A Characterization of Eventual Byzantine Agreement*

Joseph Y. Halpern†
Computer Science Dept.
Cornell University
Ithaca, NY 14853
halpern@cs.cornell.edu
http://www.cs.cornell.edu/home/halpern

Yoram Moses‡
Weizmann Institute
Rehovot, Israel
yoram@cs.weizmann.ac.il

Orli Waarts§
Stanford University
Stanford, CA 94035
orli@cs.stanford.edu

Abstract: We investigate eventual Byzantine agreement (EBA) in the crash and omission failure modes. The emphasis is on characterizing optimal EBA protocols in terms of the states of knowledge required by the processors in order to attain EBA. It is well known that common knowledge among the nonfaulty processors is a necessary and sufficient condition for attaining *simultaneous* Byzantine agreement (SBA). We define a new variant that we call *continual* common knowledge, and use it to provide necessary and sufficient conditions for attaining EBA. Using our characterization, we provide a technique that allows us to start with any EBA protocol and convert it to an optimal EBA protocol using a two-step process.

*An early version of this work appeared in the Proceedings of the Ninth Annual ACM Symposium on Principles of Distributed Computing, 1990.

†Much of this work was carried out while the author was at the IBM Almaden Research Center. IBM's support is gratefully acknowledged. The work was also supported in part by NSF under grant IRI-96-25901 and by the Air Force Office of Scientific Research under contract F49620-91-C-0080 and grant F49620-96-1-0323.

‡Supported by a grant from the Israel Academy of Science.

§Supported in part by contract ONR N00014-88-K-0166.

1 Introduction

In a distributed system, it is important to be able to design protocols that attain their goals despite the presence of unreliable components. The essence of the difficulties in doing this is captured by the well-studied problem of *Byzantine agreement* (BA), first introduced in [PSL80]. Roughly speaking, a Byzantine agreement protocol provides a means for n processors, at most t of which may be faulty, to agree on a value, in such a way that all nonfaulty processors decide on the same value, and when all processors start with the same initial value, the nonfaulty processors decide on this value. The version of the problem formulated in [PSL80] did not require all processors to decide on the value simultaneously, yet the algorithms given in [PSL80] (and many later papers) had the property that the processors did indeed decide simultaneously. In [DRS90], it was pointed out that requiring simultaneous agreement could significantly affect the problem. Since then, both simultaneous Byzantine agreement (SBA), where processors are required to decide simultaneously, and eventual Byzantine agreement (EBA), where they are not, have received a great deal of attention.

We will be particularly interested here in the problem of reaching EBA as quickly as possible. How many rounds it takes for a protocol to reach a decision depends in general on the pattern of failures, that is, how and when failures occur. Thus, roughly speaking, we say that protocol P_1 *dominates* protocol P_2 if every nonfaulty processor in a run of P_1 decides at least as soon as it does in the *corresponding* run of P_2 (where two runs are said to correspond if the initial values of all processors and the pattern of failures are the same in both); we say that P_1 *strictly dominates* P_2 if it dominates P_2 and in some run of P_1, at least one nonfaulty processor decides earlier than it does in the corresponding run of P_2. *Optimal* protocols are given for Byzantine agreement in the case of *crash failures* [DM90], where a processor that is faulty sends no messages after it has failed, and in the case of the more general *(sending) omission* failures [MT88], where a processor may continue to send messages after it has failed. These protocols are actually *optimum* protocols, in the sense that they dominate every other protocol for SBA.

The construction of these protocols, as well as the analysis of their optimality, is done in terms of reasoning about knowledge [Hal87, HM90]. The key observation is that simultaneous actions are intimately related to *common knowledge*: it can be shown that a necessary and sufficient condition for optimal SBA is that the nonfaulty processors decide once they have common knowledge of an initial value.

By definition of EBA, the processors have the freedom to decide at different times. This freedom makes it possible to construct EBA protocols that typically decide much faster than SBA protocols [DRS90]; it is also the cause of a number of subtle, but significant, differences between SBA and EBA. In particular (as already pointed out in [MT88]) although there are *optimal* protocols for EBA—ones that are not strictly dominated by any other protocol—there are no *optimum* protocols for EBA.

The differences between EBA and SBA can be related to differences in the state of knowledge required to achieve them. The fact that SBA can be reduced to attaining common knowledge about some initial value suggests that it should be possible to reduce EBA to attaining some variant of common knowledge, along the lines of the variants discussed in [HM90]. A plausible choice might be *eventual* common knowledge. While common knowledge corresponds to “everyone knows that everyone knows that everyone knows ...”, eventual common knowledge essentially corresponds to “eventually everyone will know that eventually everyone will know ...”. However, eventual common knowledge about some initial value turns out to be too weak a requirement for decision on that value. For example, consider a protocol where a processor decides on the value v ($v \in \{0,1\}$) once it knows that there is eventual common knowledge that someone started with initial value v. Such a decision rule will typically lead to inconsistency: it is quite possible that at some point, processor 1 knows that there is eventual common knowledge that some initial value was 0 and does not know that there is eventual common knowledge that some initial value was 1, while processor 2 knows that there is eventual common knowledge that some initial value was 1 and does not know that there is eventual common knowledge that some initial value was 0. Thus, such eventual common knowledge cannot guarantee consistency. (Note that, by way of contrast, with common knowledge it is the case that if one processor has common knowledge of φ, then all the processors do, so consistency is assured.)

The obvious solution is to require a processor to decide 0 when it knows that there is eventual common knowledge that some initial value was 0, but to decide 1 only when it knows that there will never be eventual common knowledge that some initial value was 0. Although this state of knowledge is indeed sufficient for consistency, it is not necessary. That is, this protocol can be modified so that processors will be able to decide 1 earlier while still preserving consistency.

What is needed here is some condition that says “decide 1 as long as you are sure that everybody that ever has eventual common knowledge that some initial value was 0, also has (at the same time) eventual common knowledge that some initial value was 1, and hence can decide 1.” It turns out that in order to capture this type of condition, we need a new variant of common knowledge that we call *continual common knowledge*. Roughly speaking, a fact φ is continual common knowledge if throughout the run everyone knows that throughout the run everyone knows that ... φ.

We show that continual common knowledge plays a role in EBA that is somewhat analogous to that played by common knowledge in SBA. In particular, we can characterize the state of knowledge needed to attain EBA in terms of continual common knowledge, although it turns out that the precise characterization is more complicated than that of SBA. Using our characterization, we are also able to characterize optimal EBA protocols. Moreover, we provide a general technique for converting *any* EBA protocol P to an optimal EBA protocol P' dominating P. All of the analysis and the conversion is done at the knowledge level, by working with high-level protocols with tests for knowledge.

This paper is organized as follows. In the next section we present our formal model of protocols and review the basic definitions of Byzantine agreement. In Section 3, we review the knowledge formalism and introduce the notion of continual common knowledge. In Section 4 we show that continual common knowledge is both a necessary and sufficient state of knowledge for an agreement protocol. We use these results in Section 5 to characterize optimal knowledge-based protocols for EBA and then use our characterization to show how we can construct optimal EBA protocols, starting with arbitrary EBA protocols. Section 6 uses the technique described in Section 5 to construct examples of optimal protocols, including a polynomial time optimal EBA protocol for the case of crash failures. We conclude with some discussion in Section 7. Most proofs are deferred to the appendix.

2 Byzantine agreement and full-information protocols

In this section, we review the Byzantine agreement problem, define the notion of optimality we are interested in, present our formal model of protocols, and show that, if optimality is our only concern, we can restrict to special protocols known as *full-information protocols.*

2.1 The Byzantine agreement problem

Suppose we are given a system with n processors, at most t of which might be faulty. Each processor i has an initial value $v_i \in \{0, 1\}$. A *Byzantine agreement protocol* is one that satisfies the following properties:

1. *Decision:* Every nonfaulty processor i eventually decides (irreversibly) on a value $y_i \in V$.
2. *Agreement:* All nonfaulty processors decide on the same value.
3. *Validity:* If all initial values v_i are identical, then all nonfaulty processors decide v_i.

The problem as stated above is actually *Eventual* Byzantine agreement (EBA). In order to define *Simultaneous* Byzantine agreement (SBA), we need to add one more condition:

4. *Simultaneity:* All the nonfaulty processors decide at the same round.

It is well known that the Byzantine agreement problem is sensitive to the types of failures that might occur. We consider two basic failure modes in this paper:

1. *Crash failures:* a faulty processor behaves according to the protocol, except that it might commit a crash failure at an arbitrary round $k > 0$. If a processor commits a crash failure in round k (or simply *fails* in round k), then it obeys its protocol in all

rounds preceding round k, it does not send any message in the rounds following k, and in round k it sends an arbitrary (not necessarily strict) subset of the messages it is required to send by its protocol.

2. *Omission failures:* a faulty processor behaves according to the protocol, except that it may omit to send an arbitrary set of message in any given round. (What we are calling "omission failures" here are what were termed *sending omission failures* in [MT88]; we do not consider *general omission failures* [PT86] here, where a processor may omit to receive a message as well as omitting to send one.)

We do not consider the more general *Byzantine failures*, where faulty processors may behave arbitrarily, although we believe that our techniques will extend to that case. For simplicity, in this paper we focus on the case of *binary agreement*, where $V = \{0, 1\}$. Extending our methods to the general case is straightforward.

The first subtlety in comparing EBA and SBA already arises if we consider what we mean by a *nonfaulty* processor. Is a processor nonfaulty in round k in a run (i.e., execution) of the protocol if it does not fail at or before round k, or should we call it nonfaulty if it does not fail throughout the run? Since all *nonfaulty* processors are required to decide consistently, if we choose the first interpretation, then a processor can decide on some value only when it is guaranteed that even if it fails after its decision, all nonfaulty processors will finally decide upon the same value. On the other hand, choosing the second interpretation enables the processor to decide on a value as long as it knows that, provided it does not fail, then all nonfaulty processors will decide on the same value. Clearly, in the crash or omission failure modes, when considering protocols in which the processors do not send any message after they decide, there is essentially no difference between these choices. Consequently, there is no difference between the choices in the case of SBA. However, there is a substantial difference in the case of EBA. For the purposes of this paper, we call a processor "nonfaulty" in a particular run only if it is nonfaulty throughout the run. This usage is consistent with the usual usage in other papers on EBA for crash and omission failures (e.g. [Fis83, LF82, PT86]), although it differs somewhat from that of [DM90], where Dwork and Moses concentrate on the set of "active" processors, which can decrease over time.

Notice that in EBA we require all the nonfaulty processors to eventually decide on some value. It is occasionally useful to consider weaker notions. An *agreement protocol* is a protocol that satisfies

$2'$. *Weak agreement:* Nonfaulty processors do not decide on different values.

An agreement protocol is a *nontrivial agreement protocol* if it also satisfies a weaker form of validity:

$3'$. *Weak validity:* If all initial values v_i are identical, then all nonfaulty processors that decide, decide v_i.

Note that a nontrivial agreement protocol does not necessarily satisfy the decision property; a nonfaulty processor may not decide at all.

An *optimal protocol* for EBA (SBA, nontrivial agreement, etc.) is one that is not strictly dominated by any other protocol. An *optimum protocol* is one that dominates every other protocol. Although it is possible to find optimum protocols for SBA [DM90, MT88], this is not the case for EBA. We give a proof here for the sake of completeness.

Proposition 2.1: [MT88] *There are no optimum EBA protocols.*

Proof: Consider a variant of an EBA protocol for the crash failure mode that was first introduced in [LF82]: When a processor first learns that some processor has an initial value of 0, it decides 0, relays 0 (i.e., sends 0 to all the other processors), and halts; if by time $t+1$ a processor does not know of any processors with initial value 0, it decides 1 and halts. It is easy to see that this protocol achieves EBA. Moreover, all nonfaulty processors with initial value 0 decide at time 0. Call this protocol $P0$. There is a symmetric protocol $P1$ where the roles of 0 and 1 are reversed. In $P1$, all nonfaulty processors with initial value 1 decide at time 0. An optimum EBA protocol would have to dominate both $P0$ and $P1$. Thus, in an optimum EBA protocol, all processors would have to decide at time 0. But this is provably impossible [DS82]. ∎

This proof shows even more. Since it is well known [DS82] that in any EBA protocol there will always be some run in which some processor takes $t+1$ rounds to decide, it follows that given any EBA protocol P, there must be some run in which some processor takes at least $t+1$ rounds longer to decide than it does in one of $P0$ and $P1$. Thus, no protocol is guaranteed to even be close to optimum in all runs.

2.2 Optimal protocols: an example

We have just seen that there is no hope of obtaining optimum protocols for EBA. The existence of optimal protocols is, however, guaranteed. Intuitively, the reason is that there are a finite number of initial configurations of votes and a finite number of behaviors of faulty processors in the first k rounds, for every finite k. An easy application of König's Lemma thus shows that there must be a bound on the time at which all processors halt in all executions of any given EBA protocol. Thus, there cannot exist an infinite sequence $P_1, P_2, \ldots$ of correct EBA protocols where P_{i+i} strictly dominates P_i. It follows that if we start with an EBA protocol and repeatedly generate a strictly dominating protocol, then this process will terminate after a finite number of steps. Moreover, the last protocol in this process (i.e., one for which no strictly dominating protocol can be found) will be optimal.

One of the main results of this paper is an effective method of generating an optimal protocol dominating a given protocol. We now describe a simple optimal protocol in the crash failure mode that is generated in this way. Consider the protocol $P0$ defined in the proof of Proposition 2.1. In this protocol, processors decide 0 and send 0 to everyone else as soon as they learn that some processor had initial value 0. A processor that has not learned this by time $t + 1$, decides 1 at time $t + 1$. While this protocol is fairly efficient, it is not optimal.

Clearly, the fact that some processor had an initial value of 0 is propagated as fast as possible in this protocol. It follows that no correct EBA protocol can decide 0 any faster than $P0$ does, because, by the specification of EBA, in order to decide 0, it is necessary that some processor started with an initial value of 0, and in $P0$, processors decide 0 as soon as they learn that some processor had an initial value of 0. How about deciding on 1? Here, it seems, the protocol $P0$ is not being as efficient as possible. For example, we know from the specification of EBA that if all initial values are 1, then a decision of 1 is forced. Obviously, in order to reach this decision as soon as possible in runs with no failures, a processor should notify the other processors in the first round about its having an initial value of 1 as well as about its having an initial value of 0.

One way to get an optimal EBA protocol is to find a rule for deciding 1 as soon as possible without changing the rule used by $P0$ for deciding on 0. We now design a protocol with this property. The idea is that processors decide 1 as soon as they know that nobody is ever going to know that some processor had an initial value of 0. The protocol uses the same rule for deciding on 0 as $P0$. Each processor i maintains a list of processors and its information about their initial values and sends this list to all others in every round. The rule for deciding on 1 is based on the observation that processor i knows that no processor will ever know that some processor had an initial value of 0 if either

(a) processor i knows that all initial values are 1 or

(b) processor i hears from the same set of processors in two consecutive rounds, and still does not know that some initial value was 0.

Thus, each processor i decides 1 and communicates for one more round if one of the two properties above holds. We call this protocol $P0_{\mathrm{opt}}$. It is easy to check that by time $t + 1$, every nonfaulty processor i will either learn that some processor had an initial value of 0, or that all processors started with 1, or will hear from the same set of processors in two consecutive rounds in the fashion described by property (b) above. $P0_{\mathrm{opt}}$ thus dominates $P0$. Indeed, as we formally show in Section 6, $P0_{\mathrm{opt}}$ is an optimal EBA protocol for the crash failure mode. In fact, it is the unique optimal protocol dominating $P0$.

