A Characterization of Consensus Solvability for Closed Message Adversaries

Kyrill Winkler
TU Wien, Vienna, Austria
kwinkler@ecs.tuwien.ac.at

Ulrich Schmid
TU Wien, Vienna, Austria
https://ti.tuwien.ac.at/ecs/people/schmid
s@ecs.tuwien.ac.at

Yoram Moses
Technion, Haifa, Israel
moses@ee.technion.ac.il

Abstract

Distributed computations in a synchronous system prone to message loss can be modeled as a game between a (deterministic) distributed algorithm versus an omniscient message adversary. The latter determines, for each round, the directed communication graph that specifies which messages can reach their destination. Message adversary definitions range from oblivious ones, which pick the communication graphs arbitrarily from a given set of candidate graphs, to general message adversaries, which are specified by the set of sequences of communication graphs (called admissible communication patterns) that they may generate. This paper provides a complete characterization of consensus solvability for closed message adversaries, where every inadmissible communication pattern has a finite prefix that makes all (infinite) extensions of this prefix inadmissible. Whereas every oblivious message adversary is closed, there are also closed message adversaries that are not oblivious. We provide a tight non-topological, purely combinatorial characterization theorem, which reduces consensus solvability to a simple condition on prefixes of the communication patterns. Our result not only non-trivially generalizes the known combinatorial characterization of the consensus solvability for oblivious message adversaries by Coulouma, Godard, and Peters (Theor. Comput. Sci., 2015), but also provides the first combinatorial characterization for this important class of message adversaries that is formulated directly on the prefixes of the communication patterns.

2012 ACM Subject Classification Theory of computation → Distributed algorithms

Keywords and phrases Dynamic networks, Consensus, Message Adversary

Digital Object Identifier 10.4230/LIPIcs.OPODIS.2019.17

Funding *Kyrill Winkler*: Austrian Science Fund (FWF): ADynNet (P28182) and RiSE/SHiNE (S11405)

1 Introduction

With the increasing pervasiveness of mobile wireless devices that are subject to energy constraints and unreliable communication, there is a growing need for a profound theoretical understanding of what can and what cannot be computed in such *dynamic networks* [15].

One popular way to model these systems is by assuming a set of n synchronous, infallible[1] agents (called processes in the sequel), which communicate by exchanging messages over directed links whose behavior is under the control of a *message adversary* (MA) [1]. Whereas

[1] Of course, the typical behavior exhibited by a crashed process (i.e., no messages sent after it failed) can easily be mimicked by an appropriate message adversary as well.

the power of such a message adversary must be restricted somehow, as it may otherwise simply suppress all messages and make any distributed task trivially impossible, it is considered omniscient and may hence seek to actively foil the success of any given solution algorithm within its constraints. We stress that an algorithm that has been proved to solve some task, like consensus, under a certain message adversary MA provides a correct result, irrespective of whether erroneous communication stems from random faults or malicious interference, as long as it remains within the specification of the MA.

Even more fundamental than solution algorithms are characterizations of message adversaries with respect to certain tasks, that is, statements about when a message adversary becomes too powerful for solving a task: If a message adversary can be shown to cause any solution algorithm to fail under some admissible communication pattern, there is no hope to find a correct algorithm. In this paper, we provide a complete characterization of the important class of closed message adversaries (see below for its definition) with respect to the classic *deterministic consensus problem*, where each process starts with some input value and, eventually, every process has to decide the same value that was the input of some process. Thanks to the close relation between the solvability of consensus and variants of common knowledge in epistemic reasoning [13], our findings are also relevant for the latter (c.f. Section 5.2).

Classes of Message Adversaries

The fundamental objects in the message adversary model are *communication graphs* G_r, which, for a given round r, determine which messages are delivered and which are lost (see Section 2 for details). A *communication pattern* is a sequence $G_1, G_2, \dots$ of such communication graphs for round $1, 2, \dots$; a *message adversary* (MA) is just the set of its *admissible* communication patterns MA.

One important and well-studied class of message adversaries are *oblivious* ones [6], where the MA may pick every G_r arbitrarily, i.e., without restrictions, from a given set $\mathbf{D}$ of candidate graphs. We can hence write $\mathsf{MA} = \mathbf{D}^\omega$ for an oblivious message adversary. A prominent example is the message adversary that allows a certain number of mobile link failures per round, which was studied in the seminal work by Santoro and Widmayer [23]. Consensus has been shown to be impossible if $\mathbf{D}$ contains all communication graphs where $\leqslant n-1$ edges are missing (excluding self-loops).

Oblivious message adversaries do not allow to change the set of candidate graphs over time, however, which makes them unsuitable to model transient performance variations in real-world dynamic networks. In wireless mobile ad-hoc networks, for example, there may be substantial periods of time where some nodes are outside the communication range of others, e.g. in disaster-relief applications [18] or under strong interference by other radio transmitters [14]. Another source of time-varying communication conditions are mode switches, caused by the nodes themselves, due to reasons such as energy-saving, boot-up completion or fault recovery.

In this paper, we therefore focus on the class of *limit-closed* message adversaries (subsequently called *closed*[2] MAs for brevity), which are a proper superset of oblivious message adversaries. Their characterizing property is that every sequence $(\sigma_i)_{i\geq 1}$, such that each σ_i

[2] In the topological framework of [21], which follows-up on [2], a closed message adversary corresponds to a compact space, which is why, in a topological context, the term "compact message adversary" is preferable. Because we do not use topological reasoning in this paper, we stick to the terminology of [19] in this paper and use the term "closed message adversary" instead.

is the *prefix* of an admissible communication pattern and each σ_i is a prefix of σ_{i+1}, has a limit $\sigma \in \mathsf{MA}$. Equivalently, for every communication pattern σ that is inadmissible under a closed message adversary, there exists a round r such that all extensions of the round r prefix of σ are inadmissible as well. Note that this definition makes the set of admissible sequences of the MA closed and hence a safety property in the spirit of [2]. Thus, for closed message adversaries, reliable message delivery implies that the message delivery is reliable and bounded and fairness implies bounded fairness. Even though this is a very strict requirement, our characterization is more general than the previously existing combinatorial[3] characterizations of consensus solvability under message adversaries with an arbitrary number of processes [6, 23].

More specifically, it follows directly from its definition that an oblivious message adversary is closed, but there are also closed message adversaries that are not oblivious. An important example are MAs that ensure *bounded instability*. Such message adversaries guarantee some "global stabilization round" r_{GST}, which must be bounded (with some known bound), such that the communication graphs become "nice" from round r_{GST} onwards. The latter could occur in a myriad of ways, however: For example, starting from r_{GST}, a benign dynamic graph structure (like a vertex-stable source component [3] that persists for sufficiently many rounds) could appear. Alternatively, the number of rounds it takes for some processes to reach all other processes could be bounded, analogous to partially synchronous systems [9][4] or MAs corresponding to the ones from [16, 17]. A different example are MAs that assemble communication patterns by concatenating finite sequences of communication graphs, picked from a set of communication graph sequences with a fixed maximal length. Such a MA can either be allowed to choose an arbitrary combination of elements from this set, similar to oblivious message adversaries, or be subject to constraints, such as bounded instability.

On the other hand, relaxing the above requirement of r_{GST} being bounded to being unbounded but finite provides an illustrative example for a *general* message adversary. A MA that guarantees stability only eventually, i.e., after *unbounded*[5] *instability*, is not closed. In fact, the communication pattern where stability *never* occurs does not have a finite prefix such that all extensions are inadmissible, as any such finite prefix could still be made admissible by attaching a stability phase. Whereas we do not aim at a combinatorial characterization of consensus solvability for non-closed MAs in this paper (a topological characterization can be found in [21]), we believe that research on this challenging problem might benefit from our result.

An example for $n = 2$

In order to illustrate the different message adversary classes, we consider the case where the set of processes is $\Pi = \{\circ, \bullet\}$, i.e., $n = 2$, and the communication graphs are $\circ\!\leftarrow\!\bullet$, $\circ\!\leftrightarrow\!\bullet$, $\circ\!\rightarrow\!\bullet$, and $\circ \;\; \bullet$ (in the last graph, there is no edge between the processes).

