Research Proposal

Improving Bitcoin Transaction Speeds with Proof-of-Trust

Jaegar Sarauer

Applied Research Methods in Software Development

Professor PhD. Borna Noureddin

Abstract

This paper introduces a research proposal for experimental improvements to the Bitcoin ecosystem. The test will focus on data transfer rates between Bitcoin versus the control group, our improvements to Bitcoin. The intention of this study is to increase the data transfer speed to match that of a centralized multiplayer game server.

To prepare for the study, an environment of interconnected local computers running the same test program will be used. Two computers will run the wallet software, allowing them to send and receive transactions, as well as two miners, which will be used to solve the transactions and identify invalid transactions. The first test will run the Bitcoin source code and document the time speeds of the two users and two miners on the network. Afterward, the modifying changes will then be applied to the source code in an attempt enhance transfer rates. The test is run again with the new source code.

It is expected that we see an increase in transaction validations per second than our control group, but imminent side effects are expected. Our testing environment will not be able to provide the same level of security and scalability testing as the live Bitcoin program, these limitations will be mentioned in the proposal body.

Table of Contents

Introduction

Background

Blockchain technology offers the potential for new interconnected user software in a market exploding with interest. Allowing software to decentralize the hosting point for its software while still being secure and reliable across all its users, opens the door for centralized services that can self-run and self-govern by its users.

Bitcoin is where it all began; the software that defines the first successful decentralized technology. Bitcoin was released as one product to the public, to be used for its decentralized currency. However, the public source code was soon separated from its commodity layer to allow advancements into new “alt-coins”. Early altcoins were just alternative coins based on blockchain technology, with their own commodity layer.

Soon after the alt-coin market was introduced, Smart Contracts were the next major innovation to cryptocurrency systems. First developed and used in the Ethereum altcoin [10], Smart Contracts allow for code to be executed on the blockchain, similar to transaction validation. Smart contracts open many doors in blockchain technology, no longer are we restricted to just data representation and transfer.

Following smart contracts is an innovation in transaction proofing. Bitcoin’s proof-of-work uses computational power to verify transaction blocks. This was improved with proof-of-stake, which puts distributed trust in its’ holders value. This allows for the future blocks to be determined by the holders of the coin.

This is the entry point for this research. We will be implementing a design update into Bitcoin’s source code to allow for faster transaction response. Please see the steps section on methodology to see the explanation of how the research will be carried out.

Problem statement

Can a replacement to the proof-of-work system be applied to a cryptocurrency ecosystem to increase the transactions per second to reach a sustainable level for real-time data transfer?

Sub-problems

Sub Problem #1:
Can the transaction processing rate increase without reintroducing vulnerabilities like double spending or monopoly control?

Sub Problem #2:
Increasing the rate updates are broadcasted increases the rate of conflicting solved blocks. Can transactions be ordered properly to avoid chain splitting?

Sub Problem #3:
Keeping transactions secure. Upholding transactions per second to a high level is easy to do if we remove brute force mining and security rules that make blockchains robust. No security measures can be removed unless the problems they solve are also removed with the implementation.

Hypotheses or research questions

Hypothesis #1:
Transaction speeds will increase to far greater than our control group. We will not see a significant increase in successful double spending attacks or monopoly control.

Hypothesis #2:
We will see an increase mining conflicts when double spending attacks are issued by users. This is unsure if it will cause more damage to the test group rather than when it happens in the control group, as it is expected to be more frequent, but to be resolved quicker.

Hypothesis #3:

We will see a decrease in chain splitting and orphan blocks being formed. Security will stay the same as we are lowering the security to a trusted selection of miners that have an automatic trust based on its users. If no security measures are altered outside of mining validation, we should not see any vulnerabilities.

Delimitations

- All tests will be run on the same computer hardware.
- Tests will be run for the control group; the test group will follow.
- Tests will be run in a local environment of two users and two miners to uphold the blockchain network.
- The study will be focused on finding speed improvements to transaction validation process. We will not compare disk storage use, CPU usage, or energy consumption.
- We will not look at efficiencies outside of redefining a proof system. This is to isolate the issue to gather information about proof-of-work specifically.

