

vFTree - Fat-Tree Routing with Virtual Lanes in InfiniBand

Wei Lin Guay¹, Bartosz Bogdanski¹, Sven-Arne Reinemo¹ and Olav Lysne^{1,2}

¹Simula Research Laboratory ²University of Oslo
E-mail: {weilin, bartoszb, svenar, olavly}@simula.no

1. Background and Motivation

Even though the bisectional bandwidth in a fat-tree is constant, **hot-spots** are still possible. Such a situation may be alleviated through adaptive routing or congestion control, but these methods are not yet fully supported in InfiniBand technology. We propose an inexpensive approach for reducing the hot-spot problem in fat-trees, which is based on the application of virtual lanes.

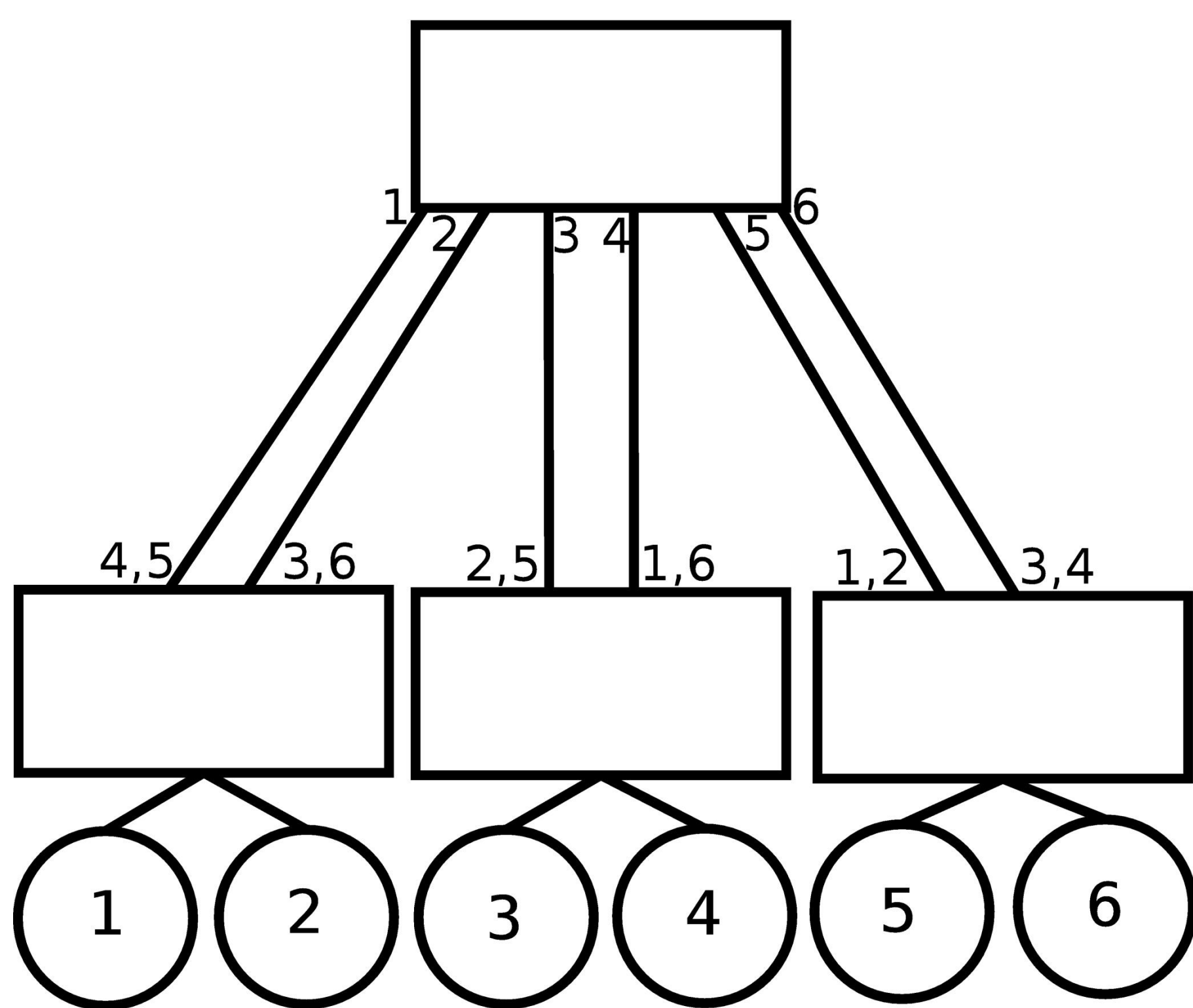


Fig. 1 A fat-tree with a given path distribution in the upward and downward directions.

