Scheduling Relaxed Loop-Free Updates Within Tight Lower Bounds in SDNs

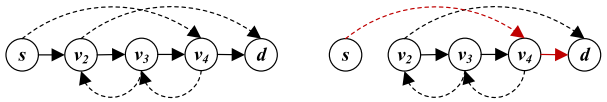
Hao Zhou, Xiaofeng Gao, *Member, IEEE*, Jiaqi Zheng, *Member, IEEE, ACM*, and Guihai Chen, *Member, IEEE*

***Abstract*— We consider a fundamental update problem of avoiding forwarding loops based on the node-ordering protocol in Software Defined Networks (SDNs). Due to the distributed data plane, forwarding loops may occur during the updates and influence the network performance. The node-ordering protocol can avoid such forwarding loops by controlling the update orders of the switches and does not consume extra flow table space overhead. However, an $\Omega(n)$ lower bound on the number of rounds required by any algorithm using this protocol with loop-free constraint has been proved, where n is the number of switches in the network. To accelerate the updates, a weaker notion of loop-freedom — relaxed loop-freedom — has been introduced. Despite that, the theoretical bound of the node-ordering protocol with relaxed loop-free constraint remains unknown yet. In this article, we solve a long-standing open problem: how to derive $\omega(1)$-round lower bound or to show that $O(1)$-round schedules always exist for the relaxed loop-free update problem. Specifically, we prove that any algorithm needs $\Omega(\log n)$ rounds to guarantee relaxed loop freedom in the worst case. In addition, we develop a fast relaxed loop-free update algorithm named Savitar that touches the tight lower bound. For any update instance, Savitar can use at most $2\lfloor \log_2 \ n \rfloor - 1$ rounds to schedule relaxed loop-free updates. Extensive experiments on Mininet using a Floodlight controller show that Savitar can significantly decrease the update time, achieve near optimal performance and save over 30% of the rounds compared with the state of the art.**

***Index Terms*— Software defined networks, relaxed loop freedom, network updates.**

I. Introduction

SOFTWARE defined networking (SDN) [12] enables flexible and global network management by decoupling the control plane from the data plane and integrating the control plane into logically centralized controllers. Compared to the static architecture of traditional networks, the centralized intelligence of the controllers in SDNs can contribute to dynamic and more efficient network configurations to improve the network performance. Consequently, more enterprises such as Microsoft SWAN [14] and Google B4 [15] have started trying to use SDN to manage their data center networks.



(a) Example for strong loop-freedom (b) Example for relaxed loop-freedom

Fig. 1. Two notions of loop-freedom. The solid lines represent the old routing path op and the dashed lines represent the new routing path np.

Despite the centralized control plane, the data plane remains distributed. Network updates are associated with network availability and performance in traffic engineering and are frequently scheduled when network conditions change, e.g., switch upgrade, link failures or traffic variations. Due to the unpredictable update orders of the switches in the data plane, network updates may cause inconsistency and introduce forwarding loops that consist of a mixture of old rules and new rules into networks. Packets entering loops cannot be forwarded out until the loops are broken, which may result in packet drops and further influence the performance of delay-sensitive applications.

Avoiding such forwarding loops can guarantee the network correctness and availability during the updates. However, given the initial route and the final one, scheduling loop-free updates remains algorithmically challenging. The node-ordering protocol is a major mechanism to guarantee loop-freedom without incurring extra flow table space overhead. This protocol partitions the switches that need to be updated into multiple disjoint subsets, and requires that the updates of each subset of the switches will not introduce any loops. Therefore, the whole network updates can be completed by scheduling the updates of each subset one by one. Actually, such an update of one subset is called one *round* and hence the number of subsets is called the number of rounds, which can reflect the update time to some degree. On the one hand, a loop-free update algorithm should guarantee the correctness, i.e., compute how to partition the switches to avoid forwarding loops. On the other hand, this algorithm should minimize the number of rounds in order to schedule loop-free updates quickly.

Strong and relaxed loop-freedom are two different definitions for loop-freedom. Strong loop-freedom requires that no loop should occur at any time during updates. In Fig. 1(a),

Manuscript received November 10, 2019; revised June 4, 2020; accepted July 17, 2020; approved by IEEE/ACM TRANSACTIONS ON NETWORKING Editor P. Giaccone. Date of publication August 26, 2020; date of current version December 16, 2020. This work was supported in part by the National Key Research and Development Program of China under Grant 2019YFB2102200; in part by the National Natural Science Foundation of China under Grant 61872238, Grant 61802172, and Grant 61972254; in part by the CCF-Huawei Database System Innovation Research Plan under Grant CCF-Huawei DBIR2019002A; and in part by the CCF-Tencent Open Research Fund. *(Corresponding author: Xiaofeng Gao.)*

Hao Zhou, Xiaofeng Gao, and Guihai Chen are with the Shanghai Key Laboratory of Scalable Computing and Systems, Department of Computer Science and Engineering, Shanghai Jiao Tong University, Shanghai 200240, China (e-mail: h-zhou@sjtu.edu.cn; gao-xf@cs.sjtu.edu.cn; gchen@cs.sjtu.edu.cn).

Jiaqi Zheng is with the State Key Laboratory for Novel Software Technology, Department of Computer Science and Technology, Nanjing University, Nanjing 210023, China (e-mail: jzheng@nju.edu.cn).

Digital Object Identifier 10.1109/TNET.2020.3017771