Definition of terms

Term	Definition
Blockchain	A publicly shared record of transaction history for a cryptocurrency.
Block	A collection of transactions to be mathematically solved and validated by miners to prove the encompassed transactions are valid.
Transaction	A structure of data containing information to send currency from one address to another address.
Real-time	Data transfer rates comparable to a centralized server listening for connections and messages from its clients, and broadcasting updates.
Test group	The modifications made to the Bitcoin source code.
Control group	Unmodified Bitcoin source code used to test our changes against.
Transaction process	The process of a user of the network building a transaction, sending it to miners to be solved and validated, and broadcasting the update to the rest of the network.

Transaction Solving	In Bitcoin, solving a transaction is calculating a resulting hash value of a block that matches or exceeds the criteria of the nonce. The nonce determines how fast a block is solved based on its difficulty and is scaled to throttle the network at 10 minutes block solution times. This is the specific methodology Bitcoin uses to resolve orphan blocks or double spending.
Monopoly control	When a select number of users on the network have the majority computing or value of the blockchain network. This can allow these users to apply changes to the network.

Assumptions

- Implementing a different proofing system or alterations on the proof-of-work system allows us to increase the speed of block releases into the ecosystem. Increasing the speed which blocks are released increases issues with chain splitting if too many miners release blocks that cause a conflict.
- If finding a proper solution to more frequent block solving causes chaining issues too often, we may have to remove the block system into a per-transaction solving method as opposed to blocks of transactions.
- Results will not match that of a live blockchain system, as we are running on a small local network of four computers.
- It is assumed that computer hardware will not cause a conflict with the study as long as the computers used are all identical in specifications.
- Changes will be applied to blockchain source code, the research will not involve writing its own cryptocurrency.

Importance of study

The importance of this study is to drive research forward for decentralized systems. Potential uses for real-time communication through a decentralized network can open a new window for self-governed APIs, game servers, and chat programs.

Related Literature Review

A Literature Review on Blockchain Inefficiencies and Improvisations

Decentralized technology, more commonly recognized as blockchain technology, is on a steep rise in market share. With growth comes innovation, and this shows with newly emerging cryptocurrencies and services that revolve around decentralized technology [3, 5, 6, 8]. However, there are a lot of improvements to be made yet. Bitcoin itself relies on a lot of inefficient systems to provide security, which rides on a type of "majority rules" system to ensure the security of trades between one another [4]. This makes transferring Bitcoins from one person to another slow and inefficient, maxing at an average speed of 10 minutes per grouping of transactions [3]. The goal of this study is to answer the question: "Can redefining the way blockchain technology verifies its transactions allow us to process enough transactions to upkeep a real-time, secure, gameplay network?". To find a solution, we need to improve two issues; increasing the amount of transactions the blockchain can handle, while reducing the time for a transaction to be processed and verified with the rest of the blockchain. We'll be looking at the negatives and positives of other cryptocurrency mining methods, and improvements applied to other cryptocurrencies that allow for these advances.

The idea behind Bitcoin mining is to validate transactions between users to ensure transactions are not malicious or invalid [3]. Each miner will race to see who can find a correct encryption key by brute forcing the key against a block of transactions (known as a "block") until a certain number of criteria in the resulting hash is met, this process is known as proof-of-work [2]. New keys are made and tried again in rapid succession if not met by the criteria. Once a valid key is found, it is sent out to other miners, all competing to find a valid key. Once a miner get a block from someone else, they will test its contents with past transaction data, and ensure they all make sense and match the criteria. Attempts at finding the key for this block are now forfeited if the key is proven true. Once proven true, the miner will build a new block out of queued transactions, and attempt to find a key for this one [3].

Maximum speeds of Bitcoin mining is throttled at one block per approximately 10 minutes, a requirement to ensure the integrity of the system. However, mining with proof-of-

work calculations is very computationally intensive on computers, uses a lot of energy in the process, and slow [4]. Proof-of-work purposely makes it hard for the miners to find a solution so that there aren't conflicts with who has mined the transactions [3, 4].

