האוניברסיטה העברית בירושלים
THE HEBREW UNIVERSITY OF JERUSALEM

GOOD, BETTER, BEST! — UNBEATABLE PROTOCOLS FOR CONSENSUS AND SET CONSENSUS

By

ARMANDO CASTAÑEDA,
YANNAI A. GONCZAROWSKI,
and YORAM MOSES

Discussion Paper # 653 (November 2013)

מרכז לחקר הרציונליות

CENTER FOR THE STUDY
OF RATIONALITY

Feldman Building, Givat-Ram, 91904 Jerusalem, Israel
PHONE: [972]-2-6584135 FAX: [972]-2-6513681
E-MAIL: ratio@math.huji.ac.il
URL: http://www.ratio.huji.ac.il/

Good, Better, Best! — Unbeatable Protocols for Consensus and Set Consensus[*]

Armando Castañeda[†]
Technion
armando@cs.technion.ac.il

Yannai A. Gonczarowski[‡]
The Hebrew University of Jerusalem
and Microsoft Research
yannai@gonch.name

Yoram Moses[§]
Technion
moses@ee.technion.ac.il

November 12, 2013

Abstract

While the very first consensus protocols for the synchronous model were designed to match the *worst-case* lower bound, deciding in exactly $\boldsymbol{t}+1$ rounds in all runs, it was soon realized that they could be strictly improved upon by *early stopping* protocols. These dominate the first ones, by always deciding in at most $\boldsymbol{t}+1$ rounds, but often much faster. A protocol is *unbeatable* if it can't be strictly dominated. Namely, if no protocol Q can decide strictly earlier than P against at least one adversary strategy, while deciding at least as fast as P in all cases. Unbeatability is often a much more suitable notion of optimality for distributed protocols than worst-case performance. Halpern, Moses and Waarts in [17], who introduced this notion, presented a general logic-based transformation of any consensus protocol to an unbeatable protocol that dominates it, and suggested a particular unbeatable consensus protocol. Their analysis is based on a notion of *continual common knowledge*, which is not easy to work with in practice. Using a more direct knowledge-based analysis, this paper studies unbeatability for both consensus and k-set consensus. We present unbeatable solutions to *non-uniform* consensus and k-set consensus, and *uniform* consensus in synchronous message-passing contexts with crash failures. Our consensus protocol strictly dominates the one suggested in [17], showing that their protocol *is* in fact beatable.

The k-set consensus problem is much more technically challenging than consensus, and its analysis has triggered the development of the topological approach to distributed computing. Worst-case lower bounds for this problem have required either techniques based on algebraic topology [13], or reduction-based proofs [1, 12]. Our proof of unbeatability is purely combinatorial, and is a direct, albeit nontrivial, generalization of the one for consensus. We also present an alternative topological unbeatability proof that allows to understand the connection between the connectivity of protocol complexes and the decision time of processes. All of our protocols make use of a notion of a ***hidden path*** of nodes relative to a process i at time m, in which a value unknown to i at m may be seen by others. This is a structure that can implicitly be found in lower bound proofs for consensus going back to the '80s [7]. Its use in our protocols sheds light on the mathematical structure underlying the consensus problem and its variants.

For the synchronous model, only solutions to the *uniform* variant of k-set consensus have been offered. Based on our unbeatable protocols for uniform consensus and for non-uniform k-set consensus, we present a uniform k-set consensus protocol that strictly dominates all known solutions to this problem in the synchronous model.

Keywords: Consensus, k-set consensus, uniform consensus, majority consensus, optimality, knowledge, topology.

[*] Part of the results of this paper were announced, without details and without proof, as a brief announcement in PODC 2013 [2].

[†] Supported in part at the Technion by an Aly Kaufman Fellowship.

[‡] Supported in part by ISF grant 230/10, by the Google Inter-university center for Electronic Markets and Auctions, and by the European Research Council under the European Community's Seventh Framework Programme (FP7/2007-2013) / ERC grant agreement no. [249159].

[§] Supported in part by ISF grant 1520/11.

