A logic of blockchain updates

KAI BRüNNLER, *Bern University of Applied Sciences, Abt. Informatik, 2502 Biel, Switzerland.*
E-mail: kai.bruennler@bfh.ch

DANDOLO FLUMINI, *ZHAW School of Engineering, Forschungsscshwerpunkt Applied Complex Systems Science, 8400 Winterthur, Switzerland.*
E-mail: dandolo.flumini@zhaw.ch

THOMAS STUDER, *University of Bern, Institute of Computer Science, 3012 Bern, Switzerland.*

Abstract

Blockchains are distributed data structures that are used to achieve consensus in systems for cryptocurrencies (like Bitcoin) or smart contracts (like Ethereum). Although blockchains gained a lot of popularity recently, there are only few logic-based models for blockchains available. We introduce BCL, a dynamic logic to reason about blockchain updates, and show that BCL is sound and complete with respect to a simple blockchain model.

Keywords: blockchain, dynamic epistemic logic, modal logic

1 Introduction

Bitcoin [19] is a cryptocurrency that uses peer-to-peer technology to support direct user-to-user transactions without an intermediary such as a bank or credit card company. In order to prevent double spending, which is a common issue in systems without central control, Bitcoin maintains a complete and public record of all transactions at each node in the network. This ledger is called the *blockchain*.

The blockchain is essentially a growing sequence of blocks, which contain approved transactions and a cryptographic hash of the previous block in the sequence. Because the blockchain is stored locally at each node, any update to it has to be propagated to the entire network. Nodes that receive a transaction [1, 4, 21]

1. first verify its validity (i.e. whether it is compatible with all preceeding transactions);
2. if it is valid, then it is added to the blockchain and
3. sent to all other nodes.

Blockchain technology, as a general solution to the Byzantine generals' problem [16], is now not only used for financial transactions but also for many other applications like, e.g. smart contracts [7].

Herlihy and Moir [11] proposed to develop a logic of accountability to design and verify blockchain systems. In particular, they discussed blockchain scenarios to test (i) logics of authorization, (ii) logics of concurrency and (iii) logics of incentives.

Halpern and Pass [10] provided a characterization of agents' knowledge when running a blockchain protocol using a variant of common knowledge. Other characterizations in terms of (probabilistic) epistemic logic are given in [17, 18].

Advance Access Publication on 25 September 2020 doi:10.1093/logcom/exaa045

In the present paper, we are not interested in accountability or aspects of common knowledge. We study the local, single agent perspective of a blockchain. That is, we investigate steps 1 and 2 of the above procedure for receiving a transaction. Our approach is inspired by dynamic epistemic logic [25]. A given state of the local blockchain entails knowledge about the transactions that have taken place. We ask the following: *how does this knowledge change when a new block is received that might be added to the blockchain?* We develop a dynamic logic, BCL, with a semantics that is based on a blockchain model. The update operators of BCL are interpreted as receiving new blocks. It is the aim of this paper to investigate the dynamics of local blockchain updates.

The deductive system for BCL includes reduction axioms that make it possible to establish completeness by a reduction to the update-free case [13]. However, since blockchain updates are performed only if certain consistency conditions are satisfied, we use conditional reduction axioms similar to the ones developed by Steiner to model consistency preserving updates [22]. Moreover, unlike traditional public announcements [25], blockchain updates cannot lead to an inconsistent state, i.e. updates are total, like in [23].

We do not base BCL on an existing blockchain implementation but use a very simple model. First of all, the blockchain is a sequence of propositional formulas. Further, we maintain a list of provisional updates. Our blocks consist of two parts: a sequence number (called the index of the block) and a propositional formula. If a block is received, then the following case distinction is performed where i is the index of the block and l is the current length of the blockchain:

1. $i \leq l$. The block is ignored.
2. $i = l+1$. If the formula of the block is consistent with the blockchain (i.e. it does not contradict blocks that are already accepted in the blockchain), then it is added to the blockchain; otherwise the block is ignored. If the blockchain has been extended, then this procedure is performed also with the blocks stored in the list of provisional updates.
3. $i > l + 1$. The block is added to the list of provisional updates.

Although this is a simple model, it features two important logical properties of blockchains: consistency must be preserved and blocks may be received in the wrong order, in which case they are stored separately until the missing blocks have been received.

The main contribution of our paper from the point of view of dynamic epistemic logic is that we maintain a list of provisional updates. That means we support updates that do not have an immediate effect but that may lead to a belief change later only after certain other updates have been performed. BCL is the first logic that features provisional updates of this kind.

The paper is organized as follows. The next section introduces our blockchain model, the language of BCL and its semantics. In Section 3, we introduce a deductive system for BCL. We establish soundness of BCL in Section 4. In Section 5, we show a normal form theorem for BCL, which is used in Section 6 to prove completeness of BCL. The final section studies some key principles of the dynamics of our blockchain logic and discusses future work.

The present paper is an extended version of a paper presented at LFCS [5]. Note that the conference version did not include any proofs.

2 A simple blockchain logic

The set of all natural numbers is denoted by $\mathbb{N} := \{0, 1, 2, \ldots\}$. The set of positive natural numbers is denoted by $\mathbb{N}^+ := \{1, 2, \ldots\}$. We use ω for the least ordinal such that $\omega > n$, for all $n \in \mathbb{N}$.

Let $\sigma = \langle \sigma_1, \ldots, \sigma_n \rangle$ be any finite sequence. We define its *length* by $\mathsf{len}(\sigma) := n$. For an infinite sequence $\sigma = \langle \sigma_1, \sigma_2, \ldots \rangle$, we set $\mathsf{len}(\sigma) := \omega$. For a (finite or infinite) sequence

$\sigma = \langle \sigma_1, \sigma_2, \ldots, \sigma_i, \ldots \rangle$, we set $(\sigma)_i := \sigma_i$ for $i \leq \mathsf{len}(\sigma)$. The case $i > \mathsf{len}(\sigma)$ can be safely ignored. The *empty sequence* is denoted by $\langle\rangle$ and we set $\mathsf{len}(\langle\rangle) := 0$. We can append x to a finite sequence $\sigma := \langle \sigma_1, \ldots, \sigma_n \rangle$, in symbols we set $\sigma \circ x := \langle \sigma_1, \ldots, \sigma_n, x \rangle$. We will also need the set of all components of a sequence σ and define

$$\mathsf{set}(\sigma) := \{x \mid \text{there is an } i \text{ such that } x = \sigma_i\}.$$

In particular, we have $\mathsf{set}(\langle\rangle) := \emptyset$. Moreover, we use the shorthand $x \in \sigma$ for $x \in \mathsf{set}(\sigma)$.

We start with a countable set of atomic propositions $\mathcal{AP} := \{P0, P1, \ldots\}$. The set of formulas $\mathcal{L}_{\mathsf{cl}}$ of classical propositional logic is given by the following grammar

$$A ::= \bot \mid P \mid A \to A \quad ,$$

where $P \in \mathcal{AP}$.

In order to introduce the language $\mathcal{L}_{\mathsf{B}}$ for blockchain logic, we need another countable set of special atomic propositions $\mathcal{AQ} := \{Q1, Q2, \ldots\}$ that is disjoint with $\mathcal{AP}$. We will use these special propositions later to keep track of the length of the blockchain. The formulas of $\mathcal{L}_{\mathsf{B}}$ are now given by the grammar

$$F ::= \bot \mid P \mid Q \mid F \to F \mid \Box A \mid [i, A]F \quad ,$$

where $P \in \mathcal{AP}$, $Q \in \mathcal{AQ}$, $A \in \mathcal{L}_{\mathsf{cl}}$ and $i \in \mathbb{N}^+$. The operators of the form $[i, A]$ are called *blockchain updates* (or simply *updates*).

