Experimental Comparison of Blockchain Scaling Protocols Using Virtual Machines (Phase II)

Research Plan, 2019-2020

Michael Lin

1. Rationale

This research will entail the process of deriving performance metrics (a.k.a. "scalability") from blockchains, a combination of peer-to-peer networks and append-only cryptographic data structures for financial or general-purpose state transitions (changes in the state or current progress of a system, such as currency transactions). Blockchains (linearly hashed blocks containing state transitions) are consistently updated and replicated across the network in such a way that liveness is maintained. In this way, blockchains are a special type of state machine. Using blockchains, any computational task can be accomplished without a central managing authority to regulate, such as a bank or state institution. Due to strict consensus rules, any computation can be done with determinism, lack of trust, and a negligible probability of failure.

A blockchain is a promising environment to host a multitude of applications in various private sectors, government agencies, and other areas where permissioned and reliable computation is required for servicing customers and denizens. Although blockchains are a desirable place to keep Internet applications, there lies the problem that [in general] blockchains cannot handle high loads to the extent of current servers such as those used in banks or online payment services. Generalized as a trilemma (a choice between three desirable properties, in which only two may be chosen and the third sacrificed), blockchains should have a secure, scalable, and decentralized nature.

One goal of research in this field is to establish an objective comparison between the ability of different blockchains to scale. The specific task of this Phase II study is to

correct and demonstrate the comparison procedure on the most current version of Bitcoin using a private Bitcoin network in a controlled environment.

2. Experiment Outline (Subject to change through *Project Summary Addendum* of Phase II):

   A. Question

      Will the modified model implementation of *Phase II* (hereby known as `bitcoin-scale-test-v2`) cause the Phase I independent variables to exhibit statistical significance?

   B. Hypothesis

      H_0: When bitcoin-scale-test-v2 is run on a randomly generated network of Bitcoin Core virtual machines, will not exhibit any statistical significance[1] in the regression analysis of *MinFee*, *MedFee*, and *MemPool*.

      H_a: bitcoin-scale-test-v2 will exhibit a statistically significant polynomial or exponential correlation with TPS (or linear, polynomial, or exponential TPS derivative) for *MinFee* and *MedFee*, and *MemPool* will exhibit statistical significance in the custom PoolSize regression curve for each TPS period.

   C. Apparatus Design Goals

      The main goal of this project was to correct the previous model implementation of *Phase I*. The two modifications of `bitcoin-scale-test-v2` from the previous design cannot be used without one another. The first change, the use of `throttle-proxy`, segregates the JSON-RPC server from the actual node listener such that only the node listener experiences latency, using the `proxy` option of `bitcoind`. The second change, the use of multiprocessing,

[1] Read §2(G) for the project's definition of statistical significance.

allows the master node to parallelize RPC calls in such a way that each call is precisely timed with no latency. Both design choices potentially allow the network to act more idealized yet more realistic, in the control and gossip aspects of the blockchain, respectively.

D. Expected Outcomes

The two-part modification, paired with a higher-specification master node, is expected to more accurately detail fee growth. When measured in either atomic or reward units, the rate of change (derivative) of fee growth vs. the network load measured in transactions per second (TPS) should show a continuous growth that can be generalized by a polynomial function or an exponential (power) function. Transaction rate itself is a derivative of load vs. time, so assuming transaction volume is a dimension *D*, and fees are a dimension *S*, then the multiplication of the quantities *S* and $(DT^{-1})^{-1}$ yield a new quantity STD^{-1} (reward unit-seconds per transaction). The parameters of this derivative can be compared with other blockchains. If *S* is a feerate and D is digital storage size, the unit is instead reward unit-seconds per byte.

The procedure requires the following table for symbol reference (*Table 1*):

Table 1. Elementary Symbols and Functions

CT_s	Median block time in seconds
CT_b	Minimum number of block confirmations before safety in a selfish mining attack
SafeTime	Minimum time before safety, in seconds — $SafeTime \equiv CT_s \times CT_b$
Tx	A single transaction (determined by context)
Blk	A single block (determined by context)
Blk_α	Genesis block of the applicable blockchain
B	The blockchain, a vector of blocks, which are vectors of transactions, all sorted by time
r	The current instantaneous count of transactions per second
B_r	The blockchain segment of TPS *r*, such that $B_r \in B$

Blk_r	The first block of the TPS segment B_r.
R()	TPS interval of *Blk* or *Tx*.
min()	Returns smallest valued argument of input arguments
max()	Returns largest valued argument of input arguments
mod(*x*,*y*)	Calculate remainder of the Euclidean division of *x* by *y*
Ω()	Best-case scenario (asymptotic)
O()	Worst-case scenario (asymptotic)
BT()	Number of blocks before inclusion of *Tx*, returns a signed integer (int)
BH()	Height of input *Tx* or *Blk*, returns an unsigned integer (uint) or float.
Fee()	Returns the fee of *Tx* in atomic cryptocurrency units, such as satoshis or wei.
T()	Returns the timestamp of *Tx* or *Blk* as seconds past January 1st, 1970, 00:00:00 UTC as a uint.
|*x*|	Returns the cardinality of set/tuple/vector *x*. Not to be confused with absolute value.
⌈*x*⌉	Ceiling Function. If *x* is an integer, return *x*. If *x* is a float, truncate the decimal portion and add 1.
hash(*x*,*y*)	Calculates hash of x using a hash function, identified by *y*.
Σ(*x*)	Returns $T(Tx \text{ or } Blk) - T(Blk_\alpha)$, or the time from genesis.
Π(x)	Returns $T(Tx \text{ or } Blk) - T(Blk_{R(Tx \text{ or } Blk)})$, or the time from the first block of the TPS interval.
Rwd()	If block, return total block reward without transaction fees. If blockchain, return original block reward.

