Blockmania QED.*

Maria A Schett[1][0000−0003−2919−5983]

University College London mail@maria-a-schett.net

Abstract. Blockchains have become ubiquitous. At the core of blockchains lies the ability to reach consensus among distributed and malicious participants. One recently developed and efficient consensus protocol is *Blockmania*. In this abstract we describe how we aim to *(1)* develop a formal specification of Blockmania, *(2)* give a formal proof of correctness, and *(3)* relate and re-use Blockmania with other protocols.

Keywords: blockchain · consensus · verification · correctness.

1 Motivation

Blockchains touch many areas of our life, most notably transferring money such as Bitcoin [17] or Facebook's Libra [2]. Therefore, it is important to get blockchains "right". Together with a sharing and open community—ideas are shared in "white papers" and source code is often open—this makes blockchains ideal for applying formal methods. At the heart of blockchains lies the promise to reach consensus even in the face of malicious, *a.k.a.*, *Byzantine* participants. A multitude of consensus protocols have been developed in the past years, best witnessed by several survey papers on blockchain consensus, *e.g.* [3,7,21].

In our work we focus on one consensus protocol: Blockmania [9]. The core ideas of Blockmania are *(1)* for participants to efficiently build a shared *blockgraph* instead of a blockchain, and then *(2) interpret* the shared blockgraph to derive received messages and consensus (*cf.* Section 2). Our research objectives are to *(1)* develop a formal specification of Blockmania, then *(2)* give a formal proof of correctness, and *(3)* relate and re-use Blockmania with other protocols (*cf.* Section 3).

2 Background: Blockmania

Through the Blockmania protocol distributed servers can reach consensus on the execution order of transactions issued by external clients. To give an example: Eve wants to issue two transactions, *(a)* to pay Alice for a flat in Moscow, and *(b)* to pay Bob for a pizza. The distributed servers have to agree on either executing transaction *(a)* or *(b)* first—especially if Eve issues the transactions to different servers, and worse, tries to double-spend her funds. Reaching consensus

* Partially supported by funding from Google.

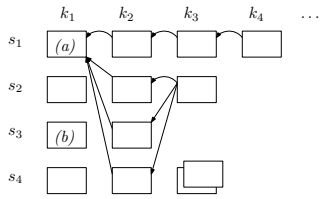

Fig. 1. From blocks to messages.

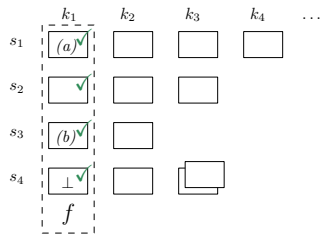

Fig. 2. From blockgraph to consensus.

is made difficult by Byzantine servers and clients not behaving according to the protocol: either intentionally such as Eve trying to double-spend while possibly even collaborating with some servers, or unintentionally such as a server failing abruptly. Moreover, the network itself can be Byzantine. In an asynchronous network even a single failing server may prohibit consensus (FLP theorem [11]).

Blockmania reaches consensus based on two novel ideas. For one, by moving from a blockchain to a blockgraph, the servers efficiently build a shared blockgraph based on light-weight validation rules by employing a gossip protocol. In Fig 1 and Figure 2 we see a schematic view of the same block graph. The servers s_i with $i \in \{1, \ldots, 4\}$ try to reach consensus on which transactions to execute at position k_j with $j \in \mathbb{N}$. Every s_i is responsible for producing a block b for s_i's own slot at position k_j. Unforgeable digital signatures prove the authenticity of a block b at position k_j being created by s_i. Every s_i includes in their blocks transactions received from clients, such as s_1 and s_3 in Figure 1 and 2 including Eve's transactions *(a)* and *(b)* from before. Additionally, s_i includes hashes of blocks received from other servers. In Figure 1, we indicate some backward references to blocks as shown by the arrows. The hash proves the integrity of backward references to previous blocks. This brings us to the second idea of Blockmania. The servers interpret the built blockgraph to *(1)* reach consensus on the transactions to execute by *(2)* deriving messages from the sent blocks. For *(1)* consider Figure 2. Assume the servers can agree on the 4 blocks from every server s_i at position k_1, as indicated by ✓. To reach consensus on the transactions to execute, every correct s_i simply calls a deterministic function f to combine the transactions in these blocks. But how can the servers agree on each of these 4 blocks? They *(2)* interpret consecutive blocks as messages to drive state machines according to a PBFT-inspired [8] consensus protocol, where references to previous blocks are interpreted as messages sent by the block. If the servers cannot agree on some of the blocks, they simply agree to disagree, as indicated by $\perp$. To avoid the FLP theorem, Blockmania assumes partial synchronicity [10].

3 Research Objectives

Objective I*: Formal Specification.* Our objective is to give a formal specification of Blockmania by *(1)* formally defining the underlying data structure

and construction of a *blockgraph*, and *(2)* formally specifying the protocols run by clients and servers for the gossip and interpretation phase. From the prose description [9] and a prototype implementation in Python[1] we extract a specification with well-understood semantics relying on formal language and distributed system primitives [6]. To demonstrate viability, we will implement a prototype in the functional programming language OCaml. The formal specification serves as a basis for the formal proof in OBJECTIVE II. Moreover, the formal specification of interfaces and invariants enables decomposition and re-usability which we will explore in OBJECTIVE III.

OBJECTIVE II*: Formal proof.* Our objective is to formally define and prove correctness of Blockmania[2] by *(1)* defining a notion of correctness in the context of blockchains relying on classical notions of safety and liveness [1], and *(2)* give a formal proof of correctness of the specification of Blockmania in OBJECTIVE I. After showing Blockmania's correctness by pen-and-paper proofs, we aim to use proof assistants. Here we will look into automation *e.g.* through ByMC [13] or Ivy [18], or mechanized proofs similar to the Coq framework Velisarios [19], which reasons about Byzantine-fault tolerant systems to prove safety properties of PBFT [8].

