Globules

Globular Specifications for Ethereum

Micah Dameron

November 13, 2017

Beautiful is better than ugly.
Explicit is better than implicit.
Simple is better than complex.
Complex is better than complicated.

The Zen of Python

Contents

Acknowledgements

Thanks to the founders of Ethereum for creating a product worth writing about in minute detail, thanks to the ConsenSys Mesh for reviewing and contributing to the creation of this work, and thanks to Dr. Gavin Wood for going above and beyond in expressing the political, philosophical, cultural, and contractual implications for Ethereum in his Yellowpaper.

1 Preface

Ethereum's formal specification, *The Yellowpaper*, goes above and beyond the understanding of most lay-readers. With a plethora of astute formal proofs, its author offers nothing short of Newton's *Principia* for Distributed Ledger Technology. But most readers of the Yellowpaper can only partially appreciate the depth of Dr. Wood's understanding, since the paper is written as a historical memoir but also contains its own mathematical alphabet and unique formal structure. Ethereum's inner workings do not avail themselves to most readers to begin with, but the same inner workings are made even more complicated to readers if they must be committed to applying an array of incumbent symbols at the same time.

Readers of the Yellowpaper are compelled to split their time and attention between the desire to understand the content and their need to become fluent with the new formal alphabet the content is written in. This would not be so much of a problem if *formalisms* Yellowpaper were limited to necessarily formal statements, but they are not. Formalisms are expanded and increased to the point that there are multiple tiers of equations, all in this same new alphabet, that are supposed to explain each other and the finer details of the Ethereum Protocol at the same time. The problem is that once again the reader's attention is split. This lack of clarity contributes to a skepticism about Ethereum in general, and deters many potential adopters from the platform. Therefore, there is a need for a firmer integration between the technical *facts* of the Ethereum platform and the *challenges* its developers must face in light of those facts.

In this paper, we aim to make formal descriptions semantically obvious by invoking a procedurally correct pseudocode, saving time and effort for the reader by using concepts they already know, rather than using a customized formal alphabet. In cases where formal proofs use well-established and widely understood standards from Mathematics and Computer Science, we capitulate to those standards where necessary, and take some measures to draw them out for the reader.

2 Introduction

Ethereum is a decentralized virtual machine that is accessible to anyone and yet not owned by anyone. The crypto-economic rules of Ethereum stipulate, through thoughtful *mechanism-design*, the conditions under which interaction with the network is possible. The rules of the protocol, which are set out in this paper, revolve around a cryptographically-secured consensus mechanism that ensures fair conduct (conduct that is in line with the rules) is contracted and enforced on the network.

This paper, covering different topics (globules) about Ethereum, recognizes a need to integrate the disparate fields which are at play in Ethereum's operation and unify them under one or more common headings. This is the second part of a joint project, the first part (mod-

ules) covering the establishment of accurate and well-sourced modular documentation for each of Ethereum's different aspects, and located here: https://www.github.com/chronaeon/modules.git.

1. `message-passing`
2. `shared-state concurrence` and
3. `transaction processing`

the **Ethereum Protocol** fosters an environment where software developers are able to precisely execute machine instructions that have the same level of veridicity and certainty as do the financial transactions on other secure blockchains like Bitcoin.

3 FAQ

Q: Who Is This Paper For?

This paper is for anyone who wants a readable specification for Ethereum.

Q: Why is The Ethereum Virtual Machine (EVM) transaction based?

A: Since resources on The EVM are limited, they must have a computational cost associated with them. If computational costs are transactional, then they will not go through even partially unless the transaction goes through fully.

Q: Bitcoin doesn't have a formal specification, why does Ethereum need one?

A: The goal of Bitcoin was to create a decentralized payment network; at that it has succeeded marvelously. The goal of Ethereum is to create a decentralized computer. A formal specification allows developers to build new clients that aren't part of the core wallet. This keeps there from being an unofficial monopoly on the protocol by those who are able to manipulate it.

Q: Is there a good way to define Ethereum in a few words?

A: Yes:

"Ethereum is a transaction-based virtual machine, following a singleton design pattern and adhering to a shared state."

Don't worry about understanding all of that yet. It will be unpacked and precisely explained throughout the paper.

Q: How is this paper organized?

A: The main body aims to reproduce Yellowpaper content in a more readable form, while the appendices critique Yellowpaper content in its original form.

4 Key Terms and Concepts

3

This section Key Terms section aims to cover the highest-level and most important concepts you need to know to understand Ethereum. Each of the following sections breaks down the EVM one of these abstractions at a time.

A few key terms: , , , transaction, state machine

Ethereum is a transaction-based state machine. A state-machine can only be in one of a set number of configurations depending on its inputs. Picture a gearshift, and picture each change in gears as a change of the machine's (car/truck's) state. Picture the hand on the gearshift as the input. Since all of the possible states in Ethereum are determined beforehand, impossible states are excluded necessarily.

