**Paper Evaluation, Jellyfish: Network Data Centers Randomly**

Jiani Jiang <jianij@kth.se>

1. **Paper summary**

Jellyfish is proposed to meet the call of industry experience for incremental expansion. In the paper, authors tend to describe approaches to solve the unstructured design of Jellyfish. To solve the routing and congestion control problems, authors made a comparison on ECMP and k-shortest paths that developed by MPTCP simulator and also made a contrast test with fat-tree topology which provides in-depth results for Jellyfish’s throughput and fairness using the best setting—k-shortest paths and MPTCP. For physical wiring, the situations of small clusters and container data centers, and massive-scale data centers are considered and were carried out to prove that it’s acceptable under the key parameters of number of cables, length of cables and cabling complexity.

1. **Top 3 contributions**

The authors introduced the principle and significance of Jellyfish in detail which shows that it adopts a degree-bounded random graph (RRG) topology among top-of-rack (ToR) switches. It achieves a high-capacity, scalability, flexible, cost-efficient, construction of arbitrary-size networks compared with other network topologies. Meanwhile, the authors gave particular data and trends by conducting a variety of comparative experiments with other network topologies to verify the advantages of Jellyfish. The lightspot is that authors considered the challenges (unstructured design) of Jellyfish and raised their ideas by performing experiments which contribute to the improvement of Jellyfish.

1. **Problems**

1. fat-tree topology is inferior to Jellyfish in many ways. However, fat-tree can easily isolate failure or virus due to the hierarchical centralized control mode and Jellyfish can’t do that.

2. Since Jellyfish is a random graph topology, it’s not easy to implement for it needs network designers to design the connections between switches.

3. It depends on a good routing protocol and performs bad on protocols like ECMP and TCP.