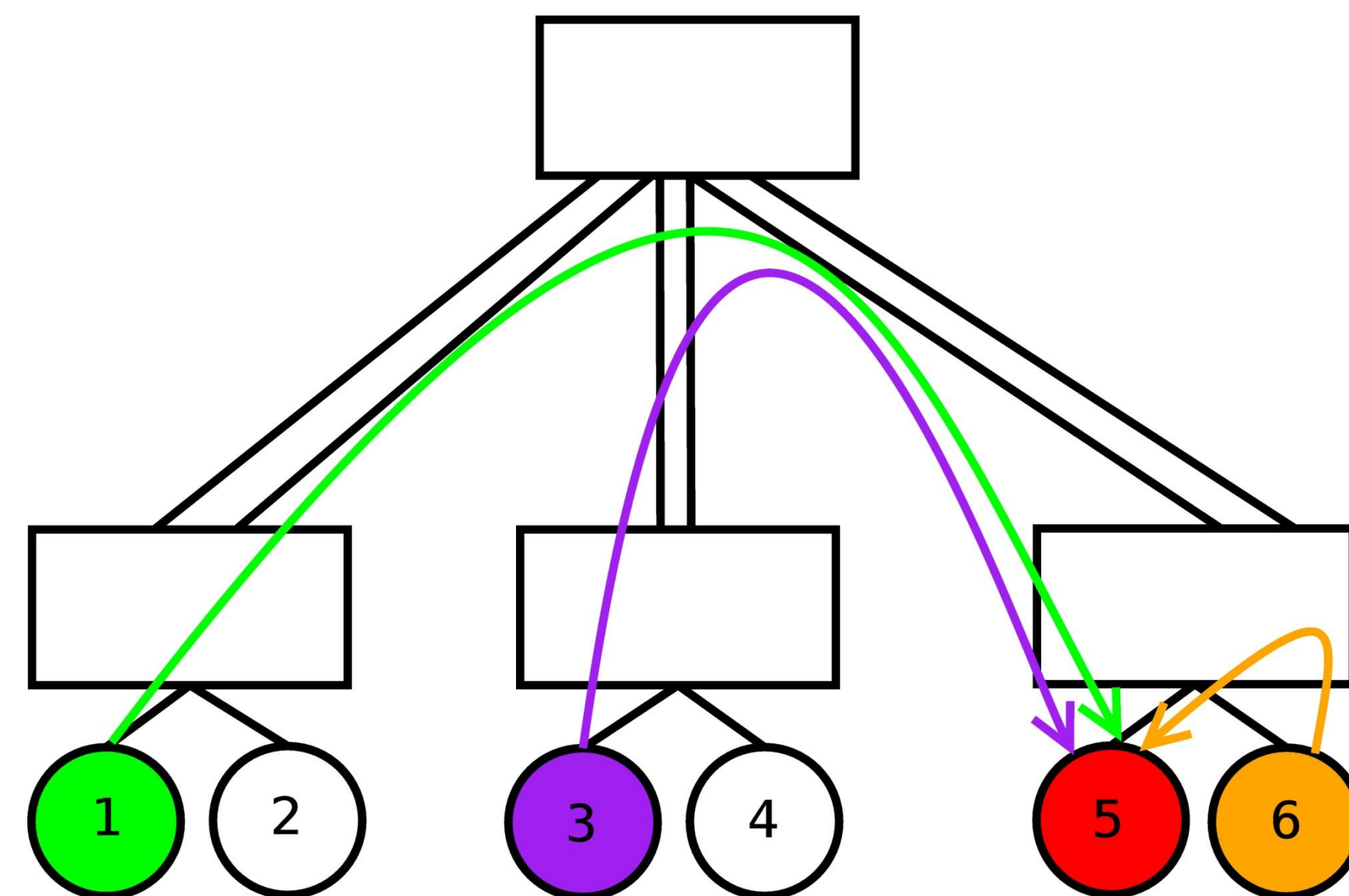


Fig. 2 A hot-spot scenario.

