Machine Learning View on Blockchain Parameter Adjustment

Vladislav Amelin*
AI & IT lab Z-union, Innovation center Skolkovo, Moscow and Velas, Zurich
Switzerland
cto@z-union.ru

Nikita Romanov
AI & IT lab Z-union, Innovation center Skolkovo, Moscow and Velas, Zurich
Switzerland
nromanov@z-union.ru

Robert Vasilyev
AI & IT lab Z-union, Innovation center Skolkovo, Moscow, Russia and Velas, Zurich
Switzerland
ceo@z-union.ru

Rostyslav Shvets
AI & IT lab Z-union, Innovation center Skolkovo, Moscow, Russia and Velas, Zurich
Switzerland
shvets.rostyslav@gmail.com

Yury Yanovich
Center for Computational and Data-Intensive Science and Engineering, Skolkovo Institute of Science and technology, Moscow, Russia and Sirius University of Science and Technology, Sochi
Russia
y.yanovich@skoltech.ru

Viacheslav Zhygulin
AI & IT lab Z-union, Innovation center Skolkovo, Moscow, Russia and Velas, Zurich
Switzerland
s.zhygulin@gmail.com

ABSTRACT

A fundamental problem in distributed computing is achieving agreement among many parties for a single data value in the presence of faulty processes–to get consensus. The consensus mechanism is an underlying part of blockchain design and commits new blocks and changes protocol itself. In addition to classic correctness requirements, blockchains need specific ones: high performance regarding transactions per second, fast transaction confirmation, etc. Blockchains control the requirements with parameters. But how to meet qualitative and optimize quantitative requirements? Typically we have the main blockchain network without access to try different parameters and the test network to do whatever we want. In the paper, we provide a machine learning view on the blockchain parameter adjustment. We list the blockchain parameters for Solana blockchain and apply feature importance to select the most significant parameters during the forthcoming optimization.

CCS CONCEPTS

• **Computer systems organization** → **Peer-to-peer architectures**; • **Computing methodologies** → **Modeling and simulation**; **Machine learning**.

KEYWORDS

blockchain, machine learning, simulation, optimization, consensus

*The authors are listed in alphabetical order.

BIOTC 2021, July 8–10, 2021, Ho Chi Minh City, Vietnam

ACM ISBN 978-1-4503-8951-8/21/07…$15.00
https://doi.org/10.1145/3475992.3475998

ACM Reference Format:
Vladislav Amelin, Nikita Romanov, Robert Vasilyev, Rostyslav Shvets, Yury Yanovich, and Viacheslav Zhygulin. 2021. Machine Learning View on Blockchain Parameter Adjustment. In *2021 3rd Blockchain and Internet of Things Conference (BIOTC 2021) (BIOTC 2021), July 8–10, 2021, Ho Chi Minh City, Vietnam.* ACM, New York, NY, USA, 6 pages. https://doi.org/10.1145/3475992.3475998

1 INTRODUCTION

A fundamental problem in distributed computing is achieving agreement among many parties for a single data value in the presence of faulty processes–to get consensus [22, 25, 38]. The consensus mechanism is an underlying part of blockchain design [47]: it commits new blocks and changes protocol itself. In addition to classic correctness requirements, blockchains need specific ones [13, 37, 58, 65]: high performance regarding transactions per second, fast transaction confirmation, low block production time, etc. The requirements compete, for example, transaction confirmation time and performance. So the blockchain has a Pareto front [19] of optimal operation regimes and needs to pick the proper regime through a trade-off between the current system needs.

Consensus mechanisms control the requirements with parameters. But how to meet qualitative requirements and optimize quantitative properties? For a given system we have the main blockchain network without access to change parameters for research purposes. The source code of the blockchain is publicly available. So we can launch a test system in our own environment and vary parameters as we wish. We observe parameters for both main and test network systems in time together with the resulting operation quantities. The main network provides historical data while the test network is an interactive black box.

The blockchain system runs in a distributed network. Both network and protocol parameters define the blockchain operation regime. Some of the network parameters are observable, for example, the current pool of unconfirmed transactions; some of the network parameters are not observable (unobservable, latent), for example, blockchain network graph, connection latency and bandwidth [21, 27]. Unobservable parameters play an essential role in the performance, for example, average propagation delay affects transaction latency, and it is useful to estimate them with a certain accuracy [63]. Latent parameters result in the impossibility to emulate the main network with the test network but only simulate. So we have a multi-objective optimization problem with multifidelity sources: high accuracy dataset and low accuracy interactive black box.

In this paper, we present a machine learning view on a blockchain parameters adjustment problem. As of our knowledge, such a view is not discussed in the scientific literature yet. We also perform feature importance analysis for Solana blockchain [64] by SHAP algorithm [42] to showcase how to apply machine learning to the blockchain parameters adjustment subtasks. The rest of the paper is organized as follows. Section 2 provides few insides on the blockchain parameters and their relation to the operating regime. Section 3 lists machine learning problems for blockchain parameter adjustment. We consider the Solana blockchain a model example for an adjustment and sum up its parameters in Section 4. SHAP algorithm provides feature importance analysis for Solana's parameters in Section 5. Finally, Section 6 presents our conclusions.