2.3 Protocols and systems

Up to now we have spoken informally of "protocols". We now formalize this notion.

We consider a synchronous distributed system consisting of a finite collection of $n \geq 2$ processors (automata) $\{1, \ldots, n\}$, each pair of which is connected by a two-way communication link. The processors share a global clock that starts out at time 0 and advances by increments of one. Communication in the system proceeds in a sequence of *rounds*, with round k taking place between time $k-1$ and time k. In each round, every processor first sends the messages it needs to send to other processors and then receives the messages that were sent to it by other processors in the same round. The identity of the sender and destination of each message, as well as the round in which it is sent, are assumed to be part of the message. At any given time, a processor's *message history* consists of the set of messages it has sent and received. Every processor p starts out in some *initial state* σ.

We think of the processors as following a *protocol*, which specifies the actions of each of the nonfaulty processors as a *deterministic* function of the processor's state. Following [Coa86, LF81] and others, we define a protocol in terms of a message generation function (which describes the messages that each processor sends as a function of its local state), a state transition function, and an output function. Formally, a protocol P is described by the following:

- V is the set of input values. Since we have assumed that the inputs in EBA are 0 and 1, we take $V = \{0, 1\}$ here.
- Q is the set of states. We assume $V \subseteq Q$.
- σ_i, for $i \in \{1, \ldots, n\}$, is the initial state of processor i.
- L is the set of messages. (We assume that L contains the null message Λ, corresponding to no message being sent.)
- $\mu_{i,j} : Q \to L$, for $i, j \in \{1, \ldots, n\}$, is the message generation function for messages sent from processor i to processor j.
- $\delta_i : Q \times L^n \to Q$, for $i \in \{1, \ldots, n\}$, is the state transition function for processor i.
- O is the set of output values; since we are interested in decisions on either 0 or 1, we take $O = \{\perp, 0, 1\}$ here, where $\perp$ denotes no output.

We take a *run* of a protocol to be a complete description of the system at each time step. This includes each processor's initial state (which we assume is an element of Q), message history, the decisions, and the behavior of the faulty processors. A *system* $\mathcal{R}$ is just a set of runs. We can associate with a protocol P the systems $\mathcal{R}_P^{cr}$ and $\mathcal{R}_P^{om}$, consisting of its possible runs under the assumption of crash failures and omission failures, respectively. For the most part, our results do not depend on whether we consider crash failures or omission failures. We omit the superscript and just write $\mathcal{R}_P$ whenever the type of failure is not relevant to the discussion. A *point* is a pair (r, k) consisting of a run r and a time k. We use $r_i(k)$ to denote processor i's state at the point (r, k). For technical reasons, it is convenient to assume that communication happens during a round (that is, between two points (r, m) and $(r, m+1)$), but that a decision is actually made at a given point, not during the round. Thus, we talk about a message being sent in round k and a decision being made at time k (of some run r).

As we said in Section 2.1, a processor is considered *nonfaulty* in a given run if it follows the protocol throughout the run.

The *faulty behavior* of processor i in a run is a complete description of the processors to whom i omits sending required messages at each round. The *failure pattern* of a run contains the faulty behavior of all the processors that fail in the run. We call the list of the processor's initial states the system's *initial configuration.* A protocol P, an initial configuration and a failure pattern uniquely determine a run. Two runs r and r' of protocols P and P' are called *corresponding runs* if they have the same initial configuration and failure pattern. Two points (r, m) and (r', m') of protocols P and P' are *corresponding points* if r and r' are corresponding runs of the two protocols and $m = m'$. Given two protocols P and P', we say that P *dominates* P' if every nonfaulty processor that makes a decision (i.e., outputs 0 or 1 at some point in a run) in a run r' of P' also does so in the corresponding run r of P, and in fact decides at least as soon in r as it does in r'. We say P *strictly dominates* P' if P dominates P' and in some run r of P there is a nonfaulty processor that decides sooner in r than it does in the corresponding run of P'.

Notice that our definition of “dominate” focuses on when the processors decide, and not when they halt. However, in all our protocols the processors can always halt after they know that all the nonfaulty processors have decided. In many cases they can in fact halt one round after they decide.

2.4 Full-information protocols

We now review the notion of a full-information protocol, and some of its implications. The treatment in this subsection follows the lines of [Coa86].

A protocol is said to be a *full-information* protocol if each processor is required to send its current state to all processors at each round. The state of a processor in a full-information protocol consists of the processor's name, initial state, message history, and the time on the global clock. Thus, the states of processors following a full-information protocol are completely independent of their decision function; at corresponding points in two full-information protocols, processors have the same states. We assume that for all the protocols we consider here, a processor's initial state is its initial value, either 0 or 1. Thus, full-information protocols differ only in their output functions.

Intuitively, the states of the processors in a full-information protocol make the finest possible distinctions among histories. That is why the full-information protocol is particularly well suited for proving possibility and impossibility of achieving certain goals in distributed systems, and for the design and analysis of distributed protocols. The following proposition and corollary formalize this intuition. The proposition is similar to a theorem stated and proved in [Coa86].

Proposition 2.2: *Let P be an arbitrary protocol in a system with omission failures. Then for each processor i there is a function f_i from i's state in a full-information protocol F to its state in P, such that for every pair (r, m) and (r', m) of corresponding points of F and P, we have $f_i(r_i(m)) = r'_i(m)$.*

Proof: See the appendix. ∎

As an immediate consequence, we get

Corollary 2.3: *Let P be an arbitrary protocol in a system with omission (resp., crash) failures. Then there is a full-information protocol that dominates P.*

This proposition shows that, just as in [DM90, MT88] for the case of SBA, we can restrict attention to full-information protocols when looking for optimal EBA protocols.

3 Continual common knowledge

As we mentioned in the introduction, we want to perform a knowledge-theoretic analysis of EBA. In this section, we introduce continual common knowledge, the key tool for doing so. We start with a review of the basic knowledge formalism.

3.1 The knowledge formalism

We want to be able to reason about the states of knowledge of the processors in the system. In order to do so, we use the formalism first introduced in [HM90].

We start with a collection of basic facts (which can essentially be thought of as primitive propositions). For each such basic fact, it will typically be clear whether or not it is true at a given point.[1] We define various basic facts as we go along. Among the basic facts of interest is $\exists 0$, which is true at a point (r, k) in $\mathcal{R}$ if some processor started with initial value 0 in r; we similarly define $\exists 1$. We close this language under the standard Boolean connectives $\wedge$, $\neg$ and $\Rightarrow$, interpreted as conjunction, negation and implication, as well as various knowledge operators. The basic operator is K_i, where $K_i\varphi$ is read "processor i knows φ".

A processor is said to *know* a fact at a given point (r, m) exactly if the fact holds at all of the points in which the processor has the same state as at (r, m). Thus, we have

[1] Strictly speaking, we need not just the system, but an *interpreted system*, which is a system together with an *interpretation* of the primitive propositions (see [FHMV95] for details). We assume that the interpretation of the basic facts will be clear from context, so we do not explicitly use interpretations here.

$(\mathcal{R}, r, m) \models K_i\varphi$ if $(\mathcal{R}, r', m') \models \varphi$ for all points (r', m') such that $r_i(m) = r'_i(m')$. Given a system $\mathcal{R}$, a formula φ is said to be *valid in* $\mathcal{R}$, denoted $\mathcal{R} \models \varphi$, if it holds at all points in $\mathcal{R}$. It is well known that our definition of knowledge satisfies the following properties (which correspond to the modal system S5):

Proposition 3.1: [HM92] *For every system $\mathcal{R}$ we have:*

(a) if $\mathcal{R} \models \varphi$ then $\mathcal{R} \models K_i\varphi$ (knowledge generalization)

(b) $\mathcal{R} \models (K_i\varphi \wedge K_i(\varphi \Rightarrow \psi)) \Rightarrow K_i\psi$ (distribution axiom)

(c) $\mathcal{R} \models K_i\varphi \Rightarrow \varphi$ (knowledge axiom)

(d) $\mathcal{R} \models K_i\varphi \Rightarrow K_iK_i\varphi$ (positive introspection axiom)

(e) $\mathcal{R} \models \neg K_i\varphi \Rightarrow K_i\neg K_i\varphi$ (negative introspection axiom).

Having defined knowledge for individual processors, we now extend this definition to states of knowledge of a group of processors. We are mainly interested in the state of knowledge called *common knowledge*, since this has been shown to be closely related to coordination and agreement [HM90]. Given a set G of processors, let $E_G\varphi$ be an abbreviation for $\wedge_{i\in G}K_i\varphi$. Thus, $E_G\varphi$ holds if every processor in G knows φ. $C_G\varphi$—φ is common knowledge among the processors in G—holds if everyone in G knows φ, everyone knows that everyone knows, and so on. Formally, taking $E_G^1\varphi$ to be an abbreviation for $E_G\varphi$ and $E_G^{k+1}\varphi$ to be an abbreviation for $E_GE_G^k\varphi$, we define

$$(\mathcal{R}, r, m) \models C_G\varphi \text{ iff } (\mathcal{R}, r, m) \models E_G^k\varphi \text{ for } k = 1, 2, 3, \ldots.$$

As shown in [DM90, MT88], what is of interest in simultaneous Byzantine agreement is not common knowledge among a fixed set G of processors, but common knowledge among the nonfaulty processors. The set of nonfaulty processors is a *nonrigid* set: its elements vary from one run to another. More generally, we allow nonrigid sets to vary from point to point. Formally, given a system $\mathcal{R}$, a nonrigid set $\mathcal{S}$ of processors in the system is a function associating with every point of the system a subset of the processors. In other words, $\mathcal{S}(r, m)$ is a (possibly different) set of processors for every point (r, m) of $\mathcal{R}$. We denote by $\mathcal{N}$ the nonrigid set of nonfaulty processors. Since a nonfaulty processor does not necessarily know that it is nonfaulty, we would like a notion of knowledge that is appropriate even when processors are not guaranteed to know whether they belong to the nonrigid set. Thus, following [MT88], given a nonrigid set $\mathcal{S}$ and a processor i, we define $B_i^{\mathcal{S}}\varphi = K_i(i \in \mathcal{S} \Rightarrow \varphi)$. More formally,

$$\begin{array}{l}(\mathcal{R}, r, m) \models B_i^{\mathcal{S}}\varphi \text{ iff } (\mathcal{R}, r', m') \models \varphi \text{ for all } (r', m') \\ \text{such that } r_i(m) = r'_i(m') \text{ and } i \in \mathcal{S}(r', m').\end{array}$$

In other words, $B_i^{\mathcal{S}}\varphi$ holds if i knows that *if it is in* $\mathcal{S}$ then φ holds. $B_i^{\mathcal{S}}$ is a notion of *belief* because $B_i^{\mathcal{S}}\varphi$ does not imply φ when $i \notin \mathcal{S}$.

We define $E_{\mathcal{S}}\varphi$ as $\bigwedge_{i\in\mathcal{S}} B_i^{\mathcal{S}}\varphi$. In other words, *everyone in $\mathcal{S}$ knows φ* if every processor in $\mathcal{S}$ knows that if it is in $\mathcal{S}$ then φ holds. Notice that if $\mathcal{S}(r,m)$ is empty then, by definition, $E_{\mathcal{S}}\varphi$ holds.

The notion of $C_{\mathcal{S}}\varphi$ is now defined as an infinite conjunction in terms of $E_{\mathcal{S}}\varphi$. Defining $E_{\mathcal{S}}^{k+1}\varphi$ inductively as an abbreviation for $E_{\mathcal{S}}E_{\mathcal{S}}^k\varphi$, we have

$$(\mathcal{R},r,m) \models C_{\mathcal{S}}\varphi \text{ iff } (\mathcal{R},r,m) \models E_{\mathcal{S}}^k\varphi \text{ for } k=1,2,\ldots$$

It is easy to see that if $\mathcal{S}$ is the constant function that always returns G, then $C_{\mathcal{S}}\varphi$ is equivalent to $C_G\varphi$. Thus, this definition extends the original definition of $C_G\varphi$ to nonrigid sets.

3.2 Eventual common knowledge and agreement

As shown in [DM90, MT88], common knowledge among the nonfaulty processors is just the right tool for characterizing optimal SBA. It is not quite right for EBA though, since in EBA it is clear that a processor can decide 0 (or 1) well before it knows that other processors know about this value and thus, *a fortiori*, before this value is common knowledge. As we said in the introduction, we might have hoped that *eventual common knowledge* would be the appropriate replacement for common knowledge, but that does not quite work.

To understand why, consider the following full-information protocol F_0 that uses eventual common knowledge in its decision rule. While we have not given a formal definition of eventual common knowledge here (one can be found in [FHMV95, HM92]), for the purposes of this discussion it suffices to know that if φ is eventually common knowledge, then it is eventual common knowledge. That is, if we denote eventual common knowledge of φ by $C^\diamond\varphi$, then $\diamond C\varphi \Rightarrow C^\diamond\varphi$ is valid. (As usual, $\diamond\psi$ denotes "eventually ψ"; later we also use $\Box\psi$, which denotes "always ψ".) Acccording to F_0, a processor decides 0 when it has eventual common knowledge that some initial value was 0 (i.e., when it knows that $C^\diamond\exists 0$ holds), and decides 1 when it knows not only that there is eventual common knowledge that some initial value was 1, but also that there can never be eventual common knowledge that some initial value was 0 (i.e., when it knows that $C^\diamond\exists 1 \wedge \Box\neg C^\diamond\exists 0$ holds). Clearly F_0 is a nontrivial agreement protocol.

Although the state of knowledge required for decision according to this rule is sufficient for nontrivial agreement, it is not necessary. Just as in the case of the protocol $P0$ of Section 2.2, it is possible to decide 1 earlier than F_0. For example, consider a run of the full-information protocol in the sending omissions failure mode, in which all processors start with initial value 1, there are t faulty processors, and the t faulty processors send no messages in the first two rounds. Techniques of [MT88] show that in this case it is common knowledge at the end of the second round that there exists an initial value of 1 (indeed, the values of all nonfaulty processors are common knowledge), but no processor knows that all initial values are 1. Indeed, each processor considers it possible that one of the faulty processors has an initial value of 0, and that it will send it to some nonfaulty processor in the third round. The analysis in [MT88] can be used to show that this will make the existence of a value of 0 common knowledge (and thus eventual common knowledge) at the end of the fourth round. Thus, according to the protocol F_0, no nonfaulty processor decides at the end of the second round in such a run. However, it is not hard to construct a protocol which dominates F_0 and does decide at this point.

To understand how this can be done, note that the reason we require a processor to wait until it knows that $C^{\diamond}\exists 1 \wedge \Box\neg C^{\diamond}\exists 0$ holds before it decides 1 is that otherwise we may have inconsistency: some processor may decide 0 while another may decide 1. Can we decide earlier while still avoiding inconsistency? One thought might be to have a processor decide 1 if knows that $C^{\diamond}\exists 1$ holds and that every nonfaulty processor that learns $C^{\diamond}\exists 0$ does so after it learns $C^{\diamond}\exists 1$. But to avoid inconsistency, we then have to modify the rule for deciding 0 so that a processor decides 0 if it knows that $C^{\diamond}\exists 0$ holds and it considers it possible that some nonfaulty processor will learn $C^{\diamond}\exists 0$ no later than $C^{\diamond}\exists 1$. It is easy to see that this gives us a nontrivial agreement protocol: it is impossible for one nonfaulty processor to decide 0 and another to decide 1. Moreover, it is not hard to show that this protocol dominates F_0: all processors decide no later in runs of this protocol than in the corresponding run of F_0 (although their decisions in corresponding runs may be different).