A state is a particular static configuration that Ethereum is in. In order to affect the state, valid transaction(s) must occur. Only commit-

[a] The Yellowpaper denotes the concept of Ethereum's state with a Greek symbol; a lowercase Sigma:σ

ted transactions change the state, and committed transactions must be mined.[a] Transactions which throw an error, whether an out of gas error or some other error, are transactions that have failed in their execution and thus fail to initiate a valid **state transition**. A valid state transition is when the hand applies enough energy to the gearshift to push it into another gear and make it switch. An invalid state transition, like say, out of gas errors, would be if the hand didn't have enough energy in its push to get it into the next gear. Alternatively, the hand might try pushing it toward a gear its already in, or towards a gear that doesn't exist (bad jump dest) and fail to change the state. Ethereum is a model, that excludes these things by definition. It would be as unlikely for someone to hack Ethereum and make counterfeit Ether as it would that your car would pop out another gear and behave autonomously like something out of a Steven King novel. It's just not in Ethereum's instructions to do that. State machines are limited for this exact reason.

State Machine

priorstate The state immediately preceding the current state.

presentstate The present state of the Blockchain i.e. now.

poststate The state immediately following the current state.

transaction An atomic operation; a message, data modification, or other procedure that is guaranteed to perform completely or not at all.

state-transition The process occurring between two inert states that constitutes the creation of a new state.

state-transition[ing] function One of several possible functions that took Ethereum from a priorstate to a presentstate or takes Ethereum from a presentstate to a poststate.

state-machine A state machine rests in a universal, stable, singular condition, called a state. State machines can transition to new states given certain compatible inputs.

state As a whole, the state is the sum total of database relationships in the **state database**. The state is an inert position on the chain, a position between prior state and post state; a block's frame of reference, and a defined set of relationships to that frame of reference.

More key content to come...Hello, here is some text without a meaning. This text should show what a printed text will look like at this place. If you read this text, you will get no information. Really? Is there no information? Is there a difference between this text and some nonsense like "Huardest gefburn"? Kjift – not at all! A blind text like this gives you information about the selected font, how the letters are written and an impression of the look. This text should contain all letters of the alphabet and it should be written in of the original language. There is no need for special content, but the length of words should match the language.

5 Transactions

2

5.1 Message Calls

2

5.2 Contract Creation

6 Gas

2

6.1 Gasprice

6.2 Gaslimit

Any unused gas is refunded to the user.

6.3 Gasused

6.4 Execution

6.5 Execution Model

2

6.6 Execution Environment

6.7 State Machines

2

6.8 Miner Choice

Miners choose which gas prices they want to accept.

7 States

7.1 Substate

8 State Transitions

9 Currency

The smallest unit of currency in Ethereum is the Wei, which is $1 * 10^18$ Ether. All units at the machine level are counted in Wei.

The MAPPING of **addresses** and **account states** (RLP data structures), this is also known as *state*, or σ. This mapping is not stored on the blockchain, rather it is stored as a Merkle-Patricia trie in a DATABASE BACKEND[a] that maintains a mapping of bytearrays to bytearrays.[b] The cryptographic internal data going back to the root node represents the *State* of the Blockchain at any given root, i.e. at any given *time*.[c]

[a] A database backend is accessed by users indirectly through an external application, most likely an Ethereum client; see also: state database

[b] A bytearray is specific set of bytes [data] that can be loaded into memory. It is a structure for storing binary data, e.g. the contents of a file.

[c] This permanent data structure makes it possible to easily recall any previous state with its root hash keeping the resources off-chain and minimizing on-chain storage needs.

[a] Formally **world state** $= \sigma$.

25 Account State

The account state is the state of any particular account during some specified world state σ.[a]

25.1 Nonce

The **nonce** aspect of an ACCOUNT'S STATE is the number of transactions sent from, or the number of contract-creations by, the address of that account.[b]

25.2 Storage Root

The **storage root** aspect of an ACCOUNT'S STATE is the hash of the trie[c]

25.3 Code Hash

The **code hash** aspect of an ACCOUNT'S STATE is the HASH OF THE EVM CODE of this account. Code hashes are STORED in the **state database**. Code hashes are permanent and they are executed when the address belonging to that account RECEIVES a message call.[def]