2. What is the Problem?

When a **hot-spot** (node 5 in Fig. 2) exists in a network, the flows destined for the hot-spot might reduce the performance for other flows called victim flows (node 2 and 4), not destined to the hot-spot. This is due to the head-of-line blocking caused by the congested hot-spot. Furthermore, the static routing approach, which is predominant in most interconnection networks, including InfiniBand, makes it hard to use the idle resources to increase the bandwidth.

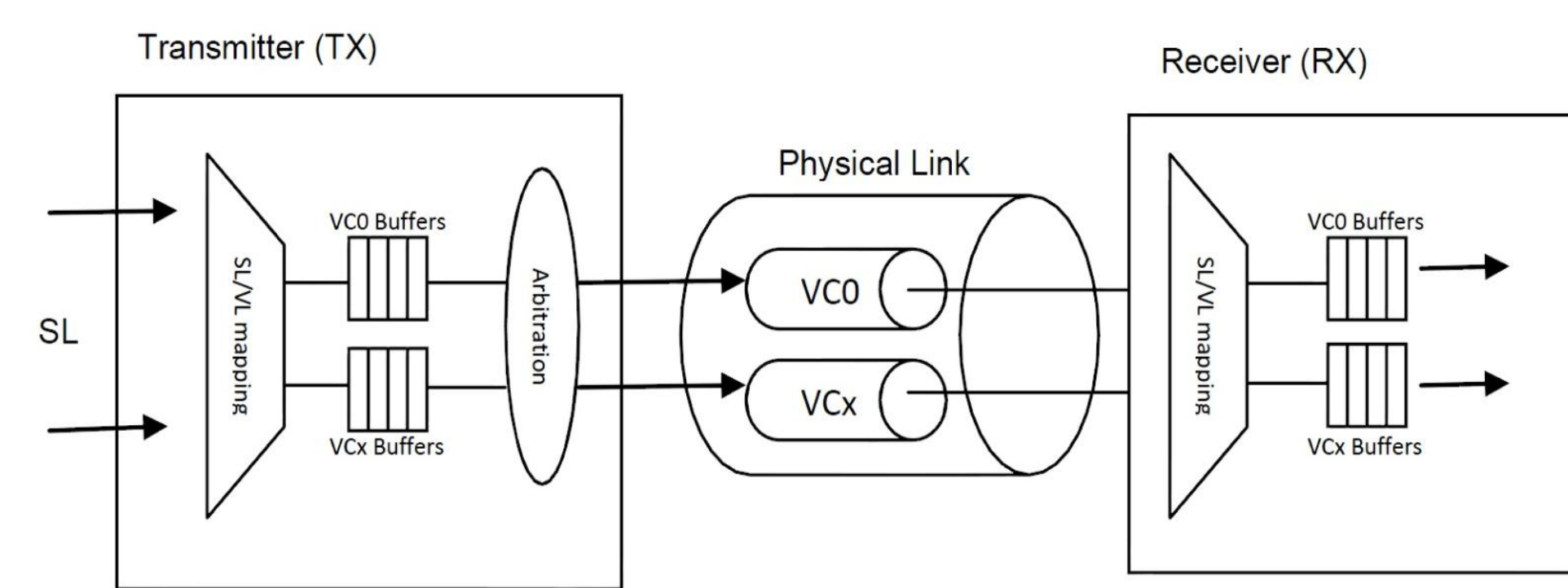


Fig. 3 An unidirectional link with virtual lanes.

3. Virtual Lanes

As illustrated in Fig. 3, the virtual lane mechanism creates several virtual channels (channel states and flit buffers) within a single physical link. Through multiplexing multiple independent data streams onto a single physical link, it provides Quality of Service support and avoids head-of-line blocking.