This protocol itself can be improved though, by refining the rule for deciding 0. It turns out that an optimal protocol is obtained when the process of refining the decision rules reaches a fixed point. These ideas can be formally captured by use of a new state of knowledge that we call *continual common knowledge*. This is the subject of the next section.

3.3 The definition of continual common knowledge

Roughly speaking, a fact φ is continual common knowledge in a run r among the processors in the nonrigid set $\mathcal{S}$ if at all points (r, m'), every processor that belongs to $\mathcal{S}(r, m')$ knows that at every point in the run, every processor that belongs to $\mathcal{S}$ at that point knows that ... φ holds. More formally, we first define

$$(\mathcal{R}, r, m) \models \boxdot\psi \text{ iff } (\mathcal{R}, r, m') \models \psi \text{ for all } m' \geq 0.$$

Thus, $\boxdot$ is analogous to the standard temporal logic operator $\Box$, except that instead of restricting attention to the present and future as we do with $\Box$, with $\boxdot$ we consider all times past, present and future.

We define $E_{\mathcal{S}}^{\boxdot}\varphi$ as an abbreviation for $\boxdot E_{\mathcal{S}}\varphi$. In other words, $E_{\mathcal{S}}^{\boxdot}\varphi$ holds at the point (r, m) if for all times m', every processor in $\mathcal{S}(r, m')$ knows at time m' that if it is in $\mathcal{S}$ then φ holds. The twist here is, of course, that because $\mathcal{S}$ is nonrigid, the $\mathcal{S}(r, m')$ may be different for different values of m'. Notice that if $\mathcal{S}(r, m')$ is empty for all $m' \geq 0$, then by definition $E_{\mathcal{S}}^{\boxdot}\varphi$ holds at (r, m) (and, in fact, throughout the run r).

The notion of continual common knowledge of φ is now defined as an infinite conjunction in terms of $E^{\boxdot}_{\mathcal{S}}\varphi$. Defining $(E^{\boxdot}_{\mathcal{S}})^{k+1}\varphi$ inductively as an abbreviation for $E^{\boxdot}_{\mathcal{S}}(E^{\boxdot}_{\mathcal{S}})^{k}\varphi$, we define

$$(\mathcal{R}, r, m) \models C^{\boxdot}_{\mathcal{S}}\varphi \text{ iff } (\mathcal{R}, r, m) \models (E^{\boxdot}_{\mathcal{S}})^{k}\varphi \text{ for every } k \geq 1.$$

Thus, $C^{\boxdot}_{\mathcal{S}}\varphi$ holds if at all times everyone in $\mathcal{S}$ knows that at all times everyone in $\mathcal{S}$ knows that $\ldots\varphi$ holds. We remark that just as $C_{\mathcal{S}}\varphi$ can be shown to be a greatest fixed point of the equation $X \Leftrightarrow E_{\mathcal{S}}(\varphi \wedge X)$ [FHMV95, HM90], so $C^{\boxdot}_{\mathcal{S}}\varphi$ is the greatest fixed point of the equation $X \Leftrightarrow E^{\boxdot}_{\mathcal{S}}(\varphi \wedge X)$; we omit further details here.

We can characterize continual common knowledge as follows. A point (r', m') is said to be *$\mathcal{S}$-$\boxdot$-reachable in k steps* from a point (r, m) if there exist runs $r^0, \ldots, r^k$, times $m_0, \ldots, m_k$, and processors $i_0, \ldots, i_{k-1}$ such that $r^0 = r$, $r^k = r'$, and, for $j = 0, \ldots, k-1$, we have that $i_j \in \mathcal{S}(r^j, m_j) \cap \mathcal{S}(r^{j+1}, m_{j+1})$ and $r^j_{i_j}(m_j) = r^{j+1}_{i_j}(m_{j+1})$. We say that (r', m') is *$\mathcal{S}$-$\boxdot$-reachable from* (r, m) if it is $\mathcal{S}$-$\boxdot$-reachable from (r, m) in k steps for some k. It is not too hard to check that

Proposition 3.2: *$(\mathcal{R}, r, m) \models (E^{\boxdot}_{\mathcal{S}})^{k}\varphi$ iff $(\mathcal{R}, r', m') \models \varphi$ for all points (r', m') that are $\mathcal{S}$-$\boxdot$-reachable from (r, m) in k steps.*

Corollary 3.3: *$(\mathcal{R}, r, m) \models C^{\boxdot}_{\mathcal{S}}\varphi$ if and only if $(\mathcal{R}, r', m') \models \varphi$ for all points (r', m') that are $\mathcal{S}$-$\boxdot$-reachable from (r, m).*

We remark that this result is a variant of the characterization of $C_{\mathcal{S}}\varphi$ given in [DM90]. Using that characterization, it was shown that $C_{\mathcal{S}}$ satisfied all the S5 axioms except the knowledge axiom[2] and that $C_{\mathcal{S}}\varphi \Rightarrow \varphi$ holds at points where $\mathcal{S}$ is nonempty.[3] $C_{\mathcal{S}}$ was also shown to satisfy two additional properties, known as the *induction rule* and the *fixed-point axiom*. Using Corollary 3.3, we can prove the analogous properties for continual common knowledge:

Lemma 3.4: *For every system $\mathcal{R}$ we have:*

(a) if $\mathcal{R} \models \varphi$ then $\mathcal{R} \models C^{\boxdot}_{\mathcal{S}}\varphi$ (common knowledge generalization)

(b) $\mathcal{R} \models (C^{\boxdot}_{\mathcal{S}}\varphi \wedge C^{\boxdot}_{\mathcal{S}}(\varphi \Rightarrow \psi)) \Rightarrow C^{\boxdot}_{\mathcal{S}}\psi$ (distribution axiom)

(c) $\mathcal{R} \models C^{\boxdot}_{\mathcal{S}}\varphi \Rightarrow C^{\boxdot}_{\mathcal{S}}C^{\boxdot}_{\mathcal{S}}\varphi$ (positive introspection)

(d) $\mathcal{R} \models \neg C^{\boxdot}_{\mathcal{S}}\varphi \Rightarrow C^{\boxdot}_{\mathcal{S}}\neg C^{\boxdot}_{\mathcal{S}}\varphi$ (negative introspection)

(e) $\mathcal{R} \models C^{\boxdot}_{\mathcal{S}}\varphi \Leftrightarrow E^{\boxdot}_{\mathcal{S}}(\varphi \wedge C^{\boxdot}_{\mathcal{S}}\varphi)$ (fixed-point axiom)

[2] The modal system satisfying precisely these properties is known as K45 [Che80].

[3] Actually, in [DM90] an instance of $C_{\mathcal{S}}$ was used that *does* satisfy the knowledge axiom. This is because the nonrigid sets $\mathcal{S}$ of interest in their application are always nonempty. However, $C_{\mathcal{S}}$ does not satisfy the knowledge axiom in the more general case in which the nonrigid set $\mathcal{S}(r, m)$ might occasionally be empty.

(f) *if* $\mathcal{R} \models \varphi \Rightarrow E_{\mathcal{S}}^{\boxdot}(\varphi \wedge \psi)$ *then* $\mathcal{R} \models \varphi \Rightarrow C_{\mathcal{S}}^{\boxdot}\psi$ *(induction rule)*

(g) $\mathcal{R} \models C_{\mathcal{S}}^{\boxdot}\varphi \Rightarrow \boxdot C_{\mathcal{S}}^{\boxdot}\varphi$.

Proof: The proof of parts (a)–(f) is essentially identical to the proof of the analogous properties for $C_{\mathcal{S}}$ in [DM90], so it is omitted here. Part (g) follows from the observation that for all m, m', the set of points $\mathcal{S}$-$\boxdot$-reachable from (r, m) and (r, m') is identical. We leave details to the reader. ∎

It is easy to see that $\models C_{\mathcal{S}}^{\boxdot}\varphi \Rightarrow C_{\mathcal{S}}\varphi$ for all formulas φ. It is also not hard to show that the converse does not hold in general. Thus, continual common knowledge is a variant of common knowledge that is in a precise sense strictly stronger than common knowledge.

4 Continual common knowledge and nontrivial agreement

In this section, we begin our analysis of EBA in terms of continual common knowledge. Let $decide_i(y)$, $y = 0, 1$, be the basic fact which is true at all points where processor i decides y. Since the truth of $decide_i(y)$ at a point depends only on processor i's local state at that point (we assumed that processors follow a deterministic protocol), when $decide_i(y)$ holds, processor i knows it. Moreover, since we have assumed that a processor cannot output 0 and 1 at the same time, we cannot have both $decide_i(0)$ and $decide_i(1)$ holding at any point. Formally,

Proposition 4.1: *If P is an agreement protocol, then for $y = 0, 1$, we have*

(a) $\mathcal{R}_P \models decide_i(y) \Rightarrow \neg decide_i(1-y)$

(b) $\mathcal{R}_P \models (K_i decide_i(y) \Leftrightarrow decide_i(y)) \wedge (K_i \neg decide_i(y) \Leftrightarrow \neg decide_i(y))$

(c) $\mathcal{R}_P \models i \in \mathcal{N} \Rightarrow ((B_i^{\mathcal{N}} decide_i(y) \Leftrightarrow decide_i(y)) \wedge (B_i^{\mathcal{N}} \neg decide_i(y) \Leftrightarrow \neg decide_i(y)))$

Lemma 4.2: *Suppose processors i, j are nonfaulty in run r of some agreement protocol P and for some point (r, m) we have $(\mathcal{R}_P, r, m) \models decide_i(0)$. Then $(\mathcal{R}_P, r, m) \models \boxdot \neg decide_j(1)$.*

Proof: Immediate from the definitions. ∎

We want to focus on the output component of a protocol P. To do this, the following definitions will be helpful. We say that a *decision set* $\mathcal{A}$ is a tuple $(\mathcal{A}_1, \ldots, \mathcal{A}_n)$, where $\mathcal{A}_i \subseteq Q$. We think of the $\mathcal{A}_i$ component of a decision set as consisting of all the states where processor i has decided on a particular value. A *decision pair* is a pair $(\mathcal{Z}, \mathcal{O})$ of decision sets. Intuitively, $\mathcal{Z}$ (for *zero*) describes the local states of processors at points where they are deciding or have decided on 0, while $\mathcal{O}$ (for *one*) describes the analogous states for a decision of 1. We say that $(\mathcal{Z}, \mathcal{O})$ *is the decision pair for protocol* P if $\mathcal{Z}_i$ consists of the local states at which processor i decides or has decided 0 when executing the protocol P, and $\mathcal{O}_i$ consists of the local states at which processor i decides or has decided 1. Typically, we shall describe a decision set such as $\mathcal{Z}_i$ or $\mathcal{O}_i$ by a formula of the form $K_i\varphi$ or $B_i^{\mathcal{N}}\psi$. Given a system, the decision set is then the set of local states in the system at which the formula is satisfied. For example, the decision set defined by $K_i\varphi$ is $\{r_i(m) : (\mathcal{R}, r, m) \models K_i\varphi\}$; similarly, the set defined by $B_i^{\mathcal{N}}\psi$ is $\{r_i(m) : (\mathcal{R}, r, m) \models B_i^{\mathcal{N}}\psi\}$.

Recall that $\mathcal{N}$ denotes the set of nonfaulty processors. Given a decision set $\mathcal{A}$, let $\mathcal{N} \wedge \mathcal{A}$ denote the nonrigid set described by $(\mathcal{N} \wedge \mathcal{A})(r, m) = \{i : i \in \mathcal{N}(r, m) \text{ and } r_i(m) \in \mathcal{A}_i\}$. With this machinery, we can now give a necessary condition for nontrivial agreement.

Proposition 4.3: *Let P be a nontrivial agreement protocol with decision pair $(\mathcal{Z}, \mathcal{O})$. Then*

(a) $\mathcal{R}_P \models decide_i(0) \Rightarrow B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0 \wedge \neg decide_i(1))$

(b) $\mathcal{R}_P \models decide_i(1) \Rightarrow B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}}\exists 1 \wedge \neg decide_i(0))$.

Proof: See the appendix. ∎

Proposition 4.3 shows that continual common knowledge is necessary for nontrivial agreement. The next result shows that it is also sufficient. It turns out that our sufficient condition is not quite identical to the necessary condition of Proposition 4.3, although it is quite similar.

Proposition 4.4: *Let P be a protocol with decision pair $(\mathcal{Z}, \mathcal{O})$ such that either*

(a) $\mathcal{R}_P \models decide_i(0) \Rightarrow B_i^{\mathcal{N}}\exists 0$ *and*

(b) $\mathcal{R}_P \models decide_i(1) \Leftrightarrow B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}}\exists 1)$

or

(a′) $\mathcal{R}_P \models decide_i(0) \Leftrightarrow B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0)$ *and*

(b′) $\mathcal{R}_P \models decide_i(1) \Rightarrow B_i^{\mathcal{N}}\exists 1$.

Then P is a nontrivial agreement protocol.

Proof: See the appendix. ∎

We remark that the asymmetry between the decision sets for 0 and 1 in Proposition 4.4 is essential. While the conditions in Propositions 4.3 and 4.4 are not identical, once we move to *optimal* protocols, we will be able to get a single necessary and sufficient condition, expressed in terms of continual common knowledge, that characterizes optimality.

5 Constructing and characterizing optimal agreement

It is not too hard to show that for optimal nontrivial protocols the implications ("$\Rightarrow$") in parts (a) and (b) of Proposition 4.3 become equivalences ("$\Leftrightarrow$"). More importantly, we can show that if "sufficient information" is transmitted, as is the case in a full-information protocol, then a protocol satisfying these conditions is optimal. These points are made precise and proved in this section. In the process, we also show how to construct an optimal protocol dominating any given protocol.

From here on in, we focus on full-information protocols. As we indicated in Section 2.4, full-information protocols are all we need to consider if we are interested in optimal protocols for EBA. We use $\mathit{FIP}(\mathcal{Z},\mathcal{O})$ to denote the (unique) full-information protocol with decision pair $(\mathcal{Z},\mathcal{O})$.

The following result is the core of our technique for constructing optimal protocols. It provides a knowledge-theoretic technique for constructing protocols that dominate a given (full-information) nontrivial agreement protocol and plays a key role in our characterization of optimal nontrivial agreement protocols. It can be viewed as a formalization of some of the intuitions presented in Section 3.2. Notice that in this proposition, we start with a full-information protocol $\mathit{FIP}(\mathcal{Z},\mathcal{O})$ and then use the decision sets $\mathcal{Z}$ and $\mathcal{O}$ to construct other decision sets $\mathcal{Z}'$, $\mathcal{O}'$, $\mathcal{Z}''$, and $\mathcal{O}''$, which then become the basis for two new full-information protocols.

Proposition 5.1: *Let $F = \mathit{FIP}(\mathcal{Z},\mathcal{O})$ be a full-information nontrivial agreement protocol, and define $\mathcal{Z}', \mathcal{O}', \mathcal{Z}'', \mathcal{O}''$ as follows:*

- $\mathcal{Z}'_i = B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0)$
- $\mathcal{O}'_i = B_i^{\mathcal{N}}(\exists 1 \wedge \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0)$
- $\mathcal{Z}''_i = B_i^{\mathcal{N}}(\exists 0 \wedge \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}}\exists 1)$
- $\mathcal{O}''_i = B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}}\exists 1)$.

