StepRoute - A MultiRoute Variant Based on Congestion Intervals

Ali Al-Shabibi
Kirchhoff-Institute for Physics
University of Heidelberg
Heidelberg, Germany
Email: ali.al-shabibi@cern.ch

Brian Martin
European Organization for Nuclear Research
Geneva, Switzerland
Email: brian.martin@cern.ch

***Abstract*—Congestion-aware routing protocols require network statistics which are timely and as precise as possible. Consequently, there is a need to represent congestion information efficiently and in a scalable manner. Based on our previous work, we propose a routing protocol, called StepRoute, which achieves these objectives, while retaining the functionality of routing according to local and remote network conditions. By classifying congestion values into different categories, we are able to deliver timely information to routers while retaining a meaningful estimate of the original statistical value. We compare our results with our previous work (MultiRoute) as well as shortest path only routing. We show that our new variant outperforms both shortest path and MultiRoute in terms of throughput. The protocol itself is media independent, but for test purposes we have employed Ethernet.**

***Keywords*—Communication systems, Protocols, Monitoring, Computer network performance, Communication system routing.**

I. Introduction & Previous Work

Redundant connections are common in modern networks. While these connections provide varying degrees of resiliency, they also deliver an opportunity for multipath protocols. Currently deployed routing protocols only consider shortest paths between any source-destination pair and seldom do they consider the congestion present within the network. Historically, attempts to deploy congestion-aware routing with the Arpanet [1] failed because of route flapping which would lead to out of order packets and therefore drastic performance degradation. Nevertheless, it has been shown that congestion control has a significant positive impact on routing performance [2].

Protocols, such as Open Shortest Path First (OSPF) [3], Routing Information Protocol (RIP) [4], or Interior Gateway Routing Protocol (IGRP) [5], all rely on the existence of a single path between any source-destination pair. Doing so causes them to always route packets for a given destination on the same path and thereby increase the build up of congestion. We consider networks where multiple paths exist, whether they are of equal length or within an acceptable margin of the shortest path. Traffic flows are then assigned to a particular path according to the congestion indication received from neighboring routers. In essence, we consider that both local and remote congestion information should be considered when performing a routing decision.

StepRoute is a multipath routing protocol which provides a lightweight mechanism to represent both local and remote congestion in a scalable and precise manner. The three components which make up StepRoute are:

1) **Path Construction:** By relying on the existence of a shortest path between any source and destination point, we have developed our multiple path discovery process. After establishing the shortest path cost (the reference cost), each alternate path is computed whose cost is within a reasonable delta of the reference cost. This ensures that the latency versus throughput trade off is respected. This process has been described in [6].
2) **In-Network Monitoring:** To ensure fresh and timely statistics the routers poll themselves, rather than having an external monitoring process poll them. The precision at which the congestion is represented is discussed in Section II-A, but we can safely say that our representation is significantly lighter than the one used in [7]. These statistics are then sent to neighboring routers via an aggregation protocol similar to [8], which enables the statistics to be distributed within the network efficiently.
3) **Routing Table Representation:** Each router is responsible for maintaining its own routing vector based on the congestion information of its local links and of neighboring routers. The congestion information sent, via the In-Network monitoring

 ISBN:978-1-61208-113-7

protocol, by neighboring routers must be interpreted correctly by the recipient router. We present a data structure, called a *routing mask,* which allows routers to interpret information correctly while enabling flexibility on the precision of the information it contains.

Based on the above components, routers initially discover the paths that are available to the different networks. Then, using the In-Network Monitoring protocol and the *routing masks*, routers are able to construct their routing table for each destination. It is important to notice that besides the path discovery the only factor in the routing decision is the congestion statistics, this enables the routers to update the routing tables as statistics become available. Therefore, we can precompute the routing tables which enables rapid next-hop lookups. Traffic is then grouped into flows which are identified by several parameters, and assigned to the port corresponding to their next-hop. The assignment of a flow to a port is immutable for the duration of that flow's lifetime. This simple approach avoids path oscillations and thus out of order packets.

