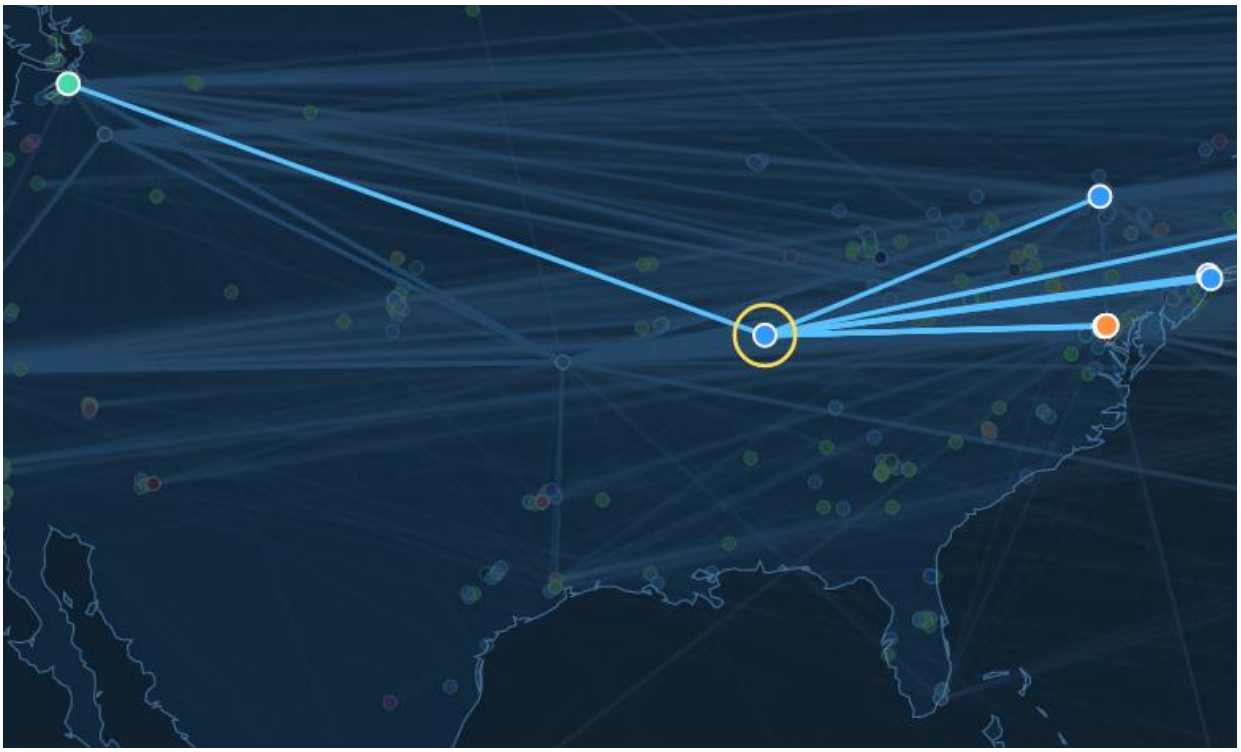
A Lightning Network Modeling and Simulation Framework

An Open-Source Development Proposal

The Lightning Network is a new and exciting technology that will likely change how Bitcoin is used all over the world. It is a complex, open network that is constantly changing and evolving. It is hard to predict its behavior due to the decentralized nature of it. As more and more people start using this network the behavior will become more advanced and even more unpredictable.

Throughout history as new technologies are developed and complex systems are built using those technologies people have built Modeling and Simulation tools alongside of them in order to better understand them and to test them. The same should be done for the Bitcoin Lightning Network.

A tool is needed that allows developers and researchers to define a Lighting Network layout and then simulate events over time to analyze how the network might be impacted. This could be used to test new products that a team is developing or to help research and predict how certain events might impact a Lightning Node or the Lightning Network as a whole.



How does the operation of the Lightning Network change if the selected node goes offline for 2 days?

