Optimal Simultaneous Byzantine Agreement, Common Knowledge and Limited Information Exchange

Ron van der Meyden
UNSW

Abstract

In order to develop solutions that perform actions as early as possible, analysis of distributed algorithms using epistemic logic has generally concentrated on “full information protocols”, which may be inefficient with respect to space and computation time. The paper reconsiders the epistemic analysis of the problem of Simultaneous Byzantine Agreement with respect to weaker, but more practical, exchanges of information. The paper first clarifies some issues concerning both the specification of this problem and the knowledge based program characterizing its solution, concerning the distinction between the notions of “nonfaulty” and “not yet failed”, on which there are variances in the literature. It is then shown that, when implemented relative to a given failure model and an information exchange protocol satisfying certain conditions, this knowledge based program yields a protocol that is optimal relative to solutions using the same information exchange. Conditions are also identified under which this implementation is also an optimum, but an example is provided that shows this does not hold in general.

2012 ACM Subject Classification Theory of computation → Modal and temporal logics; Theory of computation → Logic and verification; Theory of computation → Distributed algorithms

Keywords and phrases Logic of Knowledge, Byzantine Agreement, Consensus Protocol, Fault-tolerance

Digital Object Identifier 10.4230/LIPIcs...

Funding *Ron van der Meyden*: The Commonwealth of Australia (represented by the Defence Science and Technology Group) supported this research through a Defence Science Partnerships agreement.

1 Introduction

The logic of knowledge has been shown to be a helpful formalism for the analysis of fault-tolerant distributed algorithms [2, 3, 6, 5]. A particular focus of work in this area has been the problem of Byzantine Agreement [10], which requires a group of agents to coordinate on a decision in the face of faulty behaviour by some the agents. It has been shown that the precise conditions under which a decision can be made by an agent in such a setting can be characterized, independently of details of the fault model, in terms of what the agent knows. That characterization can then be applied to derive protocols that are *optimal* in the sense that agents decide in each possible run, at the earliest possible time. The present paper reconsiders a number of issues in these results, for Simultaneous Byzantine Agreement (SBA), which requires agents to decide simultaneously (in the same round of computation). This version of Byzantine Agreement is relevant for applications such as the fair release of stock market information, or the coordination of a set of actuators controlling physical equipment such as an airplane or motor vehicle.

In order to coordinate, agents need to exchange information. In the context of Byzantine Agreement protocols, this information is about the agents' initial preferences for the joint decision to be made, and about the faults that they have observed while running the protocol. Driven by a focus on protocols that are theoretically optimal, in the sense of deciding as early as possible, the literature has concentrated on “full information protocols” [10, 2, 6], which

Leibniz International Proceedings in Informatics
LIPICS Schloss Dagstuhl – Leibniz-Zentrum für Informatik, Dagstuhl Publishing, Germany

maximize the information exchanged by having agents store all messages that they receive, and transmitting their complete state in each round of the protocol. Agents using a full information protocol know everything that they could know in any other protocol, enabling them to make their decision at a time no later than they would in any other protocol.

However, full information protocols use agent states that grow exponentially with time. While this state can be reduced with further analysis [6], in some cases, the theoretically optimal protocols are relatively inefficient, or even intractable, in space usage or computation time [6, 7]. Full information protocols are therefore not necessarily practical, and more practical protocols need to make compromises.

Limiting the information exchanged by the protocol is one approach to obtaining a more practical protocol. However, one might still ask for a protocol that is optimal, when compared with other protocols that exchange information in the same way. A consideration of this issue was begun in [1], for the Eventual Byzantine Agreement problem in the case of sending omission failures. In the present paper, we consider optimality of limited information exchange protocols for Simultaneous Byzantine Agreement. Our particular focus is to understand the relationship between optimality of SBA protocols relative to a limited information exchange and a knowledge based program for this problem. We are interested in a general result that covers a range of different failure and information exchange models, since this kind of abstraction is one of the advantages obtainable from the knowledge based approach to distributed computing.

In addressing this question, we are lead to first revisit a number of issues. The characterization of SBA protocols using the logic of knowledge has employed a number of distinct notions of common knowledge, and there are also differences in the underlying semantic models used to represent the various failure models that have been studied. It also emerges that there are subtleties with respect to the notion of optimality guaranteed by the knowledge based program once one considers limited information exchange.

With respect to notions of common knowledge, the original analysis of Simultaneous Byzantine Agreement in the crash failures model by Dwork and Moses [2] uses a notion of common knowledge amongst the *nonfailed* (active) agents, whereas a later developed analysis by Moses and Tuttle [6] (followed by [3]), for omissions failure models, and a more general notion of agreement protocol, uses a notion of common belief amongst the *nonfaulty* agents. As generally understood, in the crash failure model, an agent may be nonfailed, but still faulty, because it will fail at a later time. There exists some gaps in reasoning in these sources related to these issues, as well as some errors in some presentations of related results (e.g., in [3]). We clarify the relationship between these notions, both at the level of specifications and the knowledge based program. Specifically, we show that both the SBA specification and the common belief condition used in the knowledge based program for SBA may refer to either the nonfaulty or the nonfailed agents, without change of meaning.

There are also some divergences between the formal modelling of the crash failure model between the original source [2] and later presentations [3]. The former uses a distinguished "crashed" state to represent when an agent has crashed, whereas the latter models crashed agents as simply failing to send messages from some point on (making this model a special case of the sending omissions model). This turns out to have an impact on the notion of common knowledge that can be used in these models. In the interests of generality, we develop a general modelling that covers both of these models of crash failures. We are then able to establish an equivalence between the different notions of common knowledge that have been used in the crash failures case.

Using the resulting unified understanding of the literature, we then turn to the main contribution of the paper, in which we provide a knowledge based characterization of an optimal protocol for SBA with respect to a limited information exchange. We work with the knowledge based program **P** that, when implemented with respect to the full information exchange, yields an SBA protocol that is an *optimum* with respect to all possible SBA protocols (for a fixed failure model). We show that if we implement **P** with respect to a given information exchange protocol, we can also obtain an implementation that is an optimum relative to protocols using that information exchange. This result requires an assumption on the information exchange, namely, that the agents do not exchange information about the specific actions that they have performed. In particular, agents should not inform others about the fact that they have made a decision, or what that decision is.

We also show that if we allow agents to also exchange the information that they have taken a decision, but not *what* that decision is, then the knowledge based program still yields an implementation that is *optimal* amongst protocols using the given information exchange, in the sense that this implementation cannot be improved upon by any SBA protocol using that information exchange. However, we show by example that, under these assumptions, we do not always get an optimum.

Our immediate motivation for developing these results was work that will be reported elsewhere, in which we have been using automated synthesis techniques to derive a concrete protocol from a knowledge based program and a description of the failure model in which it operates. The results of the present paper help us to understand the optimality guarantees satisfied by the implementations obtained using this process.

The structure of the paper is as follows. We begin in Section 2 by recalling the general *interpreted systems* semantics for the logic of knowledge, and introducing the modal operators needed for the work. Section 3 states the specification for the Simultaneous Byzantine Agreement problem. Section 4 describes how an interpreted system is generated from an underlying information exchange protocol, a model of the failures against which the solution needs to defend, and a protocol used by agents to make their decisions. In Section 5, we reconsider the knowledge based characterization of SBA in the crash failures model due to Dwork and Moses [2], and show how this is related to the later characterization of Moses and Tuttle [6] for omissions failures. The upshot of this analysis is that the Moses and Tuttle characterization can be applied in all cases. We then apply this characterization to study optimality of SBA protocols with respect to limited information exchanges in Section 6. Section 7 presents a counter-example showing that the knowledge based characterization does not always yield an optimum solution in limited information exchange contexts. Section 8 concludes with a discussion of related work and open problems. Proofs of results omitted in the body of the paper are provided in the appendix.

2 Knowledge in Interpreted Systems

We use the general semantic model of [3] to model the semantics of the logic of knowledge. We model the *global states* of a distributed system involving n agents from the set $\texttt{Agt} = \{1, \ldots, n\}$ as a set $L_e \times L_1 \times \ldots \times L_n$, where L_e is a set of states of the environment in which the agents operate, and each L_i, for $i \in \texttt{Agt}$, is a set of *local states of agent i*. A *run* of the system is a function $r : \mathbb{N} \to L_e \times L_1 \times \ldots \times L_n$ mapping times, represented as natural numbers, to global states. A *point* is a pair (r, m) consisting of a run r and a time m. An *interpreted system* is a pair $\mathcal{I} = (\mathcal{R}, \pi)$ consisting of a set $\mathcal{R}$ of runs and an *interpretation* $\pi : \mathcal{R} \times \mathbb{N} \to \mathcal{P}(Prop)$ associating a subset of the set $Prop$ of propositions to each point of the system.

The semantics of knowledge is defined using a relation $\sim_i$ on points for each agent i, given by $(r,m) \sim_i (r',m')$ if $r_i(m) = r_i(m')$. For each agent i, the logic of knowledge has a modal operator K_i, such that $K_i\phi$ is a formula for each formula ϕ. Satisfaction of formulas ϕ at points (r,m) of an interpreted system $\mathcal{I} = (\mathcal{R}, \pi)$ is defined by the relation $\models$, such that

1. $\mathcal{I},(r,m) \models p$ if $p \in \pi(r,m)$, for atomic propositions $p \in \mathit{Prop}$, and
2. $\mathcal{I},(r,m) \models K_i\phi$ if $\mathcal{I},(r',m') \models \phi$ for all points $(r',m') \sim_i (r,m)$.

The interpreted systems we consider in this paper will generally be *synchronous* in the sense that if $(r,m) \sim_i (r',m')$ then $m = m'$.

We work with a number of different notions of group knowledge, that operate with respect to an *indexical set* S of agents, which differs from point to point in the system. That is, we assume that there is a function S mapping each point of the system to a set of agents. The semantics of the atomic formula $i \in S$ is given by $\mathcal{I},(r,m) \models i \in S$ if $i \in S(r,m)$.

An agent may not know whether it is in a set S. We can define a notion of belief, relative to the indexical set S, by $B_i^S\phi = K_i(i \in S \Rightarrow \phi)$. We define the notions of "everyone in S believes" and "everyone in S knows", by $\mathit{EB}_S\phi = \bigwedge_{i\in S} B_i^S\phi$ and $\mathit{EK}_S\phi = \bigwedge_{i\in S} K_i\phi$. Common belief, relative to an indexical set S, is defined by $\mathit{CB}_S\phi = \mathit{EB}_S\phi \wedge \mathit{EB}_S^2\phi \wedge \ldots$.[1] Common knowledge, relative to an indexical set S, is is defined by $\mathit{CK}_S\phi = \mathit{EK}_S\phi \wedge \mathit{EK}_S^2\phi \wedge \ldots$.