1 Introduction

Following [18], we say that a protocol P is a ***worst-case optimal*** solution to a decision task S in a given model if it solves S, and decisions in P are always taken no later than the *worst-case* lower bound for decisions in this problem. The very first consensus protocols were worst-case optimal, deciding in exactly $\boldsymbol{t}+1$ rounds in all runs [7, 25]. It was soon realized, however, that they could be strictly improved upon by ***early stopping*** protocols [6]. The latter are also worst-case optimal, but they strictly improve upon the original ones because they can often decide much faster than the original ones. This paper is concerned with the study and construction of protocols that cannot be strictly improved upon, and are thus optimal in a much stronger sense.

In benign failure models it is typically possible to define the behaviour of the environment (i.e., the adversary) in a manner that is independent of the protocol, in terms of a pair $\alpha = (\vec{v}, \mathsf{F})$ consisting of a vector $\vec{v}$ of initial values and a failure pattern F. (A formal definition is given in Section 2.) A failure model $\mathcal{F}$ is identified with a set of (possible) failure patterns. For ease of exposition, we will think of such a pair $\alpha = (\vec{v}, \mathsf{F})$ as a particular *adversary*. A deterministic protocol P and an adversary α uniquely define a run $r = P[\alpha]$. With this terminology, we can compare the performance of different decision protocols solving a particular task in a given context $\gamma = (\vec{\mathsf{V}}, \mathcal{F})$, where $\vec{\mathsf{V}}$ is a set of initial vectors. A decision protocol Q ***dominates*** a protocol P in γ, denoted by $Q \preceq_{\gamma} P$ if, for all adversaries α and every process i, if i decides in $P[\alpha]$ at time m_i, then i decides in $Q[\alpha]$ at some time $m'_i \leq m_i$. Moreover, we say that Q ***strictly dominates*** P if $Q \preceq_{\gamma} P$ and $P \not\preceq_{\gamma} Q$. I.e., if it dominates P and for some $\alpha \in \gamma$ there exists a process i that decides in $Q[\alpha]$ *strictly before* it does so in $P[\alpha]$. In the crash failure model, the early-stopping protocols of [6] strictly dominate the original protocols of [25], which always decided at time $\boldsymbol{t}+1$. Nevertheless, these early stopping protocols may not be optimal solutions to consensus. Following [18] a protocol P is said to be an ***all-case optimal*** solution to a decision task S in a context γ if it solves S and, moreover, P dominates every protocol P' that solves S in γ. Dwork and Moses presented all-case optimal solutions to the *simultaneous* variant of consensus, in which all decisions are required to occur at the same time [10]. For the standard (*eventual*) variant of consensus, in which decisions are not required to occur simultaneously, Moses and Tuttle showed that no all-case optimal solution exists [21]. Consequently, Halpern, Moses and Waarts in [17] initiated the study of a notion of optimality that is achievable by eventual consensus protocols:

Definition 1 (Halpern, Moses and Waarts)**.** *A protocol P is an* ***unbeatable*** *solution to a decision task S in a context γ if P solves S in γ and no protocol Q solving S in γ strictly dominates P.*

Thus, P is unbeatable if for all protocols Q that solve S, if there exist an adversary α and process i such that i decides in $Q[\alpha]$ strictly earlier than it does in $P[\alpha]$, then there must exist some adversary β and process j such that j decides strictly earlier in $P[\beta]$ than it does in $Q[\beta]$. An unbeatable solution for S is $\preceq$-minimal among the solutions of S.[1]

Halpern, Moses and Waarts observed that for every consensus protocol P there exists an unbeatable protocol Q_P that dominates P. Moreover, they showed a two-step transformation that defines such a protocol Q_P based on P. This transformation is based on a notion of *continual* common knowledge that is computable, but not in a computationally-efficient manner. They also present a simple and efficient consensus protocol $P0_{\text{opt}}$ that is claimed to be unbeatable in the crash failure model.

