in both cases. In the case of random server permutations Polarized routing appears to give just a bit better throughput that using minimal routes, but it can be appreciated a sub-linear growth in the curve of minimal routing. It is seen more clearly that the average delay of minimal routing grows much more quickly. This is due to the very nature of random graphs; the irregular usage of links by minimal routing causes some packets to have very large delays, since they cannot be derouted through uncongested links. In contrast Polarized routing maintains linear growth of throughput, low average latencies, and good fairness up to the maximum accepted load.

5 Topologies with Polarized Routing

Besides being, to the best of our knowledge, the best routing algorithm for RRGs, Polarized can be applied to many other topologies of interest, as it will be shown below. But before, it is worthwhile to know that Algorithm 1 fails in certain topologies, which are focused to a different application domain. The simplest example is a path graph, described in Figure 7; it is easy to see that hops that get away from the source fail to reach destination, as they cannot be traversed again in the opposite direction. The same occurs with a mesh, where Algorithm 1 may initiate a route towards a wrong corner that cannot be completed into a whole route.

More generally, Algorithm 1 works when topology has not corners, where a router c is understood to be a corner for a source s if there are not neighbours of c farther from s (this is called a boundary vertex in [9]) but s is not at maximum distance from c. Then if t is a router at greater distance from c than s, and if a message from s to t is

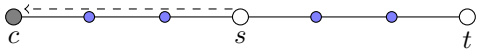

Figure 7: Illustrating wrong routing in a path.

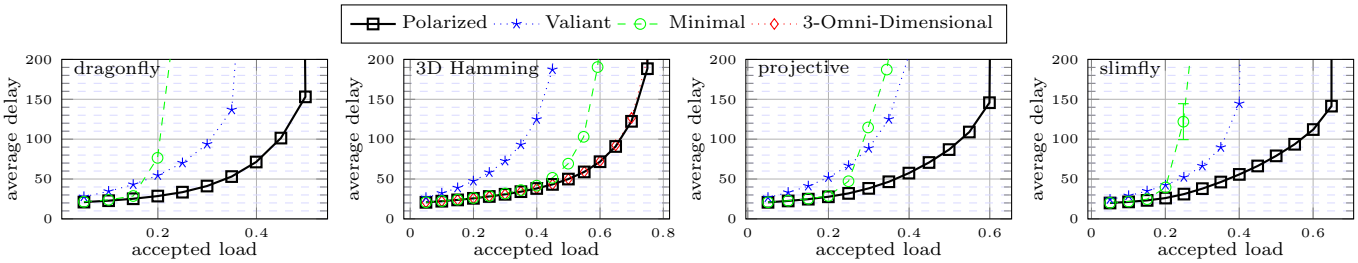

Figure 8: Random server permutation latency for Dragonfly, 3D Hamming, projective and slimfly. In the dragonfly network the shortest routing is standard hierarcal variant that uses at most global link, and Valiant uses it as base routing.

routed through c it may be the case (but not necessarily) to not having a next candidate router.

Notwithstanding, it can be assured that Polarized routing can be used in RRGs in all those cases of interest for their application to big datacenters and supercomputers. 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 Algorithm 1 can be successfully applied. The number of routers in current and forthcoming deployments, and their radix, easily meets this condition.

Moreover, Polarized routes can be found by Algorithm 1 for many other interesting topologies. This includes well-known graphs, such as cycles and complete graphs, and also their Cartesian products, which are respectively, n-dimensional tori and Hamming topologies. In addition, it can be employed by any graph of diameter 2 without superfluous vertices whose removal does not change any of the distances of other vertices. This implies that Moore graphs [24], MMS [23] [3], and projective graphs (Brown–Erdős–Rényi) [6], [12], [5], [7] can benefit from Algorithm 1. In most common topological configurations of dragonflies, as palm-tree [14], the polarized routing algorithm can be used as defined in this paper.

We provide next, partial evaluations for a subset of these notable low-diameter topologies. It includes a dragonfly, a 3D Hamming graph (flattened butterfly), a projective network and a slimfly. Their main parameters are described in Table 4. Figure 8 plots the latency curves for each topology when managing a moderately adversarial traffic. All the plots include, at least, minimal and Valiant routing as references. As it can be seen, Polarized is notably superior in three of the topologies and is even with the best routing known for Hamming graphs.

The 3D Hamming graph is a topology that is currently in the spotlight. Different recent research works propose using this kind of topologies when silicon photonics become widely adopted in future network deployments [10]. Polarized routing is also compared with Omni-Dimensional routing [22], as best reference for this topology. Polarized and Omni-Dimensional routing provide almost the same performance results. However, a deeper comparison of these routings shows that the routes used by both are different, even when providing the same throughput. This is expected since Omni-Dimensional has been specifically designed for Hamming networks and Polarized has been conceived as a general solution.

6 Conclusions

Polarized routes preserve symmetry and avoid undoing hops, which translates into better performance. As has been shown, these routes can be built using a simple distributed greedy algorithm. This algorithm can be used in almost any direct topology, and it can be determined if a particular topology can benefit from it. 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 performance gains compared to previous solutions of more than 30% in all the traffic patterns tested. 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 at the University of Cantabria, 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] Zaid ALzaid, Saptarshi Bhowmik, and Xin Yuan. Multi-path routing in the Jellyfish network. In *2021 IEEE International Parallel and Distributed Processing Symposium Workshops (IPDPSW)*, pages 832–841, 2021.

[3] 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.

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

[5] Dhananjay Brahme, Onkar Bhardwaj, and Vipin Chaudhary. SymSig: A low latency interconnection topology for HPC clusters. In *High Performance Computing (HiPC), 2013 20th International Conference on*, pages 462–471, December 2013.

