Principles of decentralized finance, June – July 2022 G. Angeris

Problem set 1: Foundations of Blockchains

This set of ‘exercises’ is mostly just a (somewhat) structured exploration of things that are publicly available. This is quite different from future exercises, which will offer concrete problems.

1 Exploring live chains

In this short problem, we will look at live data from Ethereum using a *blockchain explorer*. A blockchain explorer is a tool that allows you to look at a relatively current state of the blockchain in a human-readable way. Blockchain explorers usually allow you to see transactions, account balances, and function calls that have happened both recently and historically.

Exploring an account. Go to the Etherscan website:

`https://etherscan.io`

Etherscan is an Ethereum blockchain explorer and is a relatively standard tool when viewing historical transactions. In the search bar, search for the address:

`0xd8dA6BF26964aF9D7eEd9e03E53415D37aA96045`

Scrolling down, you should be able to see the latest transactions performed by this address, listed in reverse chronological order. Take a peek around to see what Vitalik Buterin, one of the founders of Ethereum, has been up to in one of his (very well-known) accounts.

As a side note, in the ‘From’ or ‘To’ columns, you might see `vitalik.eth` instead of the address listed above. This is part of the Ethereum Name Service (ENS) which is a set of contracts that allows users to register a human-readable `.eth` ‘address’ that links to an account address. Many front-ends support using these human-readable ‘addresses’ in addition to the usual addresses (like the one shown above). Note that ENS is not part of the Ethereum protocol specification at all, it is simply a set of on-chain, public contracts that set a standard that many projects agreed to follow.

Exploring a contract. Going back to Etherscan (the site linked above), now navigate to:

`0x5777d92f208679db4b9778590fa3cab3ac9e2168`

This is the Uniswap V3 DAI-USDC pool, which is a contract that allows users to trade between DAI and USDC, two tokens approximately equivalent to a dollar. (We will learn much more about Uniswap and these types of tokens, called stablecoins, in later lectures.)

Scroll down and find the “Erc20 Token Txns” tab, click on it, and take a look at the trades that people have been performing. In many cases the ‘From’ and ‘To’ will be addresses or ENS domains, but you might see others (such as, for example, `CoW Protocol`): these are other protocols using the Uniswap pool as an intermediate function call!

Extra credit. Follow a few of the links in different transactions, and see where you end up. You can also explore what has been happening in the latest blocks by browsing around the homepage, see what projects, NFT drops, *etc.*, are popular right now. The fact that you can see every transaction that has been made on Ethereum is both a blessing and a curse, depending on your stance; either way, it’s certainly quite fun.

2 The ERC-20 token standard

The *ERC-20 token standard*, discussed briefly in lecture, is fully specified in the following proposal:

`https://github.com/ethereum/EIPs/blob/master/EIPS/eip-20.md`

Take a peek at the proposal linked above which is a much more complete interface than the one we gave in class. For example, this proposal requires a way to delegate spending authority to other accounts, among a few other methods that are useful for developing interactive applications.

A reference implementation. While the token standard above only specified the interface for an ERC-20 token, there are a number of reference implementations. One such implementation is the ConsenSys reference implementation which can be found in

`https://github.com/ConsenSys/Tokens/blob/master/contracts/eip20/EIP20.sol`

though there are others (*e.g.*, OpenZeppelin’s). This implementation is written in Solidity, a C-like language used for developing smart contracts on Ethereum, but should be relatively straightforward to understand for anyone with some programming experience in C-like languages. If there are any questions, there are (not phenomenal) docs, found in

`https://docs.soliditylang.org/`

which we recommend you peruse, but knowing Solidity is in no way necessary for the course.