E. Procedure

Role of Mentor: Diagnoses node behavior, recommendations, and debugging support, and assistance with statistical analysis.

Role of Student: Conducts implementation design, network design, and experiment deployment. Collects and analyzes all data.

1. Technical specifications will be determined to sustain the most resource-intensive private blockchain network of at least 1000 full nodes. Contrary to Phase I, each virtual machine is downsized in its specification that is adequate for a regtest network.
2. Cloud virtual machines (VMs) will be rented out for each protocol configuration from Google Cloud Platform. These virtual machines will be identical in specifications.

3. Each VM will be set up to run its own private testing network (“testnet”) of at least 1000 full nodes (exact number TBD) using premade VM images to deploy an instance group on Google Cloud Platform.
4. For each VM, the operating system (Ubuntu) will be booted up and startup scripts shall execute `bitcoind` and `throttle-proxy` in a terminal multiplexer (`screen`).
5. A master node shall instruct and coordinate all the steps below to nodes. The master node shall have a high CPU and RAM allocation to facilitate communication even in high TPS conditions such as those 60+.
6. A parameters file written in JavaScript Object Notation (JSON) shall be generated for use by the master node for transaction pool mapping. This parameters file shall be structured as a dictionary, with each TPS interval as a key, and contain an iterable of transaction details iterables.
7. For each VM, peer gossip will be started by introducing peer IPs to other peer tables to create a decentralized network graph.
8. For each VM, the genesis block (first block of the blockchain) will be deployed by the first node spun up and block generation is performed by randomly selected nodes in a method causes negligible interference to transaction gossip performance.
9. Transactions shall be sent out at a discrete interval randomly from various nodes in each testnet, with each timestamp recorded by each node. All timestamps shall be averaged using the $\Sigma()$ function. Transactions shall be executed in pools that are created and mapped for every second

asynchronously. The $\Pi()$ function shall be used to provide the pool constructor with the correct list of parameters to use.

10. After running all VMs for 5 days, derive the following (*Table 2*):

Table 2. Intermediate and Dependent Variable Definitions

The lowest safe minimum fee during TPS interval in atomic units	$AtomMinFee_r \coloneqq min\left(\Omega\left(\text{Fee}(Tx)\right)\ \forall\, Tx \in B_r\right)$
The maximum block wait time for a $MinFee$ transaction	$MaxBlock \coloneqq max\left(\mathcal{O}\left(BT(Tx \mid MinFee)\right)\right)\ \forall\, Tx \in B$
Median transaction fee	$AtomMedFee_r \coloneqq \text{med}(Fee(Tx))\ \forall\, Tx \in B_r$
Maximum time in seconds for a *MinFee*	$MaxTime \equiv MaxBlock \times CT_s$
Average of transaction timestamp vs block timestamp (Unused)	$ConfTime \coloneqq \frac{T(\lceil BH(Tx) \rceil) - T(Tx)}{\lvert B \rvert}\ \forall\, Tx \in B$
Mempool size modified sawtooth wave regression	$PoolSize \equiv poolSize(t, r, m, b) \coloneqq mod(r \times t, \text{CT}_\text{s}) + mt + b$
Block timestamp (s)	$sum(T(Blk, node1), T(Blk, node2), \ldots, T(Blk, node1000))/1000$
Atomic units per Reward Unit	$RU \coloneqq (Rwd(\text{B}) \times CT_b)$
Cross-chain comparable *MinFee*	$RUMinFee_r \equiv AtomMinFee_r \div RU$
Cross-chain comparable *MedFee*	$RUMedFee_r \equiv AtomMedFee_r \div RU$

10. Determine if the *MinFee* and *MedFee* hypotheses are satisfied with any statistical significance, in turn satisfying the Phase II alternative hypothesis.

F. Risk and Safety:

- There will be no risks in this experiment. No personal data will be collected or stored, and all computations will be performed in a virtualized and sandboxed environment.

G. Data Analysis (Subject to change through the *Project Summary Addendum*):

- A copy of the blockchain will be stored locally on each node, and analysis is performed by querying nodes for data.