A more semantic characterization is as follows. Define the relations $\sim_S^*$ and $\approx_S^*$ on points of a system $\mathcal{I}$ to the reflexive, transitive closures of the relations $\sim_S$ and $\approx_S$ on points given by

1. $(r,m) \sim_S (r',m')$ if there exists $i \in S(r,m)$ such that $(r,m) \sim_i (r',m')$
2. $(r,m) \approx_S (r',m')$ if there exists $i \in S(r,m) \cap S(r',m')$ such that $(r,m) \sim_i (r',m')$

Then we have that $\mathcal{I},(r,m) \models \mathit{CK}_S\phi$ iff $\mathcal{I},(r',m') \models \phi$ for all points (r',m') of $\mathcal{I}$ such that $(r,m) \sim_S^* (r',m')$. Similarly, $\mathcal{I},(r,m) \models \mathit{CB}_S\phi$ iff $\mathcal{I},(r',m') \models \phi$ for all points (r',m') of $\mathcal{I}$ such that $(r,m) \approx_S^* (r',m')$.

These notions are (greatest) fixed points, satisfying $\mathit{CB}_S\phi \equiv \mathit{EB}_S\mathit{CB}_S\phi$ and $\mathit{CK}_S\phi \equiv \mathit{EK}_S\mathit{CK}_S\phi$. Provided it is valid that $S \neq \emptyset$, we have that $\mathit{EB}_S\phi \Rightarrow \phi$ and $\mathit{EK}_S\phi \Rightarrow \phi$ and $\mathit{CB}_S\phi \Rightarrow \phi$ and $\mathit{CK}_S\phi \Rightarrow \phi$ are all valid. These are therefore knowledge-like notions. Further, for each of the operators $O \in \{K_i, B_i^S, \mathit{EB}_S, \mathit{EK}_S, \mathit{CB}_S, \mathit{CK}_S\}$ we have $O\phi \Rightarrow O\psi$ valid if $\phi \Rightarrow \psi$ is valid.

▶ **Proposition 1.** *If A and B are indexical sets such that $A \subseteq B$ is valid, then the formulas $B_i^B\phi \Rightarrow B_i^A\phi$, $\mathit{CK}_B\phi \Rightarrow \mathit{CK}_A\phi$ and $\mathit{CB}_B\phi \Rightarrow \mathit{CB}_A\phi$ are valid.*

▶ **Proposition 2.** *The formulas $K_i\phi \Rightarrow B_i^A\phi$, $\mathit{EK}_A\phi \Rightarrow \mathit{EB}_A\phi$ and $\mathit{CK}_A\phi \Rightarrow \mathit{CB}_A\phi$ are valid.*

3 Simultaneous Byzantine Agreement

The specification of Simultaneous Byzantine Agreement concerns a set of agents, operating subject to faults, who are required to reach a common decision on a set of values from some set V. At each moment of time, each agent i chooses an action from the set $A_i = \{\mathtt{noop}\} \cup \{\mathtt{decide}_i(v) \mid v \in V\}$.

We may state the specification SBA(S) of Simultaneous Byzantine Agreement with respect to an indexical set S as follows:

---

[1] Moses and Tuttle [6] define this as $\phi \wedge \mathit{CB}_S\phi$. If we write this as $TCB_S(\phi)$ (for "true common belief) we have $TCB_S(\phi) \Rightarrow \phi$ valid even when $S \neq \emptyset$ is not valid. However, their application of this operator is for the set S of nonfaulty agents, which is always non-empty because they work with the assumption that the number t of faulty agents is at most the number of agents minus two. In all their applications, therefore, $TCB_S(\phi)$ is equivalent to $\mathit{CB}_S(\phi)$.

Unique-Decision: Each agent i performs an action $\texttt{decide}_i(v)$ (for some v) at most once.[2]

Simultaneous-Agreement(S): If $i \in S$ and i performs $\texttt{decide}_i(v)$ then, at the same time, all $j \in S$ also perform $\texttt{decide}_j(v)$.

Validity(S): If $i \in S$ and i performs $\texttt{decide}_i(v)$ then there exists an agent j with $init_j = v$.

There are variances in the literature as to the set S that should be used in this specification. Most work takes S to be the set $\mathcal{N}$ of nonfaulty agents. However, Dwork and Moses [2] (on the crash failure model) appears to refer to the nonfaulty agents, informally, in their introduction, but work with the active (nonfailed) agents $\mathcal{A}$ in their proofs. We consider the alternatives below in order to clarify these points.

A further point where the specification requires formal clarification is the meaning of "agent $i \in S$ performs an action $\texttt{decide}_i(v)$". Does this hold in a situation where an agent attempts to perform the action, but crashes? (In [3], this is formalised as $\texttt{deciding}_i(v)$, defined as $\neg\texttt{decided}_i(v) \land \bigcirc\texttt{decided}_i(v)$, where "$\bigcirc$" is the "next time" operator, and $\mathcal{N}$ appears to be interpreted (p. 207 and p. 213) as the set of active agents. But this combination does not support the claim (on p. 218) that the formula $\texttt{deciding}_i(v) \Rightarrow B_i^N\texttt{deciding}_i(v)$ is valid.)

4 Information Exchange Protocols and Failure Models

To model protocols for SBA under a variety of failure models, and study the effect of a range of assumptions about how agents in these protocols exchange information, we compose protocols into two parts, a *decision protocol* P and an *information exchange* $\mathcal{E}$. The environment in which the agents operate will be modelled as *failure model* $\mathcal{F}$.

An information exchange $\mathcal{E}$ associates to each agent i a tuple $\mathcal{E}_i = \langle L_i, I_i, M_i, \mu_i, \delta_i \rangle$, where

1. L_i is a set of local states for agent i;
2. $I_i \subseteq L_i$ is a set of *initial states*;
3. M_i is a set of *messages* that agent i may send, assumed to contain the value $\perp$ representing that the agent sends no message;
4. $\mu_i : L_i \times A_i \to (\texttt{Agt} \to M_i)$ is a function, such that $\mu_i(s,a)(j)$ represents the message that agent i, with local state s, sends agent j in a round in which it performs action a;
5. $\delta_i : L_i \times A_i \times \Pi_{j \in Agents} M_j \to L_i$, is a function, such that $\delta_i(s, a, (m_1, \dots, m_n))$ represents the local state of agent i immediately after a round in which the agent started in local state s, performed action a, and received messages $(m_1, \dots, m_n)$ from agents $1, \dots, n$ respectively.

A decision protocol P for an information exchange $\mathcal{E}$ consists of a function $P_i : L_i \to A_i$ for each agent i.

We focus here on *synchronous* protocols in which local states in L_i are of the form $\langle init_i, time_i, \dots \rangle$, where $init_i \in V$ represents agent i's initial preference for the decision to be made, and $time_i$ represents the current time. (In the case of the crash failures

[2] In Byzantine contexts, with S equal to the set of nonfaulty agents, it would be appropriate to change this to say that each agent $i \in S$ performs an action $\texttt{decide}_i(v)$ (for some v) at most once, since the condition as stated cannot be guaranteed. However, in benign failure models this stronger condition can be easily satisfied.

model, there is also an additional state $crashed_i$.) The update function δ_i acts so that if $\delta(\langle init_i, time_i, \ldots\rangle, a, m) = \langle init'_i, time'_i, \ldots\rangle$ then $init'_i = init_i$ and $time'_i = time_i + 1$.

In the *full information* information exchange $\mathcal{E}_{FIP}$ for SBA, agents' initial local states consist of their initial preferences, agents send their complete local states to all other agents in each round, and update their states by recording all messages received in their local state. That is, initial states are values $init_i$, for all agents i, j, states $s \in L_i$, and actions a, we have $\mu_i(s,a)(j) = s$, and $\delta(s,a,m) = s \cdot m$ for all message vectors m. (The action a and the time are not recorded explicitly in the local state in this model, but can be deduced.)

A failure model is given by a tuple $\mathcal{F} = \langle L_e, I_e, \delta_e, Adv\rangle$

1. L_e is a set of states of the environment.
2. $I_e \subseteq L_e$ is a nonempty set of initial states of the environment.
3. $\delta_e : L_e \times \Pi_{i\in\mathtt{Agt}} A_i \rightarrow L_e$, such that $\delta_e(s, (a_1, \ldots, a_n))$ represents how the state of the environment is updated in a round in which agents perform actions $a_1, \ldots, a_n$. (Dependence on agent actions allows the environment to record information about the actions performed by the agents. We could also include here a dependence on the messages sent, but we will not need this for the failure models considered in this paper.)
4. Adv is a nonempty set of adversaries, where each adversary is given by a tuple $\langle\Delta^t, \Delta^r, \Delta^s\rangle$, where
   - $\Delta^t : \mathbb{N} \times \mathtt{Agt} \times \mathtt{Agt} \times \bigcup_{i\in\mathtt{Agt}} M_i \rightarrow \bigcup_{i\in\mathtt{Agt}} M_i$ is a function, such that $\Delta^t(k, i, j, m)$ is a message resulting from a fault, if any, through which the environment perturbs the message m transmitted by agent i to agent j in round $k+1$.
   - $\Delta^r : \mathbb{N} \times \mathtt{Agt} \times \mathtt{Agt} \times \bigcup_{i\in\mathtt{Agt}} M_i \rightarrow \bigcup_{i\in\mathtt{Agt}} M_i$ is a function, such that $\Delta^r(k, i, j, m)$ is a message resulting from a fault, if any, through which the environment perturbs the message m received by agent j from agent i in round $k+1$.
   - $\Delta^s : \mathbb{N} \times \mathtt{Agt} \times \Pi_{i\in\mathtt{Agt}} L_i \rightarrow \Pi_{i\in\mathtt{Agt}} L_i$ is a function, representing effects that faults have on the agents' local states, such that $\Delta^s(k, i, s^*_i) = s'_i$ when the effect of the fault, if any, is to cause state s^*_i of agent i to be modified in round $k+1$ to state s'_i, for each agent i. (Here we write s^*_i to indicate the state of the agent *after* it has applied its state update for the round.)

Given a decision protocol P, information exchange $\mathcal{E}$ and failures model $\mathcal{F}$, we define the interpreted system $\mathcal{I}_{P,\mathcal{E},\mathcal{F}} = (\mathcal{R}_{P,\mathcal{E},\mathcal{F}}, \pi)$ with global states $L^*_e \times L_1 \times \ldots \times L_n$, where $L^*_e = L_e \times Adv$, and runs r defined by

1. $r(0) = ((s_e, (\Delta^t, \Delta^r, \Delta^s)), s_1, \ldots, s_n)$, where $s_e \in I_e$ and $s_i \in I_i$ for each $i \in \mathtt{Agt}$, and $(\Delta^t, \Delta^r, \Delta^s) \in Adv$.
2. for all times k, if $r(k) = ((s_e, (\Delta^t, \Delta^r, \Delta^s)), s_1, \ldots, s_n)$, then $r(k+1)$ is the state $((s'_e, (\Delta^t, \Delta^r, \Delta^s)), s'_1, \ldots, s'_n)$ obtained as follows.
   For each agent i, let $a_i = P_i(s_i)$ be the action selected by the decision protocol, and let $m_{i,j} = \mu_i(s_i, a_i)(j)$ be the message that agent i sends to agent j, according to the information exchange $\mathcal{E}_i$.
   Note that the adversary $(\Delta^t, \Delta^r, \Delta^s)$ is the same in $r(k)$ and $r(k+1)$. The remaining state of the environment is updated from s_e to $s'_e = \delta_e(s_e, (a_1, \ldots, a_n))$.
   For each agent i and j, let $m'_{i,j} = \Delta^r(k, i, j, \Delta^t(k, i, j, m_{i,j}))$ be the message resulting from any faults caused by the adversary in the transmission from i to j. Thus, for each agent j, the messages received by agent j are $(m'_{1,j}, \ldots, m'_{n,j})$. The expected effect of these message receptions on the agents' local states is to transition from $(s_1, \ldots, s_n)$ to $(s^*_1, \ldots, s^*_n)$, where $s^*_j = \delta_j(s_j, a_j, (m'_{1,j}, \ldots, m'_{n,j}))$. We define $s'_i = \Delta^s(k, i, s^*_i)$ for each agent i. That is, we apply the perturbation Δ^s to the local states of the agents after they have updated their local states according to the information exchange.

