On the Danger of Private Blockchains

(When PoW can be Harmful to Applications with Termination Requirements)

Vincent Gramoli
NICTA/Data61-CSIRO
University of Sydney
vincent.gramoli@sydney.edu.au

Abstract

Consensus is a fundamental problem of distributed computing. While this problem has been known to be unsolvable since 1985, existing protocols were designed these past three decades to solve consensus under various assumptions. Today, with the recent advent of blockchains, new consensus implementations were proposed to make replicas reach an agreement on the order of transactions updating what is often referred to as a distributed ledger. Very little work has however been devoted to explore its theoretical ramifications. As a result, it is often unclear whether the same systems could be adapted to work in different environments.

In this position paper, we explore the use of the Ethereum blockchain protocol in the context of a *private chain* where the set of participants is controlled. We argue that foundations are needed in order to precisely capture the guarantees of the consensus protocols of novel blockchain systems before one can deploy them safely. To this end, we define the termination of consensus to characterize when blockchain transactions commit and describe the existence of the *Blockchain Anomaly* in existing proof-of-work private chains.

1 Introduction

With the advent of cryptocurrency and distributed ledger systems, new solutions to the consensus problem were recently proposed. A *blockchain* is a chain of blocks linked to each other in reverse order, from the latest appended block to the initial, also called *genesis*, block. Blockchain users can invoke *transactions* to transfer virtual assets from one of their accounts to another account; new transactions are grouped into blocks that are appended regularly to the chain. A protocol solving the consensus problem is necessary to guarantee that blocks are totally ordered, hence preventing a block from being appended if it contains transactions that conflict with the transactions of the previous blocks.

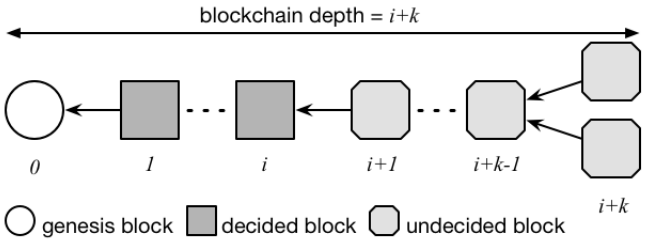

Figure 1: The blockchain structure starts with a genesis block at index 0 and links successive blocks in reverse order of their index; a new block is decided at index $i > 0$ when the blockchain depth reaches $i + k$ (note that a blockchain of depth 0 is the genesis block)

Informally, the consensus problem in a distributed system of n *nodes*, among which f are *faulty*, is the problem of having *correct* or non-faulty nodes agree on one value. The consensus problem is usually defined along three properties: (i) agreement: all nodes that decide choose the same value; (ii) validity: the common output value is an input value of some node; (iii) termination: all correct nodes eventually decide. An algorithm has to fulfil these three properties to solve the consensus problem. Limiting the number f of failures is key to solving consensus and classic consensus protocols [2] are prone to Sybil attacks, where an adversary generates fake faulty nodes to make consensus impossible.

In a blockchain system, transactions get grouped into blocks by special nodes, called *miners*. The miners then propose blocks and validating nodes must agree on a unique block to append it to the chain. Nakamoto's

consensus protocol proposed for the Bitcoin blockchain system [10] copes with Sybil attacks by requiring miners to provide the solution of a crypto-puzzle, called *proof-of-work* (PoW) [5]. It is possible in this protocol that the blockchain forks transiently, in which case multiple blocks are appended to the same block as depicted with the two blocks at index $i + k$ in Figure 1. In public blockchains, reorganizations guarantee however with high probability that the block at index i is uniquely *decided* when the chain depth reaches $i + k$. Applications consider this event as the termination of consensus for the block at index i, indicating that the transactions of this block are *committed* and, for example, that goods bought by these transactions can be shipped or that settlement is final.

Companies have recently started exploring blockchain systems, like Ethereum [14], in an environment where the set of participants are well-identified and "controlled". In contrast to the main public chain that can be accessed by anyone pseudonymously and without permissions, these blockchains are called *private chains*, and are particularly appealing to the finance industry. For example, the R3 consortium[1], which includes over 45 banks, has experimented in an Ethereum private chain across 50 nodes the business logic but not the consensus properties of the blockchain. Their initial experiment involved 11 nodes and was successful, but not all these 11 nodes were mining at the same speed during the experiment—it was intentional for some of them simply not to mine. By contrast, our private chain experiment was intended to test the consensus properties of Ethereum. We precisely exploited the difference in mining power across nodes of an Ethereum private chain and we identified the *Blockchain Anomaly*, an execution that makes it impossible to execute dependent transactions of the form "Bob transfers some coins to Carole only if it received coins from Alice" [11].

2 Communication Delays in Private Chains

There are different ways of modelling the communication among nodes, whether the communication is *synchronous* meaning that there exists a known upper bound on the delay for a message to be delivered or *asynchronous* meaning that there is no such bound. In a synchronous communication model, consensus is known to be solvable in the case of $n \geq 3f + 1$ nodes [9] and similar blockchain consensus protocols, like Nakamoto's consensus, were recently shown correct under communication synchrony [8].