The Problem

In order to develop new products and services using the Lightning Network, engineers need a quick and easy way to spin up a complete Lightning Network to test their work. This network includes ln nodes, bitcoin nodes, transaction histories, channels with different liquidities, etc. Before connecting a new service to the testnet or mainnet often a local “regtest” network is used.

There are lots of tools out there to help create regtest networks, but what these tools lack is the ability to simulate all kinds of events over a long period of time and gather data during those events. As time goes on nodes might go offline, close channels or run out of liquidity. All these events will change how the network operates.

For example, let's say a large routing node that handles lots of transactions and has lots of open channels goes offline for an hour. This has the potential to have a big impact on the network:

- How would this event impact transaction fees during the outage?
- Which nodes would be most affected?
- How many failed transactions might occur during this outage?
- How would all these statistics be different if that outage was 30 minutes instead of an hour? What about a day instead of an hour?

All these things could be simulated and then analyzed. Charts and reports could be generated so that the user can make decisions based on this data.

The Solution

A solution could be built as an open-source library that models the behavior of the Lightning Network. This library would allow a developer to specify characteristics of the network and all its nodes. As well as defining the initial state of the network, a user could define events that will take place sometime in the future. Then by running the simulation the user could see how those events impact the network over the specified time period.

Projects like “Polar” allow users to design a test network and manually interact with it by doing things like creating an invoice, opening a channel, etc. A lot of these projects use full Lightning Network node implementations and/or docker containers to set up a realistic network. While this is a good solution for small networks it does not allow a user to create very large networks and automate the operations on that network over a period of time.

Projects like “Sensei” let users setup "lightweight" lightning network nodes. The simulation framework could be built on top of a project like this in order to keep it lightweight. This would allow the library to be integrated into other projects (Polar, for example) and model big networks.

A user could define all the nodes in the network and choose which ones are connected to each other. Along with the number of nodes and connections, a user could define events that take place at different timestamps, for example:

- A node closes all its channels
- A node opens new channels
- A node raises its routing fees
- A node goes offline

The Project

A back-end API could be built so that users could quickly write extensions/plugins that would create different types of networks with the user's desired qualities and events. A front-end UI could also be built so that a user could visualize a network while they design it for their simulations. Along with the simulated network an analysis/reporting tool could be built to report network statistics, events, and the overall state of the network. This would allow developers to see how their products are impacting the network as time passes and as different events take place. In addition, template networks could be defined to model the existing Lightning Network.

The proposed project would be divided into two main parts:

1. **ln_ms_lib** - a Rust library that exposes an API that will let developers quickly create and run simulations. Part of this library would be a way to parse network definitions from several different sources. For example, projects like Polar... where a developer has defined a test network using Polar and has been testing their app. Polar could integrate this library and the user could quickly turn their test network into a simulation, define their events and run it.

2. **ln_ms_server** - a Rust back-end web server that exposes the ln_ms_lib library API to a web client. This could be used to build a web front end that would serve as a standalone simulation tool where a user could define their network through the UI (instead of another project like Polar) and then run simulations.

By keeping the simulation library outside of any other projects and keeping it lightweight and flexible, it could serve several different purposes and be used differently by different projects.

Proof-Of-Concept GitHub Repository

https://github.com/bjohnson5/ln-ms-framework

About Me

Name: Blake Johnson

Email: brjohnson5@icloud.com

LinkedIn: https://www.linkedin.com/in/blake-johnson-11aa05206

I am an experienced Software Engineer who has worked on several different Modeling and Simulation projects in my career. Most of my experience has been in the defense industry building simulations and models for the military. The defense industry has seen firsthand the benefits of high-fidelity simulation products that accurately represent real world systems. New products and features can be tested in a closed, simulated environment before being tested in the real world (which can be expensive and inefficient). The same benefits could be realized by the bitcoin development community. A large, flexible, open-source simulation framework would allow the Lightning and Bitcoin communities to more efficiently test and release new services. It could also help create stability in the network over the long term.