Currently, routing protocols make inefficient or no use of congestion to route onto alternative paths. That said, there has been significant research in this field. Using Constrained Shortest Path First [9] over Multiprotocol Label Switching (MPLS) [10] is a traffic engineered [11] approach to load balanced routing, which requires the *a priori* knowledge of the traffic matrix. Our approach is generic and requires no *a priori* knowledge about the network topology nor traffic distribution. In [7], the authors propose a method which is similar to our approach but in which all the router-router link statistics for a given device are packed into a single value. We believe that such an approach causes a significant loss of precision and propose a method for providing statistics for all router-router links. In [12], the authors mainly address the problem of route oscillations and propose to route long-lived IP flows on different paths than short-lived ones, whereas we propose no such distinction.

The remainder of this paper is structured as follows, first we will give more detail about the components that make up StepRoute. Then, we will detail StepRoute's routing algorithm. Finally, we present StepRoute's results when compared to Shortest Path Routing and MultiRoute.

II. Component Description

Each of the components described above handles a different area of the overall routing protocol, some are performed at the initialization time while others run continuously.

A. In-Network Monitoring

When designing a congestion-aware routing protocol, it is crucial to deploy a mechanism which delivers statistics as quickly as possible. Clearly, using standard centralized monitoring tools like SNMP [13] or sFlow [14] would not be appropriate as the time to gather the statistics and redistribute them to the router would exceed any timing constraints.

In order to guarantee the freshness requirement expressed above, we have decided to implement our own monitoring protocol based on the idea presented in [15]. The main advantages of our approach are expressed below:

- **Distributed:** Each router polls itself locally and generates a value representing the current level of congestion.
- **Update on change:** Updates are only sent when a change in congestion occurs, similarly to [15], thereby reducing the overhead needed by the protocol.
- **No Flooding:** All updates are only sent to neighboring routers which do not forward them.

Relying on the property that routers update their interface counters frequently (based on our research, interface counters can be safely queried at one second intervals [16].), we compute the difference between two consecutive updates and use the following formula to derive a congestion value.

$$\gamma = \Phi \cdot \frac{\Delta}{\Gamma} \tag{1}$$

where Δ is the difference between two consecutive updates, Γ represents the capacity of the link and finally Φ is a constant, which represents the sensitivity of this measure; the larger it is, the more rapidly the congestion measure will increase.

Taking the result obtained from Equation 1, we classify the congestion value into a number of categories according to the degree of precision required (shown in Table I). This classification simplifies the route calculation process [17]. In particular, as we will see in Section III, it allows for a very simple routing algorithm.

Class 1	$0.0 < \gamma \leq \alpha(1)$
Class 2	$\alpha(1) < \gamma \leq \alpha(2)$
Class i	$\alpha(i-1) < \gamma \leq \alpha(i)$
Class P	$\alpha(P-1) < \gamma \leq \alpha(P) = 1.0$

TABLE I: Congestion Classification.

Classifying the congestion values allows us to simply represent the status of the network to neighboring

 ISBN:978-1-61208-113-7

routers. Consider the situation where we would want to have m classes of congestion where m can be expressed as 2^n, then we only n bits per router-router link are needed to represent these four possibilities. Therefore as the number of possible classes grows exponentially, the space required to represent them only grows linearly. This approach allows us to describe many different congestion levels while employing a lightweight method for describing them. Moreover, routers which receive this information do not need to know any extra information, such as link speed or duplex status, about the sending router.

A router then packs all the values corresponding to its router-router links into a data structure which will be represented in Section II-B.

B. Routing Table Representation

The statistics which are received by a router take the form of a sequence of bits. While this representation is extremely space efficient, it poses one major issue: how does a receiving router interpret this information? More precisely, each router may present multiple links to a destination and varying precision for any given router-router link, therefore a mechanism is needed to allow routers to accurately interpret this information.

We propose the notion of a *routing mask*, which relies on the ordering of the routing tables. The actual ordering relation can be arbitrary as long as all participating routers use the same one, for example our implementation orders entries in the routing table by destination network. Initially, each router sends its *routing mask* to all its neighbors and this is only done once unless there is a failure in which case the entire algorithm recomputes.