4. Our Approach

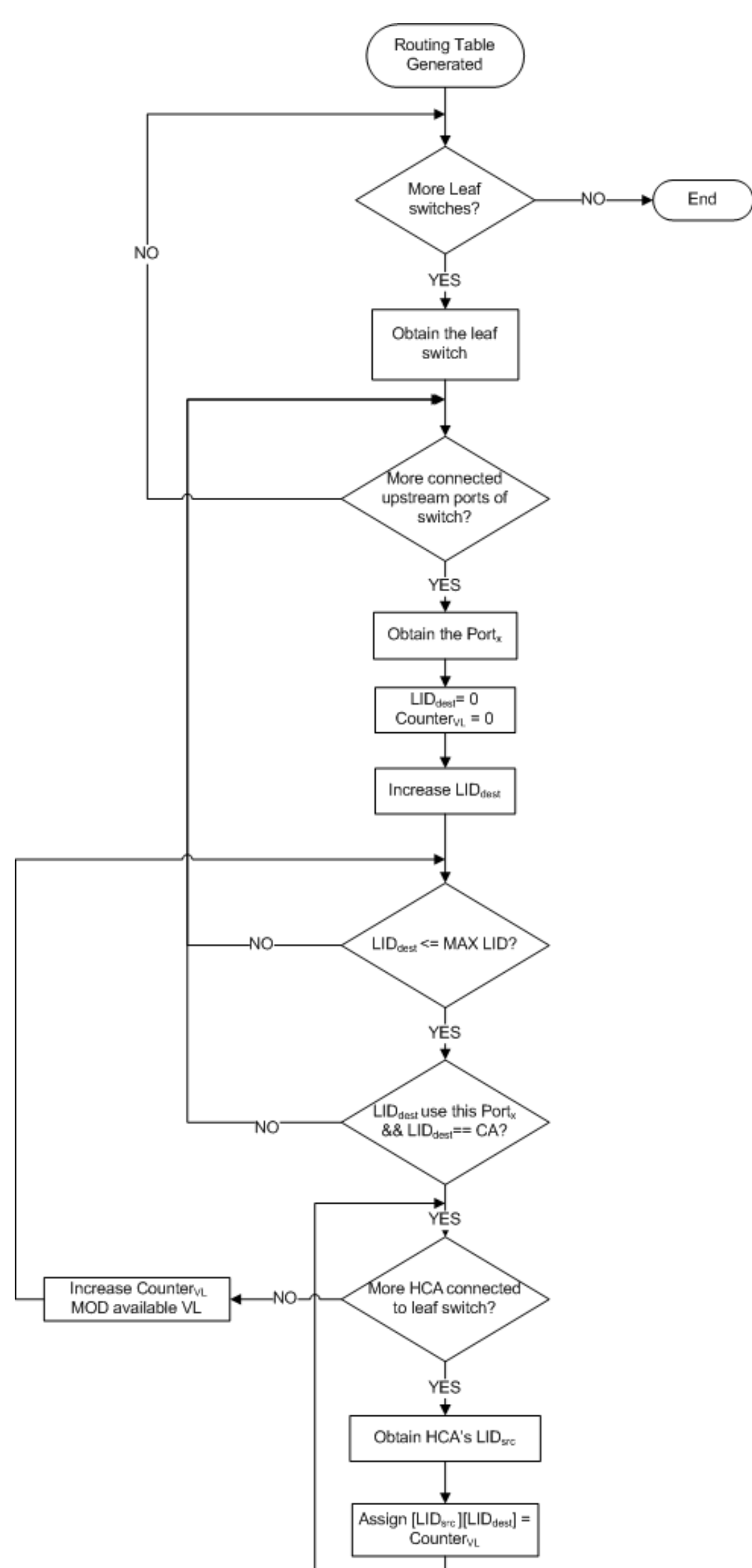


Fig. 4 The vFTree Routing Flow Chart.

Our vFTree Routing Algorithm

Our proposed algorithm will distribute the unused VLs evenly across multiple destination paths that share the same link for upstream ports, as described in Fig. 4 (i.e for link 1, VL0 for destination 5 and VL1 for destination 4 in Fig. 1).

