CMPE 252A Course Project

B. Implementing a FatTree topology and corresponding routing protocols introduced in the following paper and the SDN lecture notes:

[2] M. Al-Fares, et al., A Scalable, Commodity Data Center Network Architecture, ACM SIGCOMM, 2008.
using the simulator Mininet (http://mininet.org/)

The tutorial of Mininet can be found online (https://github.com/mininet/mininet/wiki/Documentation)

In this project we will use Mininet to emulate a fat-tree data center topology (K=4). Please follow the addressing strategy described in the paper (Section 3.2). After you have finished creating the topology, you may choose the SDN controller based on the language you are most familiar with programming.

Tasks and grading rubrics:
Total: 100 points

1) Construct a K=4 fat-tree topology using mininet. (20 pts)
2) Run Dijkstra's algorithm to calculate shortest paths for each pair of end hosts. You can treat each link with the same weight in fat-tree topology. (20 pts)
3) Dijkstra's algorithm only employs a single next hop even when multiple equal-cost next hops exist, which is common in data center topologies. Implement the two-level routing algorithms proposed in the far-tree paper to improve the link bandwidth utilization. (20 pts)
4) Implement ECMP. When there exist multiple equal-cost paths, use a hash function to choose a path based on the headers of packets. (20 pts)
5) Evaluate each routing scheme using the Iperf and the given traffic patterns. (20 pts)
6) Optional: implement a flow scheduling algorithm and compare the performance with the static routing schemes above. (external 20 pts)

The external credits if applicable will be added to the total grade for the course project.

Hints:

1. We have provided some code skeletons based on Ripl-POX (https://github.com/brandonheller/riplpox, a library for POX). Feel free to start with them if you are using POX controller.

2. For suffix match in the two-level routing scheme, consider use “Nicira Extended Match (or NXM)” to extend POX to support OpenFlow 1.2+, as shown in POX Wiki: https://openflow.stanford.edu/display/ONL/POX+Wiki

3. Bisection bandwidth of fat-tree under a certain routing scheme can be evaluated by aggregating incoming traffic of all hosts for all bijective communication mappings.