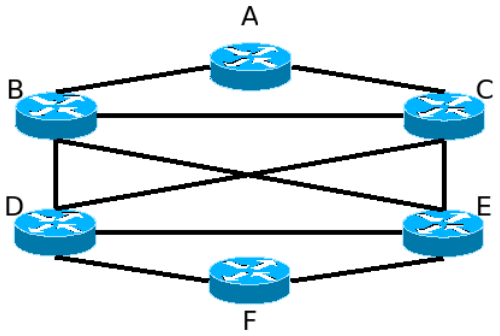

Fig. 1: The experimental network.

A *routing mask* consists of a sequence of zeros separated by ones. A consecutive sequence of zeros indicates remote links (plus the number of bits to represent the congestion class) which can be used to send packets to the same destination. A one indicates a transition to the next network in the routing table. The *routing mask* indicates to the receiving router the sequence of bits to expect for update messages received from remote routers.

		Destination					
		A	B	C	D	E	F
Source	A	L	1	1	2	2	2
	B	1	L	1	1	1	2
	C	1	1	L	1	1	2
	D	1	1	1	L	1	1
	E	2	1	1	1	L	1
	F	2	2	2	1	1	L

TABLE II: The connectivity table. Each entry shows the number of possible paths.

Considering the network given in Figure 1 and the connectivity map given in Table II, where each entry represents the number of paths to a destination network and L stands for locally connected, and finally assume that there are four classification classes and thus two bits are required to represent them. Under these condition router A would send the following routing mask to its neighbors **01001001000010000010000**, which indicates that A has two paths for networks D, E, and F. All updates send from routers will follow the same format, thereby enabling routers to immediately be able to interpret the incoming information and know exactly which part of the update vector is of interest to them.

III. Protocol Description

None of the components described above solve the main problem encountered by multipath protocols, namely, out of order packets which cause a drastic performance deterioration as is explained in [18]. We use the same approach that was used in MultiRoute [6] and in the OSPF variant Equal Cost MultiPath, where packets are classified into a flow by hashing the packet headers. We then use this classification to bind the flow to a given path. Once a flow is assigned to a path, it is bound to it for the duration of its lifetime and cannot be moved. Therefore, it is impossible to obtain out of order packets at the destination.

The three components described previously rely on each other to perform the objective function of the algorithm. The Routing Table representation depends on the In-Network monitoring protocol to deliver its message, and the monitoring protocol depends on the Path construction component to know to whom to send the statistics. All these components deliver their information to the routing algorithm, which takes routing decisions based on the available paths and their congestion status.

 ISBN:978-1-61208-113-7

StepRoute's routing algorithm requires the knowledge of the available paths (provided by the path construction component), local and remote statistics (provided by the In-Network Monitoring), and finally the *routing mask* (from the Routing Table Representation). The algorithm executes whenever new statistics are available, and re-populates the routing table for each destination. This pre-calculation enables rapid next hop lookup during the routers operation.

The routing algorithm consists of three cases corresponding to the state of the links local to a router:

- *Case I - All local paths uncongested.* In this case, the algorithm only looks at the statistics received from neighboring routers. Assuming there are multiple paths available, the router searches for the least congested path which is simple due to the classification of the congestion values discussed in Section II-A. Clearly, if the algorithm finds a remote path with is not at all congested, it immediately selects this path for forwarding. On the other hand, if all the remote congestion counts are equal or if none is found, the shortest path is selected.
- *Case II - Some local paths are congested while others are not.* This case is slightly more complex because a local path, even if it is carrying some traffic, may still be amongst one of the better options. This is due to the fact that remote paths, which lay beyond a completely uncongested link, may be completely congested. In this case, the algorithm ranks the candidate paths by summing their local congestion with the remote congestion. The path with the lowest congestion value is then selected. As with Case I, if there are multiple candidates, the shortest one is selected.
- *Case III - All local paths are completely congested.* This case is very much similar to the first case. The idea here is to look at the congestion values of remote routers and determine the least congested path, in an effort to use up all the available bandwidth. Again, if multiple candidates are found, the algorithm defaults to the shortest path.

