Update Latency Optimization of Packet Classification for SDN Switch on FPGA

Chenglong Li[1], Tao Li[1], Junnan Li[1], Zilin Shi[1], and Baosheng Wang[1]
[1]School of Computer, National University of Defense Technology, Changsha, China
{lichenglong17, taoli, lijunnan, shizilin18, bswang}@nudt.edu.cn

***Abstract*—FPGA is widely used in real-time network processing such as packet classification in SDN switches due to high performance and programmability. BV-based approaches on FPGA provide a performance guarantee for multi-field packet classification, but no update latency guarantee. We thus present SplitBV for the efficient update by splitting the ruleset into sub-rulesets that can be performed in parallel. Results show that our approach can reduce 73% and 36% update latency on average for synthetic 5-tuple rules and OpenFlow1.0 rules respectively.**

I. Introduction

Due to its high performance with flexible reconfigurability, Field Programmable Gate Array (FPGA) has been utilized for accelerating OpenFlow-based [1] Software-Defined Networking (SDN) switches. The core problem of SDN switches is multi-field packet classification with match-action rules (called rulesets). SDN switches must achieve both fast packet classification and fast rule updates. The state-of-the-art Bit-Vector-based (BV-based) algorithm named TPBV [2] exploits hardware parallelism of FPGA to achieve line-speed lookup and implements microsecond-level data structure updates on FPGA. However, the worst-case update latency of TPBV algorithm is unbearable. Excessive update latency raises both process pressure of the control plane and losing packets on the forwarding plane. In this paper, we present SplitBV to ensure sufficiently low update latency for high-performance SDN switches on FPGA.

II. Proposed Scheme and Resuls

The update latency of BV-based approaches consists of two parts: (1) *propagation time*, the time from entering the pipeline to finding the updated location; (2) *processing time*, the time from the execution of the updated instruction at the updated location to the new rule effective. For large-scale rulesets, the propagation time caused by multi-level pipeline determines the worst-case update latency. SplitBV selects several distinguishable exact-bits to split the ruleset into independent sub-rulesets without rule replication. These diminutive sub-rulesets can be implemented parallelly in BV-based pipelines, thus reducing update latency tremendously. SplitBV contains a recursive algorithm for selecting split-bits to balance the size of these sub-rulesets. The worst-case update latency of SplitBV is determined by the largest sub-ruleset. Figure 1 shows the hardware framework of SplitBV. Each SE-p_i corresponds to a sub-ruleset, obtained by splitting on $field_i$ with p_i bits. SE-p_i associates different rules corresponding to

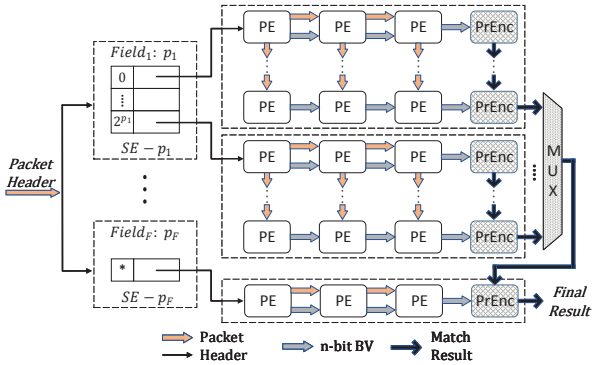

Fig. 1. Hardware Architecture of SplitBV.

2^{p_i} items of the HASH table to different horizontal pipelines subsequently. The bottom SE-p_F is prepared for special cases. Rules in each table item are matched using a two-dimensional pipeline consisted of processing elements (PEs) and priority encoders (PrEncs) in [2]. We use a multiplexer (MUX) to integrate the matching results of the SE-p_i and send it to PrEnc in the bottom horizontal pipeline. Table I shows the comparison of SplitBV with TPBV for 10K rules. SplitBV reduces worst-case update latency by an average of 73% and 36% for 5-tuple rules (Access Control List (ACL), Firewall (FW), and IP Chain (IPC)) and 12-tuple OpenFlow1.0 (OF1.0) rules, respectively. The cost of reducing update latency is the negligible expansion of memory consumption.

TABLE I
Comparison of Memory(Mem.) Consumption and Worst-case Update Latency(Lat.) for 10K Rules (s=4 and n=8).

Algorithm	ACL	FW	IPC	OF1.0
Mem. of TPBV (KB)	768.19	1707.76	588.94	794.83
Mem. of SplitBV (KB)	769.33	1710.04	589.95	796.31
Lat. of TPBV (µs)	15.49	33.99	11.96	7.09
Lat. of SplitBV (µs)	3.66	9.15	3.48	4.56

References

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

[2] Y. R. Qu and V. K. Prasanna, "High-performance and dynamically updatable packet classification engine on FPGA," *IEEE Trans. Parallel Distrib. Syst.*, vol. 27, no. 1, pp. 197–209, 2016.

2576-2621/20/$31.00 ©2020 IEEE
DOI 10.1109/FCCM48280.2020.00044