Algorithm Applications

Jaydon Conder, Nashea Wiesner, Sawyer Payne, Seth Kreitinger, AJ Gayler, William Dittman

September 30, 2015

TODO: brief introduction

1 Algorithm 1: AUDIO STREAMING PATCHING

Nowadays, much of the data transferred online is through audio and video streaming. This can take a toll on the reliability and speed of the user's network. This, therefore, causes much more data to be lost and the transfer of the stream to be interrupted. To overcome this issue, an algorithm has been developed to patch the streaming audio (particularly music) with stored recordings or expected repeat snippets from earlier in the track. This is for patching a loss that is unacceptable to the user(15-20 seconds). Basically, along with existing audio compression techniques, a method is used to monitor the syntax of the music when streaming over a low bandwidth network. Using a method called Song Form Intelligence(SoFI), the packet loss is determined and then a scan is performed through portions of the song already received in the buffer to see if a possible match exists. The song is first divided into chunks (i.e. Intro, Verse, Chorus) and then each chunk holds the corresponding packets received. If, for instance, a packet is lost on the second repeat of the chorus, the matching packet from the first chorus will replace it. The goal is to make the loss undetectable to the user and produce a smooth stream. Reference: ACM Transactions on Intelligent Systems and Technology, Vol. 6, No. 2, Article 25, Publication date: March 2015. Pattern Matching Techniques for Replacing Missing Sections of Audio Streamed across Wireless Networks JONATHAN DOHERTY, University of Ulster KEVIN CURRAN, University of Ulster PAUL McKEVITT, University of Ulster

2 Algorithm 2: RUBIK'S CUBE CIPHER

Data security is one of the bigger issues facing people today. With so much of society's interactions taking placing online, encryption algorithms are needed to make sure that only the people for which the message is intended can access it. The Rubik's Cube Image Encryption algorithm takes images sent by user one, changes it into something completely unrecognizable by shuffling the pixels' rows and columns and then doubly ciphering the image using two different ciphering matrices created from chaotic systems. Once the algorithm reaches the person it was intended for, that person decrypts the image using a related algorithm that transforms the image back into the original. This algorithm helps to ensure the only the users with the correct key can decrypt the image, ensuring privacy throughout the whole transaction. Reference: Mathematical Problems in Engineering, Vol. 2013, Article ID 848392, Publication date: February 2013. An Improved Secure Image Encryption Algorithm Based on Rubik's Cube Principle and Digital Chaotic Cipher ADRIAN-VIOREL DIACONU, University Politehnica of Bucharest KHALED LOUKHAOUKHA, Laval University

3 Algorithm 3: Raft Consensus

The Raft Consensus Algorithm is one that solves the problem of clustering a group of machines and having those machines agree on what commands are in each node's logs. This algorithm runs on each machine (node) and provides it a method to keep the cluster fault tolerant. What this means is that if one or more nodes disconnects or is lost and a majority of the cluster member nodes are still connected, then services can still be provided; and user experience wont be lost. This algorithm is identical to a similar algorithm called the Paxos algorithm, in that its for consensus in a fault tolerance system, but is a bit more intuitive and is decomposed into subproblems that each node can solve for itself. Paxos being notoriously difficult is one of the motives for Raft. The paper that we read explored the notion of the Raft being an improvement on Paxos by implementing both in a test virtual environment. The authors also looked at how to optimize the Raft algorithm.

There are three possible roles of the nodes in a Raft cluster: followers, candidates, and leaders. The leader is the node "in charge" of the cluster of nodes. Everything outside of the cluster must communicate through the leader in order to have the cluster execute tasks (or whatever the purpose of the cluster is). Followers are passive, and only respond to RPCs (Remote Procedure Call) from the leader (except when recovering from failure). A candidate is an intermediate role for a node that has the potential to become a leader. A node can be represented by a state machine diagram, where it is in one of these three states.

Since one of the selling points of Raft is fault tolerance, the algorithm accounts for message/computation delays of arbitrary time, packet loss, duplication, and re-ordering. As a result, nodes are able to dynamically elect/find a leader, as well as synchronize states. Synchronization of states is done by using the previously mentioned log that exists on every node (and is the same on every node). When one node makes a change to the log it informs the other nodes who then update their own logs thereby keeping the clustered nodes in the same state.

4 Discussion

This is an optional section, for this assignment.