[3] We use the term "combinatorial" to distinguish classic approaches that essentially enumerate combinatorial objects (like execution prefixes) from the topological characterization provided in [21], which is more general but rests on fairly abstract topological concepts like connected subspaces of infinite executions. Among the advantages of combinatorial characterizations is that they are often directly amenable to algorithmic implementations and hence more operational.

[4] In contrast to [9], message adversaries allow us to restrict precisely which processes may communicate with each other and which may not.

[5] Note that finite but unbounded r_{GST} is equivalent in terms of task solvability to r_{GST} being bounded with an unknown bound: The code of a solution algorithm $\mathcal{A}$ for the latter cannot depend on any bound on r_{GST}, yet must work for every finite value of r_{GST}.

In this example, an oblivious message adversary is represented by a subset of these communication graphs and considers all sequences admissible that consist exclusively of graphs from this subset. Thus, every oblivious message adversary that permits the communication graph $\circ \quad \bullet$ makes solving consensus trivially impossible. Furthermore, the oblivious message adversary that permits the communication graphs $\circ\leftarrow\bullet$, $\circ\leftrightarrow\bullet$, and $\circ\rightarrow\bullet$ is known to make consensus impossible at least since [23]. However, removing only one graph from this set of possible communication graphs already makes consensus solvable: If the message adversary permits $\circ\leftarrow\bullet$ and $\circ\leftrightarrow\bullet$, both processes may decide the input of $\bullet$ after the first round. If it permits $\circ\leftarrow\bullet$ and $\circ\rightarrow\bullet$, a process may decide on the other's input if it received the other's message in the first round and on its own input otherwise.

We get an example of a closed message adversary by allowing the communication graphs $\circ\leftarrow\bullet$, $\circ\leftrightarrow\bullet$, and $\circ\rightarrow\bullet$, provided there is some known round r_{GST} by which it is guaranteed that the same communication graph has occurred consecutively, in rounds r and $r+1$. This message adversary is closed, because every limit of a sequence $(\sigma_i)_{i\geq 1}$ of admissible prefixes, s.t. σ_i is a prefix of σ_{i+1}, is admissible here: Intuitively, the reason is that every prefix in the sequence that is longer than r_{GST} rounds, in order to be admissible, must have rounds $r, r+1 \leq r_{GST}$ that satisfy the property described above, and every continuation of such a sequence, which corresponds to a limit sequence, is also admissible. We can hence use Algorithm 1 introduced in Section 4 for solving consensus.

The same message adversary, however with the property that r_{GST} is finite but unbounded, provides an example of a non-closed message adversary. Consensus is solvable even under this message adversary; a suitable algorithm has been provided in [25]. Still, our Algorithm 1 does not work anymore, since the MA is not closed: Since the repetition of the same graph twice in a row may occur arbitrarily late, there is a limit sequence $(\sigma_i)_{i\geq 1}$ that consists entirely of admissible prefixes where it never occurs. This is an inadmissible communication pattern for this message adversary, however.

Contributions and Paper Organization

In this paper, we provide a complete combinatorial characterization of consensus solvability under closed message adversaries. Compared to the topological characterization provided in [21] (see the related work below), it is considerably less abstract and, more importantly, also fully operational: it utilizes an easy to check property of (finitely many) *prefixes* of admissible communication patterns, rather than properties of (uncountably many) infinite communication patterns. Compared to the combinatorial characterization for oblivious message adversaries developed in [6], our characterization applies to the larger class of closed MAs, and relies on checking the simple dynamic graph property "non-empty kernel intersection" (see Theorem 1 below), rather than on an involved algorithm that exploits certain properties of the so-called "β-classes".

In more detail, we present a condition on the admissible communication patterns of a message adversary, which we prove to be both necessary and sufficient for solving consensus. The condition, given in Theorem 1 below, rests on two main ingredients:

- **(i)** An equivalence relation $\sigma|_r \sim \rho|_r$ on the r-round prefixes of the communication patterns σ, ρ, which is the transitive closure of the per-process equivalence relations $\sigma|_r \sim_{p_i} \rho|_r$; the latter holds if process p_i cannot distinguish $\sigma|_r$ from $\rho|_r$, in every round up to r.
- **(ii)** The set of processes (called the *kernel* of $\sigma|_r$, denoted by $\mathrm{Ker}(\sigma|_r)$) that influence every process in the system within $\sigma|_r$. The processes in $\mathrm{Ker}(\sigma|_r)$ are the ones that manage to broadcast their initial value to all processes within $\sigma|_r$, and are hence sometimes called *broadcasters*.

Whereas it is not too difficult to prove (see Theorem 2) that solving consensus within r rounds under a communication pattern σ requires the existence of a broadcaster in $\sigma|_r$, i.e., $\mathrm{Ker}(\sigma|_r) \neq \emptyset$, this is only a *necessary* condition. What needs to be added to also make it sufficient is that actually *all* transitively indistinguishable prefixes $\rho|_r$ must have some *common* element(s) in their kernels $\mathrm{Ker}(\rho|_r)$: With $[\sigma|_r]$ denoting the equivalence class w.r.t. $\sim$ containing $\sigma|_r$, which can be computed from the message adversary specification, our main result is the following:

▶ **Theorem 1.** *Consensus is solvable under a closed message adversary MA if and only if for each $\sigma \in MA$ there is a round r such that $\bigcap_{x\in[\sigma|_r]} Ker(x) \neq \emptyset$.*

Theorem 1 is not only interesting from a theoretical point of view, but may also have practical implications in that it allows to avoid attempts to develop consensus algorithms for dynamic networks that do not allow any solution. The remainder of this paper is devoted to the proof of Theorem 1 and is organized as follows: In Section 2, we present our system and computation model, the definition of our message adversaries and their properties, and the specification of the consensus problem. In Section 3, we use indistinguishability arguments and König's Infinity Lemma to prove that consensus is impossible if the condition in Theorem 1 does not hold. In Section 4, we prove the sufficiency of our condition, by specifying an algorithm that solves consensus under a message adversary that satisfies Theorem 1. A comparison to other approaches in Section 5 and some conclusions in Section 6 round-off our paper.

Related Work

Dynamic networks have been studied in a wide variety of different forms in the distributed computing literature, e.g. as predicates on "heard-of" sets [5], in the form of dynamic connectivity constraints (T-interval connectivity) [16, 17], as time-varying graphs [4], and many more (c.f. the overview in [15]). The term message adversary was introduced in [1], as an intuitive way to understand message loss in distributed computing systems. [22] investigated the relation between message adversaries and failure detectors, whereas [3, 25] studied eventually stabilizing message adversaries.

Perhaps one of the earliest characterizations of consensus solvability in synchronous distributed systems prone to communication errors is the seminal work by Santoro and Widmayer [23], where it was shown that consensus is impossible if up to $n-1$ messages may be lost in each round. This classic result was refined in [24] and, more recently, by Coulouma et al. in [6], where a property of an equivalence relation on the sets of communication graphs was found that captures exactly the source of consensus impossibility under an oblivious message adversary. The authors also showed how this property can be exploited in order to develop a generic consensus algorithm.

The first characterization of consensus solvability under general message adversaries was provided in [11], albeit only for systems that consist of two processes. A bivalence argument was used there to show that certain communication patterns, namely, a fair or a special pair of unfair communication patterns, must be excluded by the MA for consensus to become solvable. In [21], Nowak et al. provided a complete topological characterization for systems of arbitrary size, which relies on non-trivial extensions of the seminal work by Alpern and Schneider [2]. It focuses on the space of communication patterns (actually, the corresponding infinite sequences of process-time graphs resp. configurations), and defines topologies based on variants of the well-known common-prefix metric. The authors show that consensus is solvable for a given MA if and only if this space partitions into multiple components $PS(v)$

consisting of the executions with decision value v. For closed message adversaries, these sets are shown to be compact and hence closed, while for non-closed MAs they are only relatively compact. The existence of this partitioning is tied to the non-empty kernel intersection of every individual $PS(v)$ in both cases. For closed MAs, it has been shown that $PS(v)$ can be replaced by some approximation $PS^{\varepsilon}(v)$, which consist of the k-prefixes of the sequences in $PS(v)$ for $\varepsilon = 2^{-k}$. In a way, the condition given in Theorem 1 can hence be viewed as the combinatorial counterpart of this topological characterization.