Assume we reduce the criteria to find the correct key, increasing mining speed. Many miners will then be able to find a correct key quickly, and broadcast their solutions more frequently [5]. This becomes an issue, as each miner builds a block of transactions in different ways, depending on the order they read new transaction requests. Blocks that have transactions in different order will be branched into unverified subchains, leading to orphan blocks[1]. Keeping the time throttled behind difficult calculations help mitigate this risk, and avoid creating orphan blocks.

Not only do we need to throttle the time to avoid conflicts in solutions, it also helps to solve issues with double spending [1, 2, 5, 7]. Double spending can happen when someone attempts to send two transfers of conflicting number of Bitcoins they own quickly. These two transactions (A and B) may be validated at different times, invalidating transaction A once transaction B is validated. This results in a rollback from transaction A and B, which may be too late for the recipient if they are accepting blocks with fewer validations for increased speed [2, 5, 7].

The consensus with typical proof-of-work verification is that when one trait of the chain is improved other traits are reduced [1, 2, 4, 5, 6, 7]. Proof-of-work is a delicate system that is hard to improve without redefining the methodology used for validating transactions.

A proposed improvement for Bitcoin is the implementation of the GHOST protocol [4]. GHOST (Greedy Heaviest Observed Subtree) drastically improves the speed of block processing through accepting the longest sub tree of conflicting blocks once it reaches a minimum length. GHOST will choose the branch that has had the most validated blocks, and after a minimum number of blocks are queued. This will group work into block lists instead of trying to fix conflicting single blocks [4].

---

[1]Orphan blocks are ignored by the consensus of the blockchain when it conflicts with another accepted block which has more miners validating that one than the orphan.

Not only do we require a faster validation solution, it must be *significantly* faster, as the average user of our chain will be sending many more transactions than the typical user of other blockchains. With Bitcoin, miners must process a transaction when someone wants to send coins to another user. We will need to send transactions several times a second to update our player location, actions by the player, external environment changes (AI), and trades.

We can increase the speed of updates by focusing on the important parts to us, what we see on our client, such as other characters and NPCs on the screen and area around us. This will allow players of the game to have faster updates by worrying about what is seen by the player, rather than a conventional server that would need to calculate updates and send them back to the clients all as one update.

Dealing with different types of transactions can be achieved by following a convention introduced in Namecoin, a direct predecessor to Bitcoin [8]. Namecoin is a branch of Bitcoin that replaces the commodity layer of Bitcoin with a key-value dictionary. This is will be essential to store the different variables of each character on the chain.

Since we'll be dealing with a lot more data than a typical single-currency blockchain, excess stored data will be discarded eventually to keep the blockchain size lower. Dead NPC's, dropped items, and game updates outside of any player's view on the world will be candidates for discarding. Namecoin follows a similar convention for removing old keys so that the blockchain stays a manageable size [8]. If an entry in the namecoin table isn't updated, it will eventually be removed.

While solutions are proposed to improve speed and transaction space, combining the solutions may cause conflicting issues. This still leaves the question "Can we introduce multiple changes to the proof-of-work concept without trade-offs?". Other cryptocurrencies have proven they can redefine their methodology of mining, but are often fine-tuned to meet their needs as a blockchain. I suspect defining a new method of validating our transactions will require a completely new system to avoid emerging drawbacks.

Methodology

Instruments

Time-Series design will be used, as the research data will be gathered and stored throughout a test duration of 24 hours. The test and control groups will have code added to them to keep a log of when transactions arrive for focus points dependent on each operating program.

For miners, the focus points will be when transactions are received, when they completed their validation computation, and broadcasting these changes to its listeners.

For wallets (users), the focus points are when transactions are created and sent to miners, and when transactions are received from miners.

This data will be displayed in a time/motion log, comparing our groups' data with pre-solving, solving, and post-solving times of a miner's transaction throughput, and the wallets send and receive rates.