[6] William G Brown. On graphs that do not contain a Thomsen graph. *Canad. Math. Bull*, 9(5):281–285, 1966.

[7] Cristóbal Camarero, Carmen Martínez, Enrique Vallejo, and Ramón Beivide. Projective networks: Topologies for large parallel computer systems. *IEEE Transactions on Parallel and Distributed Systems*, 28(7):2003–2016, July 2017.

[8] CAMINOS. Technical report.

[9] Gary Chartrand and Linda Lesniak. *Graphs and Digraphs*. California Wadsworth and Brooks, 2nd edition, 1986.

[10] Qixiang Cheng, Meisam Bahadori, Madeleine Glick, Sébastien Rumley, and Keren Bergman. Recent advances in optical technologies for data centers: a review. *Optica*, 5(11):1354–1370, Nov 2018.

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

[12] Paul Erdős, Alfréd Rényi, and Vera T. Sós. On a problem of graph theory. *Studia Sci. Math. Hungar*, 1:215–235, 1966.

[13] Mario Flajslik, Eric Borch, and Mike A Parker. Megafly: A topology for exascale systems. In *International Conference on High Performance Computing*, pages 289–310. Springer, 2018.

[14] Marina García. *Routing mechanisms for dragonfly interconnection networks*. PhD thesis, University of Cantabria, 2014.

[15] Klaus D. Günther. Prevention of deadlocks in packet-switched data transport systems. *IEEE Transactions on Communications*, 29(4):512–524, 1981.

[16] Raj Jain, Dah-Ming Chiu, and W. Hawe. A quantitative measure of fairness and discrimination for resource allocation in shared computer systems. *CoRR*, cs.NI/9809099, 1998.

[17] Nan Jiang, John Kim, and William J Dally. Indirect adaptive routing on large scale interconnection networks. In *ISCA '09: 36th International Symposium on Computer Architecture*, pages 220–231, 2009.

[18] John Kim, William J. Dally, and Dennis Abts. Flattened butterfly: a cost-efficient topology for high-radix networks. In *Proceedings of the 34th annual international symposium on Computer architecture*, ISCA '07, pages 126–137, New York, NY, USA, 2007. ACM.

[19] 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.

[20] Michihiro Koibuchi, Hiroki Matsutani, Hideharu Amano, D. Frank Hsu, and Henri Casanova. A case for random shortcut topologies for HPC interconnects. In *Proceedings of the 39th Annual International Symposium on Computer Architecture*, ISCA '12, pages 177–188, Washington, DC, USA, 2012. IEEE Computer Society.

[21] Vijay Lakamraju, Israel Koren, and C. Mani Krishna. Filtering random graphs to synthesize interconnection networks with multiple objectives. *Parallel and Distributed Systems, IEEE Transactions on*, 13(11):1139–1149, November 2002.

[22] Nie McDonald, Mikhail Isaev, Adriana Flores, Al Davis, and John Kim. Practical and efficient incremental adaptive routing for HyperX networks. In *Proceedings of the International Conference for High Performance Computing, Networking, Storage and Analysis*, SC '19, New York, NY, USA, 2019. Association for Computing Machinery.

[23] Brendan D McKay, Mirka Miller, and Jozef Širáň. A note on large graphs of diameter two and given maximum degree. *Journal of Combinatorial Theory, Series B*, 74(1):110–118, 1998.

[24] Mirka Miller and Jozef Širáň. Moore graphs and beyond: A survey of the degree/diameter problem (2nd ed). *The Electronic Journal of Combinatorics*, 5 2013.

[25] Md Atiqul Mollah, Peyman Faizian, Md Shafayat Rahman, Xin Yuan, Scott Pakin, and Mike Lang. A comparative study of topology design approaches for HPC interconnects. In *2018 18th IEEE/ACM International Symposium on Cluster, Cloud and Grid Computing (CCGRID)*, pages 392–401, 2018.

[26] Alexander Shpiner, Zachy Haramaty, Saar Eliad, Vladimir Zdornov, Barak Gafni, and Eitan Zahavi. Dragonfly+: Low cost topology for scaling datacenters. In *2017 IEEE 3rd International Workshop on High-Performance Interconnection Networks in the Exascale and Big-Data Era (HiPINEB)*, pages 1–8. IEEE, 2017.

[27] Arjun Singh. *Load-Balanced Routing in Interconnection Networks*. PhD thesis, Stanford University, 2005.

[28] 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 17–17, Berkeley, CA, USA, 2012. USENIX Association.

[29] Angelika Steger and Nicholas C Wormald. Generating random regular graphs quickly. *Combinatorics, Probability and Computing*, 8(04):377–396, 1999.

[30] Leslie G. Valiant and Gordon J. Brebner. Universal schemes for parallel communication. In *Proceedings of the thirteenth annual ACM symposium on Theory of computing*, STOC '81, pages 263–277, New York, NY, USA, 1981. ACM.

[31] Jin Y Yen. An algorithm for finding shortest routes from all source nodes to a given destination in general networks. *Quarterly of Applied Mathematics*, 27(4):526–530, 1970.

[32] Xin Yuan, Santosh Mahapatra, Wickus Nienaber, Scott Pakin, and Michael Lang. A new routing scheme for jellyfish and its performance with HPC workloads. In *Proceedings of the International Conference on High Performance Computing, Networking, Storage and Analysis*, pages 1–11, 2013.

[33] P. Yébenes, J. Escudero-Sahuquillo, P. J. García, F. J. Quiles, and T. Hoefler. Improving non-minimal and adaptive routing algorithms in slim fly networks. In *2017 IEEE 25th Annual Symposium on High-Performance Interconnects (HOTI)*, pages 1–8, 2017.