Note that in $\mathcal{L}_{\mathsf{B}}$ we cannot express higher-order knowledge, i.e. we can only express knowledge about propositional facts but not knowledge about knowledge of such facts. Since in the present paper, we only deal with the local (i.e. single agent) perspective on blockchains, higher-order knowledge and introspection are not important. Of course, for the multi-agent perspective on blockchains, higher-order notions, in particular common knowledge, are essential epistemic concepts [4, 17, 18].

For all languages in this paper, we define further Boolean connectives (e.g. for negation, conjunction, and disjunction) as usual. Moreover, we assume that unary connectives bind stronger than binary ones.

For $\mathcal{L}_{\mathsf{cl}}$, we use the semantics of classical propositional logic. A *valuation* v is a subset of $\mathcal{AP}$ and we define the truth of an $\mathcal{L}_{\mathsf{cl}}$-formula A under v, in symbols $\mathsf{v} \models A$ as usual. For a set Γ of $\mathcal{L}_{\mathsf{cl}}$-formulas, we write $\mathsf{v} \models \Gamma$ if $\mathsf{v} \models A$ for all $A \in \Gamma$. The set Γ is *satisfiable* if there is a valuation v such that $\mathsf{v} \models \Gamma$. We say Γ *entails* A, in symbols $\Gamma \models A$, if for each valuation v we have

$$\mathsf{v} \models \Gamma \quad \text{implies} \quad \mathsf{v} \models A.$$

Now we introduce the blockchain semantics for $\mathcal{L}_{\mathsf{B}}$.

DEFINITION 1

A *block* is a pair $[i, A]$ where A is a formula of $\mathcal{L}_{\mathsf{cl}}$ and i is an element of $\mathbb{N}^+$. We call i the *index* and A the *formula* of the block $[i, A]$. We define functions ind and fml by $\mathsf{ind}[i, A] := i$ and $\mathsf{fml}[i, A] := A$.

DEFINITION 2

A *model* $\mathsf{M} := (\mathsf{I}, \mathsf{BC}, \mathsf{PU}, \mathsf{v})$ is a quadruple where

1. I is a set of $\mathcal{L}_{\mathsf{cl}}$-formulas
2. BC is a sequence of $\mathcal{L}_{\mathsf{cl}}$-formulas

3. PU is a finite sequence of blocks
4. v is a valuation, i.e. $\mathsf{v} \subseteq \mathcal{AP}$

such that

$$\mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \text{is satisfiable} \tag{1}$$

and

$$\text{for each block } [i, A] \in \mathsf{PU} \text{ we have } i > \mathsf{len}(\mathsf{BC}) + 1. \tag{2}$$

The components of a model $(\mathsf{I}, \mathsf{BC}, \mathsf{PU}, \mathsf{v})$ have the following meaning:

1. I models initial background knowledge.
2. BC is the blockchain.
3. PU stands for *provisional updates*. The sequence PU consists of those blocks that have been announced but that could not yet be added to the blockchain because their index is too high. Maybe they will be added to BC later (i.e. after the missing blocks have been added).
4. v states which atomic propositions are true.

We assume that for each index, eventually a block will be added to the blockchain. If a missing block remained missing forever, then the blockchain would remain fixed from then on.

We need some auxiliary definitions in order to precisely describe the dynamics of the blockchain.

DEFINITION 3

1. Let PU be a finite sequence of blocks. Then we let $\mathsf{find}(i, \mathsf{PU})$ be the least $j \in \mathbb{N}^+$ such that there is an $\mathcal{L}_{\mathsf{cl}}$-formula A with $[i, A] = (\mathsf{PU})_j$.
2. Let $\sigma = \langle \sigma_1, \ldots, \sigma_{i-1}, \sigma_i, \sigma_{i+1}, \ldots \rangle$ be a sequence. We set

$$\mathsf{remove}(i, \sigma) := \langle \sigma_1, \ldots, \sigma_{i-1}, \sigma_{i+1}, \ldots \rangle.$$

3. Given a set of $\mathcal{L}_{\mathsf{cl}}$-formulas I, a sequence of $\mathcal{L}_{\mathsf{cl}}$-formulas BC and a finite sequence of blocks PU, the *chain completion* $\mathsf{complete}(\mathsf{I}, \mathsf{BC}, \mathsf{PU})$ is computed according to Algorithm 1.

Let us comment on the chain completion procedure. The numbers refer to the lines in Algorithm 1.

- 1: n is the index a block must contain so that it could be added to the blockchain BC.
- 2: $[n, A] \in \mathsf{PU}$ *for some formula* A means that PU contains a block that could be added to BC.
- 3–5: find the next formula B that could be added to BC and remove the corresponding block from PU.
- 6: $\mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \cup \{B\}$ *is satisfiable* means that B is consistent with the current belief. This test guarantees that (1) will always be satisfied.
- 7, 8: update the blockchain BC with B.
- 11–15: remove all blocks from PU whose index is less than or equal to the current length of the blockchain BC. Because the blockchain never gets shorter, these block will never be added. Removing them guarantees that (2) will be satisfied after executing the algorithm.

Note if BC and PU satisfy condition (2) in the definition of a model, then the chain completion algorithm will return BC and PU unchanged.

Algorithm 1 Chain completion algorithm: complete
Input:(I, BC, PU)

1. $n \leftarrow \mathsf{len}(\mathsf{BC}) + 1$
2. **while** $[n, A] \in \mathsf{PU}$ for some formula A **do**
3. $\quad i \leftarrow \mathsf{find}(n, \mathsf{PU})$
4. $\quad B \leftarrow \mathsf{fml}((\mathsf{PU})_i)$
5. $\quad \mathsf{remove}(i, \mathsf{PU})$
6. $\quad$ **if** $\mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \cup \{B\}$ is satisfiable **then**
7. $\quad\quad \mathsf{BC} \leftarrow \mathsf{BC} \circ B$
8. $\quad\quad n \leftarrow \mathsf{len}(\mathsf{BC}) + 1$
9. $\quad$ **end if**
10. **end while**
11. **for** $i \in \mathsf{len}(\mathsf{PU}), \ldots, 1$ **do**
12. $\quad$ **if** $\mathsf{ind}((\mathsf{PU})_i) < n$ **then**
13. $\quad\quad \mathsf{remove}(i, \mathsf{PU})$
14. $\quad$ **end if**
15. **end for**
16. **return** $(\mathsf{BC}, \mathsf{PU})$

LEMMA 1
Let I be a set of $\mathcal{L}_{\mathsf{cl}}$-formulas and let BC be a sequence of $\mathcal{L}_{\mathsf{cl}}$-formulas such that $\mathsf{I} \cup \mathsf{set}(\mathsf{BC})$ is satisfiable. Let PU be an arbitrary finite sequence of blocks. For $(\mathsf{BC}', \mathsf{PU}') := \mathsf{complete}(\mathsf{I}, \mathsf{BC}, \mathsf{PU})$, we find that

1. $\mathsf{I} \cup \mathsf{set}(\mathsf{BC}')$ is satisfiable and
2. for each block $[i, A] \in \mathsf{PU}'$ we have $i > \mathsf{len}(\mathsf{BC}') + 1$.