This paper is concerned with the construction of concrete unbeatable protocols for a number of variants of consensus in synchronous, message-passing systems with crash failures. A new knowledge-based analysis [11, 16]

[1] All-case optimal protocols are called *"optimal in all runs"* in [10]. They are called *"optimum"* protocols by Halpern, Moses and Waarts in [17], and unbeatable ones are simply called *"optimal"* there.

allows a simpler and more intuitive approach to unbeatability than that used in [17]. Our main contributions are:

1. A knowledge-based approach to the design and presentation of consensus protocols is employed, based on the knowledge of preconditions principle.
2. The first unbeatable protocols are presented for (non-uniform) consensus and k-set consensus, and uniform consensus in the crash failure model. A protocol that strictly dominates all known protocols for uniform k-set consensus is presented.
3. The unbeatable consensus protocol strictly dominates the $P0_{\rm opt}$ protocol from [17], proving that $P0_{\rm opt}$, which was claimed to be beatable is *not* unbeatable.
4. The proofs of unbeatability are combinatorial, and do not require topological or reduction-based arguments even for the k-set consensus protocol. A second, topological, proof for the k-set consensus protocol is presented in the appendix, and is compared with the combinatorial proof. This is the first result that we know to have proofs of both kinds, and the comparison sheds light on the relationship between these two approaches.
5. While the proof for consensus is strikingly succinct, both the proofs for k-set consensus and for uniform consensus are technically challenging and highly nontrivial.

Full proofs of all technical claims stated in the paper are given in the Appendix.

In the rest of this section we sketch the intuition behind, and the structure of, our unbeatable protocols for consensus and k-set consensus in the crash failure model. The technical development substantiating this sketch is presented in the later sections.

Denote by $\exists v$ the fact that at least one of the processes started out with initial value v. In the standard (non-uniform) version of consensus, there is an *a priori* bound of ***t*** on the number of failures, initial values are $v_i \in \{0,1\}$, and the following properties must hold in every run r:

Agreement: All correct processes that decide in r must decide on the same value.
Decision: Every correct process must decide on some value, and
Validity: For every value v, a decision on v is allowed only if $\exists v$ holds.

Since $\exists v$ is a precondition for deciding v by the **Validity** property, a process cannot decide v unless it ***knows*** that $\exists v$ is true. Indeed, Dolev presented a consensus protocol B (for *"Beep"*) for the crash failure model in which a process decides on the particular value $v = 0$ if and only if it *knows* $\exists 0$ [8]. It follows from [17] that there must exist an unbeatable protocol dominating B. Clearly, B decides on 0 as soon as possible. When is the earliest time at which it is possible to decide 1 in a protocol in which decisions on 0 use the rule employed by B? Intuitively, a process should decide 1 once it knows that the rule for 0 will never hold for any correct process. Namely, let $\mathsf{never\text{-}known}(\exists 0)$ be the fact that no correct process will *ever* know that $\exists 0$ in the current run. Clearly, a process cannot decide 1 before it knows $\mathsf{never\text{-}known}(\exists 0)$, as this would allow a run violating the **Agreement** property. On the other hand, deciding 1 when $\mathsf{never\text{-}known}(\exists 0)$ is known *is sound*, since no process will ever decide 0, because knowing $\exists 0$ is a precondition for deciding 0. To turn this argument into a protocol, we need to present a concrete test for when a process knows $\exists 0$ and when it knows $\mathsf{never\text{-}known}(\exists 0)$. This is facilitated by considering message chains between processes at different times.

A *process-time node* is a pair $\langle i, m\rangle$ referring to process i at time m. We say that $\langle j, \ell\rangle$ is *seen by* $\langle i, m\rangle$ (in a given run r) if there exists a message chain from j at time ℓ to i at time m. It will be convenient to consider $\langle j, \ell\rangle$ as being *hidden from* $\langle i, m\rangle$ (in r) if both (a) i does not know that j has failed before time ℓ (it sees no node $\langle j', \ell\rangle$ that did *not* see $\langle j, \ell - 1\rangle$, which would prove that j failed earlier), and (b) $\langle j, \ell\rangle$ is not seen by $\langle i, m\rangle$. It is straightforward to efficiently compute whether $\langle j, \ell\rangle$ is *hidden from* $\langle i, m\rangle$ in a run with adversary α based on the communication graph $\mathcal{G}_\alpha$. Finally, we say that *there is a* ***hidden path*** *with respect to* $\langle i, m\rangle$ in run r if there exists a sequence of processes $j_0, \ldots, j_{m-1}, j_m$ such that $\langle j_\ell, \ell\rangle$ is hidden from $\langle i, m\rangle$, for all $\ell = 0, \ldots, m$. For an illustration of hidden paths (indeed, of three disjoint hidden paths) with respect to $\langle i, 2\rangle$, see Figure 1(b). Observe that there does not exist a hidden path with respect to $\langle i, m\rangle$ precisely if, for some time $\ell < m$, no node $\langle j, \ell\rangle$ is hidden from $\langle i, m\rangle$. I.e., if for all processes $j = 1, \ldots, n$, either $\langle j, \ell\rangle$ is seen by $\langle i, m\rangle$, or process i knows at time m that j crashed *before* time ℓ.