Let us consider, as an example, the network given in Figure 1 and its associated Table II, when multiple flows enter at router F destined for network A. We also assume that each flow is long lived and that it immediately consumes half of the available bandwidth. There are four paths between networks F and A, namely F-D-B-A (1), F-D-C-A (2), F-E-C-A (3), and F-E-B-A (4), path (1) is considered to be the shortest followed by path (2) and so on. As there are only two distinct paths we can expect to double the network throughput with expect to a shortest path algorithm. It is important to note that, with respect to the real implementation the routing tables are pre-computed as statistics become available and not when a flow arrives.

When the first flow arrives, it is bound to path (1) as this path is considered to be the shortest, and due to Case I, this then causes an update from the routers F, D, and B indicating that their links are partially congested. When the next flow arrives, the decision is taken by Case II of the algorithm, and therefore the algorithm will consider the paths with next hop E as this link is not congested. The path (4) is excluded, because the link between B and A is congested due to the first flow. Therefore the second flow is bound to path (3). Upon arrival of the third flow, Case II will rank the available paths according to the congestion level and will choose path (2) as the link between D and C is not congested. Similarly, when the fourth flow arrives, Case II ranks the available paths again and picks path (4). As subsequent flows arrive at router F, Case III attempts to find available bandwidth to send the flow on and if this is not possible it sends it onto the shortest path.

IV. Results and Discussion

In this section, we present results obtained from our real world implementation. The experimental setup is shown in Figure 1 which operates at 100Mb/s and all link costs are equal to one. It was achieved using commodity routers running an OpenFlow [19] enabled firmware and using NOX [20] as the OpenFlow controller. The tests are performed by running 30-second UDP streams simultaneously between hosts connected to routers F and A using IPerf. The streams are run at 30 Mb/s and their number is increased to observe the behavior of StepRoute. We compare StepRoute (SR) (using four classes of congestion) with shortest path routing (SP), and MultiRoute(MR). Due to space requirements we are only able to present one set of results.

Figure 2 shows the performance of shortest path routing, MultiRoute and StepRoute under the scenario described above. Shortest Path routing is first to be limited, this is because for each flow SP routes onto the same path which means that by the third flow the path is nearly at saturation. MultiRoute performs better than SP, but still worse that SR. This is due to the fact that MR only uses a single bit to represent congestion and therefore a link may be marked as congested while it still has "spare" bandwidth and thus the protocol avoids using it. Finally, StepRoute does not suffer from this problem, because it classifies congestion into several classes, it has a finer control over the congestion levels of the different paths from F to A.

 ISBN:978-1-61208-113-7

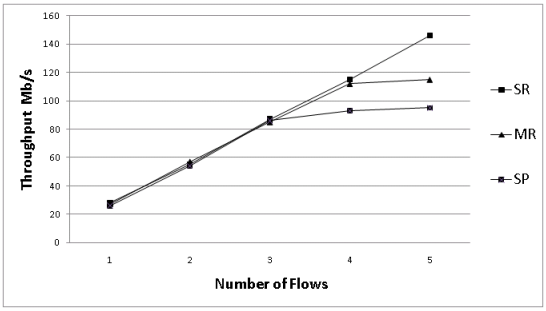

Fig. 2: StepRoute versus Shortest Path Routing and MultiRoute.

V. Conclusion and Future Work

In this paper, we have shown that by using congestion values, we can increase the overall throughput of a network. By using an in band monitoring protocol, we are able to deliver statistical information to routers in a timely manner. The use of classes of congestion greatly increases the performance of a protocol. More importantly, we are able to represent these classes efficiently with respect to a minimal one-bit marking. Our *routing mask* approach reduces to a minimum the amount of overhead required to signal router with detailed information about the network.

The results show that StepRoute performs significantly better than shortest path routing, and to our previous work; MultiRoute. Our results show that StepRoute is a promising protocol, but more work is needed to determine how well StepRoute scales when deployed onto a large-scale network. Also, it would be interesting to experiment with full-mesh traffic to observe StepRoute's behavior.

Future work is required to better understand the effects of the different parameters which make up StepRoute. Also, a future implementation of the *routing masks* would be to extend them to describe entire paths, from a source to a destination.