PROOF. By assumption,

$$\mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \text{ is satisfiable} \tag{3}$$

holds for the arguments passed to the algorithm. Moreover, the condition in line 6 guarantees that (3) is a loop invariant of the while loop in lines 2–10, i.e. it holds after each iteration. Since BC is not changed after line 10, (3) also holds for the final result, which shows the first claim of the lemma.

It is easy to see that

$$n = \mathsf{len}(\mathsf{BC}) + 1 \tag{4}$$

also is a loop invariant of while loop in lines 2–10. In particular, (4) holds after line 10 and thus the for loop in lines 11–15 removes all blocks $[i, A]$ from PU with $i < \mathsf{len}(\mathsf{BC}) + 1$. Moreover, after the while loop in lines 2–10 has terminated, its loop condition must be false, which means that PU cannot contain a block $[i, A]$ with $i = \mathsf{len}(\mathsf{BC}) + 1$. This finishes the proof of the second claim. □

DEFINITION 4
Let $\mathsf{M} := (\mathsf{I}, \mathsf{BC}, \mathsf{PU}, \mathsf{v})$ be a model and $[i, A]$ be a block. The *updated model* $\mathsf{M}^{[i,A]}$ is defined as $(\mathsf{I}, \mathsf{BC}', \mathsf{PU}', \mathsf{v})$ where

$$(\mathsf{BC}', \mathsf{PU}') := \mathsf{complete}(\mathsf{I}, \mathsf{BC}, \mathsf{PU} \circ [i, A]).$$

REMARK 1
Note that $\mathsf{M}^{[i,A]}$ is well defined: by Lemma 1, we know that $\mathsf{M}^{[i,A]}$ is indeed a model.

DEFINITION 5
Let $\mathsf{M} := (\mathsf{I}, \mathsf{BC}, \mathsf{PU}, \mathsf{v})$ be a model. We define the *truth* of an $\mathcal{L}_{\mathsf{B}}$-formula F in M, in symbols $\mathsf{M} \models F$, inductively by

1. $\mathsf{M} \not\models \bot$;
2. $\mathsf{M} \models P$ if $P \in \mathsf{v}$ for $P \in \mathcal{AP}$;
3. $\mathsf{M} \models Qi$ if $i \leq \mathsf{len}(\mathsf{BC})$ for $Qi \in \mathcal{AQ}$;
4. $\mathsf{M} \models F \to G$ if $\mathsf{M} \not\models F$ or $\mathsf{M} \models G$;
5. $\mathsf{M} \models \Box A$ if $\mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \models A$;
6. $\mathsf{M} \models [i, A]F$ if $\mathsf{M}^{[i,A]} \models F$.

A formula $\Box A$ means that A follows from the blockchain, i.e. A is a logical consequence from the propositions stored in the blockchain. We can consider $\Box$ to be an epistemic operator since the blockchain represents our *knowledge* about which transactions have happened.

We define validity only with respect to the class of models that do not have provisional updates.

DEFINITION 6
We call a model $\mathsf{M} = (\mathsf{I}, \mathsf{BC}, \mathsf{PU}, \mathsf{v})$ *initial* if $\mathsf{PU} = \langle\rangle$. A formula F is called *valid* if $\mathsf{M} \models F$ for all initial models M.

3 The deductive system BCL

In order to present an axiomatic system for our blockchain logic, we need to formalize an *acceptance condition* stating whether a received block can be added to the blockchain. That is, we need a formula $\mathsf{Acc}(i, A)$ expressing that the formula A is consistent with the current beliefs and the current length of the blockchain is $i - 1$. Thus, if $\mathsf{Acc}(i, A)$ holds, then the block $[i, A]$ will be accepted and added to the blockchain. The truth definition for the atomic propositions $Qi \in \mathcal{AQ}$ says that Qi is true if the blockchain contains at least i elements. That means the formula $Q(i-1) \wedge \neg Qi$ is true if the blockchain contains exactly $i - 1$ elements. This leads to the following definition of $\mathsf{Acc}(i, A)$ for $i \in \mathbb{N}^+$:

$$\mathsf{Acc}(i, A) := \begin{cases} \neg Qi \wedge \neg\Box\neg A & \text{if } i = 1 \\ Q(i-1) \wedge \neg Qi \wedge \neg\Box\neg A & \text{if } i > 1. \end{cases}$$

As desired, we find that if $\mathsf{Acc}(i, A)$ is true, then the chain completion algorithm can append the formula A to the blockchain (see Lemma 2 later).

(PT)	Every instance of a propositional tautology
(K)	$\Box(F \to G) \to (\Box F \to \Box G)$
(D)	$\neg\Box\bot$
(Q)	$Qi \to Qj$ if $i > j$
(A1)	$[i, A]\bot \to \bot$
(A2)	$[i, A]P \leftrightarrow P$ for $P \in \mathcal{AP}$
(A3.1)	$\mathsf{Acc}(i, A) \to ([i, A]Qi \leftrightarrow \top)$ for $Qi \in \mathcal{AQ}$
(A3.2)	$\neg\mathsf{Acc}(i, A) \to ([i, A]Qi \leftrightarrow Qi)$ for $Qi \in \mathcal{AQ}$
(A3.3)	$[i, A]Qj \leftrightarrow Qj$ for $Qj \in \mathcal{AQ}$ and $i \neq j$
(A4)	$[i_1, A_1] \dots [i_k, A_k](F \to G) \leftrightarrow ([i_1, A_1] \dots [i_k, A_k]F \to [i_1, A_1] \dots [i_k, A_k]G)$
(A5.1)	$\mathsf{Acc}(i, A) \to ([i, A]\Box B \leftrightarrow \Box(A \to B))$
(A5.2)	$\neg\mathsf{Acc}(i, A) \to ([i, A]\Box B \leftrightarrow \Box B)$
(A6)	$[h_1, C_1] \dots [h_k, C_k][i, A][j, B]F \leftrightarrow [h_1, C_1] \dots [h_k, C_k][j, A][i, B]F$ for $i \neq j$

An $\mathcal{L}_{\mathsf{B}}$-formula is called compliant if the blockchain updates occur in the correct order. Formally, we use the following definition.

DEFINITION 7
An $\mathcal{L}_{\mathsf{B}}$-formula F is called *compliant* if no occurrence of a $[i, A]$-operator in F is in the scope of some $[j, B]$-operator with $j > i$.

Now we can define a deductive system for BCL. It is formulated in the language $\mathcal{L}_{\mathsf{B}}$ and consists of the following axioms:

We need a little arithmetic: axiom (Q) is used to compare indexes. However, we do not need anything else.

Note that in (A6), we may choose k to be 0, in which case the axiom has the form $[i, A][j, B]F \leftrightarrow [j, A][i, B]F$ for $i \neq j$.

In order to formulate the rules of BCL, we need the following notation. Let $H(P)$ be a formula that may contain occurrences of the atomic proposition P. By $H(F)$, we denote the result of simultaneously replacing each occurrence of P in $H(P)$ with the formula F. The rules of BCL are

$$(\mathsf{MP})\ \frac{F \qquad F \to G}{G} \qquad (\mathsf{NEC})\ \frac{A}{\Box A} \qquad (\mathsf{SUB})\ \frac{F \leftrightarrow G}{H(F) \leftrightarrow H(G)}$$

where (SUB) can only be applied if $H(F) \leftrightarrow H(G)$ is a compliant formula.