Blockchain systems, however, operate over a network, like the Internet, in which the assumption of communication synchrony can be unrealistic. The network is shared by different applications making congestion and message delays unpredictable from the point-of-view of the blockchain application running on top of it. Unfortunately, consensus is known to be unsolvable in asynchronous networks even in the case of a simple crash failure [7]. Although these delays impact experimentally the quality of the Bitcoin protocol [4], under reasonable assumptions, they do not compromise the consistency of Bitcoin [12].

Private chains, despite running on a controlled set of participants, rarely run in a controlled network. Instead, private chains are typically used across different organizations to arbitrate ownerships among different parties, often with opposite interests: provider-consumer or competitors.[2] As an example, the R3 consortium experimented an Ethereum private chain on Microsoft Azure. In such a cloud environment and even within the same data center where multiple applications from different consumers can share network resources, message delays cannot be perfectly controlled. As opposed to a public chain, these message delays combined with the heterogeneous power of miners in such a private chain could easily allow a 51% attack and lead to the blockchain anomaly as we explain below.

3 Malicious Behaviors and Double-Spending Attack

To maximize their gains, participants may act maliciously by for example executing a *double-spending* attack, spending the same coins in two distinct transactions. Malicious behaviors are generally modeled by an arbitrary (or Byzantine) failure model [9]. In the Byzantine failure model, the consensus problem is usually referred to as the *Byzantine agreement* problem, whose practical solutions, like PBFT [2], involve a number of messages that limits their scalability [3, 13].

[1] http://r3cev.com/.

[2] These private chains are sometimes called *consortium chains* to distinguish them from a *fully private chain* that is maintained by a single organization.

In a public environment, an adversary could potentially generate fake identities to execute a Sybil attack. Creating fake identities would however be useful if they help creating more blocks to be proposed for consensus. In PoW based consensus protocols, the difficulty of the crypto-puzzle is adjusted to limit the frequency in expectation at which each node creates new blocks, regardless of the number of identities they have. These protocols can also be used to implement a Byzantine-tolerant replicated state machine [15].

Other attacks against these blockchain protocols exist and some involve delaying messages: selfish mining [6], network partition or split brain situations can impact the agreement or termination property of the consensus. Similarly, the blockchain anomaly requires malicious miners to mine independently from the rest of the network during a transient period of time to reorder committed transactions that could lead to a double-spending attack.

4 Observing the Blockchain Anomaly

We identified the Blockchain anomaly that prevents someone from executing dependent transactions like "Bob transfers some coins to Carole only if it received coins from Alice" in a private chain. This issue is named the Blockchain anomaly after the Paxos anomaly [1] and is described in detail in our companion technical report [11]. In contrast to the Paxos anomaly, the Blockchain anomaly occurs even if Bob waits for the reception of the money from Alice to be successfully committed before transferring to Carole.

The Blockchain anomaly was experienced on an Ethereum private chain with 2 mining pools running geth v1.4.0 in our controlled network. Although two miners mine on the same chain starting from the same genesis block, a long enough delay in the delivery of messages could lead to having the miners seemingly agree separately on different branches containing more than k blocks each, for any $k \in \mathbb{N}$. When messages get finally delivered, the results of the disagreement creates inconsistencies, like the reordering or deletion of transactions from previously decided blocks.

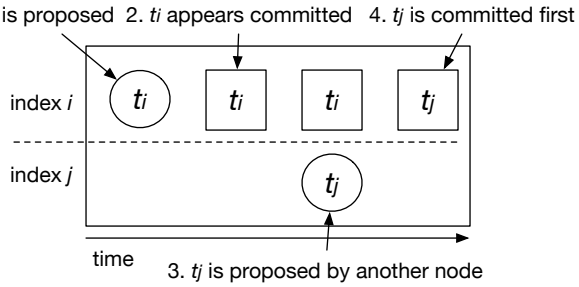

Figure 2: The blockchain anomaly: a first client issues t_i that gets successfully mined and committed then a second client issues t_j, with t_j being conditional to the commit of t_i (note that $j \geq i + k$ for t_i to be committed before t_j gets issued), but the transaction t_j gets finally reorganized and successfully committed before t_i, hence violating the dependency between t_i and t_j

Precisely because the length of the branch could be adjusted to any k, it guarantees that there is no way for an application to choose k' sufficiently large to guarantee that consensus is reached. To take the classic example of exchanges, choosing $k'_{btc} = 5$ for Bitcoin and $k'_{eth} = 11$ for Ethereum cannot be sufficient, as there exist a $k = 12$ for which the Blockchain anomaly can occur. This anomaly is dramatic as it can lead to simple double-spending attacks within a network where users have an incentive to maximize their profits—in terms of coins or arbitrary ownership.