Regarding the study of closed message adversaries, the seminal works by Dolev et al. [8] and Dwork et al. [9] on partially synchronous systems introduced important abstractions like eventual stabilization and eventually bounded message delays, and provided a characterization of consensus solvability under various combinations of synchrony and failure models (including byzantine-faulty processes).

In [19], Lubitch and Moran provided a general consensus impossibility result for asynchronous distributed systems. For this purpose, they studied the configuration tree of an asynchronous system, in which each node corresponds to a configuration and a node is the child of another, if the corresponding configuration is a successor configuration of the other. They focused on closed subsets of runs of asynchronous distributed algorithms, defined by the property that every path in the configuration tree corresponds to an admissible execution (which is equivalent to limit-closure). Using a model-independent construction of closed schedulers that generate closed sets of runs, they provided a unified impossibility proof for consensus using a bivalence argument. The simplicity of their approach rests on the fact that the forever bivalent run so constructed belongs to the closed set of runs, and is hence admissible. This convenient property of closed sets of communication patterns is also used in our paper, albeit we use König's Infinity Lemma rather than a bivalence argument for constructing a non-deciding run. We briefly explain at the end of Section 3 why our model is not suitable for a bivalence argument like the one in [19].

2 Model of Computation

We consider a finite set $\Pi = \{p_1, \ldots, p_n\}$ of deterministic network-coupled state machines (processes) with unique identifiers (for brevity we assume the identifier of p_i is its index i), unlimited memory and infinite computational power. Processes, which are assumed to be fault-free, operate in lock-step synchronous rounds where they can exchange messages with each other via unidirectional pairwise links. A message can only arrive at the destination in the same round in which it is sent. However, not all messages are guaranteed to reach their recipient. Instead, a message adversary controls which messages arrive and which get lost in a round. The message adversary is omniscient, yet restricted by a set of rules that are known to the processes. The processes need to cooperate to solve the distributed consensus problem (to be defined later), which the message adversary, in turn, seeks to foil. Throughout this paper, we study the conditions under which there is a winning strategy for either side.

Message Adversaries

A round r *communication graph* G_r is a directed graph in which each vertex corresponds to a process and there is an edge $(p_i \to p_j)$ in G_r if and only if a message sent by p_i to p_j in round r is not lost. We denote by $\mathrm{In}_{G_r}(p_i)$ the incoming edges of p_i in G_r. We assume that every process always receives a message from itself, hence G_r contains self-loops at all nodes. A finite sequence of consecutive communication graphs $G_r, G_{r+1}, \ldots, G_{r+k}$ is called a *finite communication pattern*. We say that a finite communication pattern of the

form $\sigma = G_r, \ldots, G_{r+k}$, has *length* $|\sigma| = k+1$ and *range* $[r, r+k]$. Infinite communication patterns $\sigma = G_1, G_2, G_3, \ldots$ are called communication patterns for brevity. If σ is an infinite or finite communication pattern of range at least $[1, r]$, then $\sigma|_r$ denotes the r-round *prefix* of σ, i.e., the first r graphs $G_1, G_2, \ldots, G_r$ of σ. A message adversary is a set of communication patterns MA, (the specification of) which is assumed to be common knowledge.

For a given communication pattern $\sigma = G_1, G_2, \ldots$, we define influence in the usual way (see e.g. [17]): we say process p_i at round r influences p_j in round r', written as $(p_i, r) \rightsquigarrow_\sigma (p_j, r')$, if there is a chain of messages, starting at p_i no earlier than the beginning of round $r+1$ and ending at p_j no later than at the end of r'. Formally, influence is just the transitive closure (over rounds) of the relation $(p_i, r) \rightarrow_\sigma (p_j, r+1)$, which holds if the edge $(p_i \rightarrow p_j) \in G_{r+1}$. The *view* of a process in round r for communication pattern σ is a graph $\mathsf{view}_{\sigma|_r}(p_i) = \langle V, E \rangle$, such that V is the collection of process-round pairs that have influenced p_i by round r, i.e., $V = \{(p_j, r') \colon (p_j, r') \rightsquigarrow_\sigma (p_i, r)\}$ and there is an edge in $E \subseteq V \times V$ precisely if a corresponding message was successfully delivered and subsequently the recipient of this message influenced p_i, i.e., $E = \{((p_j, r') \rightarrow (p_k, r'+1)) : (p_j \rightarrow p_k) \in G_{r'+1} \wedge (p_k, r'+1) \rightsquigarrow_\sigma (p_i, r)\}$.

We say that two finite communication patterns ρ, σ are indistinguishable for p_i, written as $\rho \sim_{p_i} \sigma$, if $\mathsf{view}_\rho(p_i) = \mathsf{view}_\sigma(p_i)$. We note that $\mathsf{view}_\rho(p_i) = \mathsf{view}_\sigma(p_i)$ implies that $|\rho| = |\sigma|$ because we assume that every communication graph contains self-loops. We use $\rho \sim \sigma$ to denote the transitive closure of the above relation, over all processes and sequences, i.e., w.r.t. a set of finite communication patterns S, we write $\rho \sim \sigma$ if, for some multiset of processes $\{p_{i_1}, \ldots, p_{i_k}\}$ of Π and some set $\{\tau_1, \ldots, \tau_{k-1}\} \subseteq S$, we have $\rho \sim_{p_{i_1}} \tau_1 \sim_{p_{i_2}} \ldots \sim_{p_{i_{k-1}}} \tau_{k-1} \sim_{p_{i_k}} \sigma$. We note that $\sim$ is an equivalence relation and, given the set S (usually, a set of prefixes of admissible communication patterns), we denote by $[\sigma]$ the equivalence class of σ over the set S. Given $S' \subseteq S$ with $\sigma \in S'$, the subclass $[\sigma]_{S'}$ of σ is the equivalence class of σ on S'; we will sloppily write $[\sigma]_{S'} \subseteq [\sigma]$ in this case. Note carefully that the transitive closure $\sim$ of $[\sigma]_{S'}$ runs over sequences in S' only. Hence, there may be $\rho \in S'$ with $\rho \in [\sigma]$ but $\rho \notin [\sigma]_{S'}$, namely, when every path between σ and ρ contains some sequence in $S \setminus S'$. Finally, we extend $\sim_p$ (resp. $\sim$) to infinite communication patterns σ, ρ, by writing $\sigma \sim_p \rho$ (resp. $\sigma \sim \rho$) if and only if $\sigma|_r \sim_p \rho|_r$ (resp. $\sigma|_r \sim \rho|_r$), for every finite $r \geq 1$.

We define the *kernel* of a prefix $\sigma|_r$ by $\mathrm{Ker}(\sigma|_r) = \{p_i \in \Pi \mid \forall p_j \in \Pi \colon (p_i, 0) \rightsquigarrow_\sigma (p_j, r)\}$ to be the set of those processes that reach every other process, directly or via transitive messages by the end of round r. The kernel intersection $\mathrm{KI}[\sigma|_r]$ of an equivalence class $[\sigma|_r]$ is defined as $\mathrm{KI}[\sigma|_r] = \bigcap_{x \in [\sigma|_r]} \mathrm{Ker}(x)$. For an infinite communication pattern σ, we define $\mathrm{Ker}(\sigma) = \bigcup_{r>0} \mathrm{Ker}(\sigma|_r)$.

Executions

Starting from its initial state $\mathsf{state}_0(p_i)$, the state of p_i at the end of its round r computation is denoted as $\mathsf{state}_r(p_i)$. The collection of all round r states is called a round r configuration C_r. The state transition function that guides the evolution of $\mathsf{state}_r(p_i)$ to $\mathsf{state}_{r+1}(p_i)$ (which also depends on $\mathrm{In}_{G_{r+1}}(p_i)$), as well as the message sending function that computes (from $\mathsf{state}_r(p_i)$) the message to be broadcast[6] in round $r+1$, are specified by an algorithm in pseudo-code. Note that if $(p_j, r) \rightsquigarrow_\sigma (p_i, s)$ then $\mathsf{state}_s(p_i)$ may depend on $\mathsf{state}_r(p_j)$ but not necessarily on $\mathsf{state}_{r+1}(p_j)$. An execution is an infinite sequence of configurations

[6] We assume that the same message is sent to every receiver for simplicity, which makes sense since the algorithm does not necessarily know even n.