Then $F' = \mathit{FIP}(\mathcal{Z}',\mathcal{O}')$ and $F'' = \mathit{FIP}(\mathcal{Z}'',\mathcal{O}'')$ are both nontrivial agreement protocols that dominate F.

Proof: See the appendix. ∎

Proposition 5.1 suggests a way to construct an optimal nontrivial agreement protocol. Namely, start with a full-information nontrivial agreement protocol $F = \mathit{FIP}(\mathcal{Z},\mathcal{O})$. Then construct a new protocol $F^1 = \mathit{FIP}(\mathcal{Z}^1,\mathcal{O}^1)$ where $\mathcal{Z}^1 = \mathcal{Z}'$ and $\mathcal{O}^1 = \mathcal{O}'$, using the notation of Proposition 5.1. Notice that the new protocol is completely determined by the decision set $\mathcal{O}$ of the original protocol. We then proceed to create a second protocol $F^2 = \mathit{FIP}(\mathcal{Z}^2,\mathcal{O}^2)$, where $\mathcal{Z}^2 = (\mathcal{Z}^1)''$ and $\mathcal{O}^2 = (\mathcal{O}^1)''$. Protocol F^2 is thus completely determined by $\mathcal{Z}^1 = \mathcal{Z}'$. We can, of course, continue in this fashion and define protocols $F^{2,1}, F^{2,2}, \ldots$. By Proposition 5.1, each new protocol we construct dominates the previous ones. Furthermore, it can be shown that if the protocol we started with is not optimal, two such steps will yield a protocol that strictly dominates it. By the observations in Section 2.2, if we start with an EBA protocol, this process eventually terminates (from some point on the sequence of protocols it generates is constant), giving us an optimal EBA protocol. However, it is far from clear how long it takes this process to terminate. Indeed, it is not clear whether the process will terminate *at all* if we start with an arbitrary nontrivial agreement protocol. We now show that, in fact, the process always terminates in two steps, so the resulting protocol F^2 is indeed optimal. (By symmetry, it also follows that the analogous construction, exchanging the roles of $\mathcal{Z}$ and $\mathcal{O}$, results in an optimal protocol.)

Theorem 5.2: *Let $F = FIP(\mathcal{Z}, \mathcal{O})$ be a full-information nontrivial agreement protocol, let $F^1 = FIP(\mathcal{Z}^1, \mathcal{O}^1)$, and let $F^2 = FIP(\mathcal{Z}^2, \mathcal{O}^2)$. Then F^2 is an optimal nontrivial agreement protocol. Moreover, if F is an EBA protocol, then F^2 is an optimal EBA protocol dominating F.*

Proof: See the appendix. ∎

We can also use Proposition 5.1 to help us show, as claimed earlier, that the necessary conditions of Proposition 4.3 actually characterize optimal (full-information) nontrivial agreement protocols.

Theorem 5.3: *Let $F = FIP(\mathcal{Z}, \mathcal{O})$ be a full-information nontrivial agreement protocol. Then F is optimal iff both of the following conditions hold:*

(a) $\mathcal{R}_F \models i \in \mathcal{N} \Rightarrow (decide_i(0) \Leftrightarrow B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0 \wedge \neg decide_i(1)))$

(b) $\mathcal{R}_F \models i \in \mathcal{N} \Rightarrow (decide_i(1) \Leftrightarrow B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1 \wedge \neg decide_i(0)))$.

Proof: See the appendix. ∎

There is a subtle point to be brought out here. Up to now, we have ignored the issue of whether we are dealing with crash failures or omission failures. Notice, however, that the sets $\mathcal{Z}_i'$, $\mathcal{O}_i'$, $\mathcal{Z}_i''$, and $\mathcal{O}_i''$ are defined in terms of what processors know at certain points. Of course, what they know at a given point will depend in part on whether we are dealing with crash failures or omission failures. Suppose, for example, we start with a nontrivial agreement protocol F that works in the case of omission failure. Then it clearly also works in the case of crash failures. However, $\mathcal{R}_F^{cr}$ and $\mathcal{R}_F^{om}$ are different systems. Thus, we have different decision pairs $(\mathcal{Z}^{cr}, \mathcal{O}^{cr})$ and $(\mathcal{Z}^{om}, \mathcal{O}^{om})$, depending on whether we consider crash failures or omission failures. Because there are more runs in $\mathcal{R}_F^{om}$ than $\mathcal{R}_F^{cr}$, it follows that $\mathcal{Z}_i^{cr} \subseteq \mathcal{Z}_i^{om}$ and $\mathcal{O}_i^{cr} \subseteq \mathcal{O}_i^{om}$. However, once we apply the procedure of Proposition 5.1 to these sets, we may no longer have, for example, $(\mathcal{Z}_i')^{cr} \subseteq (\mathcal{Z}_i')^{om}$.

Nevertheless, all the results in this section apply whether we are working with crash failures or omission failures, so we continue to suppress the failure mode here. However, as we shall see, the issue of the failure mode plays a major role in Section 6; starting with the same protocol, our techniques lead to different protocols depending on the type of failures considered.

6 Examples of optimal protocols

In this section, we apply the technique of Theorem 5.2 to construct some optimal agreement protocols. We start with a nontrivial agreement protocol $F(\mathcal{Z},\mathcal{O})$ and proceed as described in Section 5. Though each protocol constructed in this way is optimal, different choices of $\mathcal{Z}$ and $\mathcal{O}$ yield optimal protocols with different properties and different performance. Moreover, even if we start with the same basic protocol, the protocol we end up with depends on the failure mode considered.

6.1 A simple optimal protocol

A particularly trivial nontrivial agreement protocol is one in which no processor ever decides. Let F^Λ be the full-information protocol in which no processor ever decides. That is, we define $\mathcal{Z}_i^\Lambda = \mathcal{O}_i^\Lambda = \emptyset$ for $i = 1,\ldots,n$, and let $F^\Lambda = \mathit{FIP}(\mathcal{Z}^\Lambda,\mathcal{O}^\Lambda)$. Suppose we apply our optimization technique to F^Λ.

The first step of the construction consists of having the processors decide 0 as soon as possible, given the criterion for deciding 1. Thus,

$$\begin{aligned} \mathcal{Z}_i^{\Lambda,1} &= B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}^\Lambda}\exists 0) \text{ and} \\ \mathcal{O}_i^{\Lambda,1} &= B_i^{\mathcal{N}}(\exists 1 \wedge \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}^\Lambda}\exists 0). \end{aligned}$$

But since in F^Λ nonfaulty processors never decide 1, we have that $\boxdot(\mathcal{N}\wedge\mathcal{O}^\Lambda=\emptyset)$ is valid in $\mathcal{R}_{F^\Lambda}$, and hence so is $C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}^\Lambda}\exists 0$. It follows that $\mathcal{Z}_i^{\Lambda,1} = B_i^{\mathcal{N}}\exists 0$ and $\mathcal{O}_i^{\Lambda,1} = B_i^{\mathcal{N}}(\exists 1 \wedge \mathit{false})$, which is equivalent to $B_i^{\mathcal{N}}\mathit{false}$.

The second step of the construction optimizes the decision on 1, given the definition of $\mathcal{Z}^{\Lambda,1}$. Following Proposition 5.1, we define

$$\begin{aligned} \mathcal{Z}_i^{\Lambda,2} &= B_i^{\mathcal{N}}(\exists 0 \wedge \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}^{\Lambda,1}}\exists 1) \text{ and} \\ \mathcal{O}_i^{\Lambda,2} &= B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}^{\Lambda,1}}\exists 1). \end{aligned}$$

We denote $\mathit{FIP}(\mathcal{Z}^{\Lambda,2},\mathcal{O}^{\Lambda,2})$ by $F^{\Lambda,2}$.

The analysis we have performed so far applies equally well to both the crash and the omissions failure mode. Thus, $F^{\Lambda,2}$ is an optimal nontrivial agreement protocol in both cases. As we shall see, however, while F^Λ behaves in essentially the same way in both settings, the properties of $F^{\Lambda,2}$ are dramatically different in the two failure modes. The crux of the analysis will involve figuring out when $\mathcal{Z}^{\Lambda,2}$ and $\mathcal{O}^{\Lambda,2}$ hold in runs of $F^{\Lambda,2}$. This, in turn, will be determined by when $C^{\boxdot}_{\mathcal{N} \wedge B^{\mathcal{N}} \exists 0} \exists 1$ $(= C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}} \exists 1)$ holds.

We begin with an analysis of $F^{\Lambda,2}$ in the crash failure mode. As the next theorem shows, the protocol $F^{\Lambda,2}$ is quite simple in this case. Let $\mathcal{Z}_i^{cr} = B_i^{\mathcal{N}} \exists 0$ and let $\mathcal{O}_i^{cr} = B_i^{\mathcal{N}}((\mathcal{N} \wedge \mathcal{Z}^{cr}) = \emptyset)$.

Theorem 6.1: *In the crash failure mode,* $F^{\Lambda,2} = FIP(\mathcal{Z}^{cr}, \mathcal{O}^{cr})$.

Proof: See the appendix. ∎

It is easy for processor i to compute when it should decide 0 according to $F^{\Lambda,2}$ in the crash failure mode: it decides 0 if it ever hears about a 0 from any processor. It also turns out to be easy for processor i to compute when to decide 1. According to Theorem 6.1, it should decide 1 when it believes that $\mathcal{N} \wedge \mathcal{Z}^{cr} = \emptyset$, that is, when it believes that no nonfaulty processor knows that some processor started with 0. The relevant conditions for this to be the case are precisely those given for the protocol $P0_{\mathrm{opt}}$ described in Section 2.2. As we show in the proof of the next theorem, it happens when either processor i knows that all initial values are 1, or it hears from the same set of processors in two consecutive rounds and still does not know that some processor had an initial value of 0. Thus, $P0_{\mathrm{opt}}$ and $F^{\Lambda,2}$ are essentially equivalent protocols: The same decisions are made by all processors at corresponding points of the systems generated by the two protocols in the crash failure mode. (The two protocols are not identical because $F^{\Lambda,2}$ is a full-information protocol, while in $P0_{\mathrm{opt}}$ the processors send much shorter messages.) It follows from this that not only is $F^{\Lambda,2}$ an optimal nontrivial agreement protocol (this already follows from Theorem 5.2), it is an optimal EBA protocol.

Theorem 6.2: *In the crash failure mode, the same decisions are made by nonfaulty processors at corresponding points of the protocols* $P0_{\mathrm{opt}}$ *and* $F^{\Lambda,2}$. *Thus, both* $F^{\Lambda,2}$ *and* $P0_{\mathrm{opt}}$ *are optimal EBA protocols for the crash failure mode.*

Proof: See the appendix. ∎

Since $P0_{\mathrm{opt}}$ can be implemented using messages of linear size, $F^{\Lambda,2}$ gives us an efficient optimal EBA protocol in the crash failure mode. The situation is very different in the case of omission failures. While $F^{\Lambda,2}$ is still an optimal nontrivial agreement protocol, and an anlysis similar to that carried out for the case of crash failures can be used to show that it has an efficient implementation, it is no longer an EBA protocol. There are runs in which it never halts.

Proposition 6.3: *If* $t > 1$ *and* $n \geq t + 2$, *then there are runs of* $F^{\Lambda,2}$ *in the omissions failure mode in which the nonfaulty processors do not decide.*

Proof: See the appendix. ∎

6.2 Optimal EBA for omission failures

As the proof of Proposition 6.3 shows, the reason that $F^{\Lambda,2}$ does not necessarily terminate in the presence of omission failures is that there is no bound on the time at which a processor can learn that there exists an initial value of 0. Clearly, if we start out with a terminating protocol and find an optimal protocol that dominates it, then the optimal protocol we obtain will also be terminating. We now use a well-known approach to generate a (terminating) EBA protocol in the omissions failure mode. Intuitively, we say that a processor *accepts* 0 in round m only if this value was transferred by a chain of $m-1$ distinct processors (cf. [DS82]). Formally, we say that a *0-chain* exists at the point (r, m) of a full-information protocol F iff there is a sequence of m distinct processors $i_1, \ldots, i_m$, such that i_1 has initial value 0, i_{k+1} received a message from i_k at round k and $(\mathcal{R}_F, r, k) \models \neg B^{\mathcal{N}}_{i_{k+1}}(i_k \notin \mathcal{N})$, and i_m is nonfaulty. We say that $(\mathcal{R}_F, r, m) \models \exists 0^*$ if there is a 0-chain at some point (r, m') with $m' \leq m$.

Let $\mathcal{Z}^0_i = B^{\mathcal{N}}_i \exists 0^*$ and $\mathcal{O}^0_i = B^{\mathcal{N}}_i \neg\exists 0^*$, and consider the protocol $\mathit{FIP}(\mathcal{Z}^0, \mathcal{O}^0)$. It is easy to see that this is nontrivial agreement protocol. The following proposition shows that it is actually an EBA protocol.

Proposition 6.4: *In a run r of $\mathit{FIP}(\mathcal{Z}^0, \mathcal{O}^0)$ in the omission failure mode where f processors actually fail, all nonfaulty processors decide by time $f + 1$.*

Proof: See the appendix. ∎

Clearly $\mathit{FIP}(\mathcal{Z}^0, \mathcal{O}^0)$ satisfies the validity and agreement conditions, so we immediately get the following corollary.

Corollary 6.5: *$\mathit{FIP}(\mathcal{Z}^0, \mathcal{O}^0)$ is an EBA protocol.*

We can now apply the construction of Theorem 5.2 to $\mathit{FIP}(\mathcal{Z}^0, \mathcal{O}^0)$ to get an optimal EBA protocol that dominates it. We can simplify the description of the protocol that we get using the construction somewhat. Let $\mathcal{Z}^* = B^{\mathcal{N}}_i(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}^0}\exists 0)$, $\mathcal{O}^* = B^{\mathcal{N}}_i(\exists 1 \wedge \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}^0}\exists 0)$, and $F^* = \mathit{FIP}(\mathcal{Z}^*, \mathcal{O}^*)$.

Proposition 6.6: *F^* is an optimal EBA protocol in the omissions failure mode that dominates $\mathit{FIP}(\mathcal{Z}^0, \mathcal{O}^0)$.*

Proof: See the appendix. ∎

Note that the decision rules in F^* involve explicit tests for knowledge. Although it is not hard to show that these tests can be implemented effectively (by which we mean that the tests are decidable in principle; in fact, this can be done in PSPACE [MT88]), we do not know if they can be implemented efficiently. The question of whether there exists a polynomial-time protocol that is optimal for EBA in the case of omission failures is still open; we conjecture that such a protocol does exist. (We remark that in [DM90, MT88], polynomial time protocols that are optimum for SBA in the case of crash and omission failures are given.)

7 Conclusions

This paper completely characterizes optimal EBA protocols in the case of crash and omission failures. The characterization involves a new variant of common knowledge—*continual common knowledge*—in an essential way. Interestingly, continual common knowledge is a stronger state of knowledge than common knowledge. By contrast, all other variants of common knowledge considered in the literature (cf. [HM90, PT92]) are weakenings of common knowledge. The characterization is far from trivial. While it would in principle be possible to characterize and prove these properties without using knowledge, we conjecture that it would be rather difficult.

Our results can be extended in a number of ways. First, although we have considered here only crash and omission failures, we believe that our results can be extended to the case of Byzantine failures. Second, although we assumed here that the processors are synchronous, our knowledge analysis is also valid for asynchronous systems (except for the implementations in Section 6, of course). Third, although this paper focuses on EBA and nontrivial agreement protocols, it is straightforward to extend our results to general coordination problems along the lines of [MT88], including ones in which *all* processors (and not only the nonfaulty ones) are required to act consistently. See [Nei90, NB92] for a discussion of the latter issue. Finally, an interesting question is whether our two-step optimizing process can be done in a computationally efficient manner. Our examples in Section 6 show that in some cases we can obtain efficient optimal EBA protocols. But evaluating the formulas used in the optimizing process in a naive manner does not guarantee computationally efficient results.

