Realizing the Full Potential of Web3

Turing Hybrid Compute
By Boba Network

boba.network

https://bit.ly/getTuring

Fundamental premise of Ethereum:

A ledger is good, a computer is better

Our hypothesis:

More connected computer = more creative and powerful decentralized systems

boba.network

https://bit.ly/getTuring

Bitcoin

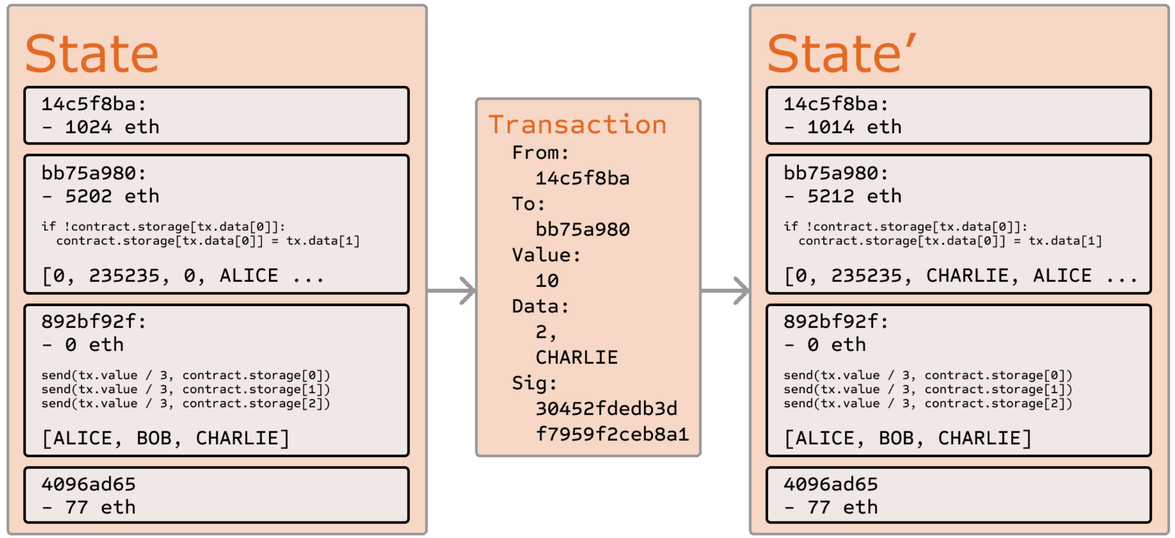
The Bitcoin Transaction Format

	Field		Type (Length)	Comments
	Version		`uint` (4 Byte)	Typically “1”
	Marker		`byte` (1 Byte)	MUST be 0x00, see BIP141
	Flag		`byte` (1 Byte)	MUST be 0x01, see BIP141
	Input count “n”		`var_int` (2 – 9 Byte)	At least 1
n×	Input #i	**TX-ID**	`byte` (32 Byte)	SHA-256d hash of the TX-ID
		TX-Index	`uint32` (4 Byte)	
		unlock script length	`var_int` (2 – 9 Byte)	
		unlock script	`byte` (variable length)	
		sequence	`uint32` (4 Byte)	
	Output count “m”		`var_int` (2 – 9 Byte)	
m×	Output #j	**value**	`uint64` (8 Byte)	Amount to transfer in Satoshi
		lock script length	`var_int` (2 – 9 Byte)	
		lock script	`byte` (variable length)	
n×	Witness	stack item count “p”	`var_int` (2 – 9 Byte)	
p×		stack item length	`var_int` (2 – 9 Byte)	
		stack item #k	`byte` (variable Byte)	NOT Bitcoin Script!
	Lock Time		`uint32` (4 Byte)	

Image by Martin Thoma

Bitcoin Script: a minimal stack-based programming language for locking and unlocking transactions