Statistics of other notable data will be stored when detected by any of the computers on the testing network. Data will be collected and stored into a tally sheet; for each our test and control group the following data will be documented:

- Dropped transactions
- Valid transactions
- Invalid transactions
- Double spending attempts
- Double spending successes
- Potential orphaning

This data will be compared with total transactions sent from the listener nodes to find comparable ratios of validation rates and success. Please see the Data Analysis section for details on what statistic methods will be used.

The data will come from recording the activity of the miners and wallets of each the test and control groups. Sources of data for all programs will use a timestamp method to keep track

on when the data was recorded. This allows data to be converted into any time interval, and see latency between communication. Final data will be delivered in intervals of one hour.

Reliability and repeatability by others

Source code from both the control and test groups will be uploaded for public use to GitHub.com with steps to recreate the experiment. Anyone with sufficient hardware and time to setup the nodes and communicating network will be able to re-run the tests themselves. Since we will be running the same source code, other tests should result in similar matching data, except for network or hardware variability.

Sampling Size

Experimental design

I will be using the Quasi Experimental Design; Control-Group Time Series Design, to structure my experiment results. Since we will be running the stock Bitcoin system as a baseline for testing to compare our improvements, that will be our control group for the experiments.

The time series design will be utilized by measuring the speeds of transaction communication over a constant span of 24 hours. Each transaction is recorded by the program utilizing it, at what time, and at what part in the program. A 24-hour test will be adequate to analyze how the test group performs on a small chain, as well as how it handles susceptible double spending attacks, and chain splitting.

Data Collection

Data will be obtained through running each of our programs on four different computers and communicating across them. Two computers will be mining blocks to validate and communicate the transactions. Two other computers will be used as communication endpoints. Computer A and D will send and receive transactions, and B and C will attempt to validate them, and broadcast the verified versions. Please see Appendix A for a visual diagram of how a transaction will be sent through the testing environment.

Data will be recorded to a log file under each program. Each time a transaction is solved, validated, made, sent, or received it is recorded by the program that is currently handling that

transaction. The data logs will be organized and condensed into a table of history for each transaction.

This user environment is not of a typical block chain environment. We are running our tests in extremely small testing situations. This means it cannot account for scalability issues, unprocessed transaction queues, or enough validations on a solved block to match a typical ecosystem. This isn't an issue as the main defining bottleneck is solving the blocks.

Data Analysis

We'll be utilizing significance testing with our quantitative research data to calculate averages for our test group to support the alternative hypothesis. Data that will be collected from our test will be used to answer hypothesis' for the main problem and three sub-problems. After data is collected and organized, the statistics of the data will be calculated and stored as explained in Appendix B.

Criteria of admissibility

This research will be conducted only by Jaegar Sarauer. The researcher is a software developer with a background in data communications, capable of understanding and modifying networking code to achieve the goal test software. The project will however also require experience with decentralized software, which the researcher has no previous experience with, but adequate understanding of the code structure. The researcher's data analysis will follow guidelines taught in his practical research course.

Outline of study

Steps

This study focusses on the optimizations that a trusted committee of miners can provide. The trusted board of miners decide on the validity of transactions. We "trust" these select miners to process the transactions so that they may do it instantly, without brute force math like proof-of-work, and allow them to quickly notify the rest of the network.

Trusting, or Proof-of-Trust, must be done by selecting a group of trustable members. These would eventually be able to be voted upon to keep the power in the hands of the people,

but irrelevant to our speed test. Transactions are then sent to these miners directly, where a group of them validate the transaction and share it with the rest of the miners. The entire group of miners broadcasts the new valid transaction to all the wallets listening on the network.

This removes block solving, arguably the largest bottleneck in Bitcoin. Calculating the correct criteria for a block to be formed in Bitcoin is completely overlooked here for there is no reason to throttle the rate blocks get released into the market when the miners communicate between each other and can trust each other partially, at least more than just any miners.

Once the test group is complete (the above changes applied to Bitcoin source code), the tests will be run. To test the differences between Bitcoin and the changes, we'll have to make sure the data is as consistent as possible by ensuring the testing is run under the same environment and variables.