A Appendix: Proofs

In this appendix, we prove all the results stated in the main text. For the convenience of the reader, we repeat the statements of the results.

A.1 Proofs for Section 2

Proposition 2.2: *Let P be an arbitrary protocol in a system with omission failures. Then for each processor i there is a function f_i from i's state in a full-information protocol F to its state in P, such that for every pair (r,m) and (r',m) of corresponding points of F and P, we have $f_i(r_i(m)) = r'_i(m)$.*

Proof: Define $f(r_i(m)) = r'_i(m)$. Clearly, this function has the required properties. We just need to show that it is well defined; that is, if $r_i(m) = s_i(m)$ and (s',m) corresponds to (s,m), then we must show that $r'_i(m) = s'_i(m)$. We proceed by induction on m.

Clearly if $r_i(0) = s_i(0)$ then $r'_i(0) = s'_i(0)$, since i has the same initial state in corresponding runs. Suppose we have proved the result for m and suppose that $r_i(m+1) =$

$s_i(m+1)$. Since r and s are runs of a full-information protocol, we must also have $r_i(m) = s_i(m)$. Moreover, we must have $r_j(m) = s_j(m)$ for all the processors j from which i receives a message in round $m+1$ of r, and these are the same processors from which it receives a message in round $m+1$ of s. Thus, by the induction hypothesis, $r'_i(m) = s'_i(m)$ and $r'_j(m) = s'_j(m)$ for all processors from which i receives a message in round $m+1$ of r. Since r' corresponds to r and s' corresponds to s, the same processors fail in round $m+1$ of r' (resp., s') as in round $m+1$ of r (resp., s). Thus, i receives messages from the same processors in round $m+1$ of r' as in round $m+1$ of s'; moreover, these messages must be the same (since the message generation function μ_{ij} for protocol P depends only on the states of the processors at time m, and these are the same in r' and s'). Since the state transition function δ_i for protocol P depends only on the messages received and i's state, it follows that $r'_i(m+1) = s'_i(m+1)$, as desired. ∎

A.2 Proofs for Section 4

Proposition 4.3: *Let P be a nontrivial agreement protocol with decision pair $(\mathcal{Z}, \mathcal{O})$. Then*

(a) $\mathcal{R}_P \models \mathit{decide}_i(0) \Rightarrow B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}} \exists 0 \wedge \neg \mathit{decide}_i(1))$

(b) $\mathcal{R}_P \models \mathit{decide}_i(1) \Rightarrow B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}} \exists 1 \wedge \neg \mathit{decide}_i(0))$.

Proposition 4.3 follows immediately from the following two lemmas.

Lemma A.1: *Let P be an agreement protocol with decision pair $(\mathcal{Z}, \mathcal{O})$. Then*

(a) $\mathcal{R}_P \models \mathit{decide}_i(0) \Rightarrow B_i^{\mathcal{N}}(C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}} \exists 0 \wedge \neg \mathit{decide}_i(1))$

(b) $\mathcal{R}_P \models \mathit{decide}_i(1) \Rightarrow B_i^{\mathcal{N}}(C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}} \exists 1 \wedge \neg \mathit{decide}_i(0))$.

Proof: We prove only part (a) here; the proof of part (b) is completely symmetric. Suppose $(\mathcal{R}_P, r, m) \models \mathit{decide}_i(0)$. We want to show that $(\mathcal{R}_P, r, m) \models B_i^{\mathcal{N}}(C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}} \exists 0 \wedge \neg \mathit{decide}_i(1))$. Let (r', m) be a point such that $r'_i(m) = r_i(m)$ and $i \in \mathcal{N}(r', m)$. Since $r'_i(m) = r_i(m)$ we have by Proposition 4.1 that $(\mathcal{R}_P, r', m) \models \mathit{decide}_i(0)$. Lemma 4.2 implies that $(\mathcal{R}_P, r', m) \models \boxdot \neg \mathit{decide}_j(1)$ for every nonfaulty processor j. Thus, for all $m' \geq 0$, we have that $(\mathcal{N} \wedge \mathcal{O})(r', m') = \emptyset$. By definition of continual common knowledge, this immediately implies that $C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}} \exists 0$ holds at (r', m). It follows that

$$(\mathcal{R}_P, r, m) \models \mathit{decide}_i(0) \Rightarrow B_i^{\mathcal{N}}(C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}} \exists 0 \wedge \neg \mathit{decide}_i(1))$$

as desired, and we are done. ∎

Lemma A.2: *A protocol P satisfies weak validity iff*

(a) $\mathcal{R}_P \models decide_i(0) \Rightarrow B_i^{\mathcal{N}} \exists 0$

(b) $\mathcal{R}_P \models decide_i(1) \Rightarrow B_i^{\mathcal{N}} \exists 1$.

Proof: Suppose that P satisfies weak validity and $(\mathcal{R}_P, r, m) \models decide_i(0)$. We want to show that $(\mathcal{R}_P, r, m) \models B_i^{\mathcal{N}} \exists 0$. There are two cases to consider. If $(\mathcal{R}_P, r, m) \models K_i(i \notin \mathcal{N})$ then, by definition, $(\mathcal{R}_P, r, m) \models B_i^{\mathcal{N}} \varphi$ for every formula φ, and hence, in particular, $(\mathcal{R}_P, r, m) \models B_i^{\mathcal{N}} \exists 0$. For the other case, assume that $(\mathcal{R}_P, r, m) \models \neg K_i(i \notin \mathcal{N})$. Moreover, assume by way of contradiction that $(\mathcal{R}_P, r, m) \models \neg B_i^{\mathcal{N}} \exists 0$. Then there exists some (r', m') such that $i \in \mathcal{N}(r', m')$, $r_i(m) = r'_i(m')$, and $(\mathcal{R}_P, r', m') \models \neg \exists 0$. Since $(\mathcal{R}_P, r, m) \models decide_i(0)$ and $r_i(m) = r'_i(m')$, we have that $(\mathcal{R}_P, r', m') \models decide_i(0)$, which contradicts the assumed weak validity of P. This proves (a). A similar argument shows that (b) holds.

For the converse, suppose that (a) and (b) hold. Then it is immediate that weak validity holds. We leave details to the reader. ∎

We next want to prove Proposition 4.4, which gives a sufficient condition for nontrivial agreement in terms of continual common knowledge. We first need the following technical lemma.

Lemma A.3: *Let P be a protocol with decision pair $(\mathcal{Z}, \mathcal{O})$. Then*

(a) $\mathcal{R}_P \models (i \in \mathcal{N} \wedge decide_i(0) \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1) \Rightarrow B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1)$

(b) $\mathcal{R}_P \models (i \in \mathcal{N} \wedge decide_i(0) \wedge \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1) \Rightarrow B_i^{\mathcal{N}} \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1$

(c) $\mathcal{R}_P \models (i \in \mathcal{N} \wedge decide_i(1) \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0) \Rightarrow B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0)$

(d) $\mathcal{R}_P \models (i \in \mathcal{N} \wedge decide_i(1) \wedge \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0) \Rightarrow B_i^{\mathcal{N}} \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0$.

Proof: First we prove part (a). Let (r, m) be a point in $\mathcal{R}_P$ such that $(\mathcal{R}_P, r, m) \models i \in \mathcal{N} \wedge decide_i(0) \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1$. From the fixed-point axiom (part (e) of Lemma 3.4), we get

$$(\mathcal{R}_P, r, m) \models B_i^{\mathcal{N} \wedge \mathcal{Z}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1).$$

Substituting the definition of $B_i^{\mathcal{N} \wedge \mathcal{Z}}$ we have

$$(\mathcal{R}_P, r, m) \models K_i((decide_i(0) \wedge i \in \mathcal{N}) \Rightarrow (\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1))$$

or, equivalently,

$$(\mathcal{R}_P, r, m) \models K_i(decide_i(0) \Rightarrow (i \in \mathcal{N} \Rightarrow (\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1))).$$

Since $(\mathcal{R}, r, m) \models decide_i(0)$, by assumption and Proposition 4.1, we have that $\mathcal{R}_P \models decide_i(0) \Leftrightarrow K_i decide_i(0)$. Applying the distribution axiom for K_i (part (b) of Proposition 3.1), we get

$$(\mathcal{R}_P, r, m) \models K_i(i \in \mathcal{N} \Rightarrow (\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1))$$

or, equivalently,

$$(\mathcal{R}_P, r, m) \models B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1).$$

For part (b), let (r, m) be a point in $\mathcal{R}_P$ such that $(\mathcal{R}_P, r, m) \models i \in \mathcal{N} \wedge \mathit{decide}_i(0) \wedge \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1$. By the negative introspection axiom for continual common knowledge (part (d) of Lemma 3.4), we get

$$(\mathcal{R}_P, r, m) \models C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1.$$

The rest of the proof follows the same lines as that of part (a).

The proofs of parts (c) and (d) are similar and are omitted here. ∎

We can now prove Proposition 4.4.

Proposition 4.4: *Let P be a protocol with decision pair $(\mathcal{Z}, \mathcal{O})$ such that either:*

(a) $\mathcal{R}_P \models \mathit{decide}_i(0) \Rightarrow B_i^{\mathcal{N}} \exists 0$ *and*

(b) $\mathcal{R}_P \models \mathit{decide}_i(1) \Leftrightarrow B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1)$

or

(a′) $\mathcal{R}_P \models \mathit{decide}_i(0) \Leftrightarrow B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0)$ *and*

(b′) $\mathcal{R}_P \models \mathit{decide}_i(1) \Rightarrow B_i^{\mathcal{N}} \exists 1.$

Then P is a nontrivial agreement protocol.

Proof: We prove the result under the assumption that (a) and (b) hold. The proof in the case that (a′) and (b′) hold is similar.

The fact that P satisfies weak validity follows directly from Lemma A.2. Suppose, by way of contradiction, that P does not satisfy weak agreement. Then there is a run r of $\mathcal{R}_P$ and there are at least two nonfaulty processors, say i and j, which decide 1 and 0 respectively in r. Since i decides 1, there must be some point (r, m) such that $(\mathcal{R}_P, r, m) \models \mathit{decide}_i(1)$. By assumption (b) in the statement of the proposition, we have that $(\mathcal{R}_P, r, m) \models B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1)$. Since i is nonfaulty in r, it follows easily from Proposition 3.2 that $(\mathcal{R}_P, r, m) \models C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1$. Since j decides 0 in r, there must be some point (r, m') such that $(\mathcal{R}_P, r, m') \models \mathit{decide}_j(0)$. Since $(\mathcal{R}_P, r, m) \models C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1$, by part (g) of Lemma 3.4, we have $(\mathcal{R}_P, r, m') \models C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1$. Thus, from Lemma A.3 it follows that

$$(\mathcal{R}_P, r, m') \models B_j^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1).$$

But then it follows from assumption (b) on P that $(\mathcal{R}_P, r, m') \models \mathit{decide}_j(1)$. Thus, $(\mathcal{R}_P, r, m') \models \mathit{decide}_j(0) \wedge \mathit{decide}_j(1)$, contradicting Proposition 4.1(i). It follows that P satisfies weak agreement. ∎

A.3 Proofs for Section 5

We next want to prove Proposition 5.1. We first need to define the notion of *corresponding formulas*. Given two protocols F and F', we say that the formula φ *corresponds* in F and F' if, whenever (r, m) and (r', m) are corresponding points of F and F', we have $(\mathcal{R}_F^{om}, r, m) \models \varphi$ iff $(\mathcal{R}_{F'}, r', m) \models \varphi$. Similarly, we say the nonrigid set $\mathcal{S}$ *corresponds* in F and F' if, whenever (r, m) and (r', m) are corresponding points of F and F', we have $\mathcal{S}(r, m) = \mathcal{S}(r', m)$.

Proposition A.4: *Let F and F' be full-information protocols, and suppose that φ and φ' are corresponding formulas in F and F', while $\mathcal{S}$ is a nonrigid set that corresponds in F and F'. Then $\varphi \wedge \varphi'$, $\neg\varphi$, $K_i\varphi$, $B_i^{\mathcal{S}}\varphi$, and $C_{\mathcal{S}}^{\boxdot}\varphi$ all correspond in F and F'.*

Proof: If φ and φ' correspond in F and F', then straightforward propositional reasoning implies that $\varphi \wedge \varphi'$ and $\neg\varphi$ correspond as well.

For $K_i\varphi$, suppose $(\mathcal{R}_F, r, m) \models K_i\varphi$ and (s, m) is the point in $\mathcal{R}_{F'}$ corresponding to (r, m). We want to show $(\mathcal{R}_{F'}, s, m) \models K_i\varphi$. So suppose $s'_i(m) = s_i(m)$. (Notice that since F is a full-information protocol, if $s_i(m) = s'_i(m')$, we must have $m = m'$.) Let (r', m) be the point in $\mathcal{R}_F$ corresponding to (s', m). Since F and F' are full-information protocols, we must have $s_i(m) = r_i(m)$ and $s'_i(m) = r'_i(m)$. Thus, $r'_i(m) = r_i(m)$, and since $(\mathcal{R}_F, r, m) \models K_i\varphi$, we have $(\mathcal{R}_F, r', m) \models \varphi$. Since φ corresponds with respect to F and F', we have $(\mathcal{R}_{F'}, s', m) \models \varphi$. It follows that $(\mathcal{R}_{F'}, s, m) \models K_i\varphi$. The proof that $(\mathcal{R}_{F'}, s, m) \models K_i\varphi$ implies $(\mathcal{R}_F, r, m) \models K_i\varphi$ is identical.

Since (a) $B_i^{\mathcal{S}}\varphi$ is equivalent to $K_i(i \in \mathcal{S} \Rightarrow \varphi)$, (b) $i \in \mathcal{S} \Rightarrow \varphi$ is equivalent to $\neg(i \in \mathcal{S} \wedge \neg\varphi)$, and (c) $i \in \mathcal{S}$ clearly corresponds in F and F' since the nonrigid set $\mathcal{S}$ does, it follows immediately from our previous results that $B_i^{\mathcal{S}}\varphi$ corresponds in F and F'.

Applying the previous results, we can easily show by induction on k that $(E_{\mathcal{S}}^{\boxdot})^k\varphi$ corresponds in F and F', for each k. It immediately follows that $C_{\mathcal{S}}^{\boxdot}\varphi$ does too. ∎

Treated as defining a nonrigid set, a decision set $\mathcal{A}$ clearly corresponds in all full-information protocols. We thus obtain the following corollary.

Corollary A.5: *Let F and F' be full-information protocols and let $\mathcal{A}$ be a decision set. Then $C_{\mathcal{N}\wedge\mathcal{A}}^{\boxdot}\exists 1$ and $C_{\mathcal{N}\wedge\mathcal{A}}^{\boxdot}\exists 0$ correspond in F and F'.*

We can now prove Proposition 5.1.

Proposition 5.1: *Let $F = FIP(\mathcal{Z}, \mathcal{O})$ be a full-information nontrivial agreement protocol, and define $\mathcal{Z}', \mathcal{O}', \mathcal{Z}'', \mathcal{O}''$ as follows:*

- $\mathcal{Z}'_i = B_i^{\mathcal{N}}(\exists 0 \wedge C_{\mathcal{N}\wedge\mathcal{O}}^{\boxdot}\exists 0)$

- $\mathcal{O}'_i = B_i^{\mathcal{N}}(\exists 1 \wedge \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0)$
- $\mathcal{Z}''_i = B_i^{\mathcal{N}}(\exists 0 \wedge \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}}\exists 1)$
- $\mathcal{O}''_i = B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}}\exists 1)$.