$C_0, C_1, \ldots$ that is in accordance with the state transition and message sending function of some deterministic algorithm. Since the processes are deterministic, the initial configuration C_0, together with the communication pattern, uniquely determines the execution, hence we write $\langle C_0, \sigma\rangle$ for the run that results when executing a given algorithm that starts from C_0 and is subject to an infinite communication pattern σ.

We say that two executions $\varepsilon, \varepsilon'$ are indistinguishable to process p_i, written as $\varepsilon \sim_{p_i} \varepsilon'$, if p_i goes through the same sequence of states in both executions. We note that, from the previous arguments, we immediately have that two executions $\langle C_0, \sigma\rangle \sim_{p_i} \langle C_0', \sigma'\rangle$ are indistinguishable for p_i if the corresponding communication patterns $\sigma \sim_{p_i} \sigma'$ are indistinguishable for p_i and, for all p_j that influence p_i in σ, we have that $\mathsf{state}_0(p_j)$ is the same in C_0 and in C_0'. Intuitively, this holds because a process can only be certain about the (successful and unsuccessful) message exchanges and initial states that it has either observed directly (by itself) or indirectly (by being influenced accordingly).

Consensus

In the classic distributed consensus problem, each process $p_i \in \Pi$ holds an input value $x_i \in \mathbb{N}$ and an output or decision value y_i, initialized to $y_i = \perp$, that can be written to at most once. In an execution of a consensus algorithm, the initial configuration C_0 is usually just an assignment of input values to each process. We say that p_i decides v if p_i assigns v to y_i. An algorithm that correctly solves consensus ensures the following:

(Decision) Every process p_i decides on a value y_i eventually.

(Agreement) If $y_i \neq \perp$ and $y_j \neq \perp$, then $y_i = y_j$.

(Validity) If $y_i = v$ and $v \neq \perp$, then $v = x_j$ for some process p_j.

We commence by identifying a crucial relation between decision values and $\mathrm{Ker}(\sigma)$.

▶ **Theorem 2.** *Let MA be an arbitrary message adversary, let C_0 be an initial configuration, let $\sigma \in \mathsf{MA}$, and let $p_i \in \Pi$. The decision y_i in execution $\varepsilon = \langle C_0, \sigma\rangle$ of any correct consensus algorithm $\mathcal{A}$ satisfies $y_i = x_j$ for some $p_j \in Ker(\sigma)$.*

Proof. Suppose that there is an initial configuration C_0 and a $\sigma \in \mathsf{MA}$, such that, in some correct consensus algorithm $\mathcal{A}$, some process decides v in execution $\langle C_0, \sigma\rangle$ where $v \neq x_i$ for any $p_i \in \mathrm{Ker}(\sigma)$. By validity, the set $\overline{K} = \{p_i \in \Pi \setminus \mathrm{Ker}(\sigma) \mid x_i = v\}$ is non-empty. Assuming some arbitrary ordering on $\overline{K} = \{p_{i_1}, \ldots, p_{i_k}\}$, let $C_0^0 = C_0$ and for $0 < m \leqslant |\overline{K}| = k$, let C_0^m be the same as C_0^{m-1} except that $x_{i_m} \neq v$ in C_0^m. We show by induction that, for any $\ell \geq 0$, some process decides v in the computation of $\mathcal{A}$ with execution $\varepsilon_\ell = \langle C_0^\ell, \sigma\rangle$. But then, $\mathcal{A}$ violates validity, because in C_0^k, $x_i \neq v$ for every process $p_i \in \Pi$.

The induction base for $\ell = 0$ follows from the initial assumption. For the induction step from $\ell - 1 \geq 0$ to ℓ, we observe that since $\overline{K} \subseteq \Pi \setminus \mathrm{Ker}(\sigma)$, there is some process p_i such that, for any round r, $(p_{i_\ell}, 0) \not\rightsquigarrow_\sigma (p_i, r)$.

By construction, therefore, $\varepsilon_{\ell-1} \sim_{p_i} \varepsilon_\ell$. The induction hypothesis asserts that some process decides v in $\varepsilon_{\ell-1}$ and thus, by agreement, p_i decides v in $\varepsilon_{\ell-1}$ and hence in ε_ℓ as well. ◀

Because of Theorem 2, it makes sense to only consider message adversaries MA where for each $\sigma \in \mathsf{MA}$ we have $\mathrm{Ker}(\sigma) \neq \emptyset$. Perhaps not surprisingly, though, this requirement alone is insufficient for solving consensus. Investigating precisely what is additionally required is the goal of the remaining paper.

3 Necessity of an Eventually Common Kernel

In this section, we prove the "only if"-direction of Theorem 1, showing that consensus is impossible under a message adversary that contains a communication pattern σ with $\mathrm{KI}[\sigma|_r] = \bigcap_{x\in[\sigma|_r]} \mathrm{Ker}(x) = \emptyset$ for all rounds r.

Our general proof strategy, as realized in Lemma 7, is to show that, because of the above condition, there is a particular subclass $X \subseteq [\sigma]$ of communication patterns with $\bigcap_{x\in X} \mathrm{Ker}(x) = \emptyset$. In Lemma 8, we show that this makes consensus impossible because it would mean that there are two indistinguishable executions with a different decision value. To prove Lemma 7, we apply König's Infinity Lemma to a tree that can be described as follows: At level r, the nodes are the subclasses of $[\sigma|_r]$ that consist of prefixes ρ such that $\mathrm{Ker}(\rho)$ is a subset of an "associate kernel" in some fixed set S of kernels. We then apply König's Infinity Lemma to show that there exists an infinite path in the tree that corresponds to a non-empty subclass X of $[\sigma]$. Because the associate kernels satisfy $\bigcap_{K\in S} K = \emptyset$, this implies $\bigcap_{x\in X} \mathrm{Ker}(x) = \emptyset$, as required.

▶ **Lemma 3** (König's Infinity Lemma, cf. [7, Chapter 6]). *Let $V_1, V_2, \ldots$ be an infinite sequence of disjoint non-empty finite sets, and let G be a graph on their union. Assume that every vertex v in a set V_r with $r > 1$ has a neighbor $f(v)$ in V_{r-1}. Then G contains an infinite path $v_1v_2\ldots$ with $v_r \in V_r$ for all r.*

We start with the definition of *kernel-restricted* classes $[\sigma]^{\mathbf{K}}$ (resp. $[\sigma]^{\subseteq\mathbf{K}}$), where all members must have a kernel that is equal to (resp. a subset of) an element of the non-empty set of kernels $\mathbf{K} = \{K_1, \ldots, K_k\}$:

▶ **Definition 4** (Kernel-restricted classes). *Given the equivalence class $[\sigma]_S$ of a communication pattern σ over the set S, and some non-empty set of non-empty kernels $\emptyset \neq \mathbf{K} = \{K_1, \ldots, K_k\}$, the kernel-restricted class $[\sigma]^{\mathbf{K}} \subseteq [\sigma]_S$ of σ is defined to be the class $[\sigma]_{S'}$ of σ over the set $S' = \{\rho \in S : Ker(\rho) \in \mathbf{K}\}$. Similarly, $[\sigma]^{\subseteq\mathbf{K}} \subseteq [\sigma]_S$ is the class $[\sigma]_{S''}$ over the set $S'' = \{\rho \in S : \exists K_i \in \mathbf{K}$ s.t. $Ker(\rho) \subseteq K_i\}$.*

Note that $[\sigma]^{\mathbf{K}} = \emptyset$ if $\mathrm{Ker}(\sigma) \notin \mathbf{K}$ (although this does not happen in Lemmas 5 to 7 below), and that there may be $\rho \in [\sigma]$ with $\mathrm{Ker}(\rho) \in \mathbf{K}$ but $\rho \notin [\sigma]^{\mathbf{K}}$, which happens if every path connecting $\sigma \sim \rho$ in $[\sigma]$ contains at least one prefix not in S'.