OBJECTIVE III*: Other Protocols.* Our objective is to *(1)* generalize Blockmania's idea to interpret to protocols, and *(2)* relate Blockmania to other graph-based protocols. As indicated [9], it may be possible to use Blockmania in non-Byzantine settings with Paxos [14] or use it with other deterministic Byzantine protocols such as Bracha's broadcast [4] instead of Blockmania's PBFT-inspired protocol. We want to formally define requirements to do so and continue to implement the prototype to demonstrate re-usability. Recently, many graph-based blockchain protocols have emerged, such as DAGCoin [15], Hashgraph [20], Aleph [12], and Casanova [5]. More recently, the Flare consensus protocol [20] combines ideas from Stellar [16] with Blockmania. Our aim is to review the white papers on graph-based consensus protocols and establish commonalities with Blockmania.

4 Progress

We are at the beginning of the project and have a first draft on the formal specification of the blockgraph and pseudo-code of the client and server protocols together with an initial prototype implementation. However, although we presented OBJECTIVE I–III as consecutive, we expect the progress to interleave and iterate. For example, we expect the formal proof to inform the formal specification and definition of correctness, and the interpretation of a different protocol to clarify interfaces in the specification.

[1] *cf.* github.com/gdanezis/blockmania
[2] With the working title: Blockmania$_\square$.

References

1. Alpern, B., Schneider, F.B.: Recognizing safety and liveness. Distr. Comp. **2**(3), 117–126 (1987)
2. Amsden, Z., Arora, R., Bano, S., Baudet, M., Blackshear, S., Bothra, A., Cabrera, G., Catalini, C., Chalkias, K., Cheng, E., Ching, A., Chursin, A., Danezis, G., Giacomo, G.D., Dill, D.L., Ding, H., Doudchenko, N., Gao, V., Gao, Z., Garillot, F., Gorven, M., Hayes, P., Hou, J.M., Hu, Y., Hurley, K., Lewi, K., Li, C., Li, Z., Malkhi, D., Margulis, S., Maurer, B., Mohassel, P., de Naurois, L., Nikolaenko, V., Nowacki, T., Orlov, O., Perelman, D., Pott, A., Proctor, B., Qadeer, S., Russi, D., Schwab, B., Sezer, S., Sonnino, A., Venter, H., Wei, L., Wernerfelt, N., Williams, B., Wu, Q., Yan, X., Zakian, T., Zhou, R.: The Libra Blockchain p. 29 (2019)
3. Bano, S., Sonnino, A., Al-Bassam, M., Azouvi, S., McCorry, P., Meiklejohn, S., Danezis, G.: Consensus in the Age of Blockchains. arXiv:1711.03936 [cs] (2017)
4. Bracha, G., Toueg, S.: Asynchronous Consensus and Broadcast Protocols. J. ACM **32**(4), 824–840 (1985)
5. Butt, K., Sorensen, D., Stay, M.: Casanova. Tech. rep. (Dec 2018)
6. Cachin, C., Guerraoui, R., Rodrigues, L.: Introduction to Reliable and Secure Distributed Programming. Springer-Verlag, Berlin Heidelberg, 2 edn. (2011)
7. Cachin, C., Vukolic, M.: Blockchain Consensus Protocols in the Wild (Keynote Talk) p. 16 pages (2017)
8. Castro, M., Liskov, B.: Practical Byzantine Fault Tolerance. pp. 173–186. OSDI'99, Berkeley, CA, USA (1999)
9. Danezis, G., Hrycyszyn, D.: Blockmania: From Block DAGs to Consensus. Tech. rep. (2018)
10. Dwork, C., Lynch, N., Stockmeyer, L.: Consensus in the Presence of Partial Synchrony. J. ACM **35**, 288–323 (1988)
11. Fischer, M.J., Lynch, N.A., Paterson, M.S.: Impossibility of distributed consensus with one faulty process. J. ACM **32**(2), 374–382 (1985)
12. Gągol, A., Świętek, M.: Aleph: A Leaderless, Asynchronous, Byzantine Fault Tolerant Consensus Protocol. Tech. rep. (2018)
13. Konnov, I., Widder, J.: ByMC: Byzantine Model Checker. pp. 327–342. Lecture Notes in Computer Science, Springer International Publishing (2018)
14. Lamport, L.: The Part-time Parliament. ACM Trans. Comput. Syst. **16** (1998)
15. Lerner, S.D.: Dag Coin Draft. Tech. rep. (2015)
16. Mazières, D.: The Stellar Consensus Protocol: A Federated Model for Internet-level Consensus. Tech. rep. (2015)
17. Nakamoto, S.: Bitcoin: A Peer-to-Peer Electronic Cash System. Tech. rep. (2009)
18. Padon, O., McMillan, K.L., Panda, A., Sagiv, M., Shoham, S.: Ivy: Safety verification by interactive generalization. In: PLDI 2016. pp. 614–630. ACM Press, Santa Barbara, CA, USA (2016)
19. Rahli, V., Vukotic, I., Völp, M., Esteves-Verissimo, P.: Velisarios: Byzantine Fault-Tolerant Protocols Powered by Coq. pp. 619–650. LNCS, Springer International Publishing (2018)
20. Rowan, S., Usher, N.: The Flare Consensus Protocol: Fair, Fast Federated Byzantine Agreement Consensus. Tech. rep. (2019)
21. Sorensen, D.: Establishing Standards for Consensus on Blockchains. Tech. rep. (2019)