Then $F' = FIP(\mathcal{Z}', \mathcal{O}')$ and $F'' = FIP(\mathcal{Z}'', \mathcal{O}'')$ are both nontrivial agreement protocols that dominate F.

Proof: We prove the result for F'; the proof for F'' is analogous. Since $B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0)$ and $B_i^{\mathcal{N}}(\exists 1 \wedge \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0)$ correspond in F and F', it follows from Proposition 4.4 that F' is a nontrivial agreement protocol. We now show that F' dominates F.

Let r be a run of $\mathcal{R}_F$ and let r' be the corresponding run in $\mathcal{R}_{F'}$. First suppose that $(\mathcal{R}_F, r, m) \models decide_i(1) \wedge (i \in \mathcal{N})$. We want to show that processor i decides by time m in r'. Clearly, either (1) $(\mathcal{R}_F, r, m) \models C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0$ or (2) $(\mathcal{R}_F, r, m) \models \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0$. If (1) holds, Lemma A.3 implies

$$(\mathcal{R}_F, r, m) \models B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0).$$

Thus, by definition of $\mathcal{Z}'$,we have that $r_i(m) \in \mathcal{Z}'_i$. If (2) holds, a similar argument shows that $(\mathcal{R}_F, r, m) \models B_i^{\mathcal{N}} \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0$. In addition, since F is a nontrivial agreement protocol, it follows from Lemma A.2 that $(\mathcal{R}_F, r, m) \models B_i^{\mathcal{N}}\exists 1$. Thus, $r_i(m) \in \mathcal{O}'_i$. In either case, we see that processor i decides at least as soon in r' as in r.

Now suppose that $(\mathcal{R}_F, r, m) \models decide_i(0) \wedge i \in \mathcal{N}$. Then by Proposition 4.3, we also have $(\mathcal{R}_F, r, m) \models B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0)$. Therefore, by definition, $r_i(m) \in \mathcal{Z}'_i$, and again it follows that processor i decides at least as soon in r' as in r. ∎

To prove Theorem 5.2, we need a technical lemma that relates the stopping conditions of two protocols F and F' where F' dominates F. It describes the state of knowledge that must hold in a run of F' in which the decision value is different from that of the corresponding run of F.

Lemma A.6: *Let $F = FIP(\mathcal{Z}, \mathcal{O})$ and $F' = FIP(\mathcal{Z}', \mathcal{O}')$ be full-information nontrivial agreement protocols such that F' dominates F. Then we must have*

(a) $\mathcal{R}_{F'} \models decide_i(0) \Rightarrow B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0)$

(b) $\mathcal{R}_{F'} \models decide_i(1) \Rightarrow B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}}\exists 1)$.

Proof: The proofs of (a) and (b) are identical, so we prove only (a). The fact that $\mathcal{R}_{F'} \models decide_i(0) \Rightarrow B_i^{\mathcal{N}}\exists 0$ follows from Lemma A.2, since F' is a nontrivial agreement protocol. So it suffices to show that

$$\mathcal{R}_{F'} \models decide_i(0) \Rightarrow B_i^{\mathcal{N}} C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}(\exists 0).$$

Proposition 4.3 shows that $\mathcal{R}_{F'} \models decide_i(0) \Rightarrow B_i^{\mathcal{N}} C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}'}(\exists 0)$. We need to show that the same result holds with $\mathcal{O}'$ replaced by $\mathcal{O}$.

Let $deciding(0)$ denote the basic fact that is true at a point (r, m) in $\mathcal{R}_{F'}$ exactly if some nonfaulty processor decides 0 at some point in r. It is clearly the case that $\mathcal{R}_{F'} \models decide_i(0) \Rightarrow B_i^{\mathcal{N}} deciding(0)$, since if $decide_i(0)$ holds then i knows $deciding(0)$, and thus i knows that if i is nonfaulty then $deciding(0)$ holds, that is, $B_i^{\mathcal{N}} deciding(0)$ holds. To complete the proof, it remains to prove the following:

Claim: $\mathcal{R}_{F'} \models deciding(0) \Rightarrow C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0$.

Proof: Since F' satisfies weak validity, it is easy to check that $\mathcal{R}_{F'} \models deciding(0) \Rightarrow \exists 0$. It therefore suffices to prove that $\mathcal{R}_{F'} \models deciding(0) \Rightarrow C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}} deciding(0)$. Using the induction rule (part (f) of Lemma 3.4), it suffices to show that

$$\mathcal{R}_{F'} \models deciding(0) \Rightarrow E^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}} deciding(0).$$

To see this, assume that $(\mathcal{R}_{F'}, r', m) \models deciding(0)$ and $j \in (\mathcal{N} \wedge \mathcal{O})(r', m')$ for some point (r', m') in the run r'. It suffices to show that $(\mathcal{R}_{F'}, r', m') \models B_j^{\mathcal{N}} deciding(0)$. Let r be the run of $\mathcal{R}_F$ that corresponds to r'. Since $F = FIP(\mathcal{Z}, \mathcal{O})$, if processor j decides in run r at all, it does so at the first time m such that $r_j(m) \in \mathcal{O}_j$. Since $j \in (\mathcal{N}\wedge\mathcal{O})(r', m')$, it follows that $r'_j(m') = r_j(m') \in \mathcal{O}_j$. Thus, processor j decides no later than at time m' in run r. Since F' dominates F, processor j must decide no later than at time m' in run r' either. Since j is nonfaulty in r', and the nonfaulty processors decide 0 in r', it follows that j decides 0 in r' at some time $m'' \leq m'$. Thus, we have $(\mathcal{R}_{F'}, r', m'') \models decide_j(0)$, from which it follows that $(\mathcal{R}_{F'}, r', m'') \models B_j^{\mathcal{N}} deciding(0)$. Finally, s in a full-information protocol, processors keep track of their history in their local state, it follows that once processor j believes that $deciding(0)$ holds, it will continue to believe this at all points in the future. (Technically, the property of *no forgetting* or *perfect recall* holds in full-information protocols [HV89].) Thus, $(\mathcal{R}_{F'}, r', m') \models B_j^{\mathcal{N}} deciding(0)$, as desired. ∎

We are now ready to prove that our construction yields an optimal protocol.

Theorem 5.2: *Let $F = FIP(\mathcal{Z}, \mathcal{O})$ be a full-information nontrivial agreement protocol, let $F^1 = FIP(\mathcal{Z}^1, \mathcal{O}^1)$, and let $F^2 = FIP(\mathcal{Z}^2, \mathcal{O}^2)$. Then F^2 is an optimal nontrivial agreement protocol. Moreover, if F is an EBA protocol, then F^2 is an optimal EBA protocol dominating F.*

Proof: Suppose that F^2 is not optimal and that F' dominates F^2. Proposition 5.1 implies that F^2 dominates both F and F^1. Therefore, F' must also dominate both F and F^1. By Lemma A.6, if processor i decides 0 at the point (r', m) of $\mathcal{R}_{F'}$, it must be the case that $(\mathcal{R}_{F'}, r', m) \models B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0)$. Thus, by definition of $\mathcal{Z}_i^1$, processor i decides at the corresponding point of $\mathcal{R}_{F^1}$ (if it has not done so earlier in the run). Since F^2 dominates F^1, it must be the case that processor i also has decided by the corresponding point of $\mathcal{R}_{F^2}$. Similarly, since F' dominates F^1, if processor i decides 1 at the point (r', m), then $(\mathcal{R}_{F'}, r', m) \models B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}^1}\exists 1)$. By definition of $\mathcal{O}_i^2$, it must be the case that processor i has decided by the corresponding point in $\mathcal{R}_{F^2}$. We conclude that F^2 dominates F'. Since F' was chosen as an arbitrary protocol dominating F^2, it follows that F^2 is indeed optimal.

Notice that, in general, F^2 is not necessarily an EBA protocol, because not all the nonfaulty processors necessarily decide. But if the protocol F we started out with is an EBA protocol, then every protocol dominating F must satisfy the decision condition of EBA. In this case, then, F^2 will in fact be an optimal EBA protocol dominating F. ∎

We now prove Theorem 5.3, which characterizes optimal EBA protocols in terms of continual common knowledge.

Theorem 5.3: *Let $F = FIP(\mathcal{Z}, \mathcal{O})$ be a full-information nontrivial agreement protocol. Then F is optimal iff both of the following conditions hold:*

(a) $\mathcal{R}_F \models i \in \mathcal{N} \Rightarrow (decide_i(0) \Leftrightarrow B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0 \wedge \neg decide_i(1)))$

(b) $\mathcal{R}_F \models i \in \mathcal{N} \Rightarrow (decide_i(1) \Leftrightarrow B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}} \exists 1 \wedge \neg decide_i(0)))$.

Proof: For the "only if" direction, we prove only (a) here; the proof of (b) is analogous. Since F is a nontrivial agreement protocol, it follows from Proposition 4.3 that

$$\mathcal{R}_F \models decide_i(0) \Rightarrow B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0 \wedge \neg decide_i(1)).$$

Hence it is enough to prove that

$$\mathcal{R}_F \models i \in \mathcal{N} \Rightarrow (B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0 \wedge \neg decide_i(1)) \Rightarrow decide_i(0)).$$

Suppose that

$$(\mathcal{R}_F, r, m) \models i \in \mathcal{N} \wedge B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0 \wedge \neg decide_i(1)).$$

Let $\mathcal{Z}'_i = B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0)$ and $\mathcal{O}'_i = B_i^{\mathcal{N}}(\exists 1 \wedge \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0)$, and consider the protocol F' defined by $F' = FIP(\mathcal{Z}', \mathcal{O}')$. Proposition 5.1 shows that F' is a nontrivial agreement protocol that dominates F. Therefore, in order for F to be optimal, it must dominate F'.

Let (r', m) be the point of $\mathcal{R}_{F'}$ corresponding to (r, m). Since $(\mathcal{R}_F, r, m) \models B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{O}} \exists 0 \wedge \neg decide_i(1))$, we have that $r_i(m) \in \mathcal{Z}'_i$. Since $r_i(m) = r'_i(m)$, it follows that $r'_i(m) \in \mathcal{Z}'_i$, so that processor i decides or has decided by (r', m) in protocol F'. Since F is optimal by assumption, we have that processor i must also decide by (r, m). Thus, $(\mathcal{R}_F, r, m) \models decide_i(0) \vee decide_i(1)$. But since $(\mathcal{R}_F, r, m) \models i \in \mathcal{N} \wedge B_i^{\mathcal{N}} \neg decide_i(1)$, it follows by Lemma A.1 that $(\mathcal{R}_F, r, m) \models \neg decide_i(1)$, and hence $(\mathcal{R}_F, r, m) \models decide_i(0)$ as desired.

For the "if" direction, suppose that both (a) and (b) hold. Proposition 2.3 implies that it is enough to show that F dominates any full-information nontrivial agreement protocol F' that dominates it. So suppose F' dominates F. Let r' be a run of $\mathcal{R}_{F'}$ and let r be the corresponding run in $\mathcal{R}_F$. First suppose that $(\mathcal{R}_{F'}, r', m) \models i \in \mathcal{N} \wedge \mathit{decide}_i(0)$. We want to show that processor i decides by time m in r. The proof that the same happens if $\mathit{decide}_i(1)$ holds is identical, and hence is omitted. By Lemma A.6, we have

$$(\mathcal{R}_{F'}, r', m) \models B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0).$$

Proposition A.4 and Corollary A.5 imply that

$$(\mathcal{R}_F, r, m) \models B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0).$$

If, in addition, $(\mathcal{R}_F, r, m) \models \mathit{decide}_i(1)$, then processor i decides by time m in run r. On the other hand, if $(\mathcal{R}_F, r, m) \models \neg\mathit{decide}_i(1)$, then Proposition 4.1 implies that $\mathcal{R}_F \models \neg\mathit{decide}_i(1) \Rightarrow B_i^{\mathcal{N}}\neg\mathit{decide}_i(1)$, so

$$(\mathcal{R}_F, r, m) \models B_i^{\mathcal{N}}(\exists 0 \wedge C^{\boxdot}_{\mathcal{N}\wedge\mathcal{O}}\exists 0 \wedge \neg\mathit{decide}_i(1)).$$

From assumption (a), it follows that $(\mathcal{R}_F, r, m) \models \mathit{decide}_i(0)$, and thus we again get that processor i decides by time m in run r. ∎

A.4 Proofs for Section 6.1

Theorem 6.1: *In the crash failure mode, $F^{\Lambda,2} = \mathit{FIP}(\mathcal{Z}^{cr}, \mathcal{O}^{cr})$.*

Theorem 6.1 follows immediately from the following two lemmas. The first shows that $\mathcal{Z}_i^{\Lambda,2} = B_i^{\mathcal{N}}\exists 0$ and the second shows that $\mathcal{O}_i^{\Lambda,2} = B_i^{\mathcal{N}}((\mathcal{N} \wedge \mathcal{Z}^{cr}) = \emptyset)$. This shows that $\mathcal{Z}^{\Lambda,2} = \mathcal{Z}^{cr}$ and $\mathcal{O}^{\Lambda,2} = \mathcal{O}^{cr}$, proving the theorem. To simplify notation in the proofs, we denote by $\mathcal{R}^{cr}$ the system $\mathcal{R}^{cr}_{F^{\Lambda,2}}$ of runs of $F^{\Lambda,2}$ in the crash failure mode by $\mathcal{R}^{cr}$.

Lemma A.7: *In $\mathcal{R}^{cr}$, $\mathcal{Z}_i^{\Lambda,2} = B_i^{\mathcal{N}}\exists 0$.*

Proof: Recall that $\mathcal{Z}_i^{\Lambda,2} = B_i^{\mathcal{N}}(\exists 0 \wedge \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}^{\Lambda,1}}\exists 1)$ and that $\mathcal{Z}_i^{\Lambda,1} = B_i^{\mathcal{N}}\exists 0$. Clearly $\mathcal{Z}_i^{\Lambda,2} \subseteq B_i^{\mathcal{N}}\exists 0$. For the opposite containment, suppose $(\mathcal{R}^{cr}, r, m) \models B_i^{\mathcal{N}}\exists 0$. We want to show that $(\mathcal{R}^{cr}, r, m) \models B_i^{\mathcal{N}}(\exists 0 \wedge \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}^{\Lambda,1}}\exists 1)$. So suppose that $r_i(m) = r'_i(m)$ and i is nonfaulty in r'. We must show that $(\mathcal{R}^{cr}, r', m) \models \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}^{\Lambda,1}}\exists 1$. Since $(\mathcal{R}^{cr}, r', m) \models i \in \mathcal{N} \wedge B_i^{\mathcal{N}}\exists 0$, it certainly suffices to show that

$$\mathcal{R}^{cr} \models (i \in \mathcal{N} \wedge B_i^{\mathcal{N}}\exists 0) \Rightarrow \neg C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}^{\Lambda,1}}\exists 1.$$

To show this, we prove by induction on k that for all runs r' of $\mathcal{R}^{cr}$ and processors i that are nonfaulty in r', if $(\mathcal{R}^{cr}, r', k) \models B_i^{\mathcal{N}}\exists 0$ then there is a point $(\hat{r}, k)$ that is $(\mathcal{N} \wedge B^{\mathcal{N}}\exists 0)$-$\boxdot$-reachable from (r', k) such that $(\mathcal{R}^{cr}, \hat{r}, k) \models \neg\exists 1$.