▶ **Lemma 5.** *Let MA be a message adversary that contains some σ s.t. $KI[\sigma|_r] = \emptyset$ holds for all rounds r. Then, there is a non-empty set $\mathbf{K} = \{K_1, \ldots, K_k\}$ of kernels, such that $\bigcap_{i=1}^{k} K_i = \emptyset$ and $[\sigma|_r] = [\sigma|_r]^{\mathbf{K}}$ for infinitely many rounds $r = r_1, r_2, \ldots$.*

Proof. For an arbitrary round $r > 0$, let $g([\sigma|_r]) = \{\kappa \subseteq \Pi\colon \exists\rho \in [\sigma|_r]$ with $\mathrm{Ker}(\rho) = \kappa\}$. Since Π is a finite set, the power set $\mathcal{P}(\Pi)$ is a finite set as well. By the pigeonhole principle, there is some set $\mathbf{K} = \{K_1, \ldots, K_k\} \subseteq \mathcal{P}(\Pi)$ such that, for infinitely many rounds r, $g([\sigma|_r]) = \mathbf{K}$, hence $[\sigma|_r] = [\sigma|_r]^{\mathbf{K}}$. Note that, since obviously $\sigma|_r \in [\sigma|_r]$ for every $r > 0$, we have $\mathrm{Ker}(\sigma|_r) \in \mathbf{K}$. By the assumption that, for all r, $\mathrm{KI}[\sigma|_r] = \emptyset$, we also have $\bigcap_{i=1}^{k} K_i = \emptyset$. ◀

▶ **Lemma 6.** *Let MA be a message adversary. If there exist a $\sigma \in MA$ and a set $\mathbf{K} = \{K_1, \ldots, K_k\} \subseteq \mathcal{P}(\Pi)$ with $[\sigma|_r] = [\sigma|_r]^{\mathbf{K}}$ for infinitely many rounds r, then there is an infinite sequence $\mathcal{V} = V_1, V_2, \ldots$ of sets $V_i \subseteq [\sigma|_i]^{\subseteq\mathbf{K}}$, such that, for all $i \geq 1$, each of the following holds:*

(1) $V_i \neq \emptyset$

(2) $V_i = \{\nu_1, \ldots, \nu_{m(i)}\}$ *for a finite $m(i) > 0$ s.t., for $1 \leqslant j \leqslant m(i)$: $\sigma|_i \in \nu_j \subseteq [\sigma|_i]^{\subseteq\mathbf{K}}$*

(3) *Each $\nu \in V_{i+1}$ has a neighbor $\nu' = f_{i+1}(\nu) \in V_i$*

Proof. Initializing $V_i = \emptyset$ for every $i \geq 1$, we construct V_i, starting from $i = 1$, as follows: For each of the infinitely many indices $i = r$ where $[\sigma|_r] = [\sigma|_r]^{\mathbf{K}}$, we set $V_i = \{[\sigma|_i]^{\mathbf{K}}\}$; note that $\mathrm{Ker}(\sigma|_r) \in \mathbf{K}$ in this case, so (1) and (2) hold for V_i. Moreover, for all $1 \leqslant j < i$, we add to V_j the set $\{\rho|_j : \rho \in [\sigma|_i]^{\mathbf{K}}\}$. As $\mathrm{Ker}(\rho|_j) \subseteq \mathrm{Ker}(\rho)$, and since $\rho \sim \tau$ implies also $\rho|_j \sim \tau|_j$, (1) and (2) continue to hold for V_j. Moreover, for $\nu \in V_{i+1}$, we define $f(\nu) = \nu|_i$, which secures (3). Thus, the infinite sequence of sets $\mathcal{V}$ with properties (1)–(3) exists, as claimed. ◀

▶ **Lemma 7.** *Let MA be a message adversary that contains some $\sigma \in MA$ such that, for all r, we have $KI[\sigma|_r] = \emptyset$. Then there is a nonempty set of kernels $\mathbf{K} = \{K_1, \ldots, K_k\}$ with $\bigcap_{i=1}^{k} K_i = \emptyset$ and a non-empty kernel-restricted subclass $X \subseteq [\sigma]^{\subseteq \mathbf{K}}$ with $\bigcap_{x \in X} Ker(x) = \emptyset$.*

Proof. We take σ and apply to it Lemma 5 and then Lemma 6. In this manner, we obtain the nonempty set of kernels $\mathbf{K}$ and an infinite tree spanning the members of the infinite sequence of sets $V_1, V_2, \ldots$ via the neighbor functions $f_i : V_i \to V_{i-1}$. König's Infinity Lemma ensures that there is an infinite path in this tree, which yields $X = \lim_{i \to \infty} V_i$: For every $\rho \in X$, we have $\rho|_r \sim \sigma|_r$ for every $r \geq 1$, and hence $\rho \sim \sigma$. ◀

▶ **Lemma 8.** *Consensus is impossible under a message adversary MA with $\sigma \in MA$ such that some non-empty subclass $X \subseteq [\sigma]$ satisfies $\bigcap_{x \in X} Ker(x) = \emptyset$.*

Proof. We show that the existence of some algorithm $\mathcal{A}$ that solves consensus under MA would lead to a contradiction. Since $X \neq \emptyset$, for some $k > 0$, there is a multiset of processes $\{p_{i_1}, \ldots, p_{i_{k-1}}\}$, which may be empty (if $k = 1$), and a non-empty multiset of communication patterns $Y = \{\sigma_1, \ldots, \sigma_k\} \subseteq X$ with $\sigma_j \in [\sigma]$ for $1 \leqslant j \leqslant k$ and $\bigcap_{\rho \in Y} \mathrm{Ker}(\rho) = \emptyset$, such that $\sigma_1 \sim_{p_{i_1}} \sigma_2 \ldots \sigma_{k-1} \sim_{p_{i_{k-1}}} \sigma_k$.

Let C_0^v be the input assignment where $x_i = v$ for all processes p_i and let $C_0^{\overline{v}}$ be the input assignment where $x_i = \overline{v}$ for all processes p_i, for some $v \neq \overline{v}$. By validity, in execution $\langle C_0^v, \sigma_1 \rangle$ every process running $\mathcal{A}$ decides v, while in $\langle C_0^{\overline{v}}, \sigma_1 \rangle$ they decide $\overline{v}$. Since input assignments are not restricted in any way, toggling the input values of $p_1, p_2, \ldots$ from v to $\overline{v}$, one after the other, reveals that there are input assignments C_0', C_0'' that differ only in the input value of a single process p_i, yet every process running $\mathcal{A}$ decides v in $\varepsilon_1^v = \langle C_0', \sigma_1 \rangle$ and $\overline{v}$ in $\varepsilon_1^{\overline{v}} = \langle C_0'', \sigma_1 \rangle$.

A simple induction shows that v is decided in $\varepsilon_j^v = \langle C_0', \sigma_j \rangle$ and $\overline{v}$ is decided in $\varepsilon_j^{\overline{v}} = \langle C_0'', \sigma_j \rangle$ for $1 \leqslant j \leqslant k$. The base case, $\ell = 1$, was already shown above. For the step from ℓ to $\ell + 1$ with $1 \leqslant \ell < k$, we have by hypothesis that v was decided in ε_ℓ^v and $\overline{v}$ was decided in $\varepsilon_\ell^{\overline{v}}$. Since $\sigma_\ell \sim_{p_{i_\ell}} \sigma_{\ell+1}$, we have $\varepsilon_\ell^v \sim_{p_{i_\ell}} \varepsilon_{\ell+1}^v$ and v is also decided in $\varepsilon_{\ell+1}^v$. A similar argument shows that $\overline{v}$ is decided in $\varepsilon_{\ell+1}^{\overline{v}}$.

We conclude the proof by showing that $p_i \in \mathrm{Ker}(\sigma_j)$ for $1 \leqslant j \leqslant k$, and hence $p_i \in \mathrm{Ker}(\rho)$ for any $\rho \in Y$, which contradicts $\bigcap_{\rho \in Y} \mathrm{Ker}(\rho) = \emptyset$. Suppose, for some j, $p_i \notin \mathrm{Ker}(\sigma_j)$. Hence there is some p_k with $p_i \not\rightsquigarrow_{\sigma^j} p_k$. Since C_0' and C_0'' are the same except for the input of p_i, $\varepsilon_j^v \sim_{p_k} \varepsilon_j^{\overline{v}}$, and p_k decides the same in ε_j^v and in $\varepsilon_j^{\overline{v}}$. This, however, contradicts our previous statement that v is decided in ε_j^v and $\overline{v}$ is decided in $\varepsilon_j^{\overline{v}}$ for some $v \neq \overline{v}$. ◀

A note on bivalence arguments

At this point, the reader might wonder why we did not resort to a bivalence argument, as introduced in [12] and used heavily in the literature (e.g. in the very closely related papers [6, 11, 19, 20, 23, 24]) to establish our impossibility result. In a nutshell, in the case of binary consensus, bivalence proofs establish impossibility by inductively constructing a

run where every reached configuration is bivalent. Reachable configurations are classified according to whether only 0-decided (resp. 1-decided) configurations are reachable from it, in which case the configuration is called 0-valent (resp. 1-valent), or whether both a 0-decided and a 1-decided configuration are reachable from it, in which case the configuration is called bivalent. Note carefully that the agreement property implies that no process can have decided in a bivalent configuration, as a single decision, say, to 0, would make the configuration already 0-valent.