- Transaction timestamps are locally stored by every network node. The network nodes are queried for these timestamps if required, and all timestamps are averaged. All timestamps are relative to genesis.
- Fee data is derived by subtracting output sums from input sums. This is generalized using the *Fee*() function.
- All data is serialized into .csv files. All fees are converted into reward units.
- From here, `SciPy` can calculate the regression and determine the residuals. The R-value is saved for the set of 120 TPS.
- Using the serialized residuals, residual analysis is performed for each TPS.
- Residual analysis is performed visually for each TPS. The analysis is "pre-screened" algorithmically. A consistent sum of high absolute errors (high SAE) is a sign of a flawed model.
- Statistical significance is defined as when the R-squared is above 95% and when the SAE is consistently below 100 nano-reward units (nRU).

H. References

Back, A. (2002, August 1). Hashcash — A Denial of Service Counter-Measure. Retrieved from http://www.hashcash.org/papers/hashcash.pdf

Bitcoin Avg. Transaction Fee chart. (n.d.). Retrieved August 15, 2018, from https://bitinfocharts.com/comparison/bitcoin-transactionfees.html#1y

Buterin, V. (2015, September 14). On Slow and Fast Block Times. Retrieved August 11, 2018, from https://blog.ethereum.org/2015/09/14/on-slow-and-fast-block-times/

Buterin, V., & Griffith, V. (2017). Casper the Friendly Finality Gadget. *Computing Research Repository (CoRR),* arXiv:1710.09437v2.

Buterin & Griffith detail a new type of PoS consensus mechanism, one that discourages forks and makes finality possible. The new system, called Casper, can only be implemented on blockchains that are Turing-complete.

Croman, K., Decker, C., Eyal, I., Gencer, A. E., Juels, A., Kosba, A., . . . Wattenhofer, R. (n.d.). On Scaling Decentralized Blockchains. Retrieved August 15, 2018, from https://fc16.ifca.ai/bitcoin/papers/CDE 16.pdf

3. Project Summary Addendum (Phase I)

- ***Many blockchains use the PoW consensus mechanism, which requires the use of computational power to produce proofs that a certain amount of effort was put into making an immutable collection of transactions (blocks). In turn, these "miners" get rewarded with the native currency of the blockchain. PoW, while preventing a blockchain from being spammed by malicious blocks from an attacker, is not efficient and costs energy. Some blockchains which will be tested use Proof of Stake (PoS), which functions by integrating the stake (or money set aside) for the purposes of creating blocks, and rewards "forgers" differently. Energy use is minimal, and blocks are minted instead.***
- ***Blockchains that have a regression test ("regtest") mode allow instantaneous minting of blocks without any proof (mainly in Bitcoin Core and forked projects such as Litecoin). This is possible in the experiment, as there is no risk of an attack on the private network. In production, this should never be used as it allows for the network to be spammed. Regtest is preferred over PoW or PoS, due to efficiency gains.***

- ***In public production chains, the difficulty is determined by previous block times, to tend toward a certain block time. This is almost never perfect, and block times in a regtest network will reflect this in a method similar to the one that will be used for adding transaction latency.***
- ***CT_s will default to difficulty retarget rule's time in a client's official release, if present. Many clients have such a rule.***
- ***If the blockchain supports sharding into smaller universes ("subchains"), an equal quantity of nodes will be dedicated to each universe, and less will be dedicated toward the unifying blockchain. Miners will perform PoW for both collations (blocks in a subchain) and blocks.***
- ***If the blockchain supports Distributed Proof of Stake (DPoS), a single node will be chosen as the forger, and all stake will go towards that node. DPoS, unlike PoS, collects the stake of endorsers to elect a single forger to produce blocks, wherein PoS, the forger uses their own stake.***
- ***Some blockchains are hybrids between these schemes. Modifications for these chains will be determined on a case-by-case basis.***
- ***CT_b will be 10 blocks for Ethereum, according to a realistic adversary model (Buterin, 2015).***
- ***Ethereum does not possess a regtest mode. However, it does possess both a "devtest" mode, which has very low difficulty, and an alternate Proof of Authority consensus mechanism used purely for testing in a non-deployment environment (Szilágyi, 2017). PoA is very similar to regtest but allows for multiple "sealers" to agree on a block. In***

this experiment, only one sealer will be used as no attacker is present. This saves on computational costs.

- ***The variables ConfTime, MaxBlock, and similar have been dropped. They can be predicted mathematically and will not be tested.***
- ***The oracle and miner nodes are replaced with a random selection of nodes. This eliminates bias by evenly spreading the functions of transaction timestamp collection and block production.***
- ***Instead of adding latency, a bandwidth limiter is added instead to simulate its effects. This is achieved with the wondershaper utility.***
- ***All scripts are in Python 3.X.X syntax. This code will be published separately.***
- ***All software used is licensed under the GNU Public License, copyright holders should find attribution in the code itself.***
- ***Raw data is stored in generic Comma Separated Values.***
- ***No Turing-complete contracts or halting scripts are used in this experiment. All transactions will be controlled by private keys.***

4. ***Project Summary Addendum (Phase II)***
    - ***Command node specifications have been reduced.***
    - ***Instead of using one command node, 30 have been used with precise start and offset timing. These changes allow each to only handle fractional TPS until 1.***
    - ***250 network nodes shall be used with lesser specifications.***