2 BLOCKCHAIN PROTOCOL PARAMETERS

Blockchains have parameters from the very beginning. Bitcoin uses a Proof-of-Work consensus mechanism: to generate a correct block, you solve a computational puzzle. The difficulty of the puzzle automatically adjusts, having one block per ten minutes on average. Ten minutes is enough time to synchronize the network at a new block height [20, 46]–we have seen no orphan block since 2017. The absence of orphan blocks is good for blockchain security: it prevents fork-related attacks, for example, double-spending [9, 66]. The transaction acceptance latency can not be less than half of the average block acceptance time under the classic assumption of the user transactions to be a random Poisson flow with a constant rate [43] at a block generation time scale. Blockchain throughput is expressed in transaction per second (tps), and it is upper bounded by
ограничена сверху соотношением емкости транзакций блока и времени принятия блока
the ratio of the block transaction capacity and block acceptance time. In Bitcoin, the block size is limited in weight [41] which is a sum of included transaction weights. The weight of a transaction is the weighted sum of its parts in Bytes, where the witness has a smaller factor. The transaction weight depends on the number of inputs and outputs together with their type. In practice, block weight limit and difficulty adjustment result in around 2500 transactions per block or 4 tps.

Bitcoin is a peer-to-peer network, and everyone is can join it as a (full) node: store their copy of its blockchain and communicate with other network participants. To discover a peer to connect the network, you request one of the six DNS servers [6, 8] for a subset of existing nodes. After it, you open 8 incoming and support up to 117 outgoing connections. Magic constants 8 and 117 are not a strict rule but rather a common trade-off between Internet traffic and connectivity. The bigger these numbers, the more robust Bitcoin is to split attacks [15, 35], but the wider Internet connection channel you need to maintain the node. While DNS servers keep the list of online nodes up-to-date, the network topology is unknown to prevent Eclipse and Sybil attacks [23, 31, 60].

Bitcoin transaction language–Script–has limited flexibility. For example, there are no cycles and recursions in it. The simple structure of Script makes transaction security analysis simple and allows to upper bound the execution time by the linear function of the transaction size. Ethereum blockchain [13]–the most popular platform for smart contracts–uses Solidity programming language for transactions. Solidity is Turing-complete. Such property both allows arbitrary algorithms as transactions and makes their security analysis a challenge [17, 24]. For example, the Solidity program can contain an infinite loop. So the transaction execution time can be a bottleneck in the network synchronization in addition to the block size. Ethereum measures the real-time execution time in a unit called gas. The block has a gas limit rather than a size limit. Ethereum has a lower average block generation time–around 15 seconds, resulting in lots of short branches. The GHOST algorithm [56] handles branches and defines the main chain in Ethereum. Only up to 7 generations of uncle blocks are allowed: unlimited GHOST includes too many complications into calculating valid uncles for a given block and does not motivate to mine on the main chain compared to alternative branches.

The Proof-of-Work (PoW) approach is not the only and, probably, not the best way to reach a consensus on a new block [2, 18]. Delegated Proof-of-Stake (Delegated PoS, DPoS) [26, 33] is a possible alternative. Like the computational power is a limited resource in the PoW, the crypto coin is a limited resource in the PoS. The computational power defines the probability to generate a new block in the PoW, the amount of coins owned defines the probability to generate a new block in the PoS. Small stakeholders may not be interested in block generation due to the nonzero cost for full node maintenance, and too seldom payouts [52, 59]. So PoW participants join mining pools, and PoS participants nominate delegates–DPoS–to generate blocks on their behalf. An example of DPoS specific parameters is the stake size to become a delegate or the maximum possible number of delegates.

Blockchains progress over time and change. Protocols can contain the flexibility to process the needs within the predefined rules (for example, Bitcoin block difficulty adjustment). Soft (for example, segregated witness [41]) and hard (for example, value overflow incident [49]) forks cover the rest of the needs. The question of whether to use a soft or a hard fork is discussional [14]. So it is better to have a mechanism for changes by design, and the timing of the changes comes as an extra set of blockchain parameters to be adjusted.

3 MACHINE LEARNING VIEW

In this Section, we consider machine learning tasks for blockchain parameter adjustment. Let

- x be the vector of observable uncontrollable blockchain parameters. For example, the number of nodes is a component of x.
- θ be the vector of (observable) controllable blockchain parameters. For example, block size is a component of θ. In general it may depends from uncontrollable parameters $\theta(x)$.

- ξ be the latent (uncontrollable) blockchain parameters. For example, nodes' computational power is a component of ξ.
- y be the vector of the blockchain operation regime with given parameters (x, θ, ξ). For example, transactions per second is a component of y.