Agents may experience a number of different types of faults. Agent i has a *transmission fault* in round $k+1$ of run r if $\Delta^t(k,i,j,m_{i,j}) \neq m_{i,j}$, where $m_{i,j}$ is the message sent by i to j in round $k+1$. Agent j has a *reception fault* in round $k+1$ of run r if $\Delta^r(k,i,j,m_{i,j}) \neq m_{i,j}$, where $m_{i,j}$ is the message delivered from i to j in round $k+1$. Agent i has a *state* fault if, in round $k+1$, we have $(s'_1,\ldots,s'_n) = \Delta^s(k,(s^*_1,\ldots,s^*_n))$ and $s'_i \neq s_i$. If none of these types of faults apply, then we say that agent i does not have a fault in round $k+1$. We say that an agent i is *faulty* in a run r if it has a fault of any type for some round $k \in \mathbb{N}$. Agent i is *nonfaulty to time* k if it does not have a fault in rounds $1\ldots k$ in run r. We define the indexical set $\mathcal{N}(r,k)$ to be the set of agents that are not faulty in r, and the indexical set $\mathcal{A}(r,k)$ to be the set of agents that are not faulty to time k.

The interpretation π will give meaning to a number of propositions dependent on the specifics of the information exchange and the failure model. In particular, for agents i, values $v \in V$, and indexical sets S,T,

- $\mathtt{decides}_i(v)$ is in $\pi(r,m)$ if $P_i(r,m) = \mathtt{decide}_i(v)$;
- $i \in S$ is in $\pi(r,m)$ if $i \in S(r,m)$;
- $S \subseteq T$ is in $\pi(r,m)$ if $S(r,m) \subseteq T(r,m)$;
- $S = \emptyset$ is in $\pi(r,m)$ if $S(r,m) = \emptyset$;
- $\exists v$ is in $\pi(r,m)$ if there exists an agent i with $init_i = v$ in $r_i(0)$.

We have noted an ambiguity in "agent performs $\mathtt{decides}_i(v)$" in the specification of SBA. In the following, we interpret this as $\mathtt{decides}_i(v)$ as defined above. We remark that our definition of $\mathtt{decides}_i(v)$ holds at a point where an agent is required by its protocol to perform $\mathtt{decide}_i(v)$, but crashes in the next round.

Plainly, $\mathcal{N} \subseteq \mathcal{A}$ is valid; any agent that never fails will not have failed before the current time. Note that $\mathcal{N}$ is independent of the time, and depends only on the run: $\mathcal{N}(r,m) = \mathcal{N}(r,m')$ for all times m,m'. This does not hold for $\mathcal{A}$.

A *context for SBA* is a pair $\gamma = (\mathcal{E},\mathcal{F})$, where $\mathcal{E}$ is an information exchange and $\mathcal{F}$ is a failure model. For brevity we may also write $\mathcal{I}_{P,\gamma}$ for the interpreted system $\mathcal{I}_{P,\mathcal{E},\mathcal{F}}$.

Commonly studied failure models from the literature can be represented in the above form. We say that Δ^s is *correct* for agent i if $\Delta^s(k,i,s_i) = s$ for all $s_i \in L_i$ and $k \in \mathbb{N}$. Similarly Δ^t is correct for agent i if $\Delta^t(k,i,j,m) = m$ for all k,j and m, and Δ^r is correct for agent j if $\Delta^r(k,i,j,m) = m$ for all k,i and m.

- In the *hard crash* failures model of [2], agents may crash at any time. In the round in which an agent crashes, it sends an arbitrary subset of the set of messages it was required to send in the round. To represent this model, we require that agents' local state sets L_i contain a distinguished state *crashed*. We always take Δ^r to be correct for all agents i. An adversary for which agent i crashes in round $k+1$ has $\Delta^s(k,i,s_i) = \mathit{crashed}$ for all $s_i \in L_i$, and there exists a set $J \subseteq \mathtt{Agt}$ such that, for all messages m, $\Delta^t(k,i,j,m) = \bot$ for $j \in J$, and $\Delta^t(k,i,j,m) = m$ for $j \in \mathtt{Agt} \setminus J$. For $k' > k$, we also have $\Delta^s(k',i,s_i) = \mathit{crashed}$, and $\Delta^t(k',i,j,m) = \bot$ for all agents j. For agents that do not crash, Δ^s, Δ^t and Δ^r are correct. We write Crash_t for the failure model in which Adv contains the adversaries in which t or fewer agents may crash.
- In the *communications crash* version of the crash failures model used in [3], again agents may crash at any time, and in the round in which an agent crashes, it sends an arbitrary subset of the set of messages it was required to send in the round. However, we do not require for this model that agents' local state sets L_i contain the distinguished state *crashed*. Instead, failures in this model can be understood as crashes of the agent's transmitter. An adversary for which agent i crashes in round $k+1$ has $\Delta^s(k,i,s_i) = s_i$ for all $s_i \in L_i$, and there exists a set $J \subseteq \mathtt{Agt}$ such that, for all messages m, $\Delta^t(k,i,j,m) = \bot$

for $j \in J$, and $\Delta^t(k,i,j,m) = m$ for $j \in \texttt{Agt} \setminus J$, and for $k' \geq k$, we also have $\Delta^t(k,i,j,m) = \bot$ for all agents j. In all other cases, Δ^s, Δ^t, and Δ^r are correct. We write $ComCrash_t$ for the failure model in which Adv contains the adversaries in which t or fewer agents may crash.
- In the *Sending Omissions* model SO_t, Δ^s and Δ^r are correct for all agents, but Δ^t may allow failures for up to t agents.
- In the *Receiving Omissions* model RO_t, Δ^s and Δ^t are correct for all agents, but Δ^r may allow failures for up to t agents.
- In the *General Omissions* model GO_t, Δ^s is correct for all agents, but Δ^r and Δ^t may allow failures may allow failures for up to t agents.

Other types of failure assumptions can also easily be modelled, such as crashing agents sending messages to a *prefix* of the list of agents $[1 \ldots n]$, atomic transmission failures in which a failing agent transmits to no other agents, message corruption, etc.

5 Crash Failures

We first consider some subtleties relating to the hard crash failures model and the knowledge based program from [2]. This modelling has consequences for the agent's knowledge, and affects the knowledge based program developed in [2]. In the context of this model, $\mathcal{N}$ represents the nonfaulty agents and the set $\mathcal{A}$ of agents that have not failed to the current time is the set of *active* agents, that have not yet crashed.

The specification for SBA for the crash failures model appears to be given by Dwork and Moses as SBA($\mathcal{N}$), i.e., with respect to *nonfaulty* agents. On the other hand, it is stated in [3] as SBA($\mathcal{A}$), i.e., for the *nonfailed* agents. Moses and Tuttle [6] consider omissions failures, and state a specification that is a generalization (to a richer set of coordinated action problems, and allowing the inclusion of a termination requirement) of SBA($\mathcal{N}$). The use of $\mathcal{N}$ appears to be the more common approach in the broader literature on distributed algorithms. We may note the following relationship between these specifications,

▶ **Proposition 3.** *Let γ be any context for SBA, and P any protocol for this context, and let S and T be indexical sets of agents such that $\mathcal{I}_{P,\gamma} \models S \subseteq T$. If $\mathcal{I}_{P,\gamma} \models SBA(T)$ then $\mathcal{I}_{P,\gamma} \models SBA(S)$. In particular if $\mathcal{I}_{P,\gamma} \models SBA(\mathcal{A})$ then $\mathcal{I}_{P,\gamma} \models SBA(\mathcal{N})$.*

Under certain conditions, we also have a converse to this result.

▶ **Proposition 4.** *Suppose that P is a protocol for the context γ, and let S and T be indexical sets of agents in $\mathcal{I}_{P,\gamma}$, such that*
(a) for all points (r,m), if $\mathcal{I}_{P,\gamma},(r,m) \models i \in T \wedge j \in T$, then there exists a run r' such that $(r,m) \sim_i (r',m)$ and $(r,m) \sim_j (r',m)$, and $\mathcal{I}_{P,\gamma},(r,m) \models i \in S \wedge j \in S$, and
(b) $\mathcal{I}_{P,\gamma} \models S \subseteq T$, and
(c) $\mathcal{I}_{P,\gamma} \models T \neq \emptyset \Rightarrow S \neq \emptyset$.
Then $\mathcal{I}_{P,\gamma} \models SBA(S)$ implies $\mathcal{I}_{P,\gamma} \models SBA(T)$.

▶ **Corollary 5.** *For crash failures and omissions failure contexts γ and protocols P, with $\mathcal{I}_{P,\gamma} \models \mathcal{N} \neq \emptyset$, we have $\mathcal{I}_{P,\gamma} \models SBA(\mathcal{N})$ implies $\mathcal{I}_{P,\gamma} \models SBA(\mathcal{A})$.*

Thus, we have SBA($\mathcal{N}$) is equivalent to SBA($\mathcal{A}$) in crash and omission failure models when $\mathcal{N} \neq \emptyset$ is valid. While SBA($\mathcal{N}$) requires only the nonfaulty agents to decide simultaneously, in fact, the stronger statement that all nonfailed agents act simultaneously is implied by this specification.

For a set S, write $\texttt{decides}_S(v)$ for $\bigwedge_{i\in S}\texttt{decides}_i(v)$. Dwork and Moses [2] Theorem 8 states that for any SBA protocol P for the crash failures model, $\texttt{decides}_i(v) \Rightarrow CK_{\mathcal{A}}(\texttt{decides}_{\mathcal{A}}(v))$ and $\texttt{decides}_i(v) \Rightarrow CK_{\mathcal{A}}(\exists v)$ are valid in $\mathcal{I}_{P,\mathcal{E},Crash_t}$. The specification of SBA is stated informally in the introduction of the paper using the term "nonfaulty" but it is not made precise in the paper whether this should be interpreted as referring to the set $\mathcal{N}$ of agents that never fail, or the set $\mathcal{A}$ of active agents, that have not yet failed. The proof of Theorem 8 appears to be using $\mathcal{A}$ as the interpretation. However, the result can also be established using the apparently weaker interpretation $\mathcal{N}$, as shown in the following result. A second subtlety is that the proof depends on the fact that crash failures have been modelled using the hard crash failures model, so that crashed agents are in a special state *crashed*, with the property that $P_i(crashed) = \texttt{noop} \neq \texttt{decide}_i(v)$ for all values v.

