Distributed Auction System - Report

rono, ghej, luvr @ ITU, Group “G.L.R”

November 25, 2024

Contents

1 Link to Project on GitHub

https://github.com/NewNorthStar/BSDISYS1KU-Assignment-Five

2 Introduction

We have implemented an auction system consisting of one or more server nodes conducting a single auction. Clients may connect to any node to bid on the auction. The solution is implemented in Go and communicates using gRPC.

- The server nodes act as a single leader system. Followers will forward new bids to the leader. The leader then updates any followers before the method returns. This ensures linearizable state changes as long as the auction is live with atleast one node.
- When a follower discovers that the leader is unreachable, it will call for an election. This is implemented using the Bully election algorithm, with priority in particular given to the most up-to-date node.
- If a client loses connection to the auction, it will try reconnecting to another known node. Clients know the nodes available their registration, or at their last bid.

3 Architecture

Clients place bids at the auction using the `PutBid(...)` rpc. This may be called on any node. If a follower node receives this call, it will forward the call to the leader using a client instance. This is also when a follower may detect the failure of the leader.

The `GetLot(...)`, `GetAuctionStatus(...)` and `GetDiscovery(...)` rpc's are used to pull information from the auction. These are for use by clients, but follower nodes also use them when registering with the leader.

`Ping(...)` is used by clients to find a new connection to the auction, should the first one fail.

`Register(...)` is used by follower nodes to register with the leader. The successful connection is confirmed by the leader by sending the first `UpdateNode` call.

`UpdateNode(...)` is used by the leader to push auction state changes to to followers. This is called as a side-effect of the `PutBid` and `Register` rpc calls.

Should the leader node fail, then the `Election(...)` and `Coordinator(...)` calls are used to elevate a follower to lead the auction.

4 Correctness

4.1 Argument 1

Argue whether your implementation satisfies linearisability or sequential consistency. In order to do so, first, you must state precisely what such property is.

4.2 Argument 2

An argument that your protocol is correct in the absence and the presence of failures.