1 Summary

This paper proposes a step towards more flexible switches whose functionality is specified yet modifiable through a high-level language for programming protocol-independent packet processors, P4. The authors have designed P4 (Programming Protocol-Independent Packet Processors) to work in conjunction with software-defined networking (SDN) protocols like the popular OpenFlow that, by themselves, do not offer the degree of flexibility appealing to the broader community.

2 Strengths of the paper

1. The authors demonstrate astute foresight by: (a) noting the increasingly complex communication between switches and controllers as specified by a growing amount of header fields, and (b) observing advancements in chip designs to develop custom purpose-fitted ASICs. But they also state the difficulty related to (b) in that it is challenging to program next-generation switch chips via existing methods. And in doing so, they set up a clear demand for P4 and motivation for bringing it to existence: by increasing the abstraction for programming the network (i.e., make the lives of developers easier), it simplifies the implementation of the interface between the controller and switches (allowing increased flexibility).
2. By stating the three design goals of P4 (i.e., *configurability, protocol independence, and target independence*) early on in the paper, the reader knows the metric against which to consider the authors' evaluations later in the paper and what to prioritize throughout the read. These goals make obvious sense, as the authors highlight a lack of any method that achieves these currently.
3. I appreciate that the authors employed the *mTag* example throughout the paper to discuss various aspects of P4, from the concepts and formats to the compiler. Even though, unlike real-world parsers in networks that have more than just four states (e.g., hundreds more), for the purpose of this paper, the authors lost virtually nothing by using such a simple example because the ideas described herein can be applied just as easily for more complex parsers thereby portraying the power of P4.

3 Weakness of the paper

Why not compete with OpenFlow? Perhaps developing an accompanying tool to sort of reset the trajectory of OpenFlow rather than replacing it entirely is more pragmatic.

4 Future work opportunities

I wonder to what extent OpenFlow development has been impacted by P4. Is there a new interface that works even better with it?

Reference
P. Bosshart et al., "P4: Programming Protocol-Independent Packet Processors," SIGCOMM Comput. Commun. Rev., vol. 44, no. 3, pp. 87–95, Jul. 2014, doi: 10.1145/2656877.2656890.