REMARK 2
Our semantics includes the case of infinite blockchains: in a given model $(\mathsf{I}, \mathsf{BC}, \mathsf{PU}, \mathsf{v})$, the sequence BC may have infinite length. If we want to exclude such models, then we have to add an infinitary

rule

$$\frac{Qi \quad \text{for all } i \in \mathbb{N}^+}{\bot}$$

to BCL. This rule states that some Qi must be false, which means that BC has finite length.

4 Soundness

Before we can establish soundness of BCL, we have to show some preparatory lemmas.

LEMMA 2
Let $\mathsf{M} := (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v})$ be an initial model. Further, let

$$(\mathsf{I}, \mathsf{BC}', \mathsf{PU}', \mathsf{v}) := \mathsf{M}^{[i,A]}$$

for some block $[i, A]$.

1. If $\mathsf{M} \models \mathsf{Acc}(i, A)$, then $\mathsf{BC}' = \mathsf{BC} \circ A$. In particular, this yields $\mathsf{len}(\mathsf{BC}') = i$ and for each j with $j \neq i$,
$$M \models Qj \quad \text{if and only if} \quad \mathsf{M}^{[i,A]} \models Qj.$$
2. If $\mathsf{M} \not\models \mathsf{Acc}(i, A)$, then $\mathsf{BC}' = \mathsf{BC}$.

PROOF. Assume $\mathsf{M} \models \mathsf{Acc}(i, A)$. That means

$$\mathsf{len}(\mathsf{BC}) + 1 = i \quad \text{and} \quad \mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \cup \{A\} \text{ is satisfiable.}$$

Hence, we find

$$\mathsf{complete}(\mathsf{I}, \mathsf{BC}, \langle\rangle \circ [i, A]) = (\mathsf{BC} \circ A, \langle\rangle).$$

Therefore, $\mathsf{BC}' = \mathsf{BC} \circ A$. This immediately yields

$$\mathsf{len}(\mathsf{BC}') = i = \mathsf{len}(\mathsf{BC}) + 1$$

and for each j with $j \neq i$,

$$M \models Qj \quad \text{if and only if} \quad \mathsf{M}^{[i,A]} \models Qj.$$

Assume $\mathsf{M} \not\models \mathsf{Acc}(i, A)$. This implies

$$\mathsf{len}(\mathsf{BC}) + 1 \neq i \text{ or } \mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \cup \{A\} \text{ is not satisfiable.}$$

Hence, for $(\mathsf{BC}', \mathsf{PU}') := \mathsf{complete}(\mathsf{I}, \mathsf{BC}, \langle\rangle \circ [i, A])$, we find $\mathsf{BC}' = \mathsf{BC}$. □

LEMMA 3
Each axiom of BCL is valid.

PROOF. We only show some cases. Let $\mathsf{M} := (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v})$ be an initial model.

1. $\neg\Box\bot$. By the definition of a model, we have that $\mathsf{I} \cup \mathsf{set}(\mathsf{BC})$ is satisfiable. Hence, $\mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \not\models \bot$, which means $\mathsf{M} \not\models \Box\bot$.
2. $Qi \to Qj$ for $i > j$. Assume $\mathsf{M} \models Qi$. That means $i \leq \mathsf{len}(\mathsf{BC})$. Hence, for $j < i$, we have $j \leq \mathsf{len}(\mathsf{BC})$, which gives $\mathsf{M} \models Qj$.
3. $\mathsf{Acc}(i, A) \to ([i, A]Qi \leftrightarrow \top)$. Assume $\mathsf{M} \models \mathsf{Acc}(i, A)$. Using Lemma 2, we get $\mathsf{M}^{[i,A]} \models Qi$. Thus, $\mathsf{M} \models [i, A]Qi \leftrightarrow \top$ as desired.

4. $\neg\mathsf{Acc}(i,A) \rightarrow ([i,A]Qi \leftrightarrow Qi)$. Assume $\mathsf{M} \not\models \mathsf{Acc}(i,A)$. We use again Lemma 2 to obtain $\mathsf{M} \models [i,A]Qi \leftrightarrow Qi$.
5. $[i,A]Qj \leftrightarrow Qj$ for $Qj \in \mathcal{AQ}$ and $i \neq j$. If $\mathsf{M} \not\models \mathsf{Acc}(i,A)$, we obtain $\mathsf{M} \models [i,A]Qj \leftrightarrow Qj$ as in the previous case. If $\mathsf{M} \models \mathsf{Acc}(i,A)$, then again by Lemma 2, $\mathsf{M} \models [i,A]Qj \leftrightarrow Qj$ for $i \neq j$.
6. $\mathsf{Acc}(i,A) \rightarrow ([i,A]\Box B \leftrightarrow \Box(A \rightarrow B))$. Assume $\mathsf{M} \models \mathsf{Acc}(i,A)$ and let

$$(\mathsf{I}, \mathsf{BC}', \mathsf{PU}', \mathsf{v}) := \mathsf{M}^{[i,A]}.$$

   By Lemma 2, we get $\mathsf{BC}' = \mathsf{BC} \circ A$. Thus, $\mathsf{set}(\mathsf{BC}') = \mathsf{set}(\mathsf{BC}) \cup \{A\}$. By classical logic, we find

$$\mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \cup \{A\} \models_{\mathsf{CL}} B \quad \text{if and only if} \quad \mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \models_{\mathsf{CL}} A \rightarrow B,$$

   which yields $\mathsf{M} \models [i,A]\Box B \leftrightarrow \Box(A \rightarrow B)$.
7. $\neg\mathsf{Acc}(i,A) \rightarrow ([i,A]\Box B \leftrightarrow \Box B)$. Assume $\mathsf{M} \not\models \mathsf{Acc}(i,A)$. From Lemma 2, we immediately get $\mathsf{M} \models [i,A]\Box B \leftrightarrow \Box B$. □

LEMMA 4
Let $\mathsf{M} = (\mathsf{I}, \mathsf{BC}, \mathsf{PU}, \mathsf{v})$ be an arbitrary model and let $[i,A]$ be a block such that $i > \mathsf{len}(\mathsf{BC}) + 1$. Then we have $\mathsf{M}^{[i,A]} = (\mathsf{I}, \mathsf{BC}, \mathsf{PU} \circ [i,A], \mathsf{v})$.

PROOF. Let

$$(\mathsf{BC}', \mathsf{PU}') := \mathsf{complete}(\mathsf{I}, \mathsf{BC}, \mathsf{PU} \circ [i,A]).$$

Since M is a model, condition (2) is satisfied. Therefore, we find that

$$\mathsf{BC}' = \mathsf{BC} \text{ and } \mathsf{PU}' = \mathsf{PU} \circ [i,A],$$

which is $\mathsf{M}^{[i,A]} = (\mathsf{I}, \mathsf{BC}, \mathsf{PU} \circ [i,A], \mathsf{v})$. □

LEMMA 5
Let $\mathsf{M} = (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v})$ be an initial model and let $[i,A]$ be a block such that $i \leq \mathsf{len}(\mathsf{BC}) + 1$. Then $\mathsf{M}^{[i,A]}$ is an initial model, too.