The function $y = f(x, \theta, \xi)$ is unknown and we approximate it. Firstly, we can collect historical data for the blockchain mainnet operation $\{(x_n, \theta_n, y_n)\}_{n=1}^{N}$, where N is the sample size. The corresponding ξ_n is not known and can be different for different n. We can try to ignore them as nuisance parameters or consider them as uncertain. Secondly, the blockchain source code is publicly available. So we can launch the testnet in our own environment. Some information about ξ can be extracted in this case or, at least, we can keep the same environment and ensure unknown but same ξ. The testnet becomes an interactive model to be considered as a black box [48, 53]: send the input (x, θ) and for a reasonable time get the output y.

The input vector (x, θ) is high dimensional, and the analysis of f is prone to the curse of dimensionality [7]. A dataset-driven feature importance score [42, 67] or black box-driven sensitivity analysis [10, 11, 50, 54] on how useful they are at predicting a target vector y may be useful for further feature extraction [29] or dimensionality reduction [3, 5, 39, 57].

The prediction of y for a given (x, θ) refers to a regression problem for continuous output components and to a classification problem for categorical output components [7, 30]. A given black box may be beneficial for the resulting model quality with the same dataset size [12, 32]. Data fusion for multifidelity sources [44, 62]–high accuracy mainnet dataset and low accuracy interactive testnet black box–is the option to transfer knowledge from testnet to mainnet.

The main task is surrogate optimization [36, 45, 51]: optimize θ to have the optimal regime under computational budget constraints. We want the robust optimization [4] to provide a solution, which is stable under small fluctuations of x.

Finally, vectors x, θ, ξ, y are in time: $y(t) = f(x(t), \theta(t), \xi(t))$. So we consider either stationary states or work directly with time series. Uncommon behaviour detection is of interest for the times series. Outlier and change point detection provide solution to the problem [1, 34].

4 SOLANA BLOCKCHAIN CASE

The approach from Section 3 is general, and one can apply it to various blockchains. Velas blockchain [61] is under development and states artificial intelligence and machine learning to be a part of its AIDPOS consensus mechanism. We have been inspired by such an idea and conducted our research with Solana blockchain [64]–a backbone for Velas.

Eight features characterize Solana among other blockchains [55]

- **Proof-of-History:** A clock before consensus
- **Tower Byzantine fault-tolerance:** A Proof-of-History optimized version of PBFT [16]
- **Turbine:** A block propagation protocol
- **Gulfstream:** Mempool-less transaction forwarding protocol
- **Sealevel:** Parallel smart contracts run-time
- **Pipelining:** Transaction processing unit for validation
- **Cloudbreak:** Horizontally-scaled accounts database
- **Archivers:** Distributed ledger storage.

Solana version 1.5.0 has 89 controllable parameters θ including a heap size (transaction synchronization parameter), count of slots per epoch (consensus parameter), default stake placed with the bootstrap validator (maximum decentralization parameter). Many extra parameters are written to the log and can be parsed. Some parameters characterize the ecosystem. For example, the total number of nodes, the number of blockchain accounts and other things that can describe the entire blockchain. Parameters from log and ecosystem are uncontrollable and result in more than 300-dimensional vector x. Latent parameters ξ introduce randomness to experiments. Generally, it is a complicated task to estimate such values but their influence does not allow us to conduct exactly identical experiments even with the same $\theta_i = \theta_j$ and $x_i = x_j$. The most striking examples from this group are bandwidth and latency of Internet connection, disk input/output operations per second.

Due to the Solana design, the ratio of sent transactions and successfully written in block, i.e. finalized, maybe not equal to 1. The percent of dropped transactions through a time interval is called droprate. The droprate and transaction per second (tps) are the main pair to characterize Solana performance like latency and tps for Bitcoin and Ethereum. Note that tps counts only successful transactions. During the current research, y is a two-dimensional vector with droprate and tps. Solana's community provides a tool to calculate these parameters. The tool creates thousands of accounts, sends tokens among them during one million iterations and after calculates average statistics.

5 NUMERICAL EXPERIMENTS

To evaluate the viability of the proposed approach, we generated a dataset on Solana testnet, applied feature importance, and examined the results with the human expert expectations.