Suppose that $k = 0$. Since i does not believe it is faulty, $B_i^{\mathcal{N}}\exists 0$ holds only if i's initial value is 0. Let $\hat{r}$ be the run where all processors have initial values 0 and no processor fails. Clearly, $(\hat{r}, 0)$ has the desired properties.

For the inductive step, let $k > 0$, and assume that the claim has been shown for all $k' < k$. If $(\mathcal{R}^{cr}, r', k') \models B_i^{\mathcal{N}}\exists 0$ for some time $k' < k$, then the point (r', k') is clearly $(\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1})$-$\boxdot$-reachable from (r', k), and the claim follows from the induction hypothesis for (r', k') and the transitivity of reachability.

Now suppose that $(\mathcal{R}^{cr}, r', k') \not\models B_i^{\mathcal{N}}\exists 0$ for all $k' < k$. Since $(\mathcal{R}^{cr}, r', k) \models B_i^{\mathcal{N}}\exists 0$, i must have received a value of 0 in one of the messages sent to it in round k, say from processor j. Since we are in the crash failure mode, if processor i receives a round k message from j, then all other processors received messages from j in all rounds preceding round k. Thus, i does not know at the point (r', k) that j is faulty. Hence, there must be a run r'' in which both i and j are nonfaulty such that $r'_i(k) = r''_i(k)$ and $r'_j(k-1) = r''_j(k-1)$. It follows that $(\mathcal{R}^{cr}, r'', k-1) \models (j \in \mathcal{N}) \wedge B_j^{\mathcal{N}}\exists 0$. Moreover, by construction, $(r'', k-1)$ is $(\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1})$-$\boxdot$-reachable from (r', k) (since $\mathcal{Z}_j^{\Lambda,1} = B_j^{\mathcal{N}}\exists 0$). The claim now follows from the inductive hypothesis for $k-1$ (with respect to r'' and j), and (as before) from the transitivity of reachability. ∎

We now consider $\mathcal{O}^{\Lambda,2}$.

Lemma A.8: *In $\mathcal{R}^{cr}$, $\mathcal{O}_i^{\Lambda,2} = B_i^{\mathcal{N}}((\mathcal{N} \wedge \mathcal{Z}^{cr}) = \emptyset)$.*

Proof: Recall that $\mathcal{O}_i^{\Lambda,2} = B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}}\exists 1)$. Observe that $\mathcal{R}^{cr} \models C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}}\exists 1 \Leftrightarrow \boxdot((\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}) = \emptyset)$. The $\Leftarrow$ direction is immediate, since $\boxdot(\mathcal{S} = \emptyset) \Rightarrow C^{\boxdot}_{\mathcal{S}}\varphi$ is valid for any nonrigid set $\mathcal{S}$ and formula φ. The proof of Lemma A.7 shows that $\mathcal{R}^{cr} \models C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}}\exists 1 \Rightarrow ((\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}) = \emptyset)$. It is immediate that $\mathcal{R}^{cr} \models \boxdot C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}}\exists 1 \Rightarrow \boxdot((\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}) = \emptyset)$. Since $C^{\boxdot}_{\mathcal{S}}\varphi \Rightarrow \boxdot C^{\boxdot}_{\mathcal{S}}\varphi$ is valid for all nonrigid sets $\mathcal{S}$ and formulas φ (Lemma 3.4), the result follows.

To complete the proof it suffices to show that

$$\mathcal{R}^{cr} \models B_i^{\mathcal{N}}(\exists 1 \wedge \boxdot((\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}) = \emptyset)) \Leftrightarrow B_i^{\mathcal{N}}((\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}) = \emptyset).$$

Setting $\psi = ((\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}) = \emptyset)$, this claim has the form

$$\mathcal{R}^{cr} \models B_i^{\mathcal{N}}(\exists 1 \wedge \boxdot\psi) \Leftrightarrow B_i^{\mathcal{N}}\psi.$$

The $\Rightarrow$ direction is straightforward, following from the validity $\models \boxdot\psi \Rightarrow \psi$ and the monotonicity of $B_i^{\mathcal{N}}$. The other direction uses properties of the crash failure mode.

Assume that i is nonfaulty and that $(\mathcal{R}^{cr}, r, m) \models B_i^{\mathcal{N}}((\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}) = \emptyset)$. Since $\mathcal{Z}_i^{\Lambda,1} = B_i^{\mathcal{N}}\exists 0$, this clearly implies that $(\mathcal{R}^{cr}, r, m) \models \neg B_i^{\mathcal{N}}\exists 0$, and hence i's own initial value could not have been 0; thus, $B_i^{\mathcal{N}}\exists 1$ holds. It remains to show that $(\mathcal{R}^{cr}, r, m) \models B_i^{\mathcal{N}}\boxdot\psi$. So suppose $r_i(m) = r'_i(m)$ and i is nonfaulty in both r and r'. Note that $(\mathcal{R}^{cr}, r', m) \models B_i^{\mathcal{N}}\psi$. We must show that $(\mathcal{R}^{cr}, r', m) \models \boxdot\psi$. Clearly $(\mathcal{R}^{cr}, r', m') \models \psi$

for $m' < m$, for if a nonfaulty processor j believed $\exists 0$ at a time $m' < m$, this fact would have appeared in j's message to i in round $m' + 1 \leq m$, and hence i would already know about the 0 at time m in r', contradicting the fact that $(\mathcal{R}^{cr}, r', m) \models \psi$. Since $(\mathcal{R}^{cr}, r', m) \models B_i^{\mathcal{N}} \psi$ and $i \in \mathcal{N}$, it follows that $(\mathcal{R}^{cr}, r', m) \models \psi$. Notice that, since we are in the crash failure mode, a processor that is known by i to be faulty at time m cannot send messages after round m. Moreover, for i to believe ψ at (r', m), it must be the case that no processor not known by i to be faulty knows $\exists 0$ at (r', m). (The formal proof of this is identical to that of the proof of Theorem 5 in [DM90].) It immediately follows that, because we are in the crash failure mode, no nonfaulty processor will know $\exists 0$ at any later time $m' > m$ in r'. ∎

Theorem 6.2: *In the crash failure mode, the same decisions are made by nonfaulty processors at corresponding points of the protocols $P0_{\text{opt}}$ and $F^{\Lambda,2}$. Thus, both $F^{\Lambda,2}$ and $P0_{\text{opt}}$ are optimal EBA protocols for the crash failure mode.*

Proof: Recall that information about the existence of a value of 0 and about the identity of processors that started out with 1 are forwarded in $P0_{\text{opt}}$ as fast as in a full-information protocol. Moreover, in $P0_{\text{opt}}$ every processor sends a message to all other processors in every round. As a result, a straightforward inductive argument shows that for every pair of corresponding runs r of $\mathcal{R}^{cr}_{P0_{\text{opt}}}$ and r' of $\mathcal{R}^{cr}_{FL^2}$, every processor i nonfaulty in these runs, and every time $m \geq 0$, we have that (1) $B_i^{\mathcal{N}} \exists 0$ holds at (r, m) exactly if it holds at (r', m), (2) $B_i^{\mathcal{N}} \neg\exists 0$ holds at (r, m) exactly if it holds at (r', m), and (3) i receives messages from the same set of processes in round m of r and in round m of r'.

Since in both protocols a nonfaulty processor i decides 0 exactly when $B_i^{\mathcal{N}} \exists 0$ first holds, it follows that decisions on 0 take place at corresponding points of the two systems. To complete the proof, we need to show that nonfaulty processors decide 1 at corresponding points of the two systems as well. Notice that the condition $\mathcal{O}_i^{\Lambda,2} = B_i^{\mathcal{N}}((\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}) = \emptyset)$ holds exactly if i knows that no nonfaulty processor can know of a value of 0. It remains to show that this holds at a point of $\mathcal{R}^{cr}_{F^{\Lambda,2}}$ exactly if one of the conditions (a) and (b) defining the decision on 1 in $P0_{\text{opt}}$ holds in the corresponding point of $\mathcal{R}^{cr}_{P0_{\text{opt}}}$.

We first argue that if the nonfaulty processor i does not decide 1 at the point (r, m) of $\mathcal{R}^{cr}_{P0_{\text{opt}}}$ then it does not do so in the corresponding point (r', m) of $\mathcal{R}^{cr}_{F^{\Lambda,2}}$. The case $m = 0$ is straightforward, since processor i has no information about values of other processors. Thus, it must consider it possible that some nonfaulty processor i_1 has initial value 0, so $\mathcal{O}_i^{\Lambda,2}$ cannot hold at $(r', 0)$ (since $\mathcal{Z}_i^{\Lambda,1} = B_i^{\mathcal{N}} \exists 0$). Now suppose that $m \geq 1$. Notice that if neither (a) nor (b) holds for i at all points (r, m') with $m' \leq m$, then there is a sequence $i_1, i_2, \ldots, i_m$ of processors such that, for all $k < m$, in round k processor i_k crashes and sends no messages to processors that survive round k, except possibly to i_{k+1}. In addition, processor i_m crashes in round m without sending a message to i. It follows that there is a run r'' in which (i) i_1 has initial value 0, (ii) for all $k < m$ we have that i_k sends a message only to i_{k+1} in round k, (iii) the same processors are nonfaulty in r

and in r'', (iv) $r_j(k) = r''_j(k)$ for all $k < m$ and all processors j nonfaulty in r (and hence also in r''), and finally (v) i_m sends a message in round m of r'' to all nonfaulty processors $j' \neq i$. Thus, $r''_i(m) = r_i(m)$ and at (r'', m) there are nonfaulty processors that know $\exists 0$. Since r' is a run of a full-information protocol in which the same processors are nonfaulty as in r, it follows that $\mathcal{O}_i^{\Lambda,2}$ will not hold at (r', m) as well, so i does not decide 1 at (r', m).

To complete the proof, we argue that if i decides 1 at the point (r, m) of $P0_{\text{opt}}$ then it also does so at the corresponding point (r', m) of $F^{\Lambda,2}$. By fact (1) above, if condition (a) holds, so that i knows at (r, m) that all initial values were 1, then it knows this fact at (r', m) as well; as a result, i knows nobody can know $\exists 0$, so $\mathcal{O}_i^{\Lambda,2}$ holds.

Suppose that condition (b) holds at (r, m); we need to show that in (r', m) process i believes that $\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1} = \emptyset$. Since condition (b) holds at (r, m), then i does not know $\exists 0$ at (r, m) and i received messages from the same set G of processes in rounds $m - 1$ and m of r. By facts (1) and (3) above, the same is true for r' as well. In the crash failure mode, a processor whose round $m - 1$ message to i is not delivered is definitely silent from round m on. Hence, no processor other than those in G sends (or receives) messages in round m of r'. Since $F^{\Lambda,2}$ is a full-information protocol, it follows that if the set of messages sent by the processes in G in round m does not contain information about the existence of a value of 0, then nobody knows $\exists 0$ at time (r', m). Finally, by assumption, processor i receives all messages from processors in G in round m, and still does not know that $\exists 0$ holds there. We conclude that $\mathcal{O}_i^{\Lambda,2}$ holds at (r', m), and we are done. ∎

We next want to prove that in the omission failure mode, $\mathcal{O}^{\Lambda,2}$ does not hold. We first need to get some insight into $\mathcal{Z}^{\Lambda,1}$ in the case of omission failures. As in the case of crash failures, we shall for simplicity denote by $\mathcal{R}^{om}$ the system $\mathcal{R}^{om}_{F^{\Lambda,2}}$ obtained by running protocol $F^{\Lambda,2}$ in the omissions failure mode. The following is a variant of Lemma A.7 modified to suit the omission failure model.

Lemma A.9: *Suppose that $n \geq t+2$, r is a run of $\mathcal{R}^{om}$ in which fewer than t processors fail, and i is nonfaulty in r. Then for all times $m \geq 0$ we have that $(\mathcal{R}^{om}, r, m) \models B_i^{\mathcal{N}} \exists 0 \Rightarrow \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}} \exists 1$.*

Proof: We proceed by induction on k to show that if k is the first time in r such that $(\mathcal{R}^{om}, r, k) \models B_i^{\mathcal{N}} \exists 0$, then $(\mathcal{R}^{om}, r, k) \models \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}} \exists 1$. This suffices since if $\neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}} \exists 1$ is true at (r, k), it is true at (r, m) for all m by part (g) of Lemma 3.4. If $k = 0$, it must be the case that i has initial value 0, and hence its state at $(r, 0)$ is the same as at $(r', 0)$, where r' is the run in which no processor started with initial value 1 and all processors are nonfaulty. Since $\exists 1$ does not hold at $(r', 0)$ and i is nonfaulty at both $(r, 0)$ and $(r', 0)$, then by Proposition 3.2, it follows that $(\mathcal{R}^{om}, r, 0) \models \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}} \exists 1$.

Now suppose that $k \geq 1$ and the induction hypothesis holds for $k - 1$. Then i must have received a 0 from some processor j in round k of r. If i does not know that j is faulty, then let r' be a run with the same failure pattern as r except that processor j is nonfaulty. (In particular, $r' = r$ if j is nonfaulty in r.) Since $(\mathcal{R}^{om}, r', k-1) \models B_j^{\mathcal{N}} \exists 0$, by the induction hypothesis, $(\mathcal{R}^{om}, r', k-1) \models \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}} \exists 1$. The claim follows since (r, k) is $(\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1})$–$\boxdot$–reachable from $(r', k-1)$. If i does consider j to be faulty at (r, k), then we must have $k > 1$. (If $k = 1$ and i gets a message from j in round 1, then i will not consider j to be faulty at time 1.) Consider a run r' that is identical to r up to time k except that there is a processor j'' faulty in r' that fails for the first time in round k of r and sends messages to all processors in round k except some nonfaulty processor $j' \neq i$. Note that this means that $r_i(k) = r'_i(k)$. (Such processors j' and j'' exist because, by assumption, $n \geq t + 2$ and less than t processors fail in r.) Finally, consider a run r'' which is identical to r' up to time k, except that whichever processor sent j a message saying $\exists 0$ in r at round $k-1$ also sends it to j', and at round k, and all processors send the same messages in r' and r'' except that j' sends i a message saying $\exists 0$. Note that $r'_j(k) = r''_j(k)$. Now, by construction, i learns about $\exists 0$ in r'' from a processor which it does not know to be faulty. Thus, by the earlier argument, $(\mathcal{R}^{om}, r'', k) \models \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}} \exists 1$. The induction hypothesis carries us through as above and we are done. ∎

Proposition 6.3: *If $t > 1$ and $n \geq t + 2$, then there are runs of $\mathcal{R}^{om}$ in which the nonfaulty processors do not decide.*

Proof: Consider the run r where all the processors start with 1, processor 1 is faulty and never sends messages to any processor, and no other processors fail. Clearly, the weak validity condition implies that any nonfaulty processor which decides in r must decide 1. We now show that no nonfaulty processor can ever decide 1 in r. Assume that a nonfaulty processor i decides 1 at (r, m) for some m. Then, by definition, $\mathcal{O}_i^2$ holds at (r, m). Let r' be a run of R^{om} in which processor 1 is the only faulty processor and where the first m rounds of r' differ from those of r only in the following two ways: (a) processor 1 has initial value 0, and (b) exactly one message sent by processor 1 is delivered in r'; it is a message sent in round m to some nonfaulty processor $j \neq i$. Clearly, $r_i(m) = r'_i(m)$. It follows that $\mathcal{O}_i^2$, which depends only on i's local state, holds at (r', m) as well. By definition of $\mathcal{O}_i^2$, we thus have that $(\mathcal{R}^{om}, r', m) \models B_i^{\mathcal{N}} C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}} \exists 1$. Since i is nonfaulty in r', we have that

$$(R^{om}, r', m) \models C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}} \exists 1.$$