PROOF. Let $\mathsf{PU} = \langle [i,A] \rangle$ and

$$(\mathsf{BC}', \mathsf{PU}') := \mathsf{complete}(\mathsf{I}, \mathsf{BC}, \mathsf{PU}).$$

If $i = \mathsf{len}(\mathsf{BC}) + 1$, then $[i,A]$ is removed from PU in line 5 of Algorithm 1. If $i < \mathsf{len}(\mathsf{BC}) + 1$, then $[i,A]$ is removed from PU in line 13. In both cases, we find $\mathsf{PU}' = \langle\rangle$, which means that $\mathsf{M}^{[i,A]}$ is initial. □

LEMMA 6
Let $(\mathsf{I}, \mathsf{BC}, \mathsf{PU}, \mathsf{v})$ be a model and F be an $\mathcal{L}_{\mathsf{B}}$-formula such that for each $[i,A]$ occurring in F we have $i > \mathsf{len}(\mathsf{BC}) + 1$. Then

$$(\mathsf{I}, \mathsf{BC}, \mathsf{PU}, \mathsf{v}) \models F \quad \text{if and only if} \quad (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v}) \models F.$$

PROOF. By induction on the structure of F and a case distinction on the outermost connective. The only interesting case is $F = [i,A]G$. Since we have $i > \mathsf{len}(\mathsf{BC}) + 1$ by assumption, we find by Lemma 4 that

$$(\mathsf{I}, \mathsf{BC}, \mathsf{PU}, \mathsf{v})^{[i,A]} = (\mathsf{I}, \mathsf{BC}, \mathsf{PU} \circ [i,A], \mathsf{v}).$$

Thus we get

$$(\mathsf{I}, \mathsf{BC}, \mathsf{PU}, \mathsf{v}) \models [i, A]G \quad \text{if and only if} \quad (\mathsf{I}, \mathsf{BC}, \mathsf{PU} \circ [i, A], \mathsf{v}) \models G. \tag{5}$$

Using I.H. twice yields

$$(\mathsf{I}, \mathsf{BC}, \mathsf{PU} \circ [i, A], \mathsf{v}) \models G \quad \text{if and only if} \quad (\mathsf{I}, \mathsf{BC}, \langle [i, A] \rangle, \mathsf{v}) \models G. \tag{6}$$

Again since $i > \mathsf{len}(\mathsf{BC}) + 1$ we find that

$$(\mathsf{I}, \mathsf{BC}, \langle [i, A] \rangle, \mathsf{v}) = (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v})^{[i,A]}$$

and thus

$$(\mathsf{I}, \mathsf{BC}, \langle [i, A] \rangle, \mathsf{v}) \models G \quad \text{if and only if} \quad (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v}) \models [i, A]G. \tag{7}$$

Taking (5), (6) and (7) together yields the desired result. □

Now we can show that the rule (SUB) preserves validity.

LEMMA 7
Let $H(P), F, G$ be $\mathcal{L}_{\mathsf{B}}$-formulas such that $H(F) \leftrightarrow H(G)$ is compliant. We have that

$$\text{if } F \leftrightarrow G \text{ is valid, then } H(F) \leftrightarrow H(G) \text{is valid, too.}$$

PROOF. We show the validity of $H(F) \leftrightarrow H(G)$ by induction on the structure of $H(P)$. We distinguish the following cases.

1. H does not contain P. Thus, $H = H(F) = H(G)$ and $H(F) \leftrightarrow H(G)$ is trivially valid.
2. $H = P$. We have $H(F) = F$ and $H(G) = G$. Thus, $H(F) \leftrightarrow H(G)$ is valid by assumption.
3. $H = H' \to H''$. Follows immediately by I.H.
4. $H = \Box H'$ By I.H., we find that $H'(F) \leftrightarrow H'(G)$ is valid. Since $\mathcal{L}_{\mathsf{B}}$ does not include nested $\Box$-operators, $H'(P)$ is an $\mathcal{L}_{\mathsf{cl}}$-formula. Since $H(F) \leftrightarrow H(G)$ is a formula, F and G must be $\mathcal{L}_{\mathsf{cl}}$-formulas, too. Hence, $H'(F) \leftrightarrow H'(G)$ is an $\mathcal{L}_{\mathsf{cl}}$-formula and we obtain $\models_{\mathsf{CL}} H'(F) \leftrightarrow H'(G)$. Hence, we have $\mathsf{M} \models \Box H'(F)$ if and only if $\mathsf{M} \models \Box H'(G)$ for any model M, which yields that $H(F) \leftrightarrow H(G)$ is valid.
5. $H = [i, A]H'$. Let $\mathsf{M} := (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v})$ be an initial model. We distinguish the following cases:
   1. $i \leq \mathsf{len}(\mathsf{BC}) + 1$. By Lemma 5, we find that $\mathsf{M}^{[i,A]}$ is an initial model. Thus by the I.H. we infer $\mathsf{M}^{[i,A]} \models H'(F) \leftrightarrow H'(G)$, from which we infer
   $$\mathsf{M} \models [i, A]H'(F) \leftrightarrow [i, A]H'(G)$$
   by the validity of (A4).
   2. $i > \mathsf{len}(\mathsf{BC}) + 1$. By Lemma 4, we find that
   $$\mathsf{M}^{[i,A]} = (\mathsf{I}, \mathsf{BC}, \langle [i, A] \rangle, \mathsf{v}).$$
   Since $H(F)$ is compliant, we obtain for each $[j, B]$ occurring in $H(F)$, that $j > \mathsf{len}(\mathsf{BC})+1$. Hence, we obtain by Lemma 6 that
   $$\mathsf{M}^{[i,A]} \models H'(F) \quad \text{if and only if} \quad (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v}) \models H'(F). \tag{8}$$
   By I.H. we get
   $$(\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v}) \models H'(F) \quad \text{if and only if} \quad (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v}) \models H'(G). \tag{9}$$

Since $H(G)$ is compliant, we find that $H'(G)$ satisfies the condition of Lemma 6. Thus, we can use that lemma again to obtain

$$(\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v}) \models H'(G) \quad \text{if and only if} \quad \mathsf{M}^{[i,A]} \models H'(G). \tag{10}$$

Taking (8), (9) and (10) together yields

$$\mathsf{M} \models [i, A]H'(F) \leftrightarrow [i, A]H'(G). \qquad \square$$

We have established that the axioms of BCL are valid and that (SUB) preserves validity. It is easy to see that the rules (MP) and (NEC) also preseve validity. Soundness of BCL follows immediately.

COROLLARY 1
For each formula F, we have

$$\vdash F \quad \text{implies} \quad F \text{ is valid.}$$

REMARK 3
The reduction axiom (A3.3) does not hold in non-initial models. Indeed, let $\mathsf{M} := (\emptyset, \langle\rangle, \langle[2, \top]\rangle, \emptyset)$. We find that $\mathsf{M}^{[1,P]} = (\emptyset, \langle P, \top\rangle, \langle\rangle, \emptyset)$. Hence, $\mathsf{M}^{[1,P]} \models Q2$, which is $\mathsf{M} \models [1, P]Q2$. However, we also have $\mathsf{M} \not\models Q2$.

REMARK 4
The above remark also implies that a block necessitation rule would not be sound, i.e. the validity of F does not entail the validity of $[i, A]F$. Indeed, the axiom $[1, P]Q2 \leftrightarrow Q2$ is valid, but the formula $[2, \top]([1, P]Q2 \leftrightarrow Q2)$ is not valid as shown in the previous remark.

REMARK 5
The rule (SUB) would not preserve validity if we drop the condition that the conclusion must be compliant. Indeed, let us again consider the valid formula $[1, P]Q2 \leftrightarrow Q2$. Without the compliance condition, the rule (SUB) would derive $[2, P'][1, P]Q2 \leftrightarrow [2, P']Q2$, which is not a valid formula.

5 Normal form

Remember that a formula is compliant if the blockchain updates occur in the correct order. In this section, we establish a normal form theorem for our simple blockchain logic.