25.4 Balance

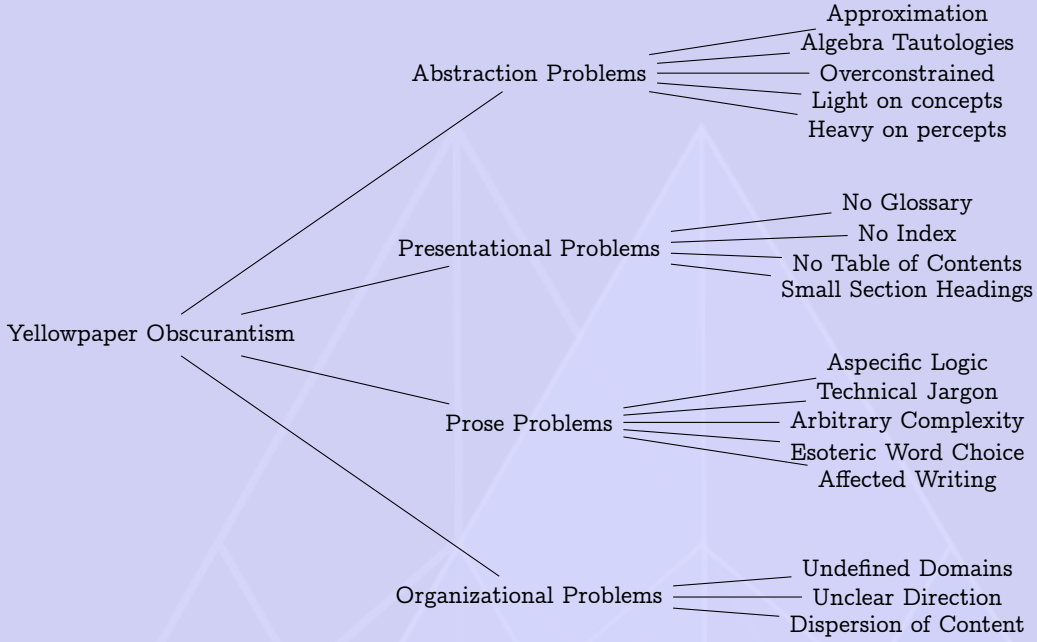
The amount of **Wei** OWNED by this account. [g]

- Key/value pair stored inside the root hash. [h]
- L_I^*, is defined as the element-wise transformation of the base function[i]
- The *element-wise transformation of the base-function* refers to all of the key/value pairs in L_I
- L_I refers to a particular trie.

[b] σ is the world state at a certain given time, and *n* is the number of transactions or contract creations by that account.
[c] A particular path from root to leaf in the **state database** that encodes the STORAGE CONTENTS of the account.
[d] A message call is any interaction with the account on-chain.
[e] Formal notation is $\sigma[a]_c$
[f] More formal notation is $\texttt{KEC}(\mathbf{b}) = \boldsymbol{\sigma}[a]_c$
[g] This formal notation is $\sigma[a]_b$
[h] $\texttt{TRIE}\big(L_I^*(\boldsymbol{\sigma}[a]_\mathbf{s})\big) \equiv \boldsymbol{\sigma}[a]_s$
[i] L_I, given as: $L_I\big((k,v)\big) \equiv \big(\texttt{KEC}(k), \texttt{RLP}(v)\big)$

.1 What problems did the original Yellowpaper have?

There are too many to list them all individually, but this map shows the general ideas:



.2 Symbol Key

Sign	Meaning
$\boldsymbol{\sigma}$	world state
Υ	state transition function
Π	block-level state transition function
Ω	block-finalization state transition function
T	substitutes any given transaction
x_t	the transaction $_t$ corresponding to x
$\equiv$	is equivalent to
B	substitutes any given block
( )	contain symbols that are logically related
...	a defined pattern in a known function

Glossary

abstract machine An abstract machine is a conceptual model of a computer that describes its own operations with perfect accuracy. Since abstract machines are theoretical, all possible outputs can be determined beforehand. 11

Address A 160-bit (20-byte) code used for identifying Accounts. 11

addresses 160-bit (20-byte) identifiers–the last 20 characters of an Ethereum address. 11

Autonomous Object A notional object existent only within the hypothetical state of Ethereum. Has an intrinsic address and thus an associated account; the account will have non-empty associated EVM Code. Incorporated only as the Storage State of that account. 11

Balance A value which is intrinsic to accounts; the quantity of Wei in the account. All EVM operations are associated with changes in account balance. 11

Bit The smallest unit of electronic data storage: there are eight bits in one byte. The Yellowpaper gives certain values in bits (e.g. 160 bits instead of 20 bytes). 11

blockchain A consensus-based record of agreement where chunks of data[a] (called blocks) are stored with cryptographic hashes linking each Block to the next, ensuring veracity. 11

Contract A piece of EVM Code that may be associated with an Account or an Autonomous Object. 11

Dapp An end-user-visible application hosted in an Ethereum browser.. 11

design pattern a pattern of design in OOP. 11

Ethereum Runtime Environment The environment which is provided to an Autonomous Object executing in the EVM. Includes the EVM but also the structure of the world state on which the relies for certain I/O instructions including CALL & CREATE. 11

