Taurus: An Intelligent Data Plane

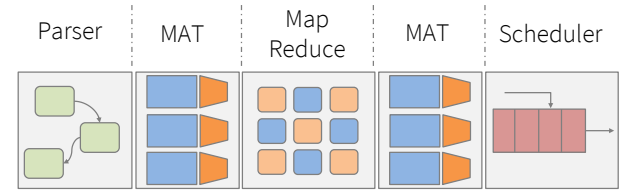
Tushar Swamy, Alexander Rucker, Muhammad Shahbaz,
Neeraja Yadwadkar, Yaqi Zhang, and Kunle Olukotun
Stanford University

ABSTRACT

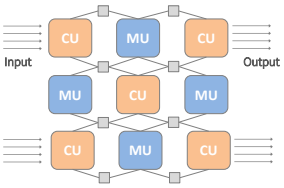
Emerging trends such as cloud computing, the internet of things, and augmented and virtual reality demand highly responsive, available, secure, and scalable networks to meet users' quality of experience expectations. Operators currently manage these networks and protocols using a variety of ad-hoc tools and scripts; however, the unpredictable and complex interactions between network conditions and workloads make such manual tuning difficult.

Machine learning (ML) can help approximate and automate these complex interactions that govern today's hyperscale datacenter networks [3, 4, 7]. Recent proposals generate ML models for networks to produce recommendations for policies like routing and congestion control [16]. At present, these models run on a logically-centralized control plane that infers learned polices, causing delays of tens of milliseconds when updating network devices [5, 9]. This is because modern reconfigurable switching devices (e.g., RMT [1]) lack the necessary operations (i.e., loops and multiplication) needed to run these ML models in the data plane. Therefore, for policies like anomaly detection where inputs to the ML model may vary over time (e.g., payload size or time-windowed features [15]), most packets—even of a single flow—need to traverse the control plane, thus, significantly increasing load on the controller and inflating flow latencies [11].

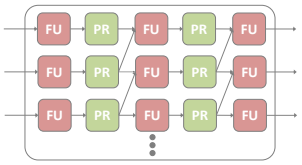
In this paper, we present *Taurus*, an intelligent data plane architecture for ML inference at line rate. Taurus extends the Protocol Independent Switch Architecture (PISA) [1, 8] by adding an ML-capable block with a map-reduce abstraction to the match-action table pipeline (Figure 1a). The map-reduce block receives pre-processed network and packet features from the preceding match-action tables and the parser, and feeds results to the following match-action tables for post processing to set the network action (e.g., drop, route, or encapsulate a packet based on the prediction). The design of the map-reduce block is based on a spatial SIMD architecture that can support a variety of ML models. It is composed of Compute Units (CU) and Memory Units (MU) interleaved in a grid, joined by a static interconnect (Figure 1b) [13]. CUs are composed of programmable Functional Units (FUs) and registers organized across lanes and stages; a CU can perform either a map, reduction, or both. This restriction allows high performance for regularly-structured applications (e.g., ML) with low configuration overhead.



(a) Modified PISA Pipeline



(b) Map-Reduce Block



(c) Compute Unit

Figure 1: Taurus Data Plane Architecture

		Perf.		Area		Power	
App	**Model**	GPkt/s	ns	mm^2	+%	mW	+%
Anomaly	SVM	1.00	68	4.59	6.1	263	1.1
Anomaly	DNN	1.00	362	8.80	11.7	506	2.0
Indigo	LSTM	0.08	380	17.73	23.6	1018	4.1

Table 1: Performance, area, and power overheads for three different application models. Overheads are calculated relative to a 300 mm^2 chip with 4 reconfigurable pipelines [6], each drawing an estimated 25 W.