A process i knows $\exists 0$ (and so can decide 0) at time m iff it starts with initial value 0, or if some $\langle j, 0\rangle$ for a process j with initial value 0 is seen by $\langle i, m\rangle$. For deciding 1, a process knows $\mathsf{never\text{-}known}(\exists 0)$ exactly if it knows that ***no active process*** currently knows $\exists 0$. Based on this, we show that a process i knows $\mathsf{never\text{-}known}(\exists 0)$ at time m precisely if both (a) i does not know $\exists 0$, and (b) no hidden path w.r.t. $\langle i, m\rangle$ exists. As we show in Section 3, this protocol can be efficiently implemented without the use of large messages. The resulting protocol is shown in Section 3 to be unbeatable. It is the first unbeatable protocol for consensus.

For k-set consensus the set $\mathtt{V}$ of possible initial values contains at least the $k+1$ values, $\{0, \ldots, d\}$, $d \geq k$, the **Validity** and **Decision** properties are as in consensus, and the **Agreement** property is replaced by

***k*-Agreement:** The correct processes that decide in r decide on at most k distinct values.

As in the case of consensus, the **Validity** condition implies that knowing $\exists v$ is also a precondition for deciding v in this variant of consensus. Our unbeatable solution to k-set consensus is a natural generalization of the one for consensus. Define $v \in \mathtt{V}$ to be a ***low*** value if $v \in \{0, \ldots, k-1\}$. At the first instance at which a process i sees a low value, it decides on the minimal low value it has seen. If i has not seen a low value by time m, it can decide on a value provided that there do not exist k process-disjoint hidden paths with respect to $\langle i, m\rangle$. Again, this translates into a simple condition regarding the existence of at least k hidden nodes from $\langle i, m\rangle$ at all times $\ell = 0, \ldots, m$. When this condition holds, i decides on the minimal value that it has seen. As discussed above, proving unbeatability of this protocol (Theorem 4) is a nontrivial challenge.

It is often of interest to consider *uniform* consensus [3, 9, 15, 20, 26, 27] in which the **Agreement** property is replaced by

Uniform Agreement: The processes that decide in r must all decide on the same value.

This forces correct processes and faulty ones to act in a consistent manner. This requirement makes sense only in a setting where failures are benign, and all processes that decide do so according to the protocol. Uniformity may be desirable when elements outside the system can observe decisions, as in distributed databases when decisions correspond to commitments to values. As we shall see, the uniformity constraint strengthens the preconditions for decision, resulting in slower protocols. Therefore, it should be avoided if possible. We present the first unbeatable protocol for uniform consensus. While it is both conceptually and structurally similar to our unbeatable consensus protocol, the proof of its unbeatability (Theorem 5) is significantly more subtle.

In an asynchronous setting, any non-uniform consensus protocol must also solve uniform consensus. Since the study of k-set consensus was initially performed in an asynchronous setting, the common version of k-set consensus in the literature is a uniform variant, in which k-Agreement is replaced by

Uniform *k*-Agreement: The processes that decide in r decide on at most k distinct values.

We present a protocol for uniform k-set consensus, generalizing our unbeatable uniform consensus protocol and building upon our unbeatable (non-uniform) k-set consensus one. This protocol strictly dominates all existing protocols in the literature [4, 12, 14, 24], and matches the worst-case bounds for this problem. Whether this protocol is unbeatable remains an open question.