5. Results

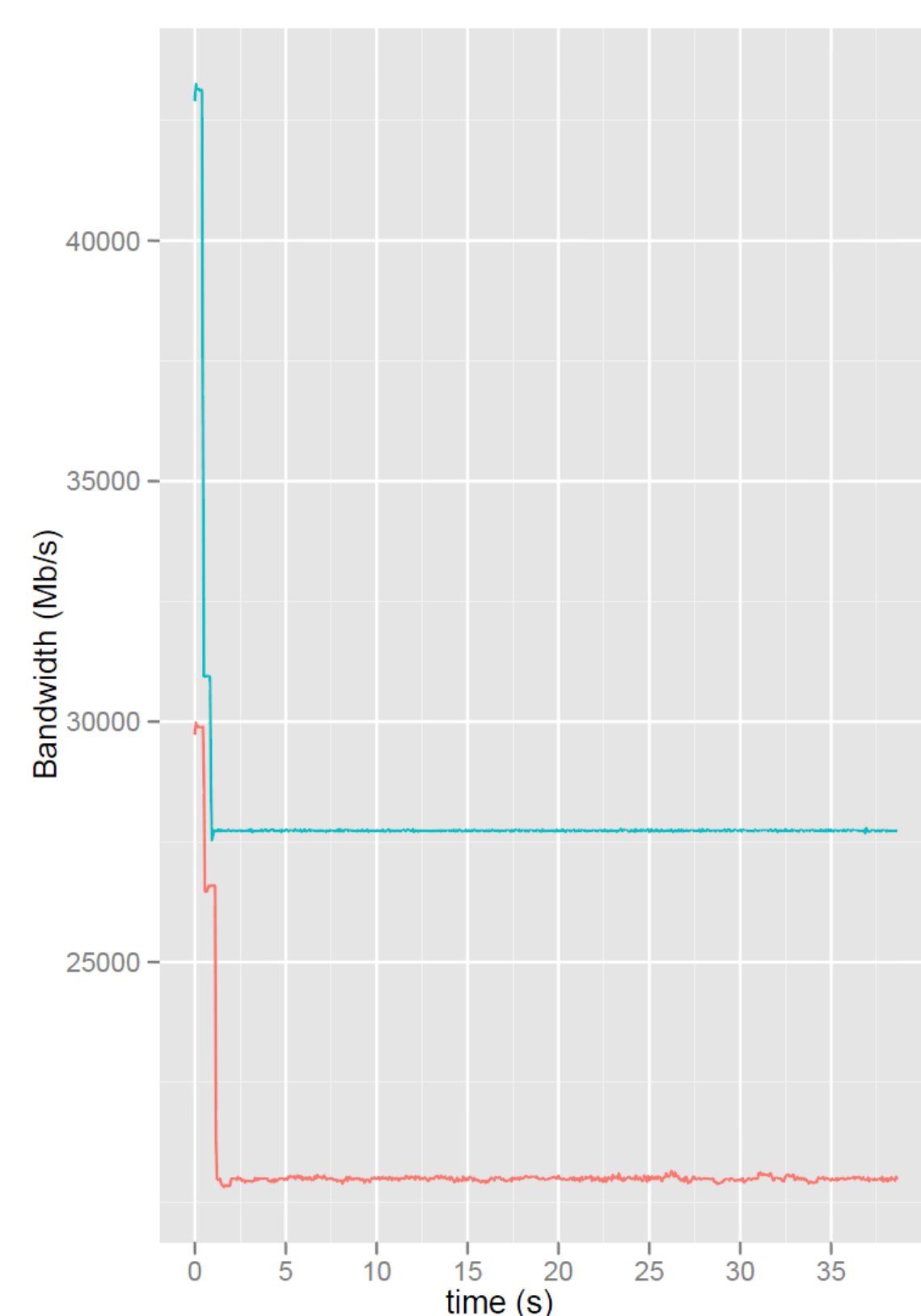
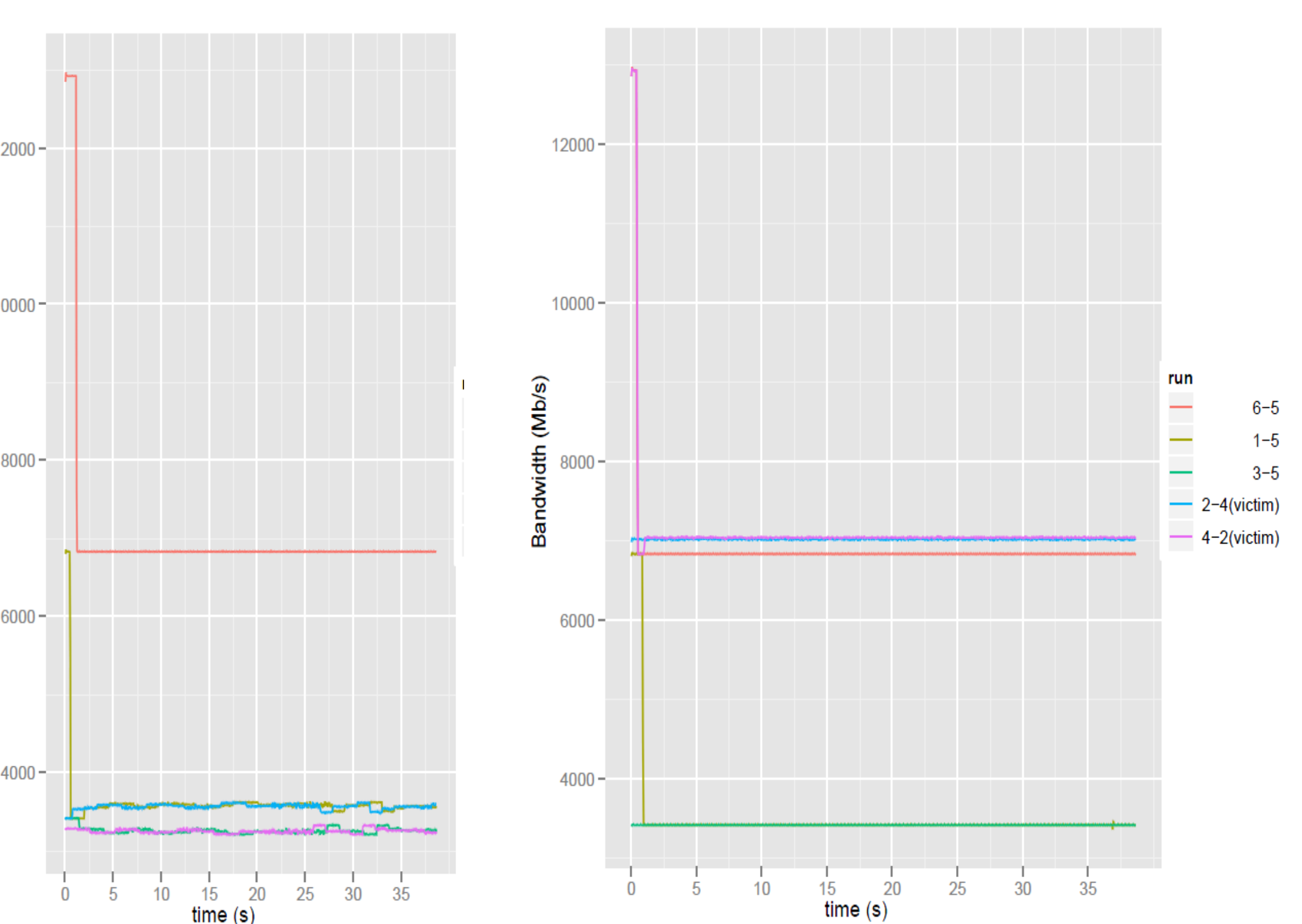