Figure 2 depicts the blockchain anomaly, where a transaction t_i is proposed as part of a block at index i from the standpoint of some nodes. Based on this observation, one waits for t_i to commit before proposing a new transaction t_j. Again, one can imagine a simple scenario where "Bob transfers an amount of money to Carole" (t_j) only if "Bob had successfully received some money from Alice" (t_i) before. However, once these nodes get notified of another branch of committed transactions, they decide to reorganize the branch to resolve the fork. The reorganization removes the committed transaction t_i from slot i. Later, the transaction t_j is successfully committed in slot i. The precise execution leading to the anomaly is given in the companion technical report [11].

Moreover, this scenario is realistic in the context of private chains where participating competitors have direct access to some of the resources. The Blockchain anomaly stems from the fact that in a private chain the reward system does not necessarily incentivize many nodes to mine correctly. Note that in the R3 experiments not all nodes were mining because the purpose of the experiment was to explore business scenarios rather than examine properties of proof-of-work consensus, which does not feature in R3s plans.

5 Conclusion

The termination of consensus is a desirable property of blockchain systems to identify the point where the prefix of a chain becomes immutable. This termination is crucial to detect when a transaction commits for an application to make it safe to ship goods, an exchange platform to convert coins into fiat currency or a bank to observe settlement finality. While tremendous efforts were recently devoted to understanding the Bitcoin public chain, most other blockchain systems are described as white papers, drafts and online documentations that sometime leave room for interpretation. Our observation is that the way mainstream blockchain systems make use of proof-of-work may be ill-suited for private chains if applications require the consensus to terminate. We urge researchers to help defining a theory of blockchains to precisely caracterize the guarantees their consensus algorithms offer and under what assumptions.

Acknowledgments

I wish to thank Richard G. Brown for going through an earlier version of this paper and for confirming that R3 have no plans to deploy proof-of-work-based solutions and that their Corda distributed ledger platform is designed with an explicit goal of providing consensus compatible with the legal notion of settlement finality. NICTA is funded by the Australian Government through the Department of Communications and the ARC through the ICT Centre of Excellence Program.

References

[1] K. Birman, D. Malkhi, and R. Van Renesse. *Guide to Reliable Distributed Systems: Building High-Assurance Applications and Cloud-Hosted Services*, chapter Appendix A: Virtually Synchronous Methodology for Building Dynamic Reliable Services, pages 635–671. Springer London, London, 2012.

[2] M. Castro and B. Liskov. Practical byzantine fault tolerance and proactive recovery. *ACM Trans. Comput. Syst.*, 20(4):398–461, Nov. 2002.

[3] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, D. Song, and R. Wattenhofer. On scaling decentralized blockchains. In *3rd Workshop on Bitcoin Research (BITCOIN), Barbados*, February 2016.

[4] C. Decker and R. Wattenhofer. Information propagation in the bitcoin network. In *Proc. of the IEEE International Conference on Peer-to-Peer Computing*, 2013.

[5] C. Dwork and M. Naor. Pricing via processing or combatting junk mail. In *Proceedings of the 12th Annual International Cryptology Conference on Advances in Cryptology*, CRYPTO '92, pages 139–147, 1993.

[6] I. Eyal and E. G. Sirer. Majority is not enough: Bitcoin mining is vulnerable. In N. Christin and R. Safavi-Naini, editors, *Financial Cryptography and Data Security*, volume 8437 of *Lecture Notes in Computer Science*, pages 436–454. Springer Berlin Heidelberg, 2014.

[7] M. J. Fischer, N. A. Lynch, and M. S. Paterson. Impossibility of distributed consensus with one faulty process. *J. ACM*, 32(2):374–382, Apr. 1985.

[8] J. A. Garay, A. Kiayias, and N. Leonardos. The bitcoin backbone protocol: Analysis and applications. In *Proceedings of the 34th Annual International Conference on the Theory and Applications of Cryptographic Technique (EUROCRYPT)*, pages 281–310, 2015.

[9] L. Lamport, R. Shostak, and M. Pease. The byzantine generals problem. *ACM Trans. Program. Lang. Syst.*, 4(3):382–401, July 1982.

[10] S. Nakamoto. Bitcoin: a peer-to-peer electronic cash system, 2008. http://www.bitcoin.org.

[11] C. Natoli and V. Gramoli. The blockchain anomaly. Technical Report 1605.05438, arXiv, May 2016.

[12] R. Pass, L. Seeman, and A. Shelat. Analysis of the blockchain protocol in asynchronous networks, 2016.

[13] M. Vukolíc. The quest for scalable blockchain fabric: Proof-of-work vs. BFT replication. In *Proceedings of the IFIP WG 11.4 Workshop on Open Research Problems in Network Security (iNetSec 2015)*, LNCS, 2016.

[14] G. Wood. Ethereum: A secure decentralised generalised transaction ledger, 2015. Yellow paper.

[15] X. Xu, C. Pautasso, L. Zhu, V. Gramoli, A. Ponomarev, A. B. Tran, and S. Chen. The blockchain as a software connector. In *Proceedings of the 13th Working IEEE/IFIP Conference on Software Architecture (WICSA'16)*, April 2016.