2 Preliminary Definitions

Our model of computation is a synchronous, message-passing model with benign crash failures. A system has $n \geq 2$ processes denoted by $\mathsf{Procs} = \{1, 2, \ldots, n\}$. Each pair of processes is connected by a two-way communication link, and each message is tagged with the identity of the sender. They share a discrete 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 $m+1$ taking place between time m and time $m+1$. Each process starts in some *initial state* at time 0, usually with an *input value* of some kind. In every round, each process first sends a set of messages to other processes, then receives messages sent to it by other processes during the same round, and then performs some local computation based on the messages it has received.

A faulty process fails by *crashing* in some round $m \geq 1$. It behaves correctly in the first $m-1$ rounds and sends no messages from round $m+1$ on. During its crashing round m, the process may succeed in sending messages on an arbitrary subset of its links. We assume that at most $\boldsymbol{t} \leq n-1$ processes fail in any given execution.

A ***failure pattern*** describes how processes fail in an execution. It is a layered graph F whose vertices are process-time pairs $\langle i, m\rangle$ for $i \in \mathsf{Procs}$ and $m \geq 0$. Such a vertex denotes process i and time m. An edge has the form $(\langle i, m-1\rangle, \langle j, m\rangle)$ and it denotes the fact that a message sent by i to j in round m would be delivered successfully. Let $\mathsf{Crash}(\boldsymbol{t})$ denote the set of failure patterns in which at most $\boldsymbol{t}$ crash failures can occur. An ***input vector*** describes what input the processes receive in an execution. The only inputs we consider are initial values that processes obtain at time 0. An input vector is thus a tuple $(v_1, \ldots, v_n)$ where v_j is the input to process j. We think of the input vector and the failure pattern as being determined by an external scheduler, and thus a pair $\alpha = (\vec{v}, \mathsf{F})$ is called an *adversary*.

A ***protocol*** describes what messages a process sends and what decisions it takes, as a deterministic function of its local state at the start of a round and the messages received during a round. We assume that a protocol P has access to the values of n and $\boldsymbol{t}$, typically passed to P as parameters.

A ***run*** is a description of an infinite behaviour of the system. Given a run r and a time m, $r_i(m)$ denote the ***local state*** of process i at time m in r and the ***global state*** at time m is defined to be $r(m) = \langle r_1(m), r_2(m), \ldots, r_n(m)\rangle$. A protocol P and an adversary α uniquely determine a run, and we write $r = P[\alpha]$.

Since we restrict attention to benign failure models and focus on decision times and solvability in this paper, Coan showed that it is sufficient to consider *full-information* protocols (*fip*'s for short), defined below [5]. There is a convenient way to consider such protocols in our setting. With an adversary $\alpha = (\vec{v}, \mathsf{F})$ we associate a ***communication graph*** $\mathcal{G}_\alpha$, consisting of the graph F extended by labelling the initial nodes $\langle j, 0\rangle$ with the initial states v_j according to α. With every node $\langle i, m\rangle$ we associate a subgraph $\mathcal{G}_\alpha(i, m)$ of $\mathcal{G}_\alpha$, which we think of as i's *view* at $\langle i, m\rangle$. Intuitively, this graph will represent all nodes $\langle j, \ell\rangle$ from which $\langle i, m\rangle$ has heard, and the initial values it has seen. Formally, $\mathcal{G}_\alpha(i, m)$ is defined by induction on m. $\mathcal{G}_\alpha(i, 0)$ consists of the node $\langle i, 0\rangle$, labelled by the initial value v_i. Assume that $\mathcal{G}_\alpha(1, m), \ldots, \mathcal{G}_\alpha(n, m)$ have been defined, and let $J \subseteq \mathsf{Procs}$ be the set of processes j such that $j = i$ or $e_j = (\langle j, m\rangle, \langle i, m+1\rangle)$ is an edge of F. Then $\mathcal{G}_\alpha(i, m+1)$ consists of the node $\langle i, m+1\rangle$, the union of all graphs $\mathcal{G}_\alpha(j, m)$ with $j \in J$, and the edges $e_j = (\langle j, m\rangle, \langle i, m+1\rangle)$ for all $j \in J$. We say that (j, ℓ) is *seen* by $\langle i, m\rangle$ if (j, ℓ) is a node of $\mathcal{G}_\alpha(i, m)$. Note that this occurs exactly if F allows a (Lamport) message chain starting at $\langle j, \ell\rangle$ and ending at $\langle i, m\rangle$.