DEFINITION 8
A *base formula* is a formula that has one of the following forms (which include the case of no blockchain updates):

1. $[i_1, A_1] \dots [i_m, A_m]\bot$
2. $[i_1, A_1] \dots [i_m, A_m]P$ with $P \in \mathcal{AP} \cup \mathcal{AQ}$
3. $[i_1, A_1] \dots [i_m, A_m]\Box B$

Formulas in *normal form* are given as follows:

1. each compliant base formula is in normal form
2. if F and G are in normal form, then so is $F \to G$.

REMARK 6
As an immediate consequence of this definition, we obtain that for each formula F,

$$\text{if } F \text{ is in normal form, then} F \text{ is compliant.}$$

The following theorem states that for each formula F, there is a provably equivalent formula in normal form. The proof is by induction on the structure of F.

THEOREM 1
For each $\mathcal{L}_{\mathsf{B}}$-formula F, there is an $\mathcal{L}_{\mathsf{B}}$-formula G in normal form such that $\vdash F \leftrightarrow G$.

PROOF. We do an induction on the structure of F and distinguish the following cases:

1. The cases when $F = \bot$, $F \in \mathcal{AP} \cup \mathcal{AQ}$ or $F = \Box B$ are trivial.
2. $F = G \to H$. By I.H., there are G' and H' in normal form such that $\vdash G \leftrightarrow G'$ and $\vdash H \leftrightarrow H'$. Hence, for $F' := G' \to H'$, we find $\vdash F \leftrightarrow F'$ and F' is in normal form.
3. $F = [i_1, A_1] \dots [i_k, A_k] G$ with G not of the form $[i_{k+1}, A_{k+1}] G'$. Subinduction on G. We distinguish
   1. $G = \bot$, $G = P \in \mathcal{AP} \cup \mathcal{AQ}$ or $G = \Box B$. In this case, F is a base formula. Using axiom (A6), we find a compliant base formula F' such that $\vdash F \leftrightarrow F'$.
   2. $G = G' \to G''$. Then by axiom (A4)

      $$\vdash F \leftrightarrow ([i_1, A_1] \dots [i_k, A_k] G' \to [i_1, A_1] \dots [i_k, A_k] G'').$$

      Moreover, by I.H., there are H' and H'' in normal form such that

      $$\vdash H' \leftrightarrow [i_1, A_1] \dots [i_k, A_k] G'$$

      and

      $$\vdash H'' \leftrightarrow [i_1, A_1] \dots [i_k, A_k] G''.$$

      We find that $H := H' \to H''$ is in normal form and $\vdash F \leftrightarrow H$. □

6 Completeness

We first show that BCL is complete for modal formulas. The modal language $\mathcal{L}_{\mathsf{M}}$ consists of all update-free $\mathcal{L}_{\mathsf{B}}$-formulas. Formally, $\mathcal{L}_{\mathsf{M}}$ is given by the following grammar

$$F ::= \bot \mid P \mid Q \mid F \to F \mid \Box A \quad ,$$

where $P \in \mathcal{AP}$, $Q \in \mathcal{AQ}$ and $A \in \mathcal{L}_{\mathsf{cl}}$.

We need the collection $\mathsf{BCL}^{\Box}$ of all BCL axioms that are given in $\mathcal{L}_{\mathsf{M}}$. The usual satisfaction relation for Kripke models is denoted by $\models_{\Box}$.

LEMMA 8
For each $\mathcal{L}_{\mathsf{M}}$-formula F, we have

$$F \text{ is valid} \quad \text{implies} \quad \vdash F.$$

PROOF. We show the contrapositive. Assume $\nvdash F$. Since F is a modal formula, there is a Kripke model K with a world w such that

$$\mathsf{K}, w \not\models_{\Box} F \tag{11}$$

and

$$\mathsf{K}, w \models_{\Box} G \qquad \text{for all } G \in BCL^{\Box}. \tag{12}$$

Based on the Kripke model K, we construct an initial update model $\mathsf{M} = (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{V})$ as follows. Note that because of (12), we have $\mathsf{K}, w \models_{\Box} Qi \to Qj$ if $j < i$. Let k be the least $i \in \mathbb{N}^+$ such that $\mathsf{K}, w \not\models_{\Box} Qi$ if it exists and $k := \omega$ otherwise. We set

1. $\mathsf{I} := \{A \in \mathcal{L}_{\mathsf{cl}} \mid \mathsf{K}, w \models_{\Box} \Box A\}$;
2. $\mathsf{BC} := \begin{cases} \langle \top, \ldots, \top \rangle \text{ such that} \mathsf{len}(\mathsf{BC}) = k - 1 & \text{if } k < \omega \\ \langle \top, \top, \ldots \rangle & \text{if } k = \omega \end{cases}$
3. $\mathsf{V} := \{P \in \mathcal{AP} \mid \mathsf{K}, w \models P\}$.

This definition of BC means that BC is an infinite sequence of $\top$ if $k = \omega$.

For each $\mathcal{L}_{\mathsf{M}}$-formula G, we have

$$\mathsf{K}, w \models_{\Box} G \quad \text{if and only if} \quad \mathsf{M} \models G. \tag{13}$$

We show (13) by induction on the structure of G and distinguish the following cases:

1. $G = P \in \mathcal{AP}$. Immediate by the definition of V.
2. $G = Qi \in \mathcal{AQ}$. If $k = \omega$, we have $\mathsf{K}, w \models_{\Box} Qi$ and, since $\mathsf{len}(\mathsf{BC}) = \omega$, also $\mathsf{M} \models Qi$. If $k < \omega$, we have $\mathsf{K}, w \models_{\Box} Qi$ iff $i \leq k - 1 = \mathsf{len}(\mathsf{BC})$ iff $\mathsf{M} \models Qi$.
3. $G = \bot$. Trivial.
4. $G = G_1 \to G_2$. By induction hypothesis.
5. $G = \Box A$. If $\mathsf{K}, w \models \Box A$, then $\mathsf{M} \models \Box A$ by the definition of I. If $\mathsf{M} \models \Box A$, then $\mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \models A$. By the definition of BC, this is $\mathsf{I} \models A$. Because I is deductively closed, we get $A \in \mathsf{I}$, which yields $\mathsf{K}, w \models \Box A$.

By (11) and (13), we conclude $\mathsf{M} \not\models F$ as desired. □

We establish completeness for compliant formulas using a translation from compliant formulas to provably equivalent update-free formulas. We start with defining a mapping h that eliminates update operators.

DEFINITION 9
The mapping h from $\{[i, A]F \mid F \in \mathcal{L}_{\mathsf{M}}\}$ to $\mathcal{L}_{\mathsf{M}}$ is inductively defined by

$$\begin{aligned}
\mathsf{h}([i, A]\bot) &:= \bot \\
\mathsf{h}([i, A]P) &:= P \quad \text{for } P \in \mathcal{AP} \\
\mathsf{h}([i, A]Qi) &:= \mathsf{Acc}(i, A) \vee Qi \\
\mathsf{h}([i, A]Qj) &:= Qj \quad \text{for } Qj \in \mathcal{AQ} \text{ and } i \neq j \\
\mathsf{h}([i, A](F \to G)) &:= \mathsf{h}([i, A]F) \to \mathsf{h}([i, A]G) \\
\mathsf{h}([i, A]\Box B) &:= (\mathsf{Acc}(i, A) \wedge \Box(A \to B)) \vee (\neg\mathsf{Acc}(i, A) \wedge \Box B)
\end{aligned}$$

The mapping h corresponds to the reduction axioms of BCL. Thus, it is easy to show the following lemma by induction on the structure of F.