Ethereum



Ethereum: a blockchain with a built-in Turing-complete programming language

boba.network

Unfortunately…

Ethereum Compute Performance July 30, 2015

Ethereum Compute Performance February, 2022

(there are lots of good reasons for that… Ethereum is not really designed for raw compute performance)

boba

boba.network

Fundamental issues

Getting many computers to agree on anything **is really hard** unless you put strong constraints on the type of compute you allow

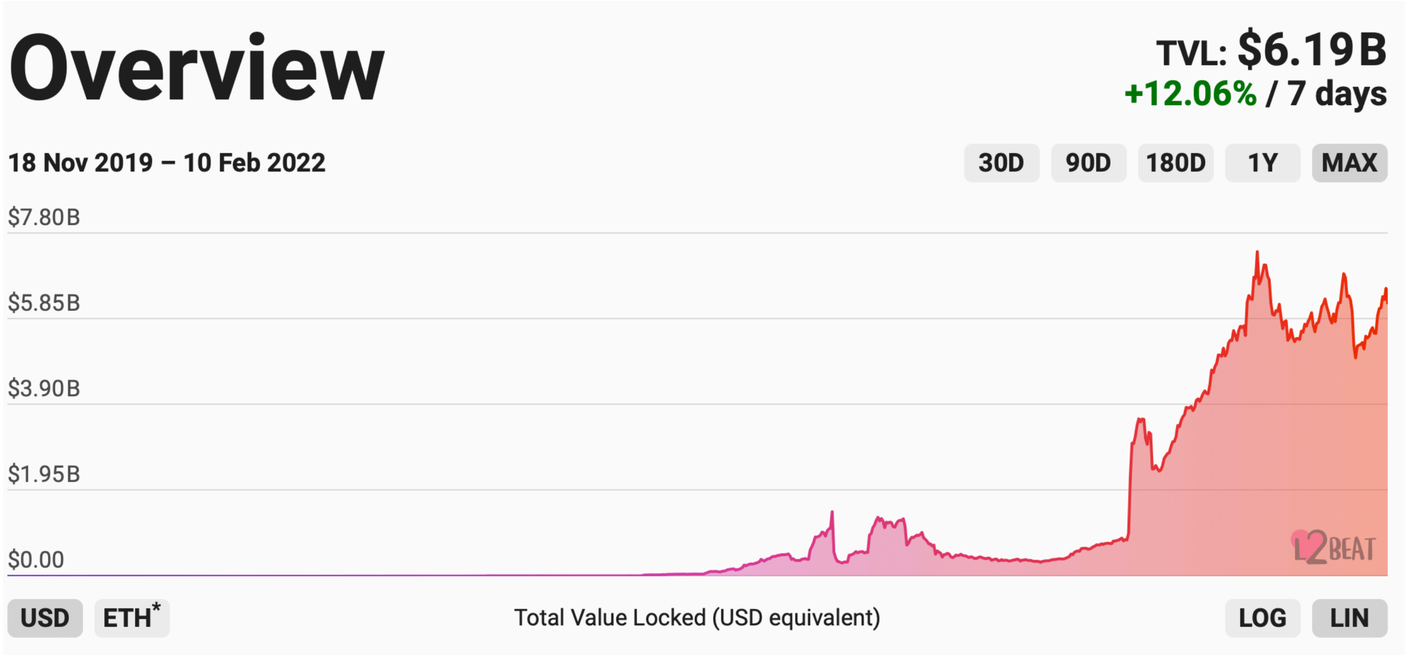
Ethereum's main simplifications are that it (**1**) allows only integers and (**2**) prohibits RAND()

There are more constraints - gas limit, limited memory/stack, everything takes place inside the EVM, but those two are enough for now

That makes it possible for hundreds/thousands/millions of computers to agree on the "right answer" and thus what a correct state/block is

boba.network

The rise of L2s - unlocking compute?



