INTRODUCTION

The fuzzy log project is an attempt to create a better abstraction for distributed metadata storage, in particular for use by control plane services, including but not limited to SDN controllers (ONOS, NSX), filesystem namespaces (HDFS namenode), big data runtime schedulers (Google Omega), and general-purpose coordination services (e.g. ZooKeeper, Chubby). All of these services need to ensure that state is managed with strong consistency, concurrency control, fault-tolerance and performance.

Currently, your typical service of this nature might use ARIES for failure atomicity and durability with regard to reboots, some version of MultiPaxos for availability and durability despite machine crashes and some form of two-phase commit protocol for transactional isolation and failure atomicity across partitioned data, requiring the services of a number of competent distributed systems engineers to comprehend, implement and manage.
The ultimate aim of the project is to abstract away the snarl of complex protocols, each providing, for the most part, a small subset of these properties separately, that are currently utilized by such services, allowing the application developer to concentrate on the application and leave the distributed systems stuff completely to the experts.

A relatively recent development in this field is the use of the shared log abstraction to simplify the creation and maintenance of control plane services. Projects such as Professor Balakrishnan’s own CorfuDB [corfudb] and Tango [tango2013] and others such as Hyder[hyder2011], Calvin[calvin2012], and Confluent [confluent] have leveraged a single, globally shared log to derive properties such as consistency, durability, failure atomicity and transactional isolation via simple append/read operations. This is considerably more simple, from the view of the application developer, than mucking about with the distributed protocols mentioned above - it is possible to build a shared log version of ZooKeeper in only a few thousand lines of code, and order of magnitude less than the Paxos-based version.

Enter the fuzzy shared log: a new abstraction allowing for applications to define arbitrary partial orders on updates, allowing it to, in principle, pick exactly the level of consistency it needs in every case.

The weaker consistency guarantee we’re interested in is causal consistency, similar to the definition in the COPS [cops2011] and Eiger [eiger2013] papers. In our version, we will
allow client apps to specifically define the causal dependencies of each mutation by way of edges on the DAG.

**CURRENT PROJECT STATUS**

We have a working implementation of the color layer in Rust. I have written an implementation of HDFS over the color layer.

**PLANNED WORK**

Over the course of this semester, I will test, improve and benchmark the HDFS implementation, as well as possibly implement another application or two over the Fuzzy Log, in preparation for our SOSP ‘17 submission.