Notice, however, that in r' the nonfaulty processor $j \neq i$ receives a message reporting a value of 0 from processor 1 in round m. As a result, we have that $(R^{om}, r', m) \models B_j^{\mathcal{N}} \exists 0$. Applying Lemma A.9 with respect to r' and j now yields $(R^{om}, r', m) \models \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^{\Lambda,1}} \exists 1$, giving us a contradiction. ∎

A.5 Proofs for Section 6.2

Proposition 6.4: *In a run r of FIP$(\mathcal{Z}^0, \mathcal{O}^0)$ in the omission failure mode where f processors actually fail, all nonfaulty processors decide by time $f + 1$.*

Proof: Let $F = \mathit{FIP}(\mathcal{Z}^0, \mathcal{O}^0)$. Suppose that r is a run in $\mathcal{R}_F^{om}$ in which f processors fail. Then, for each nonfaulty processor i, there is a round $m \leq f+1$, such that i does not learn of any new failures in round m. Since a nonfaulty processor sends its state to everybody at each round, if processor j does not send a message to i in round $m' < m$, in round m (in fact, in all rounds between m' and m) i tells all processors that j is faulty. Thus, for each such j, we have

$$(\mathcal{R}_F^{om}, r, m) \models B_i(E_{\mathcal{N}}(j \notin \mathcal{N})).$$

We claim that $(\mathcal{R}_F^{om}, r, m) \models B_i^{\mathcal{N}} \exists 0^* \vee B_i^{\mathcal{N}} \neg \exists 0^*$, so that i decides at (r, m). Clearly, if i receives a message in round m from some processor j that is not known by i to be faulty which implies $\exists 0^*$, then $(\mathcal{R}_F^{om}, r, m) \models B_i^{\mathcal{N}} \exists 0^*$, and i can decide 0. Otherwise, we claim that $(\mathcal{R}_F^{om}, r, m) \models B_i^{\mathcal{N}} \neg \exists 0^*$. Suppose by way of contradiction that there exists a run r' such that $r'_i(m) = r_i(m)$, i is nonfaulty in r', and $(\mathcal{R}_F^{om}, r', m) \models \exists 0^*$. Thus, there must be a 0-chain in r' ending with some nonfaulty processor j at some time $m' \leq m$. Thus, j hears $\exists 0^*$ at some round $m' \leq m$. If $m' < m$, since j is nonfaulty in r', it will tell i $\exists 0^*$ in round $m'+1$ of r', so there will be a 0-chain ending with i at $(r', m'+1)$. Thus, i decides 0 at $(r', m'+1)$. Since $r_i(m) = r'_i(m)$ and F is a full-information protocol, we must have $r_i(m'+1) = r'_i(m'+1)$, and i decides 0 at $(r, m'+1)$, contradicting our assumption. So suppose that j hears $\exists 0^*$ in round m from j'. Thus, j considers j' to be nonfaulty at (r', m). But this means that i must consider j' to be nonfaulty at $(r', m-1)$ (and hence at $(r, m-1)$), since otherwise i would tell j that j' was faulty in round m of r'. Since i does not learn of any new failures in round m of r, i must also consider j' to be nonfaulty at (r, m). Thus, i also hears from j' in round m of r', and receives the same message in both r and r'. Again, this means that i hears $\exists 0^*$ in r, a contradiction. ∎

Proposition 6.6: *F^* is an optimal EBA protocol in the omissions failure mode that dominates $\mathit{FIP}(\mathcal{Z}^0, \mathcal{O}^0)$.*

Proof: Applying the first step of our construction, optimizing the decision on 1, given the rule in $F(\mathcal{Z}^0, \mathcal{O}^0)$ for deciding 0, we get

$$\begin{aligned} \mathcal{Z}_i^1 &= B_i^{\mathcal{N}}(\exists 0 \wedge \neg C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^0} \exists 1) \text{ and} \\ \mathcal{O}_i^1 &= B_i^{\mathcal{N}}(\exists 1 \wedge C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^0} \exists 1). \end{aligned}$$

We claim that $\mathcal{Z}^1 = \mathcal{Z}^0$ and $\mathcal{O}^1 = \mathcal{O}^0$. This follows from the next two lemmas.

Lemma A.10: *Let F be any full-information protocol. Then*

$$\mathcal{R}_F^{om} \models C^{\boxdot}_{\mathcal{N} \wedge \mathcal{Z}^0} \exists 1 \Leftrightarrow \boxdot(\mathcal{N} \wedge \mathcal{Z}^0) = \emptyset.$$

Proof: The "if" direction follows immediately from the definition, so we prove the "only if" direction. Assume that $(\mathcal{R}_F^{om}, r, m) \models \neg \boxdot(\mathcal{N} \wedge \mathcal{Z}^0) = \emptyset$. Then there is a first time

$l = l(r)$ at which we have $(\mathcal{R}_F^{om}, r, l(r)) \models (\mathcal{N} \wedge \mathcal{Z}^0) \neq \emptyset$. We prove by induction on $l(r)$ that $C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}^0}\exists 1$ does not hold in r. If $l(r) = 0$, since $\mathcal{Z}_i^0 = B_i^{\mathcal{N}}\exists 0^*$, it must be the case that all the nonfaulty processors in r have initial value 0. Thus, all the nonfaulty processors in r initally consider it possible that $\neg\exists 1$ holds; the result follows immediately.

Now suppose that $l(r) > 0$ and the induction hypothesis holds for $l(r) - 1$. Suppose $B_i^{\mathcal{N}}\exists 0^*$ holds at $(r, l(r))$ for some nonfaulty processor i. Then there must be some processor j that is not known to i as faulty at (r, l) that tells i $B_j^{\mathcal{N}}\exists 0^*$ in round l of r. Let r' be the run with the same failure pattern and initial states as r except that j is nonfaulty in r'. It is easy to see that $r_i(l(r)) = r'_i(l(r))$. But $l(r') = l(r) - 1$, and hence by the induction hypothesis, $C^{\boxdot}_{\mathcal{N}\wedge\mathcal{Z}^0}\exists 1$ does not hold in r'. By Proposition 3.2, it follows that $C^{\boxdot}_{\mathcal{N}\wedge B^{\mathcal{N}}\exists 0^*}\exists 1$ does not hold in r either, and we are done. ∎

Lemma A.10 implies that $\mathcal{Z}^1, \mathcal{O}^1$ reduce to

$$\begin{aligned} \mathcal{Z}_i^1 &\equiv B_i^{\mathcal{N}}(\exists 0 \wedge \neg\boxdot((\mathcal{N} \wedge \mathcal{Z}^0) = \emptyset)) \text{ and} \\ \mathcal{O}_i^1 &\equiv B_i^{\mathcal{N}}(\exists 1 \wedge \boxdot((\mathcal{N} \wedge \mathcal{Z}^0) = \emptyset)). \end{aligned}$$

As the following lemma shows, even further simplification is possible.

Lemma A.11: *Let F be any full-information protocol. Then*

$$\mathcal{R}_F^{om} \models B_i^{\mathcal{N}}(\exists 0 \wedge \neg\boxdot((\mathcal{N} \wedge \mathcal{Z}^0) = \emptyset)) \Leftrightarrow B_i^{\mathcal{N}}(\exists 0^*)$$

and

$$\mathcal{R}_F^{om} \models B_i^{\mathcal{N}}(\exists 1 \wedge \boxdot((\mathcal{N} \wedge \mathcal{Z}^0) = \emptyset)) \Leftrightarrow B_i^{\mathcal{N}}(\neg\exists 0^*).$$

Proof: We first need the following

Claim: $\mathcal{R}_F^{om} \models (\mathcal{N} \wedge \mathcal{Z}^0) \neq \emptyset \Leftrightarrow \exists 0^*$.

Clearly if $\exists 0^*$ holds at (r, m), then there is a 0-chain ending at some nonfaulty processor j at some point (r, m') with $m' \leq m$. Then $j \in (\mathcal{N} \wedge \mathcal{Z}^0)(r, m'')$ for all $m'' \geq m'$. Conversely, if $j \in (\mathcal{N} \wedge \mathcal{Z}^0)(r, m)$, then $B_j^{\mathcal{N}}(\exists^* 0)$ holds at (r, m) and j is nonfaulty in r. Thus, $\exists^* 0$ holds at (r, m) too. This completes the proof of the claim.

For the first half of the lemma, clearly $B_i^{\mathcal{N}}(\exists 0^*)$ implies $B_i^{\mathcal{N}}(\exists 0)$. It is immediate from the claim that $\mathcal{R}_F^{om} \models B_i^{\mathcal{N}}(\exists 0^*) \Rightarrow B_i^{\mathcal{N}}(\neg\boxdot((\mathcal{N} \wedge \mathcal{Z}^0) = \emptyset))$. For the converse, the claim implies that $\mathcal{R}_F^{om} \models B_i^{\mathcal{N}}(\neg\boxdot((\mathcal{N} \wedge \mathcal{Z}^0) = \emptyset)) \Rightarrow B_i^{\mathcal{N}}(\neg\boxdot\neg\exists^* 0)$. However, it should be clear that the only way processor i can believe that $\exists 0^*$ holds at some point in a run r is if i currently believes $\exists 0^*$ holds.

For the second half of the lemma, note that it is immediate from the claim that $\mathcal{R}_F^{om} \models B_i^{\mathcal{N}}(\exists 1 \wedge \boxdot((\mathcal{N} \wedge \mathcal{Z}^0) = \emptyset)) \Rightarrow B_i^{\mathcal{N}}(\neg\exists 0^*)$. For the converse, suppose that $(\mathcal{R}_F^{om}, r, m) \models B_i^{\mathcal{N}}(\neg\exists 0^*)$. Clearly this means that processor i must have an initial value of 1 in r, so $(\mathcal{R}_F^{om}, r, m) \models B_i^{\mathcal{N}}\exists 1$. To see that $(\mathcal{R}_F^{om}, r, m) \models B_i^{\mathcal{N}}(\boxdot((\mathcal{N} \wedge \mathcal{Z}^0) = \emptyset))$, suppose not. Then, by the claim, there is some time m' such that $(\mathcal{R}_F^{om}, r, m') \models B_i^{\mathcal{N}}(\exists 0^*)$. Since $(\mathcal{R}_F^{om}, r, m) \models B_i^{\mathcal{N}}(\neg\exists 0^*)$, we must have $m' > m$. But this means that i believes at

(r, m') there is a 0-chain with m' processors. Let i_m and i_{m+1} be the mth and $(m+1)$st processors in that chain. Processor i must believe at time m that i_m is faulty, otherwise we would not have $(\mathcal{R}_F^{om}, r, m) \models B_i^{\mathcal{N}}(\neg\exists 0^*)$. But this means that i would tell i_{m+1} in round $m+1$ that i_m is faulty, contradicting the existence of the 0-chain. This completes the proof of the lemma. ∎

From Lemmas A.10 and A.11, we have that

$$\mathcal{Z}_i^1 \equiv B_i^{\mathcal{N}}\exists 0^* \text{ and}$$
$$\mathcal{O}_i^1 \equiv B_i^{\mathcal{N}}\neg\exists 0^*.$$

Now we apply the second step of our construction and obtain

$$\mathcal{Z}_i^2 = B_i^{\mathcal{N}}(\exists 0 \wedge C_{\mathcal{N}\wedge\mathcal{Z}^0}^{\boxdot}\exists 0) \text{ and}$$
$$\mathcal{O}_i^2 = B_i^{\mathcal{N}}(\exists 1 \wedge \neg C_{\mathcal{N}\wedge\mathcal{Z}^0}^{\boxdot}\exists 0).$$

Thus, $F^* = \mathit{FIP}(\mathcal{Z}^*, \mathcal{O}^*)$. By Theorem 5.2, F^* dominates $\mathit{FIP}(\mathcal{Z}^0, \mathcal{O}^0)$. ∎

References

[Che80] B. F. Chellas. *Modal Logic.* Cambridge University Press, Cambridge, U.K., 1980.

[Coa86] B. Coan. A communication-efficient canonical form for fault-tolerant distributed protocols. In *Proc. 5th ACM Symp. on Principles of Distributed Computing*, pages 63–72, 1986.

[DM90] C. Dwork and Y. Moses. Knowledge and common knowledge in a Byzantine environment: crash failures. *Information and Computation*, 88(2):156–186, 1990.

[DRS90] D. Dolev, R. Reischuk, and H. R. Strong. Early stopping in Byzantine agreement. *Journal of the ACM*, 34(7):720–741, 1990.

[DS82] D. Dolev and H. R. Strong. Requirements for agreement in a distributed system. In H. J. Schneider, editor, *Distributed Data Bases*, pages 115–129. North-Holland, Amsterdam, 1982.

[FHMV95] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning about Knowledge.* MIT Press, Cambridge, Mass., 1995.

[Fis83] M. J. Fischer. The consensus problem in unreliable distributed systems. Technical Report RR-273, Yale University, 1983.

[Hal87] J. Y. Halpern. Using reasoning about knowledge to analyze distributed systems. In J. F. Traub, B. J. Grosz, B. W. Lampson, and N. J. Nilsson, editors, *Annual Review of Computer Science, Vol. 2*, pages 37–68. Annual Reviews Inc., Palo Alto, Calif., 1987.

[HM90] J. Y. Halpern and Y. Moses. Knowledge and common knowledge in a distributed environment. *Journal of the ACM*, 37(3):549–587, 1990. A preliminary version appeared in *Proc. 3rd ACM Symposium on Principles of Distributed Computing*, 1984.

[HM92] J. Y. Halpern and Y. Moses. A guide to completeness and complexity for modal logics of knowledge and belief. *Artificial Intelligence*, 54:319–379, 1992.

[HV89] J. Y. Halpern and M. Y. Vardi. The complexity of reasoning about knowledge and time, I: lower bounds. *Journal of Computer and System Sciences*, 38(1):195–237, 1989.

[LF81] N. A. Lynch and M. J. Fischer. On describing the behavior and implementation of distributed systems. *Theoretical Computer Science*, 13:17–43, 1981.

[LF82] L. Lamport and M. J. Fischer. Byzantine generals and transactions commit protocols. Technical Report Opus 62, SRI International, Menlo Park, Calif., 1982.

[MT88] Y. Moses and M. R. Tuttle. Programming simultaneous actions using common knowledge. *Algorithmica*, 3:121–169, 1988.

[NB92] G. Neiger and R. Bazzi. Using knowledge to optimally achieve coordination in distributed systems. In Y. Moses, editor, *Theoretical Aspects of Reasoning about Knowledge: Proc. Fourth Conference*, pages 43–59. Morgan Kaufmann, San Francisco, Calif., 1992.

[Nei90] G. Neiger. Consistent coordination and continual common knowledge. Manuscript. 1990.

[PSL80] M. Pease, R. Shostak, and L. Lamport. Reaching agreement in the presence of faults. *Journal of the ACM*, 27(2):228–234, 1980.

[PT86] K. Perry and S. Toueg. Distributed agreement in the presence of processor and communication faults. *IEEE Transactions on Software Engineering*, SE-12(3):477–481, 1986.

[PT92] P. Panangaden and S. Taylor. Concurrent common knowledge: defining agreement for asynchronous systems. *Distributed Computing*, 6(2):73–93, 1992.