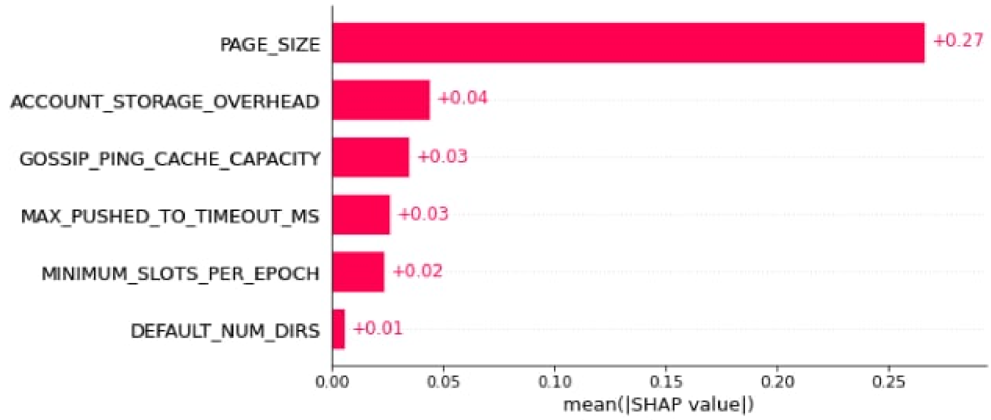
5.1 Dataset Generation

Mainnet nodes do not want to grant full access to their information due to security reasons. Parameters vary only in a narrow range during the mainnet operation. So a simulation process on our own testnet is the best way to collect all the possible data. We wrote a software development kit (SDK) that allows running each Solana node in Docker Swarm, put them in a specific environment, and set interaction rules. The modular structure of such SDK can expand for various scenarios: from the most simple case when we just start a proper configuration and nodes sign empty blocks to the simulation of different kinds of attacks. We use the following software

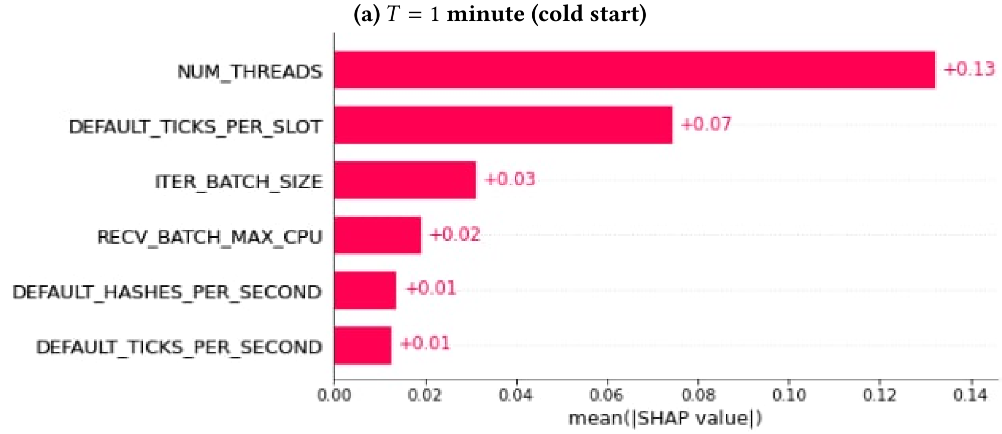
- Ubuntu 18.04
- Docker (in Swarm mode) 19.03.11
- Rust 1.48 and Solana 1.5.0
- Python 3.7 and Solana-py 0.6.4.

Experiments are performed on 3 local servers based on AMD Ryzen 9 processors with 64 GB RAM and NVMe SSD.

Parameters $\theta_n, \quad n = 1, \ldots, N$ are generated as independent uniformly distributed vectors with the ranges from Solana documentation. The vector θ has both continuous and categorial components. Some vectors θ results in a broken blockchain configuration



(a) $T = 1$ **minute (cold start)**



(b) $T = 8$ **minutes (normal start)**

Figure 1: Blockchain top 6 parameter in SHAP values [42] descending order. Timeout before measurement (a) $T = 1$ **and (b)** $T = 8$ **minutes correspondingly**

and are omitted without any postprocessing. Timeout T has sufficient impact on y_n: nodes establish connections, create accounts, signs the first blocks and etc. We use two different times: $T = 1$ minute–cold start–the short time after the system launch, $T = 8$ minutes–normal start–the long time after the system launch. The resulting dataset consists $N = 400$ points for two different times T each. The protocol to generate a sample point for a given θ is as follows

(1) Launch testnet with the θ defined in the genesis block.
(2) Wait for a predefined period of time T.
(3) Send 1 million transactions by a Solana bench tool.
(4) Measure y as average quantities during the processing of the 1 million transactions.
(5) Extract x from the initial configuration and logs for the measurement period.

A single data point generation takes from 4 to 20 minutes and depends on timeout T and concrete values of θ. The process can be easily parallelized by assigning computations for different (θ, T) to different instances.

5.2 Feature Importance

The input space θ is too high-dimensional for dense sampling, and the current dataset is relatively small (≈ 4.5 points per dimension). The effective (intrinsic) dimensionality [28, 40] of the simulation may be lower, or, at least, some of the features do not make a significant impact on the outputs. We used SHapley Additive exPlanations (SHAP) algorithm [42] to assign importance to features. SHAP takes dataset, recommender model and computes the importance scores. The scores are nonnegative and sum to 1. For a fixed T vectors $x_n, n = 1, \ldots, N$ are almost constant in our dataset. So we consider $\{(\theta_n, y_n)\}_{n=1}^{N}$ as an input for the SHAP algorithm. And the output is a vector with the mean absolute SHAP scores for each component of θ. We used a two layers fully-connected neural net as a recommender model.