LEMMA 9
Let F be an $\mathcal{L}_{\mathsf{B}}$-formula of the form $[i,A]G$ such that $G \in \mathcal{L}_{\mathsf{M}}$. We have that $\vdash F \leftrightarrow \mathsf{h}(F)$.

We define a translation t from $\mathcal{L}_{\mathsf{B}}$ to $\mathcal{L}_{\mathsf{M}}$

DEFINITION 10
The mapping $\mathsf{t} : \mathcal{L}_{\mathsf{B}} \to \mathcal{L}_{\mathsf{M}}$ is inductively defined by

$$\begin{aligned} \mathsf{t}(\bot) &:= \bot \\ \mathsf{t}(P) &:= P \quad \text{for } P \in \mathcal{AP} \cup \mathcal{AQ} \\ \mathsf{t}(F \to G) &:= \mathsf{t}(F) \to \mathsf{t}(G) \\ \mathsf{t}(\Box A) &:= \Box A \\ \mathsf{t}([i,A]F) &:= \mathsf{h}([i,A]\mathsf{t}(F)) \end{aligned}$$

LEMMA 10
For each compliant formula F, we have

$$\vdash F \leftrightarrow \mathsf{t}(F).$$

PROOF. The proof is by induction on the structure of F. There are two interesting cases.

1. $F = G \to H$. By I.H., we find $\vdash G \leftrightarrow \mathsf{t}(G)$ and $\vdash H \leftrightarrow \mathsf{t}(H)$. Thus, we have
$$\vdash (G \to H) \leftrightarrow (\mathsf{t}(G) \to \mathsf{t}(H)),$$
which yields the desired result by $\mathsf{t}(G) \to \mathsf{t}(H) = \mathsf{t}(G \to H)$.
2. $F = [i,A]G$. By I.H., we find $\vdash G \leftrightarrow \mathsf{t}(G)$. Since $[i,A]G$ is compliant by assumption, we can use (SUB) to infer $[i,A]G \leftrightarrow [i,A]\mathsf{t}(G)$. By Lemma 9, we know
$$\vdash [i,A]\mathsf{t}(G) \leftrightarrow \mathsf{h}([i,A]\mathsf{t}(G)).$$
We finally conclude $\vdash [i,A]G \leftrightarrow \mathsf{h}([i,A]\mathsf{t}(G))$, which yields the claim since by definition
$$\mathsf{t}([i,A]F) = \mathsf{h}([i,A]\mathsf{t}(F)).$$

□

THEOREM 2
For each compliant $\mathcal{L}_{\mathsf{B}}$-formula F, we have

$$F \text{ is valid} \quad \text{implies} \quad \vdash F.$$

PROOF. Assume that F is a valid and compliant $\mathcal{L}_{\mathsf{B}}$-formula. By Lemma 10, we know $\vdash F \leftrightarrow \mathsf{t}(F)$. Hence, by soundness of BCL, we get that $\mathsf{t}(F)$ is valid, too. Since $\mathsf{t}(F)$ is an $\mathcal{L}_{\mathsf{M}}$-formula, Lemma 8 yields $\vdash \mathsf{t}(F)$. Using Lemma 10 again, we conclude $\vdash F$. □

Combining Theorem 1 and Theorem 2 immediately yields completeness for the full language.

THEOREM 3
For each $\mathcal{L}_{\mathsf{B}}$-formula F, we have

$$F \text{ is valid} \quad \text{implies} \quad \vdash F.$$

PROOF. Assume that F is a valid $\mathcal{L}_{\mathsf{B}}$-formula. By Theorem 1, we find a compliant $\mathcal{L}_{\mathsf{B}}$-formula G such that

$$\vdash F \leftrightarrow G. \tag{14}$$

Hence, by soundness of BCL, we know that G is valid, too. Applying Theorem 2 yields $\vdash G$. We finally conclude $\vdash F$ by (14). □

7 Conclusion

We have presented BCL, a dynamic logic to reason about updates in a simple blockchain model. Our semantics does not have the full complexity of the blockchains used in Bitcoin or Ethereum (see, e.g. [4] for a detailed description of blockchain algorithms), yet it exhibits two key properties of blockchains: blockchain extensions must preserve consistency and blocks may be received in the wrong order. Note, however, that although receiving blocks in the wrong order is an important logical possibility, it only happens rarely in practice: in the Bitcoin protocol the average generation time of a new block is 10 minutes; the average time until a node receives a block is only 6.5 seconds [8].

In order to illustrate the dynamics of our simple blockchain logic, we state some valid principles of BCL:

Persistence: $\Box A \rightarrow [i,B]\Box A$. Beliefs are persistent, i.e. receiving a new block cannot lead to a retraction of previous beliefs.

Consistency: $[i,B]\neg\Box\bot$. Receiving a new block cannot result in inconsistent beliefs.

Success: $\mathsf{Acc}(i,A) \rightarrow [i,A]\Box A$. If a block $[i,A]$ is acceptable, then A is believed after receiving $[i,A]$.[1]

Failure: $(Qi \vee \neg Q(i-1)) \rightarrow ([i,B]\Box A \leftrightarrow \Box A)$. If the current length of the blockchain is not $i-1$, then receiving a block $[i,B]$ will not change the current beliefs.

PROOF.

1. Persistence: $\Box A \rightarrow [i,B]\Box A$. Let $\mathsf{M} := (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v})$ be an initial model and assume $\mathsf{M} \models \Box A$. That is $\mathsf{I} \cup \mathsf{set}(\mathsf{BC}) \models A$. Let $(\mathsf{I}, \mathsf{BC}', \mathsf{PU}', \mathsf{v}) := \mathsf{M}^{[i,B]}$. We find that $\mathsf{set}(\mathsf{BC}) \subseteq \mathsf{set}(\mathsf{BC}')$. Therefore, $\mathsf{I} \cup \mathsf{set}(\mathsf{BC}') \models A$; hence, we have $\mathsf{M}^{[i,B]} \models \Box A$ and $\mathsf{M} \models [i,B]\Box A$.
2. Consistency: $[i,B]\neg\Box\bot$. We let $\mathsf{M} := (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v})$ be an initial model. Further, we set $(\mathsf{I}, \mathsf{BC}', \mathsf{PU}', \mathsf{v}) := \mathsf{M}^{[i,B]}$. By Lemma 1, we know that $\mathsf{I} \cup \mathsf{set}(\mathsf{BC}')$ is satisfiable, i.e. $\mathsf{I} \cup \mathsf{set}(\mathsf{BC}') \not\models \bot$. Hence, $\mathsf{M}^{[i,B]} \models \neg\Box\bot$, which is $\mathsf{M} \models [i,B]\neg\Box\bot$.
3. Success: $\mathsf{Acc}(i,A) \rightarrow [i,A]\Box A$. Let $\mathsf{M} := (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v})$ be an initial model and assume $\mathsf{M} \models \mathsf{Acc}(i,A)$. Let $(\mathsf{I}, \mathsf{BC}', \mathsf{PU}', \mathsf{v}) := \mathsf{M}^{[i,A]}$. By Lemma 2, we know $\mathsf{BC}' = \mathsf{BC} \circ A$. Thus, $\mathsf{I} \cup \mathsf{set}(\mathsf{BC}') \models A$ and, therefore $\mathsf{M}^{[i,A]} \models \Box A$, which is $\mathsf{M} \models [i,A]\Box A$.
4. Failure: $(Qi \vee \neg Q(i-1)) \rightarrow ([i,B]\Box A \leftrightarrow \Box A)$. Again, let $\mathsf{M} := (\mathsf{I}, \mathsf{BC}, \langle\rangle, \mathsf{v})$ be an initial model and assume $\mathsf{M} \models Qi \vee \neg Q(i-1)$. We find that $\mathsf{M} \not\models \mathsf{Acc}(i,B)$. Indeed,

$$\mathsf{M} \models Qi \text{ implies } \mathsf{M} \not\models \mathsf{Acc}(i,B)$$

and

$$\mathsf{M} \models \neg Q(i-1) \text{ implies } i > 1 \text{ and } \mathsf{M} \not\models \mathsf{Acc}(i,B).$$

[1] We call this prinicple *success*, but it is not related to the notion of a *successful formula* as studied in dynamic epistemic logic, see, e.g. [24].