Ethereum Browser [b] A cross-platform GUI of an interface similar to a simplified browser (a la Chrome) that is able to host applications, the backend of which is purely on the Ethereum protocol.. 11

Ethereum Foundation The non-profit organization in charge of executing the development processes of Ethereum in line with the Whitepaper. 11

Ethereum Virtual Machine A secure, consensus-based global computer, and the subject of this paper. 11

EVM Assembly The human readable version of EVM code. 11

EVM Code The bytecode that the EVM can natively execute. Used to formally specify the meaning and ramifications of a message to an Account. 11

External Actor A person or other entity able to interface to an Ethereum node, but external to the world of Ethereum. It can interact with Ethereum through depositing signed Transactions and inspecting the blockchain

[a] As of November 13, 2017, roughly between 1 and 20 kilobytes in size[Etherscan2017]

[b] a.k.a. Ethereum Reference Client

and associated state. Has one (or more) intrinsic Accounts. 11

Gas The fundamental network cost unit. Paid for exclusively by Ether (as of PoC-4), which is converted freely to and from Gas as required. Gas does not exist outside of the internal Ethereum computation engine; its price is set by the Transaction and miners are free to ignore Transactions whose Gas price is too low. 11

hacker ethic A maxim purporting that knowledge about systems should be free and unhindered[Levy2010]. 11

leaf node the bottom-most node in a particular tree, of blocks, one half of the "key" the other half being the root node, which creates the path between. 11

Lower-Level Lisp The Lisp-like Low-level Language, a human-writable language used for authoring simple contracts and general low-level language toolkit for trans-compiling to. 11

Message Data (as a set of bytes) and Value (specified in Wei) that is passed between two Accounts, either through the deterministic operation of an Autonomous Object or the cryptographically secure signature of the Transaction. 11

Message The act of passing a message from one Account to another. If the destination account is associated with non-empty EVM Code, then the VM will be started with the state of said Object and the Message acted upon. If the message sender is an Autonomous Object, then the Call passes any data returned from the VM operation. 11

Object Synonym for Autonomous Object. 11

public key A term originating from *cryptography* and corresponding to **private key**, this is the 42-byte (i.e. 42-character) string of ASCII digits which transacts on the public network. 11

root node the uppermost node in a particular tree, of blocks, representing a single world state$^{\sigma}$ at a particular time. 8, 11

serialization Serialization is the process of converting an object into a stream of bytes in order to store the object or transmit it to memory, a database, or a file. Its main purpose is to save the state of an object in order to be able to recreate it when needed. The reverse process is called deserialization.[billwagner]. 11

singleton A design pattern in Object-Oriented Programming which specifies a class with one instance but with a global point of access to it[sourcemaking]. 11

specification Technical descriptions, instructions, and definitions from which other people can create working prototypes. 5, 11

State Database A database backend that maintains a mapping of bytearrays to bytearrays. 11

state database A database stored off-chain, [i.e. on the computer of some user running an Ethereum client] which contains a trie structure mapping bytearrays [i.e. organized chunks of binary data] to other bytearrays [other organized chunks of binary data]. The RELATIONSHIPS between each node on this trie constitute a MAP, a.k.a. a MAPPING of all PREVIOUS STATES on the **EVM** which a client might need to reference. 6, 8, 9, 11

state machine A state machine rests in a universal, stable, singular condition, called a state. State machines transition to new states given certain compatible inputs.. 5, 11

Storage State The information particular to a given Account that is maintained between the times that the Account's associated EVM Code runs. 11

transaction An input message to a system that, because of the nature of the real-world event or activity it reflects, requires to be regarded as a single unit of work and must either be processed completely or rejected[Ngondi2016]. 5, 11

Transaction A piece of data, signed by an External Actor. It represents either a Message or a new Autonomous Object. Transactions are recorded into each block of the blockchain. 11

trie A tree-structure for organizing data, the position of data in the tree contains the particular path from root to leaf node that represents the key (the path from root to leaf is "one" key) you are searching the trie structure for. The data of the key is contained in the trie relationships that emerge from related nodes in the trie structure. 8, 9, 11

Whitepaper A conceptual map, distinct from the Yellowpaper, which highlights the development goals for Ethereum as a whole[EF2017]. 11

Yellowpaper Ethereum's first formal specification,[Wood2017] written by Dr. Gavin Wood, one of the founders of Ethereum. Notorious for its difficulty to read,[reddit] it provides both a boon and an obstacle to potential adopters of Ethereum. 11

Acronyms

ERE Ethereum Runtime Environment. 11

EVM Ethereum Virtual Machine. 5, 11

LLL Lower Level Lisp. 11

OOP Object-Oriented Programming. 11

YP Yellowpaper. 11