Table 1 shows that that cost of adding ML models to a network data plane is small. Taurus can run simple models such as SVM-based anomaly detection [10] with as little as 6.1% area and 1.1% power overhead. The deep learning (DL) network [14] consumes more resources but the area and power utilization is still under 12% and 2%, respectively. Both models meet the high-end switch line rates of a billion packets per second (i.e., 1 GPkt/s). The third application, Indigo [16] is an endpoint application for congestion control that could be deployed on Taurus-based network interface cards (NICs). The Indigo's DL network is unrolled to meet 40 Gbps line rate for minimum-sized packets (i.e., 0.08 GPkt/s). While the original DL network ran once every 10ms, a Taurus-based NIC pipeline runs Indigo in 12.5ns intervals. With Taurus, we demonstrate that data plane devices can run ML models and do inference at line rate with several orders of magnitude lower latencies than traditional control-plane approaches [2, 5, 12].

REFERENCES

[1] Bosshart, P., Gibb, G., Kim, H.-S., Varghese, G., McKeown, N., Izzard, M., Mujica, F., and Horowitz, M. Forwarding Metamorphosis: Fast Programmable Match-action Processing in Hardware for SDN. In *ACM SIGCOMM* (2013).

[2] Farrington, N., and Andreyev, A. Facebook's Data Center Network Architecture. In *IEEE Optical Interconnects* (2013).

[3] Feamster, N., and Rexford, J. Why (and How) Networks Should Run Themselves. *CoRR abs/1710.11583* (2017).

[4] Geng, Y., Liu, S., Wang, F., Yin, Z., Prabhakar, B., and Rosenblum, M. Self-programming networks: Architecture and algorithms. In *IEEE CCC, Allerton* (2017), IEEE.

[5] Google. Cloud TPU: Frequently Asked Questions. https://cloud.google.com/tpu/docs/faq. Accessed on 03/15/2019.

[6] Gurevich, V. Programmable Data Plane at Terabit Speeds. https://p4.org/assets/p4_d2_2017_programmable_data_plane_at_terabit_speeds.pdf, 2017.

[7] Hornik, K. Approximation Capabilities of Multilayer Feedforward Networks. *Elsevier Neural Networks 4*, 2 (1991), 251–257.

[8] Kim, C. Programming The Network Data Plane: What, How, and Why? https://conferences.sigcomm.org/events/apnet2017/slides/chang.pdf.

[9] McKeown, N., Anderson, T., Balakrishnan, H., Parulkar, G., Peterson, L., Rexford, J., Shenker, S., and Turner, J. OpenFlow: Enabling Innovation on Campus Networks. *ACM SIGCOMM CCR 38*, 2 (2008), 69–74.

[10] Mehmood, T., and Rais, H. B. M. SVM for Network Anomaly Detection using ACO Feature Subset. In *IEEE iSMSC* (2015).

[11] Mestres, A., Rodriguez-Natal, A., Carner, J., Barlet-Ros, P., Alarcón, E., Solé, M., Muntés-Mulero, V., Meyer, D., Barkai, S., Hibbett, M. J., et al. Knowledge-Defined Networking. *ACM SIGCOMM CCR 47*, 3 (2017), 2–10.

[12] Niranjan Mysore, R., Pamboris, A., Farrington, N., Huang, N., Miri, P., Radhakrishnan, S., Subramanya, V., and Vahdat, A. PortLand: A Scalable Fault-tolerant Layer 2 Data Center Network Fabric. In *ACM SIGCOMM* (2009).

[13] Prabhakar, R., Zhang, Y., Koeplinger, D., Feldman, M., Zhao, T., Hadjis, S., Pedram, A., Kozyrakis, C., and Olukotun, K. Plasticine: A Reconfigurable Architecture for Parallel Patterns. In *ACM/IEEE ISCA* (2017).

[14] Tang, T. A., Mhamdi, L., McLernon, D., Zaidi, S. A. R., and Ghogho, M. Deep Learning Approach for Network Intrusion Detection in Software Defined Networking. In *IEEE WINCOM* (2016).

[15] Tavallaee, M., Bagheri, E., Lu, W., and Ghorbani, A. A. A Detailed Analysis of the KDD CUP 99 Data Set. In *IEEE CISDA* (2009).

[16] Yan, F. Y., Ma, J., Hill, G. D., Raghavan, D., Wahby, R. S., Levis, P., and Winstein, K. Pantheon: The Training Ground for Internet Congestion-control Research. In *USENIX ATC* (2018).