In the bivalence induction proof, it is first established that not all initial configurations can be 0-valent or 1-valent. Then, under the hypothesis that the reached round r configuration is bivalent, it is shown that not all round $r+1$ configurations can be univalent. This results in a forever bivalent run, in which no process can have decided. Care must be taken, however, to also prove that the run so constructed is also admissible, as the processes must decide only in an admissible run.

The reason why we cannot use such an argument in our setting, and need to resort to König's Lemma instead, is that the induction step might lead to a dead end later on: there is no a priori guarantee that the bivalent successor chosen in some step is one that allows the construction of an infinite admissible suffix. Technically, what would be needed in the induction step to ensure this is that two configurations that are currently indistinguishable for some process can be extended in a way that remains indistinguishable forever for this process. As the only thing we know about our message adversary is that it is closed and eventually guarantees a non-empty kernel intersection, however, it is not clear how to infer sufficient information on the possible communication graphs generated in all admissible suffixes to guarantee this.

4 Sufficiency of an Eventually Common Kernel

Lemma 8 established the "only if"-direction of Theorem 1. We now show the "if"-direction of Theorem 1, by introducing Algorithm 1. This algorithm solves consensus under any message adversary MA that guarantees, for every $\sigma \in \mathsf{MA}$, that there is a round r and a non-empty set $K \subseteq \Pi$ such that $K = \mathrm{KI}[\sigma|_r] = \bigcap_{x \in [\sigma|_r]} \mathrm{Ker}(x)$. Note that this algorithm could stop operating immediately after decision, as the decision happens in the same round at all processes.

Essentially, each process p_i executing the algorithm attempts to send a local estimation of the communication pattern, stored in array E, along with its own input value $x[i]$ to all other processes. Here, the k^{th} entry of E, $\mathsf{E}[k]$, contains the local estimate of the round k communication graph. On reception of a round r message from some process p_j, including $p_j = p_i$, p_i stores the input of p_j in $x[j]$ and adds the edge $(p_j \to p_i)$ to $\mathsf{E}[r]$ since it could only have received the message if $(p_j \to p_i) \in G_r$. Note that this implies that in every round r, the edge $(p_i \to p_i)$ is added to $\mathsf{E}[r]$, as we assumed that every process receives a message from itself in each round. Process p_i then proceeds to merge its local estimates E with the ones received and then calculates the set S of all communication patterns $G_1, \ldots, G_r$ that it considers possible. It does this by checking which communication pattern prefixes, allowed by MA, are in accordance with what p_i observed so far. Finally, p_i picks an arbitrary prefix ρ of S and checks whether all members of the equivalence class of ρ have a non-empty intersection K of their kernels. If this is the case, p_i decides on the input of the process in K with the largest identifier. We note that the equivalence class of ρ can be computed from the specification of MA, which is, according to the system model, known to the processes. Note that there are only finitely many communication graphs and hence finitely many round r prefixes at any finite r.

Algorithm 1 Consensus algorithm, code for process p_i.

Initialization:
1 $\mathsf{x}[i] \leftarrow x_i$
2 $\mathsf{x}[j] \leftarrow \perp$ for $j \neq i$
3 $\mathsf{E}[0] \leftarrow \emptyset$
4 $r \leftarrow 1$

Transmit round r messages:
5 Attempt to send $\langle \mathsf{E}, \mathsf{x} \rangle$ to all
6 Receive $\langle \mathsf{E}_j, \mathsf{x}_j \rangle$ from all p_j with $(p_j \to p_i) \in G_r$

Round r computation:
7 **foreach** p_j *from which* p_i *received a message in round* r **do**
8 | **foreach** k *with* $\mathsf{x}_j[k] \neq \perp$ **do**
9 | | $\mathsf{x}[k] \leftarrow \mathsf{x}_j[k]$
10 | Add $(p_j \to p_i)$ to $\mathsf{E}[r]$
11 | **if** $r > 1$ **then**
12 | | **for** $r'' \in \{1, \dots r-1\}$ **do**
13 | | | $\mathsf{E}[r''] \leftarrow \mathsf{E}[r''] \cup \mathsf{E}_j[r'']$
14 **for** $r' \in \{1, \dots r\}$ **do**
15 | $\mathsf{V}_{r'} \leftarrow \{p_j \in \Pi \colon \exists p_k \in \Pi \text{ s.t. } (p_k \to p_j) \in \mathsf{E}[r']\}$
16 Let $\mathrm{In}_{G_{r'}}(\mathsf{V}_{r'})$ denote the edges $(u \to v) \in G_{r'}$ with $v \in \mathsf{V}_{r'}$
17 $\mathsf{S} \leftarrow \left\{\sigma|_r = G_1, \dots, G_r \colon \sigma \in \mathsf{MA} \text{ and } \mathrm{In}_{G_{r'}}(\mathsf{V}_{r'}) = \mathsf{E}[r'] \text{ for all } 1 \leqslant r' \leqslant r\right\}$
18 Pick an arbitrary $\rho \in \mathsf{S}$
19 **if** $y_i = \perp$ *and there exists* $K \neq \emptyset$ *s.t.* $K = KI[\rho]$ **then**
20 | $m \leftarrow \max\{j : p_j \in K\}$
21 | $y_i \leftarrow \mathsf{x}[m]$ /* decide */
22 $r \leftarrow r + 1$

In the following proof of the correctness of Algorithm 1, we use $\mathsf{var}_i^r \in \mathsf{state}_r(p_i)$ to denote the value of variable var, held by process p_i at the end of its round r computation. This is clearly the same as the value of var after it was written to the last time in round r, so if the last write to var occurs in line ℓ, var_i^r is the value of var at process p_i after p_i finished line ℓ for the last time in round r.

We start with a few technical results, which essentially assert the correctness of the local estimates. Lemma 9 establishes that the set E approximates edges in the communication graph faithfully if there was an appropriate influence.

▶ **Lemma 9.** $(p_j, r') \rightsquigarrow_\sigma (p_i, r) \Leftrightarrow \mathrm{In}_{G_{r'}}(p_j) \subseteq \mathsf{E}_i^r[r']$.

Proof. For the "⇒" direction, we show inductively for $r' < \ell \leqslant r$ that $(p_j, r') \rightsquigarrow_\sigma (p_k, \ell)$ implies $\mathrm{In}_{G_{r'}}(p_j) \subseteq \mathsf{E}_k^\ell[r']$.

For $\ell = r' + 1$, $(p_j, r') \rightsquigarrow_\sigma (p_k, \ell)$ implies, by definition of the $\rightsquigarrow_\sigma$ relation, $(p_j \to p_k) \in G_{r'+1}$, i.e., p_k receives the round $r'+1$ message of p_j. By Line 10, $\mathrm{In}_{G_{r'}}(p_j) \subseteq \mathsf{E}_j^{r'}[r']$. Since p_k received $\mathsf{E}_j^{r'}[r']$ from p_j via its round $r'+1$ message, p_k incorporates $\mathsf{E}_j^{r'}[r']$ into its own $\mathsf{E}_k^{r'+1}[r']$ in Line 13.

For $r' + 1 < \ell \leqslant r$, we assume that $(p_j, r') \rightsquigarrow_\sigma (p_k, \ell - 1)$ implies $\mathrm{In}_{G_{r'}}(p_j) \subseteq \mathsf{E}_k^{\ell-1}[r']$. If for some p_m, $(p_j, r') \rightsquigarrow_\sigma (p_m, \ell)$, by definition, $(p_k \to p_m) \in G_\ell$ for some p_k with $(p_j, r') \rightsquigarrow_\sigma (p_k, \ell - 1)$. By hypothesis, $\mathrm{In}_{G_{r'}}(p_j) \subseteq \mathsf{E}_k^{\ell-1}[r']$, hence p_m receives $\mathsf{E}_k^{\ell-1}[r']$ at the beginning of round ℓ and incorporates $\mathrm{In}_{G_{r'}}(p_j)$ into $\mathsf{E}_m^\ell[r']$ in Line 13 of its round ℓ computation.