▶ **Proposition 6.** *Suppose that P is a protocol for the hard crash failures context $(\mathcal{E}, Crash_t)$ with $t < n$ such that $\mathcal{I}_{P,\mathcal{E},Crash_t} \models SBA(\mathcal{N})$. Then $\texttt{decides}_i(v) \Rightarrow CK_{\mathcal{A}}(\texttt{decides}_{\mathcal{A}}(v))$ and $\texttt{decides}_i(v) \Rightarrow CK_{\mathcal{A}}(\exists v)$ are valid in $\mathcal{I}_{P,\mathcal{E},Crash_t}$.*

On this basis, [2] use the general knowledge-based program $\mathbf{P}(\Phi)$ in which agent i operates as follows

$$\begin{array}{l}\text{do } \texttt{noop} \text{ until } \exists v \in V(\Phi_v);\\ \text{let } v \text{ be the least value in } V \text{ for which } \Phi_v\\ \text{in } \texttt{decide}_i(v)\end{array} \tag{1}$$

where Φ is a collection of formulas indexed by a values $v \in V$ such that Φ_v is the (knowledge-based) condition for each possible choice $v \in V$ given by $K_i CK_{\mathcal{A}}(\exists v)$. A concrete protocol P *implements* $\mathbf{P}(\Phi_v)$ with respect to a context γ if at all points (r, m) of $\mathcal{I}_{P,\gamma}$, and all agents i, $P_i(r_i(m))$ is the same action as would be selected by $\mathbf{P}(\Phi_v)$ at (r, m), with Φ_v interpreted as true iff $\mathcal{I}_{P,\gamma}, (r, m) \models \Phi_v$.

By contrast, [3] show that for an SBA($\mathcal{A}$) protocol, the formula $\texttt{decide}_i(v) \Rightarrow B_i^{\mathcal{A}} CB_{\mathcal{A}}\exists v$ is valid in $\mathcal{I}_{P,\mathcal{E},\mathcal{F}}$. On the basis of this, they use $\Phi_v = B_i^{\mathcal{A}} CB_{\mathcal{A}}\exists v$ in the knowledge based program $P(\Phi)$.[3] In fact, this result holds more generally, as shown in the following result.

▶ **Lemma 7.** *Let S be an indexical set of agents and suppose that P is an SBA(S) protocol for an information exchange protocol $\mathcal{E}$ and failure environment $\mathcal{F}$. Then the formula $\texttt{decide}_i(v) \Rightarrow B_i^S CB_S \exists v$ is valid in $\mathcal{I}_{P,\mathcal{E},\mathcal{F}}$.*

Beyond the use of $CK_{\mathcal{A}}$ instead of $CB_{\mathcal{A}}$ to characterize the conditions for an agent to decide, a further difference in the results of [2] and [3] is the modelling of crash failures. Whereas [2] uses the hard crash model, [3] uses the communication crash model. We now clarify the connection between these characterizations: in hard crash contexts, the two characterizations are equivalent.

▶ **Proposition 8.** *If P is an SBA($\mathcal{N}$) protocol for the hard crash context $(\mathcal{E}, Crash_t)$ with $t < n$ then $CK_{\mathcal{A}}(\texttt{decides}_{\mathcal{A}}(v)) \Leftrightarrow CB_{\mathcal{N}}(\texttt{decides}_{\mathcal{N}}(v))$ and $i \in \mathcal{A} \Rightarrow (K_i CK_{\mathcal{A}}(\texttt{decides}_{\mathcal{A}}(v)) \Leftrightarrow B_i^N CB_{\mathcal{N}}(\texttt{decides}_{\mathcal{N}}(v))$ are valid in $\mathcal{I}_{P,\mathcal{E},Crash_t}$.*

[3] Moses and Tuttle [6] show just that $i \in \mathcal{N} \wedge \texttt{decide}_i(v) \Rightarrow CB_N \exists v$ is valid, and write a program in which the condition "test for $CB_{\mathcal{N}}\exists v$" is used. This work predated the formal definition of knowledge based programs, which requires that the conditions of the program be local to an agent. The treatment of [3] is therefore more satisfactory.

Proposition 8 establishes that, in hard crash contexts, the knowledge based program using $K_i CK_{\mathcal{A}}(\texttt{decides}_{\mathcal{A}}(v))$ is equivalent to the knowledge based program using $B_i^N CB_{\mathcal{N}}(\texttt{decides}_{\mathcal{N}}(v))$, since these formulas are equivalent for active agents, and agents that have crashed take no actions in either case. However, these knowledge based programs may behave differently in a "communications crash" model, where crashed agents continue to take actions, since then $i \in A \Rightarrow K_i(i \in A)$ is no longer valid, and whether a crashed agent satisfies $K_i(i \notin \mathcal{A})$ depends on the information exchange.

Since the characterization of [6] is more general, in the sequel, we work with their belief based decision condition in the knowledge based program $\mathbf{P}(\Phi)$, assume $\mathcal{N} \neq \emptyset$ is valid, and take SBA($\mathcal{N}$) to be meaning of the specification of SBA.

However, we may also note that, similar to the equivalence at the level of the specification, the choice of $\mathcal{N}$ or $\mathcal{A}$ in the condition of the knowledge based program makes no difference to the semantics. Define a *synchronous epistemic bisimulation* on $\mathcal{I}$ with respect to a set of atomic propositions *Prop* to be a relation $\approx$ such that whenever $(r, m) \approx (r', m')$, we have

- $m = m'$,
- for all $p \in Prop$, $\mathcal{I}, (r, m) \models p$ iff $\mathcal{I}, (r', m) \models p$, and
- for all $i \in \texttt{Agt}$, $(r, m) \sim_i (r', m)$.

▶ **Proposition 9.** *Suppose that S and T are indexical sets of agents in an interpreted system $\mathcal{I}$, and let $\approx$ be a synchronous epistemic bisimulation on $\mathcal{I}$ with respect to Prop such that (a) $\mathcal{I} \models S \subseteq T$, and (b) for all points (r, m) of $\mathcal{I}$ there exists a point (r', m) such that $(r, m) \approx (r', m)$ and $S(r', m) = T(r, m)$. If $p \in Prop$ then $\mathcal{I} \models B_i^S CB_S p \Leftrightarrow B_i^T CB_T p$.*

▶ **Corollary 10.** *If p is an atomic proposition that depends only on the local states of the agents, and $\mathcal{F}$ is either a crash or omission failure model, then $\mathcal{I}_{P,\mathcal{E},\mathcal{F}} \models (B_i^{\mathcal{N}} CB_{\mathcal{N}} p) \Leftrightarrow B_i^{\mathcal{A}} CB_{\mathcal{A}} p$*

Taking $p = \exists v$, we see that we can use either the formula $B_i^{\mathcal{N}} CB_{\mathcal{N}} \exists v$ or $B_i^{\mathcal{A}} CB_{\mathcal{A}} \exists v$ in the knowledge based program, without changing its semantics.

6 Optimality with Respect to Limited Information Exchange

We now turn the the question of optimality of SBA protocols with respect to limited information exchange. The literature has concentrated on implementations P of the knowledge based program $\mathbf{P}(\Phi)$ with respect the full information exchange, because it can be shown that such implementations P are an optimum, in the sense that for every SBA protocol P' using any other information exchange, in every run the nonfaulty agents decide using P no later than they would in the corresponding run of P'. Here, a run r of P is said to *correspond* to a run r' of P' if they have the same initial state, hence the same adversary and initial states of all the agents.

Since the full information protocol P may be impractical or even require agents to perform intractable computations we are interested in alternative limited information exchanges. However, having selected an information exchange, it is still desirable to use a protocol that is optimal amongst those that use the same information exchange. In this section, we consider whether the knowledge based program $\mathbf{P}(\Phi)$ yields such implementations. We show that this is the case in several ways, subject to some assumptions about the information exchange.

In order to fairly compare two decision protocols relative to an information exchange, it helps to assume that the information exchange does not explicitly transmit information about what decisions have been taken. Say that an information exchange protocol $\mathcal{E}$ with action sets $A_i = \{\texttt{noop}\} \cup \{\texttt{decide}_i(v) | v \in V\}$ *does not transmit decision information* if for all agents i, local states $s \in L_i$, and actions $\texttt{decide}_i(v_1), \texttt{decide}_i(v_2) \in A_i$, we have

- $\mu_i(s, \mathtt{decide}_i(v_1)) = \mu_i(s, \mathtt{decide}_i(v_2))$, and
- for all message vectors m, we have $\delta_i(s, \mathtt{decide}(v_1), m) = \delta_i(s, \mathtt{decide}(v_2), m)$.

Say that an information exchange protocol $\mathcal{E}$ *does not transmit information about actions* if for all agents i, local states $s \in L_i$, and actions $a_1, a_2 \in A_i$, we have

- $\mu_i(s, a_1) = \mu_i(s, a_2)$, and
- for all message vectors m, we have $\delta_i(s, a_1, m) = \delta_i(s, a_2, m)$.

Clearly, if $\mathcal{E}$ does not transmit information about actions, then it does not transmit decision information. Intuitively, the information such protocols exchange is only about the initial states and failure pattern, and not about decisions that the protocol has taken. Similarly, the information exchange protocol does not record information about decisions in its local state. In effect, this assumption states that agents should not base their decisions on what other agents have decided, but only on what information about the initial state and failures has been exchanged. Note also that an *early stopping* protocol, which stops transmitting information once it has decided, satisfies the property of not transmitting decision information, but such a protocol may transmit information about actions, since we may still have $\mu_i(s, \mathtt{noop}) \neq \mu_i(s, \mathtt{decide}_i(v))$.

We remark that a protocol, as defined in [6], determines the messages to be sent, and actions to be performed, as a function of a *view* (corresponding to our notion of local state) that is comprised of a history of messages received, a history of other inputs from the environment, the time, and the agent identity. This means that the [6] protocols (including their full-information protocols) do not transmit information about actions. However, in the case of a full-information protocol, and other protocols that exchange sufficient information, it is in fact possible, knowing the decision protocol that the agents are running, for an agent to deduce what actions other agents have taken in the past.

We work with the following order on decision protocols: $P' \leq_{\mathcal{E},\mathcal{F}} P$ if for all runs r' of $\mathcal{I}_{P',\mathcal{E},\mathcal{F}}$, and all agents i, if agent i decides in round m in run r', then in the corresponding run r of $\mathcal{I}_{P,\mathcal{E},\mathcal{F}}$, agent i decides no earlier than round m (or not at all). An SBA protocol P is *optimal* with respect to an information exchange $\mathcal{E}$ and failure model $\mathcal{F}$, if for all SBA protocols P' with respect to $\mathcal{E}$ and $\mathcal{F}$, if $P' \leq_{\mathcal{E},\mathcal{F}} P$ then $P \leq_{\mathcal{E},\mathcal{F}} P'$. That is, there is no SBA protocol P' that decides no later than P, and sometimes decides earlier.

▶ **Theorem 11.** *Suppose the information exchange $\mathcal{E}$ is synchronous and does not transmit decision information, and that the protocol P implements $\mathbf{P}(\Phi)$ with respect to information exchange $\mathcal{E}$ and failure model $\mathcal{F}$. Then P is an optimal SBA protocol with respect to information exchange $\mathcal{E}$ and failure model $\mathcal{F}$,*

Proof. We prove optimality. Suppose that $P' \leq_{\mathcal{E},\mathcal{F}} P$. We show that there is no run where some agent i running P' decides strictly earlier than in the corresponding run of P. Moreover, we show that for all runs r' of $\mathcal{I}_{P',\mathcal{E},\mathcal{F}}$, and all times m, then for the corresponding run r of $\mathcal{I}_{P,\mathcal{E},\mathcal{F}}$, we have that $r'_i(m) = r_i(m)$ for all agents i.