References

[1] A. Khanna and J. Zinky, "The revised arpanet routing metric," *SIGCOMM Comput. Commun. Rev.*, vol. 19, no. 4, pp. 45–56, 1989.

[2] H. Rudin and H. Mueller, "Dynamic routing and flow control," *Communications, IEEE Transactions on*, vol. 28, no. 7, pp. 1030–1039, Jul 1980.

[3] J. Moy, "Ospf version 2," United States, 1998.

[4] C. L. Hedrick, "RFC 1058: Routing information protocol," Jun. 1988.

[5] B. Albrightson and J. Boyle, "Eigrp-a fast routing protocol based on distance vectors," in *Proc. Networld/Interop 94*, 1994.

[6] A. Al-Shabibi and B. Martin, "Multiroute - a congestion-aware multipath routing protocol," in *High Performance Switching and Routing (HPSR), 2010 International Conference on*, jun. 2010, pp. 88–93.

[7] I. Gojmerac, T. Ziegler, F. Ricciato, and P. Reichl, "Adaptive multipath routing for dynamic traffic engineering," *IEEE GLOBECOM*, vol. 6, pp. 3058 – 3062, Dec. 2003.

[8] R. Stadler, M. Dam, A. Gonzalez, and F. Wuhib, "Decentralized real-time monitoring of network-wide aggregates," in *LADIS*, New York, 2008, pp. 1–6.

[9] J.-L. L. Roux, J.-P. Vasseur, and J. Boyle, "Requirements for Inter-Area MPLS Traffic Engineering," RFC 4105 (Informational), Internet Engineering Task Force, June 2005. [Online]. Available: http://www.ietf.org/rfc/rfc4105.txt

[10] E. Rosen, A. Viswanathan, and R. Callon, "Multiprotocol label switching architecture," United States, 2001.

[11] D. Awduche, A. Chiu, A. Elwalid, I. Widjaja, and X. Xiao, "Overview and principles of internet traffic engineering," United States, 2002.

[12] A. Shaikh, J. Rexford, and K. G. Shin, "Load-sensitive routing of long-lived ip flows," in *SIGCOMM '99: Proceedings of the conference on Applications, technologies, architectures, and protocols for computer communication*. New York, NY, USA: ACM, 1999, pp. 215–226.

[13] J. D. Case, M. Fedor, M. L. Schoffstall, and J. Davin, "Simple network management protocol (snmp)," United States, 1990.

[14] M. Wang, B. Li, and Z. Li, "sflow: towards resource-efficient and agile service federation in service overlay networks," in *Distributed Computing Systems, 2004. Proceedings. 24th International Conference on*, 2004, pp. 628–635.

[15] A. Prieto and R. Stadler, "A-gap: An adaptive protocol for continuous network monitoring with accuracy objectives," *Network and Service Management, IEEE Transactions on*, vol. 4, no. 1, pp. 2–12, June 2007.

[16] S. Batraneanu, A. Al-Shabibi, M. Ciobotaru, M. Ivanovici, L. Leahu, B. Martin, and S. Stancu, "Operational model of the atlas tdaq network," *Nuclear Science, IEEE Transactions on*, vol. 55, no. 2, pp. 687 –694, april 2008.

[17] S. Bahk and M. El Zarki, "Dynamic multi-path routing and how it compares with other dynamic routing algorithms for high speed wide area network," *SIGCOMM Comput. Commun. Rev.*, vol. 22, no. 4, pp. 53–64, 1992.

[18] D. Thaler and C. Hopps, "Multipath issues in unicast and multicast next-hop selection," United States, 2000.

[19] N. McKeown, T. Anderson, H. Balakrishnan, G. Parulkar, L. Peterson, J. Rexford, S. Shenker, and J. Turner, "Openflow: enabling innovation in campus networks," *SIGCOMM Comput. Commun. Rev.*, vol. 38, no. 2, pp. 69–74, 2008.

[20] N. Gude, T. Koponen, J. Pettit, B. Pfaff, M. Casado, N. McKeown, and S. Shenker, "Nox: towards an operating system for networks," *Computer Communication Review*, vol. 38, no. 3, pp. 105–110, 2008.

 ISBN:978-1-61208-113-7