B4: Experience with a Globally-Deployed Software Defined WAN

Ashutosh Mittal (E-mail: amittal@kth.se)

I. Paper summary

THIS paper talks about Google's experience with software defined WAN named *B4*. Traditional WAN treat all bits as the same and average utilization is around 30% to 40% of available bandwidth due to over-provisioning of resources. Google needed a network for the high magnitude data traffic between its data centers with centralized control, thus came B4.

It uses open flow and is built on top of ONIX. Google also built its own fast and 'dumb' routers using merchant silicon for deployment of B4. Its control unit uses an novel traffic engineering algorithm. This algorithm introduces the concept *fairshare* which helps it prioritize bandwidth allocation between different application and change the priorities dynamically as well. This iterative algorithm uses max-min fairness for bandwidth allocation. However, this algorithm is merely an overlay over the shortest path routing and the system can fall back anytime to it.

B4 embodies centralized TE control, application priority, high link utilization and reduced data traffic cost.

II. Significant Contributions

- Evaluations show that B4 has been able to achieve roughly 95% utilization, thus providing optimal usage of resources. This is a remarkable achievement compared to the the fact that traditional networks provide 30%-40% utilization.
- The concept of fair share introduced in the paper has equipped the network to prioritize allocation of resources. Evaluations have shown significantly low packet loss ratio for high priority packets as compared to low priority ones.
- This implementation is a stunning proof of viability of SDN because Google has large amount of traffic and that too spread out across the world. SDN implementation has been able to improve the traffic management significantly and thus raises similar hopes for improvement in internet traffic in general.

III. Unresolved Issues

- It is impossible to distinguish between physical failures of underlying switch hardware and the associated control plane and hence it requires application level signals of broken connectivity.
- Neither the software nor switch SDKs are optimized for dynamic table programming. In particular, tunnel tables are typically assumed to be set and forget rather than targets for frequent reconfiguration. Also, the OFA to OFC link is highly queued and under high strain.
- The implementation needs much more stress testing. Like for example, it has not been tested for multiple type simultaneous failures.