The proof is by induction on m. For $m = 0$, we have that $r'_i(0) = r_i(0)$ for all agents i, by definition of correspondence. Moreover, there can be no instance of P' deciding in an earlier round than P before time $m = 0$.

For the inductive case, assume that that we have that for all agents i, $r'_i(k) = r_i(k)$ for all $k \leq m$, and there is no instance, before time m, of some agent using P' deciding in an earlier round than it would using P. We show that for each agent i, protocols P' and P either both decide (possibly on different values), or both perform $\mathtt{noop}$. It will follow from this that $r'_i(m+1) = r_i(m+1)$ for all agents i. Also, it remains true for each agent i that P' has not decided earlier than P to time $m+1$.

We first show that for each agent i, either both $P_i(r_i(m)) = P'_i(r'_i(m)) = \texttt{noop}$ or there exists $v, v' \in V$ such that $P_i(r_i(m)) = \texttt{decide}_i(v)$ and $P'_i(r'_i(m)) = \texttt{decide}_i(v')$. Obviously, this holds if $P_i(r_i(m)) = P'_i(r'_i(m)) = \texttt{noop}$, so we need only consider the cases where either protocol decides. If $P'(r'_i(m)) = \texttt{decide}_i(v')$, then by Lemma 7, we have that $\mathcal{I}_{P',\mathcal{E},\mathcal{F}}, (r', m) \models B_i^N(CB_N \exists v')$. Because the local states of corresponding runs of $\mathcal{I}_{P,\mathcal{E},\mathcal{F}}$ are identical to time m to those of $\mathcal{I}_{P',\mathcal{E},\mathcal{F}}$, it follows that $\mathcal{I}_{P,\mathcal{E},\mathcal{F}}, (r, m) \models B_i^N(CB_N \exists v')$. Because $P' \leq_{\mathcal{E},\mathcal{F}} P$, agent i has not yet decided at the point (r, m). Since P implements $\mathbf{P}(\Phi)$, it follows that $P_i(r_i(m)) = \texttt{decide}_i(v)$ for some value v. Alternately, if $P(r_i(m)) = \texttt{decide}_i(v)$, then because P' has not decided earlier, and $P' \leq_{\mathcal{E},\mathcal{F}} P$, we must have $P'(r'_i(m)) = \texttt{decide}_i(v')$ for some value v'. Thus, in either case we have that both protocols decide, as required.

Next, we show that $r'_i(m+1) = r_i(m+1)$ for all agents i. The proof considers several cases, but in each case, the fact that the local states of all agents are identical in $r'(m)$ and $r(m)$ and that for each agent i, protocols P' and P either both decide, or both perform $\texttt{noop}$, implies that the same messages are sent by each agent in round $m + 1$ of r' and r'. (In the case that both protocols decide, we use the fact that $\mathcal{E}$ does not transmit decision information.) Moreover, the failure patterns are identical in these corresponding runs, so the same vector ρ_i represents the messages received by agent i in round $m + 1$ in run r and in run r'. If $P_i(r_i(m)) = P'_i(r'_i(m)) = \texttt{noop}$, then we have $r'_i(m + 1) = \delta_i(r'_i(m), \texttt{noop}, \rho_i) = \delta_i(r_i(m), \texttt{noop}, \rho_i) = r_i(m + 1)$. Alternately, if $P_i(r_i(m)) = \texttt{decide}_i(v)$ and $P'_i(r'_i(m)) = \texttt{decide}_i(v')$ then, because $\mathcal{E}$ does not record decision information, we have $r'_i(m+1) = \delta_i(r'_i(m), \texttt{decide}(v'), \rho_i) = \delta_i(r_i(m), \texttt{decide}(v'), \rho_i) = \delta_i(r_i(m), \texttt{decide}(v), \rho_i) = r_i(m + 1)$. ◀

Note that Theorem 11 does not state that an implementation P of the knowledge-based program is an *optimum* SBA protocol, in the sense that $P \leq_{\mathcal{E},\mathcal{F}} P'$ for all SBA protocols P' with respect to $\mathcal{E}$ and $\mathcal{F}$. In fact, this is not true, as we show in Section 7. The counter-example illustrates a trade-off between information exchange and decision time: sending less information may result in making later decisions. The information exchange in this counter-example does not transmit decision information, but it does transmit information about actions. However, for information exchanges that do not transmit information about actions, we do obtain that the knowledge-based program implementation is an optimum.

▶ **Theorem 12.** *Suppose that information exchange $\mathcal{E}$ does not transmit information about actions. Let P be an implementation of the knowledge-based program $\mathbf{P}(\Phi)$ with respect to $\mathcal{E}$ and failure model $\mathcal{F}$. Then P is an optimum SBA protocol with respect to $\mathcal{E}$ and $\mathcal{F}$.*

Proof. Note first that for all SBA protocols P and P', if r and r' are corresponding runs of P and P' with respect to $\mathcal{E}$ and $\mathcal{F}$, then because $\mathcal{E}$ does not transmit information about actions, for all times m and all agents i, we have $r_i(m) = r'_i(m)$. We show that P decides no later than P' for all agents i. Suppose that P' decides in round $m + 1$ in run r'. Then $\mathcal{I}_{P',\mathcal{E},\mathcal{F}}(r', m) \models B_i^N(CB_N \exists v)$ for some value v. Since the local states are always the same with respect to P, we also have $\mathcal{I}_{P,\mathcal{E},\mathcal{F}}(r, m) \models B_i^N(CB_N \exists v)$. Because P implements $\mathbf{P}(\Phi)$ this implies that either agent i has already decided before time m in run r, or agent i also decides in round $m + 1$ in run r. ◀

7 A Counter-example

For the counter-example promised above, we demonstrate that the implementation P of $\mathbf{P}(\Phi)$ with respect to an information exchange $\mathcal{E}$ and the sending omissions failure model SO_t is not always an optimum SBA protocol with respect to $\mathcal{E}$ and SO_t. To do so, we provide an SBA protocol P' with respect to $\mathcal{E}$ and SO_t such that we do not have $P \leq_{\mathcal{E},SO_t} P'$. We take $V = \{0,1\}$ and give the description of P' for an arbitrary number n of agents of which up to $t \leq n$ are faulty, but then specialize to $n = 4$ and $t = 3$ for the counter-example.

The information exchange $\mathcal{E}$ is defined as follows. The local states L_i of agent i are tuples of the form $\langle init_i, w_i, new_i, kf_i, done_i, time_i \rangle$, where

- $init_i \in \{0,1\}$ is the agent's initial value,
- $w_i \in \mathcal{P}(\{0,1\})$ is, intuitively, the set of values that the agent knows to be the initial value of some agent,
- $new_i \in \mathcal{P}(\{0,1\})$ is, intuitively, the set of values that the agent first learned about in the most recent round,
- $kf_i \in \mathcal{P}(\mathtt{Agt})$ is, intuitively, the set of agents that the agent knows to be faulty,
- $done_i \in \{0,1\}$ indicates whether the agent has made a decision, and
- $time_i$ is the current time.

The initial local states I_i are the states with $w_i = \{init_i\}$, $new_i = \{init_i\}$, $kf = \emptyset$ and $done_i = time_i = 0$.

Agent i's set of messages M_i contains $\perp$ and messages of the form $\langle n, f \rangle$, where $n \subseteq \{0,1\}$ and $f \subseteq \mathtt{Agt}$. Intuitively, n is a set of values that agent i has just learned about, and f is a set of agents that agent i knows to be faulty. The message that agent i sends when it performs action a and has local state $s_i = \langle init_i, w_i, new_i, kf_i, done_i, time_i \rangle$ is defined as follows:

- If either $done_i = 1$ or $a = \mathtt{decide}_i(v)$ for some $v \in \{0,1\}$, then $\mu_i(s_i, a) = \langle \emptyset, \emptyset \rangle$. Intuitively, if either the agent is in the process of deciding, or it has already decided, then it sends a message carrying no information. Note that this is different from sending no message, since reception of such a message informs the recipient that agent i did not make a sending omission fault in the current round. Effectively, when an agent decides, it stops participating in the protocol, except for sending a heartbeat message in each round.
- Otherwise $\mu_i(s_i, a) = \langle new_i, kf_i \rangle$. That is, if the agent has not yet decided and in the current round performs the action $a = \mathtt{noop}$, the agent transmits the set of values it has newly learned about, and the set of agents that it knows to be faulty.

When agent i is in local state $s_i = \langle init_i, w_i, new_i, kf_i, done_i, time_i \rangle$, performs action a, and receives vector of messages $(m_1, \ldots, m_n)$ from the other agents, agent i's state update $\delta_i(s_i, a, (m_1, \ldots, m_n)) = \langle init'_i, w'_i, new'_i, kf'_i, done'_i, time'_i \rangle$ is defined as follows. Let $J \subseteq \mathtt{Agt}$ be the set of agents from which agent i actually receives a message, so that $m_j = \perp$ iff $j \notin J$. For $j \in J$, suppose $m_j = (n_j, f_j)$. Then

- $init'_i = init_i$,
- $w'_i = w_i \cup \bigcup_{j \in J} n_j$,
- $new'_i = w'_i \setminus w_i$,
- $kf'_i = kf_i \cup (\mathtt{Agt} \setminus J) \cup \bigcup_{j \in J} f_j$,
- if $a = \mathtt{decide}_i(v)$ for some $v \in \{0,1\}$, then $done_i = 1$, otherwise $done'_i = done_i$, and
- $time'_i = time_i + 1$.

Intuitively, the agent collects in w'_i the values that it has heard about, either previously or as new values transmitted by the other agents in the current round. It records an agent j as known to be faulty in kf'_i if either it already knew j to be faulty, it does not receive a message from j in the current round, or it receives a message saying that j is faulty. This completes the description of the information exchange $\mathcal{E}$.

The protocol P' is defined for agent i on a local state $s_i = \langle init_i, w_i, new_i, kf_i, done_i, time_i \rangle$, when there may be up to t faulty agents, by $P'_i(s_i) = \mathtt{decide}_i(v)$ if $done_i = 0$ and v is the least value in w_i and either $\mathit{time} = t + 1$ or $\mathit{kf}_i = \mathtt{Agt} \setminus \{i\}$, and $P_i(s_i) = \mathtt{noop}$ otherwise. That is, an agent decides if it learns that it is the only nonfaulty agent, otherwise it waits to time $t + 1$ to make a decision.