The original motivation of L2s are to help with **cost** and **scaling/congestion**

Additional major benefit of L2s: **compute**

boba.network

Single-sequencer L2s and compute

In a single-sequencer* L2, the original blockers to floats, RAND(), and talking to other APIs/computers, are **removed** since there is no mining / decentralized consensus

Hypothesis: *the next generation of L2s will allow smart contracts to interact with outside computers and act on their responses within one transaction*

***single-sequencer** - Turing can be directly extended to multiple sequencers provided they are **not** separately running their own decentralized consensus (which would be redundant to Ethereum)

boba.network

Single-sequencer L2s and compute

We overcame EVM's restrictions by modifying Geth ("L2TGeth")

with **ATOMIC** support for RAND() and client.Call()

boba.network

https://bit.ly/getTuring

L2TGeth - A Geth flavor for L2s enabling hybrid compute

From the Solidity perspective, easy to use… one line calls

`turing.getRandom()`

```
// ERC721.sol
random_number = turing.getRandom()

// Test/Debug Response
Turing NFT Random 256
  256 bit random number as a BigInt = 612455941595319977171587766669000355729927578575637133505704086435528306264 92n
  Minted an NFT with Attribute A = 135 and Attribute B = 103
  Minted a pirate with a green hat
  ✓ should mint an NFT with random attributes (65ms)
```

`turing.getCurrentQuote()`

```
urlStr = 'https://i9iznmo33e.execute-api.us-east-1.amazonaws.com/quote'
rate = lending.getCurrentQuote(urlStr, "BTC/USD")

// Test/Debug response
Bitcoin to usd price is 42406.68
timestamp 1642104413221
✓ should get the current Bitcoin - USD price (327ms)
```

boba

boba.network

https://bit.ly/getTuring

L2TGeth - System overview

Central idea

1. L2TGeth intercepts certain calls
2. Then, calls off-chain APIs/RAND()
3. Then, replaces the original call data with modified call data
4. Then, writes both the original transaction and the modified call data into Ethereum L1 for verification, replay, fraud-detection, and replication

Key: **only the sequencer calls APIs** - once the L2 block is written, verifiers and replicas use stored API response data from the block

boba.network

https://bit.ly/getTuring

1

RPC call

L2TGeth

2

isTuring?

L2TGeth

3

User-designated Endpoint

client.Call

responseString

L2TGeth

4

L2TGeth

- Replaces original callData
- run(evm, contract, updated_input)
- evm.Context.Turing = updated_input
- Forms new block*
  - L1Timestamp/blocknumber
  - rawTransaction
  - Updated calldata ('L1Turing')

L2 Blockchain

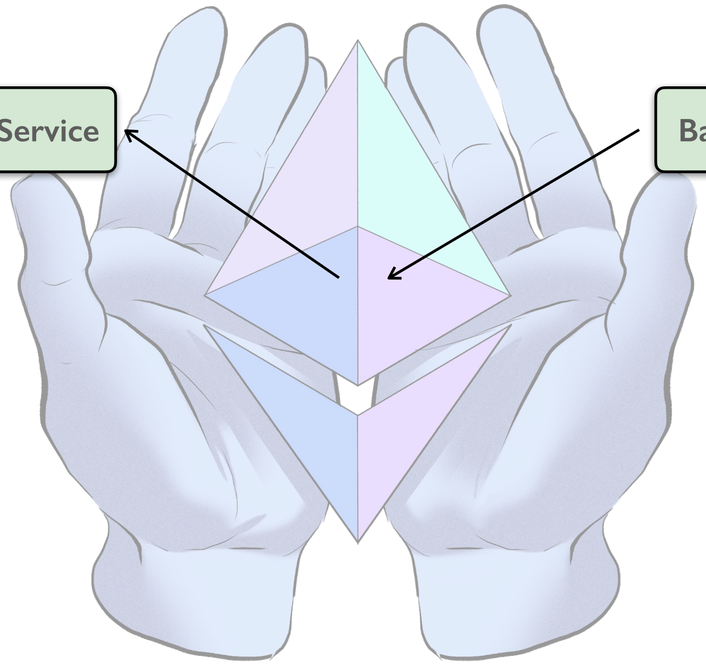
The usual L2 block with updated calldata ('L1Turing')