A full-information protocol P is one in which at every node $\langle i, m\rangle$ of a run $r = P[\alpha]$ the process i constructs

$\mathcal{G}_\alpha(i,m)$ after receiving its round m nodes, and sends $\mathcal{G}_\alpha(i,m)$ to all other processes in round $m+1$. In addition, P specifies what decisions i should take at $\langle i,m\rangle$ based on $\mathcal{G}_\alpha(i,m)$.[2] Full-information protocols thus differ only in the decisions taken at the nodes. Let $\mathsf{d}(i,m)$ be the history of decisions taken by i up to time m. Thus, in a run $r = P[\alpha]$, we define the local state $r_i(m) = \langle \mathsf{d}(i,m), \mathcal{G}_\alpha(i,m)\rangle$.

2.1 Knowledge

Our construction of unbeatable protocols will be assisted and guided by a knowledge-based analysis, in the spirit of [11, 16]. We now define only what is needed for the purposes of this paper. For a comprehensive treatment, the reader is referred to [11]. Runs are dynamic objects, changing from one time point to the next. E.g., at one point process i may be undecided, while at the next it may decide on a value. Similarly, the set of initial values that i knows about, or has seen, may change over time. In addition, whether a process knows something at a given point can depend on what is true in other runs in which the process has the same information. We will therefore consider the truth of facts at *points* (r,m)—time m in run r, with respect to a set or runs R (which we call a ***system***). The systems we will be interested will have the form $R_P = R(P,\gamma)$ where P is a protocol and $\gamma = \gamma(\mathtt{V}^n, \mathcal{F})$ is the set of all adversaries that assign initial values from $\mathtt{V}$ and failures according to $\mathcal{F}$. We will write $(R,r,m) \models A$ to state that fact A holds, or is satisfied, at (r,m) in the system R.

The truth of some facts can be defined directly. For example, the fact $\exists v$ will hold at (r,m) in R if some process had initial value v in r. We say that *(satisfaction of)* a fact A is ***well-defined in*** R if for every point (r,m) with $r \in R$ we can determine whether or not $(R,r,m) \models A$. Satisfaction of $\exists v$ is thus well defined. We will write K_iA to denote that ***process*** i ***knows*** A, and define:

Definition 2 (Knowledge)**.** *Suppose that A is well-defined in R. Then:*

$$(R,r,m) \models K_iA \quad \textit{iff} \quad (R,r',m) \models A \;\; \textit{for all} \;\; r' \in R \;\; \textit{such that} \;\; r_i(m) = r'_i(m).$$

Thus, if A is well-defined in R then Definition 2 makes K_iA well-defined in R. The definition can then be applied recursively, to define the truth of K_jK_iA etc. Moreover, any boolean combination of well-defined facts is also well-defined. Knowledge has been used to study a variety of problems in distributed computing. We will make use of the following fundamental connection between knowledge and action in distributed systems. We say that a fact A is a ***precondition*** for process i performing action σ in R if $(R,r,m) \models A$ whenever i performs σ at a point (r,m) of R.

Theorem 1 (Knowledge of Preconditions, [22])**.** *Assume that $R_P = R(P,\gamma)$ is the set of runs of a deterministic protocol P. If A is a precondition for i performing σ in R_P, then K_iA is a precondition for i performing σ in R_P.*

3 Unbeatable Consensus

We are now ready to apply knowledge to design an unbeatable protocol for consensus. We start with the standard version of consensus defined in the Introduction, and consider the crash failure context $\gamma^{\mathrm{cr}} = \langle \mathtt{V}^n, \mathsf{Crash}(\boldsymbol{t})\rangle$, where $\mathtt{V} = \{0,1\}$ — initial values are binary bits. Every protocol P in this setting determines a system $R_P = R(P,\gamma)$. The **Validity** property of consensus states that $\exists v$ is a precondition for deciding v. Theorem 1 immediately implies:

Lemma 1. *$K_i\exists v$ is a precondition for i deciding on value v in any protocol satisfying the* **Validity** *property.*

[2] Observe that in benign models fip's do not involve exponentially large states nor exponentially large messages. In the crash failure model processes need only send the new edges and nodes that the learn about in every round, rather than the graph $\mathcal{G}_\alpha(i,m)$.