The top features by their importance score are provided in Figure 1. The names of the features are from Solana implementation [55]. Two subfigures correspond to different timeouts $T = 1$ (cold start) and $T = 8$ (normal start) minutes. The major difference between the two experiments is as follows: Solana creates many accounts before sending transactions. It requires some time and memory to complete: the more time, the more chances to finish the process before the transfer begins. Vise versa, a big amount of allocated memory allows to speed up the accounts creation as the process can create more accounts simultaneously. So to achieve better y_i we need to increase memory page size (PAGE_SIZE parameter) if we want to run benchmark immediately or almost immediately. But for a big timeout, the influence of the page size parameter becomes smaller, and the number of ticks per slot (NUM_THREATS parameter for Delegated Proof-of-Stake) comes into play. So the results of feature importance are plausible.

6 CONCLUSION

Modern blockchain architectures keep a lot of parameters constant during the lifetime. This could be sub-optimal for security and performance. Adaptive parameters selection can provide a good trade-off between them. This paper proposes a machine learning view on blockchain parameter adjustment problem, considers Solana blockchain as a model example and applies the SHAP algorithm for the feature importance. The results meet experts' expectations and prove the approach's viability.

We consider testbed for data generation as a next step. The main challenge here is to fix uncertainty ξ. Dataset generation and its comprehensive study via machine learning methods are also of interest.

ACKNOWLEDGMENTS

The authors are grateful to the reviewers for their constructive input. The work of Vladislav Amelin, Nikita Romanov, Rostyslav Shvets, Robert Vasilyev and Viacheslav Zhygulin is supported by Velas as a part of AIDPOS research (https://velasblockchain.medium.com/velas-technologies-aidpos-70d0244467db). The reported study of Yury Yanovich was funded by RFBR under Grant No.: 19-37-51036 https://www.rfbr.ru/rffi/ru/contest/n_812/o_2095160.