To do this, we will be throttling the transactions sent per second to a high, constant rate. Wallets will also be programmed to periodically send transactions that may cause double spending or orphaning to see how they impact the miners. Slight variances in the amount of data recorded will be adjusted based on the difference in total transaction at the end of each test. This allows for a small buffer in error if the amount of transactions in each test are different.

Data analysis will be collected from the logs of each program and each transaction will be organized into a timeline. The timeline for every transaction is analyzed and stats are extracted from the timestamps and instructions the logs left with the transaction id.

The data is finally organized into time/motion logs from the data extracted. Tally sheets are populated from total numbers of each unit of data over the duration of the study.

Timeline

10 weeks will be allocated to the implementing and testing phase. This phase is for the development of the proof-of-trust design, data logging functions and implementation into both test and control group, and alterations to both the test and control group wallets to send our packets. Testing of the entire ecosystem will be run each implementation, first starting with the proof-of-trust.

Once the programs are tested thoroughly for reliability, testing phase will start. Testing will include the setup and execution of the data collection step. An environment for the control group will be setup and execute in our computer environment, followed by the test group. Both programs will be run for 24 hours.

the final phase, data analysis, is begun once both tests have been executed. This step is expected to take 4 weeks to write programs to organize the data, store it properly, generate graphs, and verify the data.

Required Resources

Access to four desktop computers with identical specifications. Ideal computer specifications 6^{th} gen i5 3.4GHz IMacs. No extra funding is required to conduct this research.

References

[1] Gervais, A., Ritzdorf, H., Karame, G. O., & Capkun, S. (2015). Tampering with the Delivery of Blocks and Transactions in Bitcoin. Proceedings of the 22nd ACM SIGSAC Conference on Computer and Communications Security - CCS 15. doi:10.1145/2810103.2813655

[2] Karame, G. O., Androulaki, E., & Capkun, S. (2012). Double-spending fast payments in Bitcoin. Proceedings of the 2012 ACM conference on Computer and communications security - CCS 12. doi:10.1145/2382196.2382292

[3] Witte, J. H. (2016). The Blockchain: A Gentle Introduction. SSRN Electronic Journal. doi:10.2139/ssrn.2887567

[4] Romano, D., & Schmid, G. (2017). Beyond Bitcoin: A Critical Look at Blockchain-Based Systems. Cryptography, 1(2), 15. doi:10.3390/cryptography1020015

[5] Kiayias, A., & Panagiotakos, G. (2016). Speed-Security Tradeoffs in Blockchain Protocols, 1-27. Retrieved November 1, 2017.

[6] Kiayias, A., & Panagiotakos, G. (n.d.). On Trees, Chains and Fast Transactions in the Blockchain, 1-27. Retrieved November 1, 2017.

[7] Duffield, E., Schinzel, H., & Gutierrez, F. (2014). Transaction Locking and Masternode Consensus: A Mechanism for Mitigating Double Spending Attacks, (2), 1-14.

[8] Kraft, D. (2016). Game Channels for Trustless Off-Chain Interactions in Decentralized Virtual Worlds. Ledger, 1, 84-98. doi:10.5195/ledger.2016.15