▶ **Proposition 13.** *P' is an SBA protocol with respect to $\mathcal{E}$ and SO_t.*

We now argue that for the implementation P of $\mathbf{P}(\Phi)$ with respect to $\mathcal{E}$ and SO_t, we do not have that $P \leq_{\mathcal{E},SO_t} P'$. Consider the case of $n = 4$ and $t = 3$, and let r be a run in which the only failures are that agents 1,2, and 3 omit to send their message to agent 1 in round 1. Note that the model is defined in such a way that an agent is able to detect its own faultiness by seeing that a message it sent to itself was not received. Hence, we have $\mathit{kf}_1(r, 1) = \{1, 2, 3\}$. In case of protocol P, this means that $\mathcal{I}_{P,\mathcal{E},SO_t}, (r, 1) \models K_i(i \notin \mathcal{N})$, which implies that $\mathcal{I}_{P,\mathcal{E},SO_t}, (r, 1) \models B_i^N CB_N \exists v$ for all v. According to P, therefore, agent 1 decides in round 2 and sends the message $(\emptyset, \emptyset)$ in round 2 (and all subsequent rounds). This means that at time 2, all other agents i have $\mathit{kf}_i(r, 2) = \emptyset$. The run is indistinguishable to the other agents from a run without failures. When $t = n - 1$, the earliest possible decision time in a run without failures is round $t + 1$ (see appendix), but $t = 3$, so no nonfaulty agent running P can decide in round 3 in run r.

On the other hand, for protocol P', agent 1 does not decide in round 2 of the run r' corresponding to r, since we do not have $\mathit{kf}_1(r', 1) = \mathtt{Agt} \setminus \{1\}$ or $1 = t + 1 = 4$. By the definition of $\mathcal{E}$, this means that agent 1 sends a message $(w, \{1, 2, 3\})$ in round 2, and the nonfaulty agents i have $\mathit{kf}_i(r', 2) = \{1, 2, 3\}$. This means that the nonfaulty agents all decide in round 3 of the run r'.

We therefore have a run in which the nonfaulty agents decide earlier using P' than they do when using the corresponding run of P, so it is not the case that $P \leq_{\mathcal{E},SO_t} P'$. We remark that this remains the case had we defined $\leq_{\mathcal{E},\mathcal{F}}$ by comparing decision times of only the nonfaulty agents, rather than all agents.

8 Conclusion

Our focus has been on *Simultaneous* Byzantine Agreement, in which the nonfaulty agents are required to decide at the same time. A number of variants of the specification have been studied in the literature on the knowedge based approach to distributed algorithms.

One dimension of variation is with respect to the behaviour of faulty agents. The SBA specification does not require the faulty agents to make the same decision as the nonfaulty agents. Neiger and Tuttle [9] consider the *uniform* (also called *consistent*) variant, in which the faulty agents, if they decide, must agree with the nonfaulty agents. They show that a different formulation of common knowledge captures the condition under which a decision can be made, which is equivalent to the "common belief" condition for the crash and sending omissions failure models, but may differ otherwise. Since, in general, the faulty agents cannot decide ahead of the nonfaulty agents in this problem, the example of Section 7 does not apply in this case, so it remains open to understand optimality of Uniform SBA with respect to limited information exchange.

Another dimension of variation is with respect to simultaneity. In *Eventual Byzantine Agreement* (EBA), the nonfaulty agents may decide at different times. In general, there is not an optimum protocol for this specification, but there are optimal protocols. Halpern, Moses and Waarts [5] show that a more complex notion called "continual common knowledge" is required to capture the conditions under which a decision can be made in optimal protocols for EBA. Neiger and Bazzi [8] show that adding a termination requirement to the specification further complicates the required notion of common knowledge. We do not presently have a general characterization of optimality with respect to limited information exchange for EBA. Alpturer, Halpern and van der Meyden [1] present optimal protocols, for full information exchange and for two specific limited information exchanges, but the proof of optimality for the latter uses side conditions that do not hold in general. In particular, information exchanges involving reports about faults detected, such as our example in Section 7, do not satisfy these side conditions. A satisfactory general characterization of optimality for EBA with respect to limited information exchange therefore remains open.

We have identified conditions on the information exchange under which the knowledge-based program $\mathbf{P}(\Phi)$ gives an optimum with respect to a limited information exchange that does not transmit information about actions, but also a counter-example that shows that this knowledge-based program yields an optimal implementation but does not yield an optimum implementation when the information exchange transmits information about actions. The underlying reason is that the knowledge based program forces faulty agents to decide early, and this may diminish the amount of information available to the nonfaulty agents.

Conceivably, another knowledge based program can express the optimum implementation, if one exists, with respect to an order that compares the decision times of only the nonfaulty agents only. However, it would seem that such a program would need agents that discover that they are faulty to determine when they decide based on counterfactual reasoning about the consequences, on the decision times of the nonfaulty agents, of deciding or deferring a decision. This introduces a number of complexities. For one thing, the knowledge-based program would need to refer to the future, and a unique implementation of the knowledge based program is then not guaranteed to exist. Counterfactual reasoning in knowledge based programs also requires a more complex semantic framework, which has been little studied. (The only relevant work is [4].) We therefore leave this question for future work.

References

1. Kaya Alpturer, Joseph Y. Halpern, and Ron van der Meyden. Optimal eventual byzantine agreement protocols with omission failures. In *Proc. ACM Symp. on Principles of Distributed Computing, PODC*, pages 244–252. ACM, 2023.
2. C. Dwork and Y. Moses. Knowledge and common knowledge in a Byzantine environment: crash failures. *Information and Computation*, 88(2):156–186, 1990.
3. R. Fagin, J.Y. Halpern, Y. Moses, and M.Y. Vardi. *Reasoning About Knowledge*. The MIT Press, 1995.
4. J. Y. Halpern and Y. Moses. Using counterfactuals in knowledge-based programming. *Distributed Computing*, 17(2):91–106, 2004.
5. Joseph Y. Halpern, Yoram Moses, and Orli Waarts. A characterization of eventual byzantine agreement. *SIAM J. Comput.*, 31(3):838–865, 2001.
6. Y. Moses and M. R. Tuttle. Programming simultaneous actions using common knowledge. *Algorithmica*, 3:121–169, 1988.
7. Yoram Moses. Optimum simultaneous consensus for general omissions is equivalent to an NP oracle. In *Proc. Distributed Computing, DISC 2009*, volume 5805 of *Lecture Notes in Computer Science*, pages 436–448. Springer, 2009.
8. Gil Neiger and Rida A. Bazzi. Using knowledge to optimally achieve coordination in distributed systems. *Theor. Comput. Sci.*, 220(1):31–65, 1999.
9. Gil Neiger and Mark R. Tuttle. Common knowledge and consistent simultaneous coordination. *Distributed Comput.*, 6(3):181–192, 1993.
10. M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27(2):228–234, 1980.

A Proofs for Section 2 (Knowledge in Interpreted Systems)

▶ **Proposition 1.** *If A and B are indexical sets such that $A \subseteq B$ is valid, then the formulas $B_i^B \phi \Rightarrow B_i^A \phi$, $CK_B \phi \Rightarrow CK_A \phi$ and $CB_B \phi \Rightarrow CB_A \phi$ are valid.*

Proof. Suppose that $\mathcal{I}, (r, m) \models B_i^B \phi$. Then $\mathcal{I}, (r', m') \models \phi$ for all points $(r', m') \sim_i (r, m)$ such that $i \in B(r', m')$. We show that $\mathcal{I}, (r, m) \models B_i^A \phi$. Let $(r', m) \sim_i (r, m)$ and suppose that $i \in A(r', m)$. Since $A \subseteq B$ is valid in $\mathcal{I}$, we also have $i \in B(r', m)$, and it follows that $\mathcal{I}, (r', m') \models \phi$.

Similarly, suppose that $\mathcal{I}, (r, m) \models CK_B \phi$. Then $\mathcal{I}, (r', m') \models \phi$ for all points (r', m') of $\mathcal{I}$ such that $(r, m) \sim_B^* (r', m')$. When $A \subseteq B$ is valid in $\mathcal{I}$, we have for all points (r', m') that $(r, m) \sim_A^* (r', m')$ implies $(r, m) \sim_B^* (r', m')$, hence $\mathcal{I}, (r', m') \models \phi$. This shows that $\mathcal{I}, (r, m) \models CK_B \phi$.

The proof of $CB_B \phi \Rightarrow CB_A \phi$ is similar, using instead the characterization in terms of the relations $\approx^* A$ and $\approx^* B$. ◀

▶ **Proposition 2.** *The formulas $K_i \phi \Rightarrow B_i^A \phi$, $EK_A \phi \Rightarrow EB_A \phi$ and $CK_A \phi \Rightarrow CB_A \phi$ are valid.*

Proof. Validity of $K_i \phi \Rightarrow B_i^A \phi$ is immediate from the fact that $B_i^A \phi$ is $K_i(i \in A \Rightarrow \phi)$.

For $EK_A \phi \Rightarrow EB_A \phi$, note that if $EK_A \phi$ then $\bigwedge_{i \in A} K_i \phi$, which implies $\bigwedge_{i \in A} B_i^A \phi$ by the previous paragraph, and this is $EB_A \phi$.

For $CK_A \phi \Rightarrow CB_A \phi$, we show by induction that $EK_A^k \phi \Rightarrow EB_A^k \phi$ is valid for all $k > 0$. The base case of $k = 1$ is the result of the previous paragraph. Assuming $EK_A^k \phi \Rightarrow EB_A^k \phi$ is valid, we have that if $EK_A^{k+1} \phi$ then $EK_A(EK_A^k \phi)$, which implies $EB_A(EK_A^k \phi)$ by the result of the first paragraph, and then $EB_A(EB_A^k \phi) = EB_A^{k+1} \phi$ by the inductive hypothesis. It follows that $CK_A \phi = \bigwedge_{k>0} EK_A^k \phi$ implies $\bigwedge_{k>0} EB_A^k \phi = CB_A \phi$. ◀

B Proofs for Section 5 (Crash Failures)

▶ **Proposition 3.** *Let γ be any context for SBA, and P any protocol for this context, and let S and T be indexical sets of agents such that $\mathcal{I}_{P,\gamma} \models S \subseteq T$. If $\mathcal{I}_{P,\gamma} \models SBA(T)$ then $\mathcal{I}_{P,\gamma} \models SBA(S)$. In particular if $\mathcal{I}_{P,\gamma} \models SBA(\mathcal{A})$ then $\mathcal{I}_{P,\gamma} \models SBA(\mathcal{N})$.*

Proof. The Unique-Decision is property is independent of the indexical set in the specification, so holds trivially. Validity(T) implies Validity(S) since $S \subseteq T$ is valid. Also, Simultaneous-Agreement(T) implies Simultaneous-Agreement(S) for the same reason. Thus validity of SBA(T) implies validity of SBA(S). The fact that $\mathcal{I}_{P,\gamma} \models$ SBA($\mathcal{A}$) implies $\mathcal{I}_{P,\gamma} \models$ SBA($\mathcal{N}$) follows directly from the fact that $\mathcal{N} \subseteq \mathcal{A}$ is valid. ◀

▶ **Proposition 4.** *Suppose that P is a protocol for the context γ, and let S and T be indexical sets of agents in $\mathcal{I}_{P,\gamma}$, such that*

(a) for all points (r, m), if $\mathcal{I}_{P,\gamma}, (r, m) \models i \in T \wedge j \in T$, then there exists a run r' such that $(r, m) \sim_i (r', m)$ and $(r, m) \sim_j (r', m)$, and $\mathcal{I}_{P,\gamma}, (r, m) \models i \in S \wedge j \in S$, and

(b) $\mathcal{I}_{P,\gamma} \models S \subseteq T$, and

(c) $\mathcal{I}_{P,\gamma} \models T \neq \emptyset \Rightarrow S \neq \emptyset$.