REFERENCES

[1] Samaneh Aminikhanghahi and Diane J. Cook. 2017. A survey of methods for time series change point detection. *Knowledge and Information Systems* 51, 2 (5 2017), 339–367. https://doi.org/10.1007/s10115-016-0987-z
[2] Michael Bedford Taylor. 2017. The Evolution of Bitcoin Hardware. *Computer* 50, 9 (2017), 58–66. https://doi.org/10.1109/MC.2017.3571056
[3] Mikhail Belkin and Partha Niyogi. 2003. Laplacian Eigenmaps for dimensionality reduction and data representation. *Journal Neural Computation* 15, 6 (2003), 1373–1396. https://doi.org/10.1.1.131.3745
[4] Aharon Ben-Tal, Laurent El Ghaoui, and Arkadi Nemirovski. 2009. *Robust optimization.* 541 pages.
[5] Alexander Bernstein, Alexander Kuleshov, and Yury Yanovich. 2015. Manifold Learning in Regression Tasks. In *Lecture Notes in Computer Science*. Vol. 9047. 414–423. https://doi.org/10.1007/978-3-319-17091-6_36
[6] Alex Biryukov and Ivan Pustogarov. 2015. Bitcoin over Tor isn't a Good Idea. In *2015 IEEE Symposium on Security and Privacy*. IEEE, 122–134. https://doi.org/10.1109/SP.2015.15
[7] Christopher M. Bishop. 2006. *Pattern Recognition and Machine Learning*. Springer-Verlag, New York.
[8] BitcoinProject. 2014. Bitcoin Developer Reference. https://bitcoin.org/en/developer-reference#target-nbits
[9] Bitfury Group. 2016. On Blockchain Auditability. *bitfury.com* (2016), 1–40. https://bitfury.com/content/downloads/bitfury-white-paper-on-blockchain-auditability.pdf
[10] Evgeny Burnaev, Ivan Panin, and Bruno Sudret. 2016. Effective Design for Sobol Indices Estimation Based on Polynomial Chaos Expansions. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 9653. Springer Verlag, 165–184. https://doi.org/10.1007/978-3-319-33395-3_12
[11] Evgeny Burnaev, Ivan Panin, and Bruno Sudret. 2017. Efficient design of experiments for sensitivity analysis based on polynomial chaos expansions. *Annals of Mathematics and Artificial Intelligence* 81, 1-2 (10 2017), 187–207. https://doi.org/10.1007/s10472-017-9542-1
[12] Evgeny Burnaev and Maxim Panov. 2015. Adaptive Design of Experiments Based on Gaussian Processes. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 9047. Springer Verlag, 116–125. https://doi.org/10.1007/978-3-319-17091-6_7
[13] Vitalik Buterin. 2014. Ethereum White Paper: A Next Generation Smart Contract & Decentralized Application Platform. *Etherum* January (2014), 1–36. https://github.com/ethereum/wiki/wiki/White-Paper
[14] Vitalik Buterin. 2017. Hard Forks, Soft Forks, Defaults and Coercion. https://vitalik.ca/general/2017/03/14/forks_and_markets.html
[15] Vitalik Buterin, Diego Hernandez, Thor Kamphefner, Khiem Pham, Zhi Qiao, Danny Ryan, Juhyeok Sin, Ying Wang, and Yan X Zhang. 2020. Combining GHOST and Casper. *arXiv* (3 2020). http://arxiv.org/abs/2003.03052
[16] Miguel Castro and Barbara Liskov. 1999. Practical Byzantine Fault Tolerance. *Proceedings of the Third Symposium on Operating Systems Design OSDI '99* February (1999), 173–186. http://pmg.csail.mit.edu/papers/osdi99.pdf
[17] Huashan Chen, Marcus Pendleton, Laurent Njilla, and Shouhuai Xu. 2020. A Survey on Ethereum Systems Security. *Comput. Surveys* 53, 3 (7 2020), 1–43. https://doi.org/10.1145/3391195
[18] Alex de Vries. 2018. Bitcoin's Growing Energy Problem. *Joule* 2, 5 (5 2018), 801–805. https://doi.org/10.1016/j.joule.2018.04.016
[19] Kalyanmoy Deb. 2014. Multi-objective Optimization. In *Search Methodologies*. Springer US, Boston, MA, 403–449. https://doi.org/10.1007/978-1-4614-6940-7_15
[20] Christian Decker and Roger Wattenhofer. 2013. Information propagation in the Bitcoin network. In *13th IEEE International Conference on Peer-to-Peer Computing, IEEE P2P 2013 - Proceedings*. IEEE Computer Society. https://doi.org/10.1109/P2P.2013.6688704
[21] Varun Deshpande, Hakim Badis, and Laurent George. 2018. BTCmap: Mapping Bitcoin Peer-to-Peer Network Topology. In *2018 IFIP/IEEE International Conference on Performance Evaluation and Modeling in Wired and Wireless Networks (PEMWN)*. IEEE, 1–6. https://doi.org/10.23919/PEMWN.2018.8548904
[22] Jean Dollimore, Tim Kindberg, and George Coulouris. 2005. *Distributed Systems: Concepts and Design*. Addison-Wesley. 944 pages.
[23] John R. Douceur. 2002. The Sybil Attack. In *Peer-to-Peer Systems*. Springer, Berlin, Heidelberg, 251–260. https://doi.org/10.1007/3-540-45748-8_24
[24] Thomas Durieux, João F. Ferreira, Rui Abreu, and Pedro Cruz. 2020. Empirical review of automated analysis tools on 47,587 Ethereum smart contracts. In *Proceedings of the ACM/IEEE 42nd International Conference on Software Engineering*. ACM, New York, NY, USA, 530–541. https://doi.org/10.1145/3377811.3380364
[25] Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. 1988. Consensus in the presence of partial synchrony. *J. ACM* 35, 2 (4 1988), 288–323. https://doi.org/10.1145/42282.42283
[26] Ethereum. 2020. Ethereum 2.0 Specifications. https://github.com/ethereum/eth2.0-specs
[27] Giulia Fanti, Shaileshh Bojja Venkatakrishnan, Surya Bakshi, Bradley Denby, Shruti Bhargava, Andrew Miller, and Pramod Viswanath. 2018. Dandelion++. *Proceedings of the ACM on Measurement and Analysis of Computing Systems* 2, 2 (6 2018), 1–35. https://doi.org/10.1145/3224424
[28] Marina Gomtsyan, Nikita Mokrov, Maxim Panov, and Yury Yanovich. 2019. Geometry-Aware Maximum Likelihood Estimation of Intrinsic Dimension. *Proceedings of Machine Learning Research* 101 (4 2019), 1126–1141. http://arxiv.org/abs/1904.06151
[29] Isabelle Guyon, Masoud Nikravesh, Steve Gunn, and Lotfi A. Zadeh (Eds.). 2006. *Feature Extraction*. Studies in Fuzziness and Soft Computing, Vol. 207. Springer Berlin Heidelberg, Berlin, Heidelberg. 645 pages. https://doi.org/10.1007/978-3-540-35488-8
[30] Trevor Hastie, Robert Tibshirani, and Jerome Friedman. 2009. *The Elements of Statistical Learning*. Springer New York, New York, NY. https://doi.org/10.1007/978-0-387-84858-7
[31] Ethan Heilman, Alison Kendler, Aviv Zohar, and Sharon Goldberg. 2015. Eclipse attacks on Bitcoin's peer-to-peer network. In *Proceedings of the 24th USENIX Security Symposium*. 129–144.
[32] Klaus Hinkelmann. 2012. *Design and Analysis of Experiments*. Wiley Series in Probability and Statistics, Vol. 3. John Wiley & Sons, Inc., Hoboken, NJ, USA. https://doi.org/10.1002/9781118147634
[33] Aggelos Kiayias, Alexander Russell, Bernardo David, and Roman Oliynykov. 2017. Ouroboros: A Provably Secure Proof-of-Stake Blockchain Protocol. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 10401 LNCS. Springer, Cham, 357–388. https://doi.org/10.1007/978-3-319-63688-7_12

