Report for Reading assignment on SplitStream

# Motivation

Why is the problem addressed in the paper interesting and important for the larger community to be solved?

The problem is that in conventionally tree-based multicast system, the burden of forwarding as well as duplicating multicast messages are only taken care of by interior nodes, however those nodes are small subset, and if they are not highly available nor dedicate organized, the system doesn’t work. On the other hand, the majority, those leaf nodes, contribute no resources to forwarding load, wasting their outband bandwidth. What’s more, once the tree grows deep, the system is not practical, since it becomes fault prone and introduces large delay.

# Contribution

What are the main contributions of the paper?

The main contribution of the paper is to introduce SplitStream, presenting a forest of interior-node-disjoint multicast trees that enables distribution of high-bandwidth content with application-level multicast in a cooperate environment. The goal is to make all peers contribute resources in order to distribute the forwarding load.

Another point of the paper is the idea of splitting the multicast stream into multiple stripes with separate multicast trees based on a non-dedicated infrastructure, so that it accommodates different limitations, since each node may have different bandwidth capacities, and also increases the resilience to node failure and sudden node departures.

# Solution

How did the authors solve the problem at hand?

As the name SplitStream indicates, the original stream is spited into k stripes, and there is a separate multicast tree for each stripe, and in order to achieve fault tolerance, application uses erasure coding or multiple description coding to mitigate the effects of node failure, the content is encoded in the way such that each stripe acquires nearly the same bandwidth and any subset of the stripes with adequate size can rebuild the content. The trees are interior-node-disjoint, which require nodes to be interior in at most one tree, in result node failure affects only one stripe, and the trees are built on Pastry’s routing properties with setting 2b. The stripe ids differ on the first digits, and nodes are claimed to join at least the stripe which owns the same first digit as their own ID. Regarding to the bandwidth problem, the push-down mechanism in Scribe is furthermore tailored, such that adoption always happens, but a random child share no prefix in the set, or with the shortest prefix in common with the stripe id is rejected. The orphaned child tries to attach a parent among former siblings with proper prefix by push-down; otherwise it anycasts the space capacity group and performs depth-first search for a parent. Nevertheless, two conditions must hold in forest construction: sum of desired indegrees is less than or equal to sum of forwarding capcacities and nodes whose forwarding capacity is greater than desired indegree should receive or originate all k stripes.

# Evaluation

How good is the solution? How did the authors evaluate their solution?

How good was the evaluation of their work?

# Disadvantages of the Solution

What are the disadvantages and shortcomings of the solution given by the authors?

# Disadvantages of the Evaluation

During the evaluation of their solution, did the authors overlook something?

# Further improvements

Are there any further improvements that can be made to the solution?

Nodes may keep track of the packet arrival rate for each stripe, once they find the incoming channel for a stripe is delivering the bandwidth behind expectation, they can break away from the stripe tree and look for an alternative parent.

Are there any future directions you can think of?