Let $(\mathsf{I}, \mathsf{BC}', \mathsf{PU}', \mathsf{v}) := \mathsf{M}^{[i,B]}$. By Lemma 2, we know $\mathsf{BC}' = \mathsf{BC}$. Therefore, $\mathsf{M}^{[i,B]} \models \Box A$ if and only if $\mathsf{M} \models \Box A$, which yields

$$\mathsf{M} \models [i, B]\Box A \leftrightarrow \Box A.$$

□

There are several open issues for future work. Let us only mention two of them. Although blockchains are called *chains*, the data structure that is actually used is more tree-like and there are different options how to choose the valid branch: Bitcoin currently uses the branch that has the greastest proof-of-work effort invested in it [19] (for simplicity we can think of it as the longest branch), but it is well known that the GHOST rule [21] (used, e.g. in Ethereum [26]) provides better security at higher transaction throughput. We plan to extend BCL so that it can handle tree-like structures and the corresponding forks of the chain. In particular, this requires some form of probability logic to model the fact that older transactions are less likely reversed [9, 19, 21].

In a multi-agent setting, each agent (node) has her own instance of the blockchain. Justification logics [2, 3, 15] could provide a formal approach to handle this. Evidence terms could represent blockchain instances and those instances can be seen as justifying the agents' knowledge about the accepted transactions. This approach would require to develop new dynamic justification logics [6, 14, 20]. Moreover, if the underlying blockchain model supports forks of the chain, then we need justification logics with probability operators [12].

Acknowledgements

We would like to thank Eveline Lehmann and Nenad Savic for carefully reading a previous version of this paper. We also thank the anonymous referees of the LFCS version and of the current version of this paper for many valuable comments.

Funding

Supported by the Swiss National Science Foundation grant 200021_165549.

References

[1] A. M. Antonopoulos. *Mastering Bitcoin: Unlocking Digital Crypto-Currencies*. O'Reilly Media, Inc., 2014.
[2] S. N. Artemov. Explicit provability and constructive semantics. *Bulletin of Symbolic Logic*, **7**, 1–36, 2001.
[3] S. N. Artemov and M. Fitting. *Justification Logic: Reasoning with Reasons*. Cambridge University Press, 2019.
[4] K. Brünnler. *Blockchain kurz & gut*. O'Reilly, 2018.
[5] K. Brünnler, D. Flumini and T. Studer. A logic of blockchain updates. In *Logical Foundations of Computer Science*, S. Artemov and A. Nerode, eds, pp. 107–119. Springer, 2018.
[6] S. Bucheli, R. Kuznets and T. Studer. Realizing public announcements by justifications. *Journal of Computer and System Sciences*, **80**, 1046–1066, 2014.
[7] V. Buterin. Ethereum: a next-generation smart contract and decentralized application platform, 2013. Retrieved 2 Feb. 2017.
[8] C. Decker and R. Wattenhofer. Information propagation in the bitcoin network. In *13th IEEE International Conference on Peer-to-Peer Computing*, pp. 1–10, IEEE, 2013.

[9] C. Grunspan and R. Pérez-Marco. Double spend races. *International Journal of Theoretical and Applied Finance*, **21** 2018. https://doi.org/10.1142/S021902491850053X.
[10] J. H. Halpern and P. Rafael. A knowledge-based analysis of the blockchain protocol. In *TARK 2017*, K. Lang ed. Vol. 251 of *EPTCS*, pp. 324–335, EPTCS, 2017.
[11] M. Herlihy and M. Moir. Blockchains and the logic of accountability: keynote address. In *LICS '16*, pp. 27–30, ACM, 2016.
[12] I. Kokkinis, P. Maksimović, Z. Ognjanović and T. Studer. First steps towards probabilistic justification logic. *Logic Journal of IGPL*, **23**, 662–687, 2015.
[13] B. Kooi. Expressivity and completeness for public update logics via reduction axioms. *Journal of Applied Non-Classical Logics*, **17**, 231–253, 2007.
[14] R. Kuznets and T. Studer. Update as evidence: belief expansion. In *Logical Foundations of Computer Science, International Symposium, LFCS 2013, San Diego, CA, USA, January 6–8, 2013, Proceedings*, S. N. Artemov and A. Nerode, eds. Vol. 7734 of *Lecture Notes in Computer Science*, pp. 266–279. Springer, 2013.
[15] R. Kuznets and T. Studer. *Logics of Proofs and Justifications*. College Publications, 2019.
[16] L. Lamport, R. Shostak and M. Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, **4**, 382–401, 1982.
[17] B. Marinković, P. Glavan, Z. Ognjanović, D. Doder and T. Studer. Probabilistic consensus of the blockchain protocol. In *Symbolic and Quantitative Approaches to Reasoning with Uncertainty*, G. Kern-Isberner and Z. Ognjanović, eds, pp. 469–480. Springer, 2019.
[18] B. Marinković, P. Glavan, Z. Ognjanović and T. Studer. A temporal epistemic logic with a non-rigid set of agents for analyzing the blockchain protocol. *Journal of Logic and Computation*, **29**, 803–830, 2019.
[19] S. Nakamoto. *Bitcoin: A Peer-to-Peer Electronic Cash System*, 2009. https://bitcoin.org/bitcoin.pdf
[20] B. Renne. Public communication in justification logic. *Journal of Logic and Computation*, **21**, 1005–1034, 2011. Published online July 2010.
[21] Y. Sompolinsky and A. Zohar. Secure high-rate transaction processing in bitcoin. In *Financial Cryptography and Data Security 2015, Revised Selected Papers*, R. Böhme and T. Okamoto, eds, pp. 507–527. Springer, Berlin, Heidelberg, 2015.
[22] D. Steiner. A system for consistency preserving belief change. In *Proceedings of Rationality and Knowledge, 18th ESSLLI*, S. Artemov and R. Parikh, eds, pp. 133–144. Association for Logic, Language and Information, 2006.
[23] D. Steiner and T. Studer. Total public announcements. In *LFCS 2007*, S. Artemov and A. Nerode, eds, pp. 498–511. Vol. 4514 of *LNCS*. Springer, 2007.
[24] H. van Ditmarsch and B. Kooi. The secret of my success. *Synthese*, **151**, 201–232, 2006.
[25] H. van Ditmarsch, W. van der Hoek and B. Kooi. *Dynamic Epistemic Logic*. Vol. 337 of *Synthese Library*. Springer, 2007.
[26] G. Wood. Ethereum: a secure decentralised generalised transaction ledger, EIP-150 revision, 2017. Retrieved 2 Feb. 2017.

Received 17 January 2020