# Routing Stability in Hybrid Software-Defined Networks

Shih-Hao Tseng*, Kevin Tang*, Gagan L. Choudhury†, and Simon Tse†

*School of Electrical and Computer Engineering, Cornell University †AT&T Labs

Supported by NSF grant CNS-1544761

## Objective and Key Contributions

- We consider the stability of a hybrid software-defined network (SDN) in which a centralized controller routes the traffic along with the legacy routers.
- We propose a simple algorithmic scheme to ensure the routing pattern given by the centralized controller is consistent with the legacy routers.
- Three algorithms are proposed for the scheme and their trade-offs are discussed.

## Introduction

ISPs tend to upgrade their legacy networks to support SDN instead of switching to pure SDN directly, which results in a hybrid SDN. Two issues then need to be handled appropriately in a hybrid design:

- Dual control stability: As an incremental improvement, without modifying the distributed routing, we want to ensure the centralized controller won't cause persistent routing flapping.
- Failure resilience: The system should be stable under data plane failure.

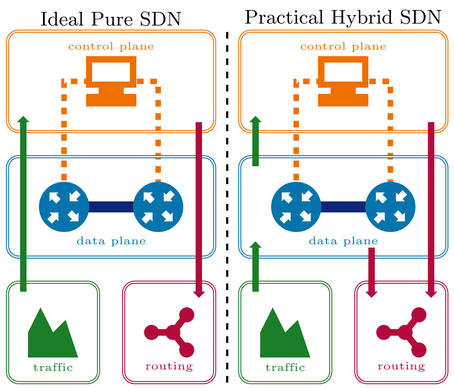
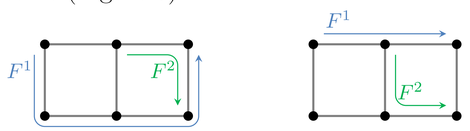



Figure 1: In a hybrid SDN, the distributed routing continues functioning along with the centralized routing. Also, the centralized controller relies on the data plane to collect information from routers instead of having a direct access to each router.

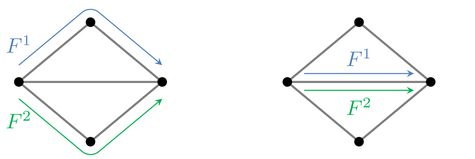
## Background

Adding a centralized controller can improve the system efficiency (Figure 2), while persistent route flapping can happen if the centralized controller and the legacy routers take turns to pursue different routing patterns (Figure 3).




(a) Stalemate (b) Optimal solution

Figure 2: Distributed routing can end up in stalemate and poor resource utilization, and the system can be better utilized by introducing a centralized controller to reroute.




(a) Routing pattern preferred by the centralized controller (b) Routing pattern preferred by the local routers

Figure 3: Inconsistency between the centralized controller and the local routers can cause instability.

## System Setup

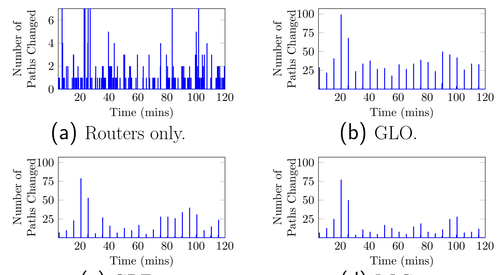
We consider an MPLS source routing network. The traffic in the network belongs to two different priority classes. Each distributed router performs constrained shortest-path first (CSPF) routing, and the found route will be overwritten only when

- The centralized controller inserts a route.
- The existing route is no longer feasible.
- A new feasible route with strictly lower cost exists.

The centralized controller collects the following data plane information:

- The path for each traffic (PCEP, RFC 5440 [1]).
- The link information including capacity, cost metric, and the aggregated traffic rate on the link per priority class (BGP-LS, RFC 7752 [2]).

## Simulation Results




(a) Routers only. (b) GLO. (c) GRE. (d) LOC.

Figure 4: Number of paths changed during $5-120$ mins after the system starts.

Using the trace from a modified but realistic wide-area network (WAN), we measure the number of paths changed in Figure 4. Starting with a stable routing pattern, distributed routing needs several cycles before reaching a stable routing pattern whenever traffic fluctuates (Figure 4(a)). While the proposed centralized algorithms can quickly stabilize the network (Figure 4(b)-(d)).

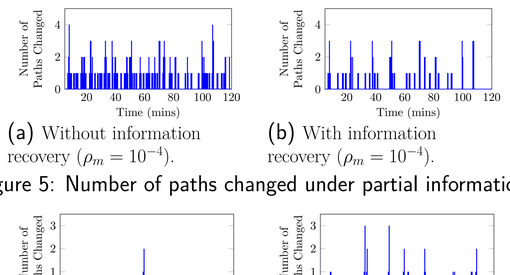
## Routing Stability

We propose the following stability definition:

**Definition**: A routing pattern is *stable* in a hybrid SDN if it won't be changed by any distributed router after the centralized controller deploys it (i.e., it is the optimal solution to each distributed CSPF).

A stable routing pattern can be achieved by finding a stable routing pattern iteratively for each priority class, from the higher prioritized to the lower. Such framework requires finding a stable routing pattern for the flows in the same priority class. That can be done by three algorithms:

- Global optimization (GLO): find the shortest aggregated path length.
- Greedy (GRE): solve CSPF one at a time.
- Local Search (LOC): improve a routing pattern until no shorter path exists.




(a) Without information recovery ($\rho_m = 10^{-4}$). (b) With information recovery ($\rho_m = 10^{-4}$).

Figure 5: Number of paths changed under partial information.




(a) Without information recovery ($\rho_i = 10^{-3}$). (b) Without information recovery ($\rho_i = 10^{-2}$).

Figure 6: Number of paths changed under inconsistent information. No path is changed if adopting information recovery.

We also design the information recovery procedure to make the system robust to different information loss scenarios: partial information (losing PCEP or BGP-LS messages) and inconsistent information (PCEP and BGP-LS are inconsistent due to asynchronous reporting time).

## Comparison of the Algorithms

Table 1 summarizes the tradeoffs.

| | GLO | GRE | LOC |
|---|---|---|---|
| Cost Effectiveness | optimal | depending on the order | depending on the order and the initial pattern |
| Time Complexity | in general $O(2^{\|N\|})$ | $O(\|N\|)$ | $O(\|N\|^2)$ |
| Initialization Allowance | no | no | yes |

Table 1: Comparison of the Algorithms.

## References

[1] JP Vasseur and JL Le Roux. RFC 5440: Path computation element (PCE) communication protocol (PCEP), 2009.

[2] H Gredler, J Medved, S Previdi, A Farrel, and S Ray. RFC 7752: North-bound distribution of link-state and traffic engineering (TE) information using BGP, 2016.