Then $\mathcal{I}_{P,\gamma} \models SBA(S)$ implies $\mathcal{I}_{P,\gamma} \models SBA(T)$.

Proof. Assume that $\mathcal{I}_{P,\gamma} \models$ SBA(S). We first show that Simultaneous-Agreement(T) is valid in $\mathcal{I}_{P,\gamma}$. Suppose $\mathcal{I}_{P,\gamma}, (r, m) \models i \in T \wedge \texttt{decides}_i(v)$ and let $j \in T(r, m)$. We show that $\mathcal{I}_{P,\gamma}, (r, m) \models \texttt{decides}_j(v)$. By (a), there exists a point (r', m) such that $(r, m) \sim_i (r', m)$ and $(r, m) \sim_j (r', m)$ and $\mathcal{I}_{P,\gamma}, (r, m) \models i \in S \wedge j \in S$. Since $(r, m) \sim_i (r', m)$, we have

$P_i(r'_i(m)) = P(r_i(m)) = \mathtt{decide}_i(v)$, so also $\mathcal{I}_{P,\gamma},(r',m) \models \mathtt{decides}_i(v)$. Since $\mathcal{I}_{P,\gamma} \models$ SBA(S) we have Simultaneous-Agreement(S) and it follows that $\mathcal{I}_{P,\gamma},(r',m) \models \mathtt{decides}_j(v)$. Because $(r,m) \sim_j (r',m)$, we also have $\mathcal{I}_{P,\gamma},(r,m) \models \mathtt{decides}_j(v)$, as required.

Next, we show Validity(T) is valid in $\mathcal{I}_{P,\gamma}$. Let (r,m) be a point where $\mathcal{I}_{P,\gamma},(r,m) \models \mathtt{decides}_i(v) \wedge i \in T$. Since Simultaneous-Agreement(T) is valid in $\mathcal{I}_{P,\gamma}$, as shown above, we have $\mathcal{I}_{P,\gamma},(r,m) \models \mathtt{decides}_j(v)$ for all $j \in T(r,m)$. Since $S \subseteq T$ is valid, by (b), we have $\mathcal{I}_{P,\gamma},(r,m) \models \mathtt{decides}_j(v)$ for all $j \in S(r,m)$. Because $S(r,m) \neq \emptyset$, by (c) and the fact that $i \in T(r,m)$, there exists $j \in S(r,m)$ such that $\mathcal{I}_{P,\gamma},(r,m) \models \mathtt{decides}_j(v)$. It now follows from Validity(S) that $\mathcal{I}_{P,\gamma},(r,m) \models \mathit{init}_k = v$ for some agent k.

The property Unique-Decision is the same in SBA(S) and SBA(T), so this is immediate. ◀

▶ **Corollary 5.** *For crash failures and omissions failure contexts γ and protocols P, with $\mathcal{I}_{P,\gamma} \models \mathcal{N} \neq \emptyset$, we have $\mathcal{I}_{P,\gamma} \models SBA(\mathcal{N})$ implies $\mathcal{I}_{P,\gamma} \models SBA(\mathcal{A})$.*

Proof. The result follows using Proposition 4 with $S = \mathcal{N}$ and $T = \mathcal{A}$. Condition (b) in Proposition 4 follows from the definitions of $\mathcal{N}$ and $\mathcal{A}$. Condition (c) is direct, by assumption. We show that condition (a) of Proposition 4 holds for these failure models. Suppose $\mathcal{I}_{P,\gamma},(r,m) \models i \in \mathcal{A} \wedge j \in \mathcal{A}$. Let r' be the run that is identical to r to time m, but in which the adversary is modified so that agents i and j never fail after time m. Since these agents did not have a failure in run r before time m either, we have $\mathcal{I}_{P,\gamma},(r',m) \models i \in \mathcal{N} \wedge j \in \mathcal{N}$ as required. Because runs are determined by their initial states, the protocol P and the adversary, there is no difference between r and r' in the adversary before time m, we have $(r,m) \sim_i (r',m)$ and $(r,m) \sim_j (r',m)$ in particular. ◀

▶ **Proposition 6.** *Suppose that P is a protocol for the hard crash failures context $(\mathcal{E}, \mathit{Crash}_t)$ with $t < n$ such that $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t} \models SBA(\mathcal{N})$. Then $\mathtt{decides}_i(v) \Rightarrow CK_{\mathcal{A}}(\mathtt{decides}_{\mathcal{A}}(v))$ and $\mathtt{decides}_i(v) \Rightarrow CK_{\mathcal{A}}(\exists v)$ are valid in $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t}$.*

Proof. From Proposition 5, we obtain from $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t} \models \mathrm{SBA}(\mathcal{N}) \wedge \mathcal{N} \neq \emptyset$ that $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t} \models$ SBA($\mathcal{A}$). Suppose that $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t},(r,m) \models \mathtt{decides}_i(v)$. Then we cannot have that $r_i(m) = \mathit{crashed}$, and thus $i \in \mathcal{A}(r,m)$. It follows from SBA($\mathcal{A}$) that $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t},(r,m) \models \mathtt{decides}_{\mathcal{A}}(v)$. Let $j \in \mathcal{A}(r,m)$ and $(r,m) \sim_j (r',m)$. Then $r_j(m) = r'_j(m) \neq \mathit{crashed}$ so also $j \in \mathcal{A}(r',m)$ and $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t},(r',m) \models \mathtt{decides}_j(v)$. Using SBA($\mathcal{A}$), we obtain $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t},(r',m) \models \mathtt{decides}_{\mathcal{A}}(v)$.This shows that for all agents i, $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t} \models \mathtt{decides}_i(v) \Rightarrow EK_{\mathcal{A}}\mathtt{decides}_{\mathcal{A}}(v)$, which implies that $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t} \models \mathtt{decides}_{\mathcal{A}}(v) \Rightarrow EK_{\mathcal{A}}\mathtt{decides}_{\mathcal{A}}(v)$. By induction, this gives $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t} \models \mathtt{decides}_{\mathcal{A}}(v) \Rightarrow CK_{\mathcal{A}}\mathtt{decides}_{\mathcal{A}}(v)$, and we derive $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t} \models \mathtt{decides}_i(v) \Rightarrow CK_{\mathcal{A}}\mathtt{decides}_{\mathcal{A}}(v)$. Next, it follows using $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t} \models$ SBA($\mathcal{N}$) $\wedge\, \emptyset \neq \mathcal{N} \subseteq \mathcal{A}$ and Validity($\mathcal{N}$) that $\mathcal{I}_{P,\mathcal{E},\mathit{Crash}_t} \models \mathtt{decides}_i(v) \Rightarrow CK_{\mathcal{A}}\exists v$. ◀

▶ **Lemma 7.** *Let S be an indexical set of agents and suppose that P is an SBA(S) protocol for an information exchange protocol $\mathcal{E}$ and failure environment $\mathcal{F}$. Then the formula $\mathtt{decide}_i(v) \Rightarrow B_i^S CB_S \exists v$ is valid in $\mathcal{I}_{P,\mathcal{E},\mathcal{F}}$.*

Proof. For brevity, we write $\mathcal{I}$ for $\mathcal{I}_{P,\mathcal{E},\mathcal{F}}$. We first show that $\mathtt{decides}_S(v) \Rightarrow CB_S\mathtt{decides}_S(v)$ is valid in $\mathcal{I}$. Suppose that $\mathcal{I},(r,m) \models \mathtt{decides}_i(v)$. Suppose $(r,m) \sim_i (r',m)$ where $i \in S(r',m)$. Then $P_i(r'_i(m)) = P_i(r_i(m)) = \mathtt{decide}_i(v)$. Since P is an SBA(S) protocol and $i \in S(r',m)$, it follows by Simultaneous-Agreement(S) that $\mathcal{I},(r',m) \models \mathtt{decides}_S(v)$. This shows that $\mathcal{I} \models \mathtt{decide}_i(v) \Rightarrow B_i^S\mathtt{decide}_S(v)$. Since this holds for all i, it follows that $\mathtt{decides}_S(v) \Rightarrow EB_S\mathtt{decides}_S(v)$ is valid in $\mathcal{I}$. It follows by induction that $\mathtt{decides}_S(v) \Rightarrow CB_S\mathtt{decides}S(v)$ is valid in $\mathcal{I}$.

Next, notice that $(S \neq \emptyset \wedge CB_S\texttt{decide}_S(v)) \Rightarrow EB_S(S \neq \emptyset \wedge CB_S\texttt{decide}_S(v) \wedge \exists v)$ is valid in $\mathcal{I}$. This is because (i) $B_i^S(s \neq \emptyset)$ is valid by definition of B_i^S, (ii) $CB_S\phi \Rightarrow EB_SCB_S\phi$ is valid for all ϕ, and because (iii) $(S \neq \emptyset \wedge \texttt{decide}_S(v)) \Rightarrow \exists v$ is valid in $\mathcal{I}$ by Validity(S). By induction, we conclude that $S \neq \emptyset \wedge CB_S\texttt{decide}_S(v)) \Rightarrow CB_S\exists v$ is valid in $\mathcal{I}$. ◀

▶ **Proposition 8.** *If P is an SBA($\mathcal{N}$) protocol for the hard crash context $(\mathcal{E}, Crash_t)$ with $t < n$ then $CK_{\mathcal{A}}(\texttt{decides}_{\mathcal{A}}(v)) \Leftrightarrow CB_{\mathcal{N}}(\texttt{decides}_{\mathcal{N}}(v))$ and $i \in \mathcal{A} \Rightarrow (K_iCK_{\mathcal{A}}(\texttt{decides}_{\mathcal{A}}(v)) \Leftrightarrow B_i^NCB_{\mathcal{N}}(\texttt{decides}_{\mathcal{N}}(v))$ are valid in $\mathcal{I}_{P,\mathcal{E},Crash_t}$.*

Proof. Suppose $\mathcal{I}_{P,\mathcal{E},Crash_t}, (r,m) \models CK_{\mathcal{A}}(\texttt{decides}_{\mathcal{A}}(v))$. Then we have $\mathcal{I}_{P,\mathcal{E},Crash_t}, (r,m) \models CB_{\mathcal{A}}(\texttt{decides}_{\mathcal{A}}(v))$ by Proposition 2. Since $\mathcal{N} \subseteq \mathcal{A}$ is valid, it follows that $\mathcal{I}_{P,\mathcal{E},Crash_t}, (r,m) \models CB_{\mathcal{N}}(\texttt{decides}_{\mathcal{A}}(v))$, and also that $\texttt{decides}_{\mathcal{A}}(v) \Rightarrow \texttt{decides}_{\mathcal{N}}(v)$ is valid. It follows that $\mathcal{I}_{P,\mathcal{E},Crash_t}, (r,m) \models CB_{\mathcal{N}}(\texttt{decides}_{\mathcal{N}}(v))$.