The "⇐" direction holds trivially if $p_i = p_j$. If $p_i \neq p_j$, then, according to Line 13, p_i can only learn about $\mathrm{In}_{G_{r'}}(p_j)$ if there is a chain of messages, starting at p_j no earlier than the end of round r' and ending at p_i before its round r computing step. By definition, hence $(p_j, r') \rightsquigarrow_\sigma (p_i, r)$. ◀

Lemma 10 shows that any $\mathsf{E}_k[r']$ is an under-approximation of $G_{r'}$, i.e., it does not contain any fabricated edges.

▶ **Lemma 10.** *If $(p_j \to p_i) \in \mathsf{E}_k^r[r']$ then (1) $(p_j \to p_i) \in G_{r'}$ and (2) $(p_i, r') \rightsquigarrow_\sigma (p_k, r)$.*

Proof. (1) Suppose $(p_j \to p_i) \in \mathsf{E}_k^r[r']$ but $(p_j \to p_i) \notin G_{r'}$. Since $(p_j \to p_i)$ could only be added to $\mathsf{E}_k^r[r']$ through Line 10 or Line 13, we have $(p_j \to p_i) \in \mathsf{E}_i^{r'}[r']$. Since Line 10 is guarded by Line 7, $(p_j \to p_i) \in \mathsf{E}_i^{r'}[r']$ can only happen when p_i received a message from p_j in round r', which implies that $(p_j \to p_i) \in G_{r'}$.

(2) If $(p_j \to p_i) \in \mathsf{E}_k^r[r']$, since $(p_j \to p_i) \in \mathrm{In}_{G_{r'}}(p_i)$, because during the loop of Line 7, every in-edge is added in Line 10, $\mathrm{In}_{G_{r'}}(p_i) \subseteq \mathsf{E}_k^{r'}[r']$. By Corollary 9 hence $(p_i, r') \rightsquigarrow_\sigma (p_k, r)$. ◀

The above lemmas can be combined to the following Corollary 11, which shows that if $\mathsf{E}_k^r[r']$ contains some non-empty fraction of the incoming round r' edges of some process p_i, then it actually contains all the incoming round r' edges of p_i and vice-versa. Furthermore, this is equivalent to the existence of an influence from p_i in round r' to p_k in round r.

▶ **Corollary 11.** *The following are all equivalent: (1) $(p_j \to p_i) \in \mathsf{E}_k^r[r']$, (2) $\mathrm{In}_{G_{r'}}(p_j) \subseteq \mathsf{E}_k^r[r']$, (3) $(p_i, r') \rightsquigarrow_\sigma (p_k, r)$.*

Corollary 11 can be used to show that S_i^r indeed contains all execution prefixes that are indistinguishable from the current execution prefix, as expressed in Corollary 12.

▶ **Corollary 12.** *In every round r of an execution $\langle C_0, \sigma\rangle$, the following holds: $\mathsf{S}_i^r = \{\rho|_r \colon \rho \in \mathsf{MA}$ and $\rho|_r \sim_{p_i} \sigma|_r\}$.*

Proof. Since we assume self-loops in every communication graph, by Corollary 11, for any processes p_i, p_j, p_k, an in-edge $(p_k \to p_j)$ to p_j is present in $\mathsf{E}_i^r[r']$ exactly when $(p_j, r') \rightsquigarrow_\sigma (p_i, r)$. Hence, S_i^r contains exactly those sequences $\rho_r = G_1, \ldots, G_r$ where each G_ℓ matches the in-neighborhood for those nodes p_m that satisfy $(p_m, \ell) \rightsquigarrow_\sigma (p_i, r)$. By definition, these are precisely those prefixes $\rho|_r$ of $\rho \in \mathsf{MA}$ where $\mathsf{view}_{\rho|_r}(p_i) = \mathsf{view}_{\sigma|_r}(p_i)$ and thus $\rho|_r \sim_{p_i} \sigma|_r$. ◀

Finally, from the assumption that the specification of MA is known to the processes, and since the finite number of processes implies a finite number of communication graphs, which, in turn, implies a finite number of round r prefixes, we have the following Corollary 13 of Lemmas 9 and 10.

▶ **Corollary 13.** *Let r be an arbitrary round. The equivalence class $[\sigma|_r]$ is finite and thus can be computed by all processes.*

We are now ready to show the correctness of Algorithm 1.

▶ **Lemma 14.** *Algorithm 1 solves consensus under every message adversary MA that ensures, for all $\sigma \in \mathsf{MA}$, that $KI[\sigma|_r] \neq \emptyset$ for some round r.*

Proof. Pick an arbitrary sequence $\sigma \in \mathsf{MA}$ and fix some input assignment C_0. We show that Algorithm 1 satisfies all properties of the consensus specification in the execution $\langle C_0, \sigma \rangle$. Let t be the earliest round such that $\mathrm{KI}[\sigma|_t] \neq \emptyset$, i.e., for all $t' < t$ we have $\mathrm{KI}[\sigma|_{t'}] = \emptyset$. We show that (1) no p_i decides before round t and (2) there is a value v, which is the input value of some process p_j, such that every undecided p_i decides v in round t.

(1) Suppose some process p_i decides in some round $t' < t$. This means p_i passes the guard of Line 19 in round t'. After executing Line 18, due to Corollary 12, at any process p_i, $\rho_i^{t'}$ is set to some prefix $\sigma'|_{t'}$ with $\sigma' \in \mathsf{MA}$ and $\sigma'|_{t'} \sim_{p_i} \sigma|_{t'}$. By definition, hence $\sigma'|_{t'} \in [\sigma|_{t'}]$ and, in fact, $[\sigma'|_{t'}] = [\sigma|_{t'}]$. This, taken together with Corollary 13, yields that, after performing the computations in the guard of Line 19, at every process p_i, we have $K_i^{t'} = \mathrm{KI}[\sigma|_{t'}]$. If p_i passes this guard, we have $K_i^{t'} \neq \emptyset$ which contradicts the assumption that t is the earliest round for which the condition holds.

(2) By assumption, we have some set K such that $\mathrm{KI}[\sigma|_t] = K \neq \emptyset$. Let m be the maximal identifier of any process in K. We show the claim for $v = x_m$. Since we assumed that p_i has not decided yet, a similar argument as above shows that every process p_i passes the guard of Line 19 in round t and decides on $\mathsf{x}^t[m]$ via Line 21. Since $p_m \in \mathrm{Ker}(\sigma|_t)$, $(p_m, 0) \in \mathsf{view}^t_{\sigma|_t}(p_i)$ and hence Lines 1 and 9 ensure that $\mathsf{x}^t[m] = x_m = v$. ◀

5 Relation to Other Approaches

In this section, we will complement our results by exploring some not so obvious relations to other approaches. Moreover, we will provide some illustrating examples.

5.1 A topological view

Our combinatorial characterization of consensus for closed message adversaries given in Theorem 1 is of course compatible with the topological one established in [21], as mentioned already in our discussion of related work in Section 1: The requirement of broadcastability of the connected components in the topological space of admissible executions made in [21] is clearly enforced by our non-empty kernel intersection condition.

5.2 A knowledge-based view

Our results are also perfectly in line some well-known results of the epistemic analysis of consensus [13]. First and foremost, it is well-known (see e.g. [13, Prop. 4]) that simultaneous consensus can only be achieved if there is common knowledge of certain facts (in particular, a decision taking place). And indeed, Lemma 14 reveals that in Algorithm 1 all processes are guaranteed to decide in the same round. Furthermore, it was established in [10] that for simultaneous consensus, common knowledge of the input values is needed. We will therefore argue now that the processes actually establish common knowledge about the initial values of the members of the kernel intersection guaranteed by Theorem 1, in the same round.

