Cole Horvitz
CS1951L Blockchains
5/10/22

Capstone Writeup

The automated market maker is an ingenious implementation of smart contracts on the Ethereum blockchain. Due to its ability to replace centralized exchanges and order books, it has become one of my favorite uses of blockchain technology. Instead of buyers and sellers dealing with each other directly, traders buy and sell assets using a liquidity pool. The liquidity pool is funded by providers, and its asset capitalization is fixed to a mathematical formula. This ensures predictable and quick trades, offering the liquidity which makes AMMs so convenient compared to other alternatives.

This capstone assignment was a productive means of becoming familiar with automated market makers. To prepare, I read the provided materials from the handout. While most of the articles provided a solid baseline understanding of automated market makers, Dr. Herlihy and Daniel Engel's article *Loss and Slippage in Networks of Automated Market Makers* offered a mathematical approach to the topic, defining the formulas needed to complete this assignment.

While at this point my conceptual understanding felt strong, in practice things turned out to be somewhat trickier. For one, Solidity is still new to me, and it was interesting to have to use safe mathematical operations to protect against overflows. Viewing the provided code for a Uniswap-like AMM was both enlightening and confusing. Thanks to the Swap assignment's usage of Solidity I was able to understand much of the code. Although since this AMM was written with a different version of Solidity some syntax had changed, and I wonder why the author did not use modifiers. While I gathered the meaning of much of the code, there was a lot that I couldn't quite figure out in the AMM implementation itself. I believe that it would be easier to understand the code if it were easier to compile and experiment with it. I struggled greatly trying to get the code to actually run. Regardless, I was able to figure out how to obtain the Ether and token reserves, which was all I needed to implement the eight view functions.

Implementing the view functions went well. Using the formulas for constant-product AMMs provided in the paper, I was able to easily translate these into Solidity code. In particular, linear slippage, divergence loss, and gain were easy to implement in both directions. The most difficult metric to implement was angular slippage. This is because I had to derive its formula for a constant product AMM, and then compute arctangents. Luckily, the provided RealMath library had an atan implementation which I used in my code. Overall this process was quick and simple, as most of the hard work went into actually understanding the conceptual basis of the functions. These particular formulas offer additional insight into trades with an AMM, as they return metrics which signify the impact of a proposed trade.

I still have some remaining questions regarding the AMM. For example, I wonder what the formulas look like for AMMs which do not use the constant product formula. Furthermore, I was somewhat confused about whether or not I was supposed to use the raw ether reserve and token reserve values, or if they were meant to be represented as a proportion of the AMM's total

capitalization. After discussing with peers, I believe it is the former. Furthermore, while I believe to have gotten the gist of the metrics, I do wish that there was a testing suite to provide validation of the code.

This capstone assignment has sparked a genuine interest in automated market makers for me. I believe that this sort of technology is better aligned with the decentralized purpose of cryptocurrencies, and offers a strong alternative to centralized exchanges. I hope to do more research and experimentation with this topic as time goes on.