Fig. 5 Throughput for the conventional fat tree algorithm and vftree routing algorithm with VLs.



(a) Throughput for the conventional fat-tree routing algorithm. (b) Throughput for the vftree algorithm with 2 virtual lanes.

Fig. 6 Per flow throughput for the conventional fat tree algorithm and vftree algorithm with VLs.

Performance Evaluation

We performed a simple experiment on a small-scale fat-tree cluster that consists of 6 nodes and 4 switches as shown on Fig. 1. Fig. 5 shows that the total network throughput using 2 VLs has increased by 35% when compared to the original performance (with 1 VL). The following synthetic traffic pattern, {1->5, 2->4, 3->5, 4->2, 6->5}, was selected to generate a hot-spot in our test bed. Node 5 is the hot spot and multiple nodes (1,3 and 6) are simultaneously sending to it. As shown in Fig. 6(a), the congestion in the network blocks and slows down the traffic on physical links 1 and 3. This makes nodes 2 and 4 victims of the congestion where the throughput is less than half of the bandwidth that they are supposed to get. On the other hand, with our vftree routing that uses two virtual lanes, the congested traffic will not affect the victims and both of them can continue to utilize half of the link bandwidth, which is shown in Fig. 6(b).

6. Conclusion

We proposed an inexpensive software based solution to reduce the congestion in the InfiniBand cluster. From our simple experiment, it is evident that our mechanism is capable of reducing the congestion effects for the synthetic traffic patterns. Future work includes studying this mechanism in a larger cluster with application traffic.