L2TGeth-Verifier

L1 Data Indexing Service

Batch Submitter

Obtain block
evm.Context.Turing = updated_input
run(evm, contract, evm.Context.Turing)
Forms new block



L2 Data Indexing Service

L2TGeth-Replica

Obtain block
evm.Context.Turing = updated_input
run(evm, contract, evm.Context.Turing)
Forms new block

L2TGeth-Replica

L2TGeth-Replica

L2TGeth-Replica

Boba research

L2TGeth - changes needed - under the hood

- Modify /geth/code/evm.go - especially (evm *EVM) Call()
- Modify the **Ethereum block format** to also include responses from Turing compute requests - *Yes - all data from Off-chain are written to Ethereum L1*
- Add the capability for geth to replay time shifted compute requests based on data written into Ethereum
- Modify transaction/receipt/block/EVM.context and other data types internal to geth
- Modify w.engine.FinalizeAndAssemble and other parts of l2geth/miner/worker.go
- Modify all services that pass data from L2 to L1, and index L1, and inject L1 data into the L2 Geth

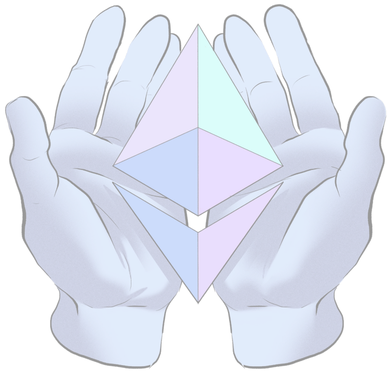
The system is live now on mainnet

boba.network

https://bit.ly/getTuring

L2TGeth - Example uses

- DeFi protocols based on off-chain assets such as real estate
- NFT lending based on off-chain ML-based valuation models
- On-chain rewards based on off-chain activities such as retweets
- NFTs or DAO memberships that are connected to off-chain identities
- Twitter activity-based token fountain
- Atomic random number generator for NFTs



\+ Any other networked computer = Quickly build interesting new stuff

boba.network

https://bit.ly/getTuring

L2TGeth - TLDR

- Detailed writeup at https://bit.ly/getTuring
- **Easy to use** on the smart contract level - e.g. if you need a random number or price quote anywhere in your smart contract, just…

```
turing.getRandom()
```

or

```
turing.getCurrentQuote()
```

boba.network

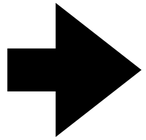
Fundamental premise of Ethereum

A more general and powerful computer unlocks innovation and creativity in decentralized systems

L2s, v0

"Scale Ethereum"

Faster
Cheaper



L2s, v1

"Augment Ethereum"

Faster
Cheaper
& Smarter

boba.network


Boba Hybrid Compute - Turing
Supercharge Ethereum with scalable compute power
What do you want to build today?
https://bit.ly/getTuring

FAQs

- Multiple sequencers - working on it - ***but <u>not</u> workable for systems with <u>decentralized</u> consensus***
- Yes, this also works with **Beacon/post merge**
- ***DoS?* -** each Turing calls costs 0.01 BOBA to (help to) prevent DoS - also have rate limiting
- **Write garbage into L1?** -> Massive ETH cost for us -> hard limit on length of off-chain responses - you cannot write LOTR into L1 with Turing
- **Misuse of your data api**? No - you designate the contracts that can use your off-chain API
- **Slow API response?** - 1200 ms timeout -> revert with the usual 404 etc reported to your contract
- **Only one Turing call per raw Transaction** - cannot chain Turing calls (for multiple reasons)

boba.network

https://bit.ly/getTuring