[34] Tung Kieu, Bin Yang, and Christian S. Jensen. 2018. Outlier Detection for Multidimensional Time Series Using Deep Neural Networks. In *2018 19th IEEE International Conference on Mobile Data Management (MDM)*, Vol. 2018-June. IEEE, 125–134. https://doi.org/10.1109/MDM.2018.00029

[35] Lyudmila Kovalchuk, Dmytro Kaidalov, Oleksiy Shevtsov, Andrii Nastenko, Mariia Rodinko, and Roman Oliynykov. 2017. Analysis of splitting attacks on Bitcoin and GHOST consensus protocols. In *Proceedings of the 2017 IEEE 9th International Conference on Intelligent Data Acquisition and Advanced Computing Systems: Technology and Applications, IDAACS 2017*, Vol. 2. Institute of Electrical and Electronics Engineers Inc., 978–982. https://doi.org/10.1109/IDAACS.2017.8095233

[36] Slawomir Koziel and Leifur Leifsson (Eds.). 2013. *Surrogate-Based Modeling and Optimization*. Springer New York, New York, NY. https://doi.org/10.1007/978-1-4614-7551-4

[37] Stanislav Kruglik, Kamilla Nazirkhanova, and Yury Yanovich. 2019. Challenges beyond blockchain: scaling, oracles and privacy preserving. In *2019 XVI International Symposium "Problems of Redundancy in Information and Control Systems" (REDUNDANCY)*. IEEE, 155–158. https://doi.org/10.1109/REDUNDANCY48165.2019.9003331

[38] Leslie Lamport, Robert Shostak, and Marshall Pease. 1982. The Byzantine Generals Problem. *ACM Transactions on Programming Languages and Systems* 4, 3 (7 1982), 382–401. https://doi.org/10.1145/357172.357176

[39] John A. Lee and Michel Verleysen. 2007. *Nonlinear Dimensionality Reduction*. Springer New York, New York, NY. 295 pages. https://doi.org/10.1007/978-0-387-39351-3

[40] Elizaveta Levina and Peter J. Bickel. 2005. Maximum Likelihood Estimation of Intrinsic Dimension. In *Advances in Neural Information Processing Systems*. MIT Press, 777–784. https://www.stat.berkeley.edu/~bickel/mldim.pdf

[41] Eric Lombrozo, Johnson Lau, and Pieter Wuille. 2015. Segregated Witness (Consensus layer). https://github.com/bitcoin/bips/blob/master/bip-0141.mediawiki

[42] Scott M Lundberg and Su-In Lee. 2017. A Unified Approach to Interpreting Model Predictions. In *Advances in Neural Information Processing Systems 30*, I. Guyon, U. V. Luxburg, S. Bengio, H. Wallach, R. Fergus, S. Vishwanathan, and R. Garnett (Eds.). Curran Associates, Inc., 4765–4774. http://papers.nips.cc/paper/7062-a-unified-approach-to-interpreting-model-predictions.pdf

[43] R. Memon, J. Li, and J. Ahmed. 2019. Simulation model for blockchain systems using queuing theory. *Electronics (Switzerland)* 8, 2 (2019). https://doi.org/10.3390/electronics8020234

[44] Tong Meng, Xuyang Jing, Zheng Yan, and Witold Pedrycz. 2020. A survey on machine learning for data fusion. *Information Fusion* 57 (5 2020), 115–129. https://doi.org/10.1016/j.inffus.2019.12.001

[45] Juliane Müller. 2017. SOCEMO: Surrogate Optimization of Computationally Expensive Multiobjective Problems. *INFORMS Journal on Computing* 29, 4 (11 2017), 581–596. https://doi.org/10.1287/ijoc.2017.0749

[46] Ryunosuke Nagayama, Ryohei Banno, and Kazuyuki Shudo. 2020. Identifying Impacts of Protocol and Internet Development on the Bitcoin Network. In *Proceedings - IEEE Symposium on Computers and Communications*, Vol. 2020-July. Institute of Electrical and Electronics Engineers Inc. https://doi.org/10.1109/ISCC50000.2020.9219639

[47] Satoshi Nakamoto. 2008. Bitcoin: A Peer-to-Peer Electronic Cash System. *www.bitcoin.org* (2008), 1–9. https://bitcoin.org/bitcoin.pdf

[48] Srinivas Nidhra and Jagruthi Dondeti. 2012. Black Box and White Box Testing Techniques - A Literature Review. *International Journal of Embedded Systems and Applications* 2, 2 (6 2012), 29–50. https://doi.org/10.5121/ijesa.2012.2204