Conversely, suppose that $\mathcal{I}_{P,\mathcal{E},Crash_t}, (r,m) \models CB_{\mathcal{N}}(\texttt{decides}_{\mathcal{N}}(v))$. Since $\mathcal{N} \neq 0$ is valid, we have $\mathcal{I}_{P,\mathcal{E},Crash_t}, (r,m) \models \texttt{decides}_i(v)$ for some $i \in \mathcal{N}(r,m)$. By Proposition 6, $\mathcal{I}_{P,\mathcal{E},Crash_t}, (r,m) \models CK_{\mathcal{A}}\texttt{decides}_{\mathcal{A}}(v)$.

This shows validity of $CK_{\mathcal{A}}(\texttt{decides}_{\mathcal{A}}(v)) \Leftrightarrow CB_{\mathcal{N}}(\texttt{decides}_{\mathcal{N}}(v))$. Validity of the formula $i \in \mathcal{A} \Rightarrow (K_iCK_{\mathcal{A}}(\texttt{decides}_{\mathcal{A}}(v)) \Leftrightarrow B_i^NCB_{\mathcal{N}}(\texttt{decides}_{\mathcal{N}}(v))$ follows from this using the fact that $K_i\phi \Rightarrow B_i^N\phi$ is valid, and that in the hard crash system $\mathcal{I}_{P,\mathcal{E},Crash_t}$, we have $i \in \mathcal{A} \Rightarrow K_i(i \in \mathcal{A})$. ◀

▶ **Proposition 9.** *Suppose that S and T are indexical sets of agents in an interpreted system $\mathcal{I}$, and let $\approx$ be a synchronous epistemic bisimulation on $\mathcal{I}$ with respect to Prop such that (a) $\mathcal{I} \models S \subseteq T$, and (b) for all points (r,m) of $\mathcal{I}$ there exists a point (r',m) such that $(r,m) \approx (r',m)$ and $S(r',m) = T(r,m)$. If $p \in Prop$ then $\mathcal{I} \models B_i^SCB_Sp \Leftrightarrow B_i^TCB_Tp$.*

Proof. We have $\mathcal{I} \models CB_T\phi \Rightarrow CB_S\phi$ and hence $\mathcal{I} \models (B_i^TCB_T\phi) \Rightarrow B_i^SCB_S\phi$ by Proposition 1. For the converse, we prove $\mathcal{I} \models (\neg B_i^TCB_T\phi) \Rightarrow \neg B_i^SCB_S\phi$. Suppose that $\mathcal{I}, (r,m) \models \neg B_i^TCB_T\phi$. Then there exists a point $(r^0,m) \sim_i (r,m)$ such that $i \in T(r^0,m)$ and $\mathcal{I}, (r^0,m) \models \neg CB_T\phi_T$. Moreover, from the latter we have that there exists a sequence $(r^0,m) \sim_{i_1} (r^1,m) \sim_{i_2} \ldots \sim_{i_k} (r^k,m)$ such that $\mathcal{I}, (r^k,m) \models \neg p$ and for $j = 1 \ldots k$ we have $i_j \in T(r^{j-1},m) \cap T(r^j,m)$. By the assumptions on $\approx$, there exists for each $j = 0 \ldots k$ a run ρ^j of $\mathcal{I}$ such that $(r^j,m) \approx (\rho^j,m)$, and $S(\rho^j,m) = T(r^j,m)$. Since $(r^k,m) \approx (\rho^k,m)$ we obtain that $\mathcal{I}, (\rho^k,m) \models \neg p$. Also for $j = 1 \ldots k$ we have $i_j \in T(r^{j-1},m) \cap T(r^j,m) = S(r^{j-1},m) \cap S(r^j,m)$. It follows that $\mathcal{I}, (\rho^0,m) \models \neg CB_Sp$.

Moreover, $i \in T(r^0,m) = S(\rho^0,m)$, and because $(r^0,m) \approx (\rho^0,m)$, we have $(r^0,m) \sim_i (\rho^0,m)$. Because $(r,m) \sim_i (r^0,m)$, we obtain $(r,m) \sim_i (\rho^0,m)$. It follows that $\mathcal{I}, (r,m) \models \neg B_i^SCB_Sp$. ◀

We remark that the above proof does not show that $\mathcal{I} \models CB_S\phi \Leftrightarrow CB_T\phi$.

▶ **Corollary 10.** *If p is an atomic proposition that depends only on the local states of the agents, and $\mathcal{F}$ is either a crash or omission failure model, then $\mathcal{I}_{P,\mathcal{E},\mathcal{F}} \models (B_i^{\mathcal{N}}CB_{\mathcal{N}}p) \Leftrightarrow B_i^{\mathcal{A}}CB_{\mathcal{A}}p$*

Proof. Define the relation $\approx$ on the points of $\mathcal{I}_{P,\mathcal{E},\mathcal{F}}$ by $(r,m) \approx (r',m)$ if for all agents i, we have that i has the same initial state in r as in r', and the behaviour of the adversary of r up to time m is the same as the the behaviour of the adversary of r' up to time m. In particular, it follows from $(r,m) \approx (r',m)$ that we have $\mathcal{A}(r,m) = \mathcal{A}(r',m)$ and $(r,m) \sim_i (r',m)$ for all agents i. If we take $S = \mathcal{N}$ and $T = \mathcal{A}$ then the assumptions of Proposition 9 are satisfied with respect to $\approx$. In particular, note that we can obtain the run r' required for condition (b) by changing the adversary so that there are no new faults after time m. The claim is then immediate. ◀

C Proofs for Section 7 (A Counter-example)

▶ **Proposition 13.** *P' is an SBA protocol with respect to $\mathcal{E}$ and SO_t.*

Proof. Unique-Decision holds because an agent performs a $\mathtt{decide}_i(v)$ action only if $done_i = 0$, and the variable $done_i$ captures whether the agent has performed a $\mathtt{decide}_i(v')$ action some time in the past. Validity($\mathcal{N}$) holds because when agent i performs $\mathtt{decide}_i(v)$, we have $v \in w_i$, which can be the case only when some agent j had $init_j = v$, by the initial condition and update rule for w_i.

For Simultaneous-Agreement, suppose that $i \in \mathcal{N}$ performs $\mathtt{decide}_i(v)$ in round $m+1$ of run r. By definition of $\mathcal{E}$, the set $kf_i(r,m)$ contains only faulty agents. Hence, in the case where $kf_i(r,m) = \mathtt{Agt} \setminus \{i\}$, we have that i is the only nonfaulty agent in run r, and Simultaneous-Agreement holds trivially. Otherwise, suppose that $m = t+1$. If any nonfaulty agent $j \neq i$ decided earlier, then j can only have done so because it is the only nonfaulty agent, contradicting the assumption that i is nonfaulty. Hence no nonfaulty agent has decided earlier. This implies that all nonfaulty agents decide in round $m+1$ also. ◀

Figure 1 shows a key part of the argument for the fact that a decision cannot be made in round three in a failure free run. The figure depicts a sequence of runs for four agents and indistinguishability relations at time 2, from a failure free run (at the top of the diagram) with both 0 and 1 values to a run (at the bottom of the diagram) with only 0 values. Dashed lines indicate messages that are *not* sent. (We omit messages that are sent in order to avoid cluttering the diagram.) N indicates nonfaulty agents. This shows that the first run, at time 2, is $\approx^*_{\mathcal{N}}$ related to the last (also at time 2). The first run is indistinguishable to the first agent from a similar run that has three 1 and one 0 value, and a similar sequence then shows that there is also a $\approx_{\mathcal{N}}$ path to a run with only 1 values. It follows that, in a failure free run such as the first, we do not have either $B_i^N CB_{\mathcal{N}} \exists 0$ or $B_i^N CB_{\mathcal{N}} \exists 1$.

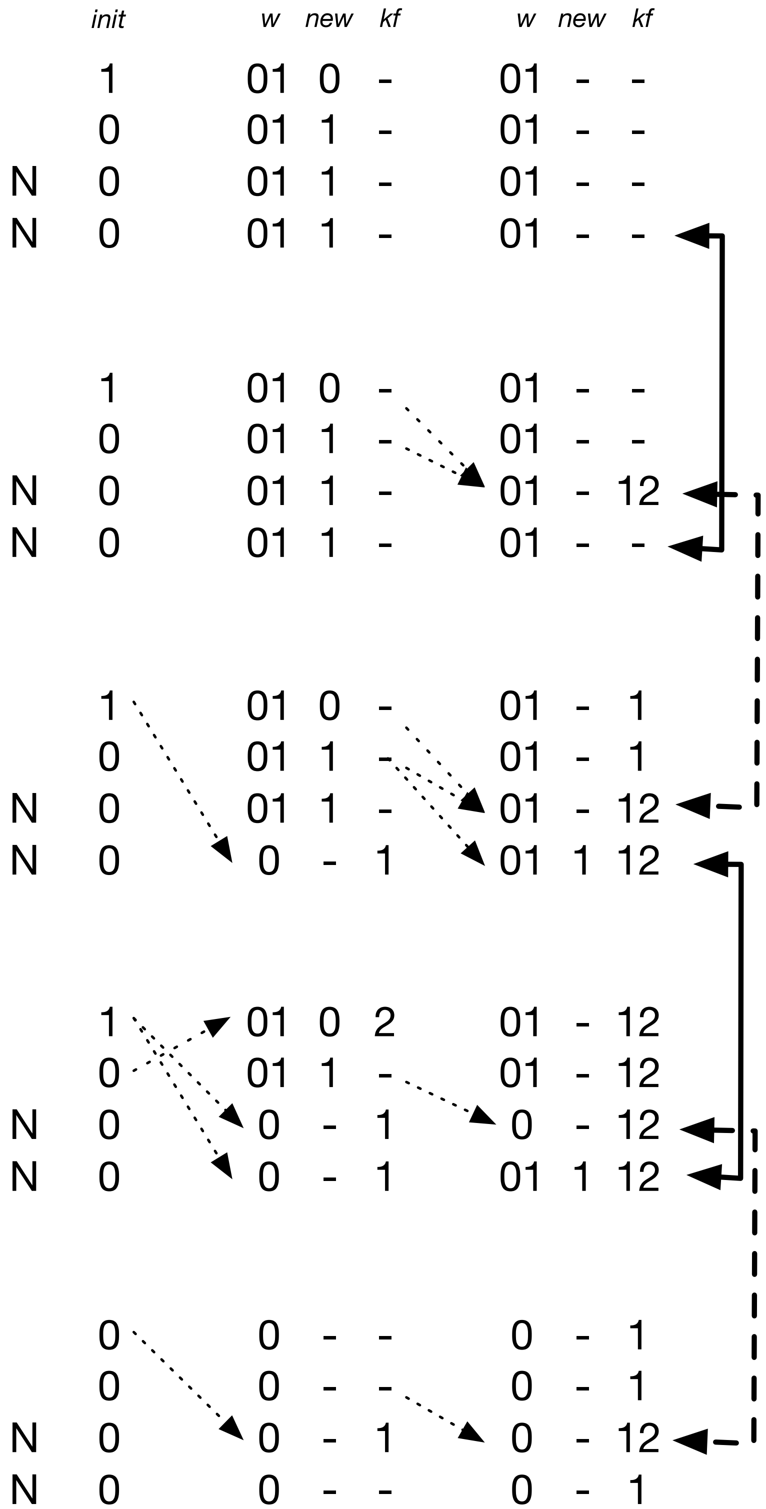

Figure 1 Runs of $\mathcal{E}$