The first thing to mention is that our similarity relation $\sim$ is actually equivalent to G-reachability in view-based interpretations in S5 models, cf. [13, p. 561]. For every σ and r, the equivalence class $[\sigma|_r]_\sim$, and hence also the kernel intersection $K_r = \bigcap_{x \in [\sigma|_r]_\sim} \mathrm{Ker}(x)$, can be computed by every process from the a priori common knowledge of MA. Since every process can unambiguously determine the equivalence class $[\sigma|_r]_\sim$ (albeit not $\sigma|_r$ itself) from its local state, all the algorithm has to do is to wait for the round where $K_r \neq \emptyset$ for the first time, which is the round r guaranteed by Theorem 1.

What is also worth exploring is the apparent paradox that our algorithm manages to attain some common knowledge in systems with unreliable communication that is not already available initially. According to [13, Thm. 5], this should be impossible. There is no contradiction here, however, since if we translate the definition of "communication cannot be guaranteed" [13, p. 566] into a corresponding message adversary MA, it would fail the condition of Theorem 1: Condition NG2 says that if some p_i does not receive a message in round r in $\sigma|_r$, then there is some $\rho|_r \in [\sigma|_r]_\sim$ where no message is delivered in round r. NG2 thus allows the non-empty kernel intersection condition of Theorem 1 to hold only if a simultaneous broadcast happens in round r in all prefixes in $[\sigma|_r]_\sim$. This, in turn, violates condition NG1, which requires that every $\sigma|_{r-1}$ can be continued with a suffix where no further messages are ever received. Consequently, closed message adversaries that satisfy our condition do not qualify as "communication cannot be guaranteed".

6 Conclusions

We have derived the, to the best of our knowledge, first combinatorial characterization of consensus solvability in the important class of closed message adversaries: Consensus is solvable here if and only if, for every communication pattern, eventually, the kernel-intersection of all transitively indistinguishable communication pattern prefixes becomes non-empty. Our consensus characterization surpasses all existing non-topological characterizations for message adversaries known so far w.r.t. the range of message adversaries covered. Moreover, unlike the existing topological characterization for closed message adversaries, it operates on prefixes of communication patterns and is hence well-suited for practical implementations as well. Moreover, our result was obtained using only very basic notions from dynamic networks, like the kernel of a communication pattern prefix and the transitive closure of the indistinguishability relation.

Regarding future work, a natural question is whether there is also a combinatorial characterization of consensus solvability for general (non-closed) message adversaries. Existing algorithms show that a simultaneous decision is sometimes impossible under such message adversaries, which suggests that our closed algorithm is definitely not applicable there. On the other hand, the principles exploited in Theorem 1 rest fundamentally only on the consensus specification itself, thus it seems unlikely that they could completely disappear even in a more general result.

References

1 Yehuda Afek and Eli Gafni. Asynchrony from Synchrony. In Davide Frey, Michel Raynal, Saswati Sarkar, RudrapatnaK. Shyamasundar, and Prasun Sinha, editors, *Distributed Computing and Networking*, volume 7730 of *Lecture Notes in Computer Science*, pages 225–239. Springer Berlin Heidelberg, 2013. doi:10.1007/978-3-642-35668-1_16.

2 Bowen Alpern and Fred B. Schneider. Defining Liveness. *Information Processing Letters*, 21(4):181–185, 1985.

3 Martin Biely, Peter Robinson, Ulrich Schmid, Manfred Schwarz, and Kyrill Winkler. Gracefully degrading consensus and k-set agreement in directed dynamic networks. *Theoretical Computer Science*, 726:41–77, 2018. doi:10.1016/j.tcs.2018.02.019.

4 Arnaud Casteigts, Paola Flocchini, Walter Quattrociocchi, and Nicola Santoro. Time-varying graphs and dynamic networks. *IJPEDS*, 27(5):387–408, 2012.

5 Bernadette Charron-Bost and André Schiper. The Heard-Of model: computing in distributed systems with benign faults. *Distributed Computing*, 22(1):49–71, April 2009. doi:10.1007/s00446-009-0084-6.

6 Étienne Coulouma, Emmanuel Godard, and Joseph G. Peters. A characterization of oblivious message adversaries for which Consensus is solvable. *Theoretical Computer Science*, 584:80–90, 2015. doi:10.1016/j.tcs.2015.01.024.

7 Reinhard Diestel. *Graph Theory (3rd ed.)*. Springer, 2006.

8 Danny Dolev, Cynthia Dwork, and Larry Stockmeyer. On the minimal synchronism needed for distributed consensus. *Journal of the ACM*, 34(1):77–97, January 1987.

9 Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. Consensus in the Presence of Partial Synchrony. *Journal of the ACM*, 35(2):288–323, April 1988.

10 Cynthia Dwork and Yoram Moses. Knowledge and common knowledge in a Byzantine environment: Crash failures. *Information and Computation*, 88(2):156–186, 1990. doi:10.1016/0890-5401(90)90014-9.

11 Tristan Fevat and Emmanuel Godard. Minimal Obstructions for the Coordinated Attack Problem and Beyond. In *25th IEEE International Symposium on Parallel and Distributed Processing, IPDPS 2011 - Conference Proceedings*, pages 1001–1011, 2011. doi:10.1109/IPDPS.2011.96.

12 Michael J. Fischer, Nancy A. Lynch, and M. S. Paterson. Impossibility of Distributed Consensus with one Faulty Process. *Journal of the ACM*, 32(2):374–382, April 1985.

13 Joseph Y. Halpern and Yoram Moses. Knowledge and common knowledge in a distributed environment. *J. ACM*, 37(3):549–587, 1990. doi:10.1145/79147.79161.

14 Wolfgang Kiess and Martin Mauve. A Survey on Real-world Implementations of Mobile Ad-hoc Networks. *Ad Hoc Networks*, 5(3):324–339, April 2007. doi:10.1016/j.adhoc.2005.12.003.

15 F. Kuhn and R. Oshman. Dynamic networks: Models and algorithms. *SIGACT News*, 42(1):82–96, 2011.

16 Fabian Kuhn, Nancy A. Lynch, and Rotem Oshman. Distributed computation in dynamic networks. In *STOC*, pages 513–522, 2010.

17 Fabian Kuhn, Rotem Oshman, and Yoram Moses. Coordinated consensus in dynamic networks. In *Proceedings of the 30th annual ACM SIGACT-SIGOPS symposium on Principles of distributed computing*, PODC '11. ACM, 2011.

18 Franck Legendre, Theus Hossmann, Felix Sutton, and Bernhard Plattner. 30 Years of Wireless Ad Hoc Networking Research: What about Humanitarian and Disaster Relief Solutions? What are we still missing? In *International Conference on Wireless Technologies for Humanitarian Relief (ACWR 11)*, Amrita, India, 2011. IEEE.

19 Ronit Lubitch and Shlomo Moran. Closed Schedulers: A Novel Technique for Analyzing Asynchronous Protocols. *Distributed Computing*, 8(4):203–210, June 1995. doi:10.1007/BF02242738.

20 Yoram Moses and Sergio Rajsbaum. A Layered Analysis of Consensus. *SIAM J. Comput.*, 31(4):989–1021, 2002.

21 Thomas Nowak, Ulrich Schmid, and Kyrill Winkler. Topological Characterization of Consensus Under General Message Adversaries. In *Proceedings of the 2019 ACM Symposium on Principles of Distributed Computing*, PODC '19, pages 218–227, New York, NY, USA, 2019. ACM. doi:10.1145/3293611.3331624.

22 Michel Raynal and Julien Stainer. Synchrony weakened by message adversaries vs asynchrony restricted by failure detectors. In *Proceedings ACM Symposium on Principles of Distributed Computing*, PODC'13, pages 166–175, 2013.

23 Nicola Santoro and Peter Widmayer. Time is not a healer. In *Proceedings of the 6th Annual Symposium on Theoretical Aspects of Computer Science (STACS'89)*, LNCS 349, pages 304–313, Paderborn, Germany, February 1989. Springer-Verlag.

24 Ulrich Schmid, Bettina Weiss, and Idit Keidar. Impossibility Results and Lower Bounds for Consensus under Link Failures. *SIAM Journal on Computing*, 38(5):1912–1951, 2009.

25 Kyrill Winkler, Manfred Schwarz, and Ulrich Schmid. Consensus in rooted dynamic networks with short-lived stability. *Distributed Computing*, 32(5):443–458, October 2019. doi:10.1007/s00446-019-00348-0.