[49] Non-github-bitcoin. 2010. Fix for block 74638 overflow output transaction. https://github.com/bitcoin/bitcoin/commit/d4c6b90ca3f9b47adb1b2724a0c3514f80635c84#diff-118fcbaaba162ba17933c7893247df3aR1013

[50] Ivan Panin. 2021. Risk of estimators for Sobol' sensitivity indices based on metamodels. *Electronic Journal of Statistics* 15, 1 (1 2021), 235–281. https://doi.org/10.1214/20-EJS1793

[51] Nestor V. Queipo, Raphael T. Haftka, Wei Shyy, Tushar Goel, Rajkumar Vaidyanathan, and P. Kevin Tucker. 2005. Surrogate-based analysis and optimization. *Progress in Aerospace Sciences* 41, 1 (1 2005), 1–28. https://doi.org/10.1016/j.paerosci.2005.02.001

[52] Matteo Romiti, Aljosha Judmayer, Alexei Zamyatin, and Bernhard Haslhofer. 2019. A Deep Dive into Bitcoin Mining Pools: An Empirical Analysis of Mining Shares. (5 2019). https://github.com/http://arxiv.org/abs/1905.05999

[53] Cynthia Rudin and Joanna Radin. 2019. Why Are We Using Black Box Models in AI When We Don't Need To? A Lesson From An Explainable AI Competition. *Harvard Data Science Review* 1, 2 (11 2019), 2019. https://doi.org/10.1162/99608f92.5a8a3a3d

[54] Andrea Saltelli. 2002. Sensitivity Analysis for Importance Assessment. *Risk Analysis* 22, 3 (6 2002), 579–590. https://doi.org/10.1111/0272-4332.00040

[55] Solana Foundation. 2018. Solana Docs. https://docs.solana.com/

[56] Yonatan Sompolinsky and Aviv Zohar. 2015. Secure High-Rate Transaction Processing in Bitcoin. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 8975. Springer Verlag, 507–527. https://doi.org/10.1007/978-3-662-47854-7_32

[57] J. B. Tenenbaum, V. de Silva, and JC. Langford. 2000. A Global Geometric Framework for Nonlinear Dimensionality Reduction. *Science* 290, 5500 (2000), 2319–2323. https://doi.org/10.1126/science.290.5500.2319

[58] Parth Thakkar, Senthil Nathan, and Balaji Vishwanathan. 2018. Performance Benchmarking and Optimizing Hyperledger Fabric Blockchain Platform. (5 2018). https://arxiv.org/pdf/1805.11390.pdf

[59] Natkamon Tovanich, Nicolas Soulié, Nicolas Heulot, and Petra Isenberg. 2021. An empirical analysis of pool hopping behavior in the Bitcoin blockchain. In *2021 IEEE International Conference on Blockchain and Cryptocurrency (ICBC)*. IEEE Communications Society (ComSoc, 1–9. https://hal.archives-ouvertes.fr/hal-03163006v3

[60] Zied Trifa and Maher Khemakhem. 2014. Sybil Nodes as a Mitigation Strategy Against Sybil Attack. *Procedia Computer Science* 32 (1 2014), 1135–1140. https://doi.org/10.1016/j.procs.2014.05.544

[61] Velas. 2021. Velas Whitepaper v1.0. , 29 pages. https://velas.com/pdf/whitepaper.pdf

[62] Yikai Wang, Wenbing Huang, Fuchun Sun, Tingyang Xu, Yu Rong, and Junzhou Huang. 2020. Deep Multimodal Fusion by Channel Exchanging. *Advances in Neural Information Processing Systems* 33 (11 2020). http://arxiv.org/abs/2011.05005

[63] Xiaoqiong Xu, Gang Sun, Long Luo, Huilong Cao, Hongfang Yu, and Athanasios V. Vasilakos. 2021. Latency performance modeling and analysis for hyperledger fabric blockchain network. *Information Processing & Management* 58, 1 (1 2021), 102436. https://doi.org/10.1016/j.ipm.2020.102436

[64] Anatoly Yakovenko. 2018. Solana: A new architecture for a high performance blockchain. , 32 pages.

[65] Yury Yanovich, Ivan Ivashchenko, Alex Ostrovsky, Aleksandr Shevchenko, and Aleksei Sidorov. 2018. Exonum: Byzantine fault tolerant protocol for blockchains. *bitfury.com* (2018), 1–36. https://bitfury.com/content/downloads/wp-consensus-181227.pdf

[66] Ehab Zaghloul, Tongtong Li, Matt W. Mutka, and Jian Ren. 2020. Bitcoin and Blockchain: Security and Privacy. *IEEE Internet of Things Journal* (2020), 10288–10313. https://doi.org/10.1109/JIOT.2020.3004273

[67] Alexander Zien, Nicole Krämer, Sören Sonnenburg, and Gunnar Rätsch. 2009. The Feature Importance Ranking Measure. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*. Vol. 5782 LNAI. Springer, Berlin, Heidelberg, 694–709. https://doi.org/10.1007/978-3-642-04174-7_45