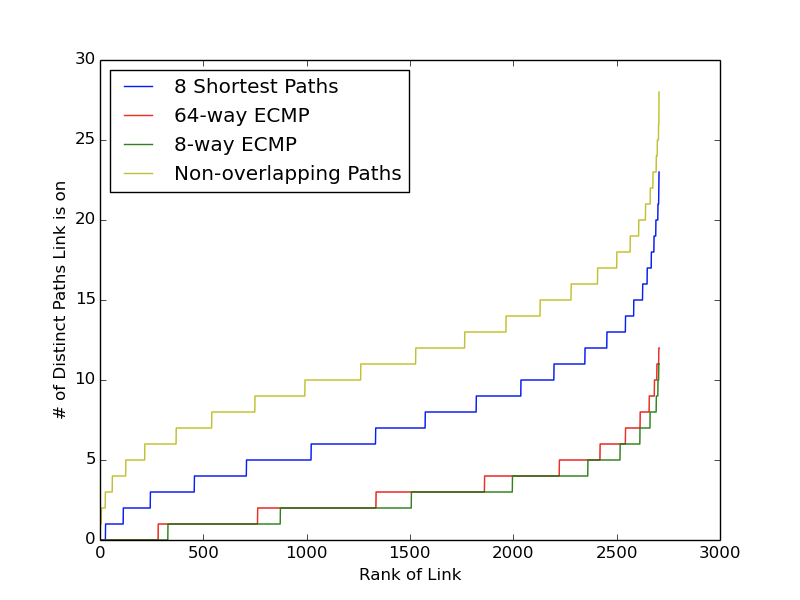
1. **Recap**: The high capacity mentioned in the Jellyfish paper inspires us to explore its potential to tackle burst flow. We want to maximize the average throughput at the expense of tolerable latency.
2. **Progress**:
   1. We leveraged several libraries (Mininet, Pox, RipL[1], RipL-POX[2]) and open-source code[3] to reproduce the Jellyfish network and k-shortest-paths routing;
   2. We put forward and implemented a new routing algorithm (*Non-overlapping Path Algorithm*), which guarantees all links on paths from A to B have no overlapping;
   3. We compared the path diversity between non-overlapping routing with k-shortest-path. Result is attached below and based on Jellyfish Paper Figure 9[4];
   4. We tested the average throughput per server, achieving 23.6% more throughput. Please check our code for more details[5];
   5. We set up our experimental environment on the Google Cloud Platform, which makes it easier to conduct future experiments.
3. **Next step**:
   1. Release strict non-overlapping restraint to balance the path length and path diversity;
   2. Perform more tests under various topologies and circumstances;
4. **Contribution**
   1. Liwei Cui: Reproduced Jellyfish network; implemented and benchmarked different routing algorithms;
   2. Mou Zhang: Prepared test environment migration; Set up the experimental environment on the Google Cloud Platform;
   3. Yifeng Yin: Surveyed similar algorithms.



[1] <https://github.com/brandonheller/ripl>

[2] <https://github.com/brandonheller/riplpox>

[3] <https://github.com/lechengfan/cs244-assignment2>

[4] <https://www.usenix.org/system/files/conference/nsdi12/nsdi12-final82.pdf>

[4] <https://github.com/Lw-Cui/Non-overlapping-Path-in-Jellyfish>