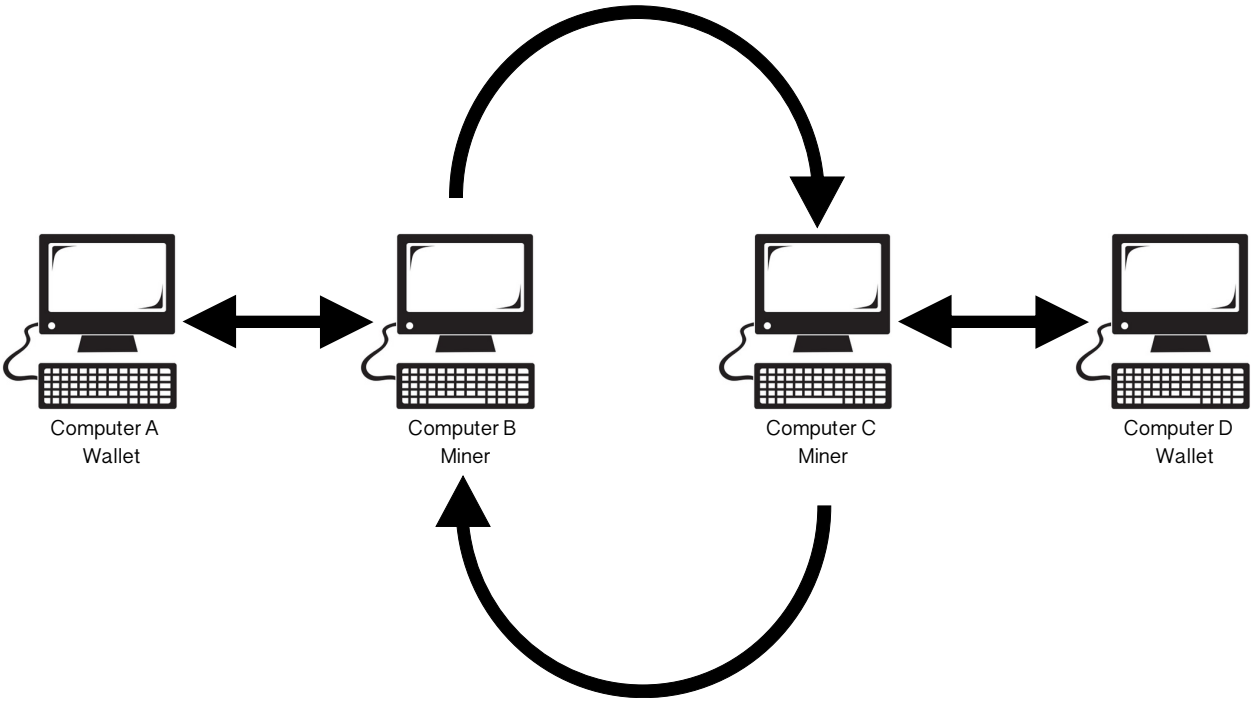
[9] Sompolinsky, Y., & Zohar, A. (2015, January). Secure high-rate transaction processing in Bitcoin. In International Conference on Financial Cryptography and Data Security (pp. 507-527). Springer, Berlin, Heidelberg.

[10] Christidis, K., & Devetsikiotis, M. (2016). Blockchains and Smart Contracts for the Internet of Things. IEEE Access, 4, 2292-2303. doi:10.1109/access.2016.2566339

Appendices

Appendix A:

A diagram of transaction flow during the test. Computers A and D will be sender and receivers of transactions, and Computer B and C will validate and manage the communication of A and D's transactions.



Appendix B:

Data Type	Problem	Usage	Display
Transaction Average Latency	Main Problem	Used to determine the average transaction speed at time position x.	Time/Motion Log
Total Transactions Sent	Main Problem	Used to determine overall transactions sent throughout the test. Used for defining a ratio in case transactions sent	Time Motion Log Tally Sheet

		are not equal between both tests.	
Total Transactions Received	Main Problem	Used to determine overall transactions received throughout the test. Used for defining a ratio in case transactions received are not equal between both tests.	Time Motion Log Tally Sheet
Total Invalid Transactions	Main Problem Sub Problem #1, #3	Total amount of transactions rejected by the miners. These are used to determine our overall fail rate of transactions.	Time Motion Log Tally Sheet
Double Spending Attempts	Sub Problem #1	A recognized attempt at double spending. An attempt is recorded when two contradicting transactions have yet to be validated on the blockchain.	Tally Sheet
Double Spending Success	Sub Problem #1	A recognized success at double spending, we hope to see none of these or it becomes an insecurity in the blockchain's trading system. A success is recorded when a sub chain formed from this separation is found.	Tally Sheet
Orphaning Attempts	Sub Problem #2	A recognized split in the blockchain, where two or more groups of miners have	Tally Sheet

		conflicting chains of transactions.	
Orphaning Success	Sub Problem #2	A recognized split in the blockchain, where two or more groups of miners have conflicting chains of transactions. Successes are regarded as when a sub chain must be backtracked and restructured instead of just dropping lesser important chains.	Tally Sheet