Since we restrict attention to full-information protocols, $(R_P, r, m) \models K_i \exists v$ exactly if a node $\langle j, 0\rangle$ with initial value v is seen by $\langle i, m\rangle$. For if not, then a run r' of the same protocol exists with $r'_i(m) = r_i(m)$ in which all initial values are $\neq v$. Notice that this depends only on the adversary $\alpha = (\vec{v}, \mathsf{F})$: If $r = P[\alpha]$ and $r' = Q[\alpha]$ then, for all i and m we have $(R_P, r, m) \models K_i \exists v$ iff $(R_Q, r', m) \models K_i \exists v$.

While $K_i \exists v$ is a necessary condition for deciding v, if $K_i \exists 0$ is used as a sufficient condition for $\mathsf{decide}(0)$ then $K_i \exists 1$ cannot be sufficient for $\mathsf{decide}(1)$, since this would violate **Agreement**: Everyone would decide on their own value at time 0. The following is a consensus protocol in which decisions on 0 are performed as soon as possible:

Protocol P_0 (for an undecided process i at time m):

```
if      $K_i \exists 0$          then decide(0)
elseif  $m = \boldsymbol{t} + 1$    then decide(1)
```

P_0 is essentially the early stopping protocol from [6]. We know from [17] that there exists an unbeatable solution to consensus that dominates P_0. A key step in establishing unbeatability in this case is based on the following lemma (see Appendix A for proofs):

Lemma 2. *If $Q \preceq P_0$ solves consensus, then every active process i decides 0 in Q when $K_i \exists 0$ first holds.*

By the **Agreement** property, a precondition for deciding 1 is that no correct process ***ever*** decides 0. By Lemma 1, in consensus protocols that dominate P_0 processes decide 0 as soon as they know $\exists 0$. It follows that a precondition for deciding 1 is that no correct process will *ever* know $\exists 0$ (denoted by $\mathsf{never\text{-}known}(\exists 0)$). Indeed, by the Knowledge of Preconditions Theorem 1, a process deciding 1 must know this fact. It turns out that this is equivalent to knowing that no active process *currently knows* $\exists 0$. Using this we can show:

Lemma 3. *In a full-information protocol in γ^{cr}, the following facts are equivalent at time m:*

- $K_i(\mathsf{never\text{-}known}(\exists 0))$ *and*
- $\neg K_i \exists 0$ & *there is no hidden path w.r.t.* $\langle i, m\rangle$.

As long as there is a hidden path w.r.t. $\langle i, m\rangle$, process i considers it possible that some process currently knows $\exists 0$. On such a path is excluded, it knows that it is safe to decide 1. This leads to an unbeatable protocol in which decisions on 0 occur as soon as possible, and on 1 as soon as a process knows that 0 will never be decided on:

Protocol $\textsc{Opt}_0$ (for an undecided process i at time m):

```
if      $K_i \exists 0$                                      then decide(0)
elseif  no hidden path w.r.t. $\langle i, m\rangle$ exists    then decide(1)
```

Since we assume for simplicity that communication in our protocols is according to the full-information protocol, we only specify how processes decide. By Lemmas 2 and 3, we have

Theorem 2. $\textsc{Opt}_0$ *is an unbeatable consensus protocol in* γ^{cr}.

It is interesting to compare $\textsc{Opt}_0$ with the protocol $P0_{\mathrm{opt}}$ that was claimed by [17] to be unbeatable. Both protocols decide 0 when $\exists 0$ is known, but they differ in the rule for deciding 1. In $P0_{\mathrm{opt}}$ a process decides 1 following a round in which it has not discovered a new failure. This condition implies the nonexistence of a hidden path, but is strictly weaker than it. E.g., in a run in which all initial nodes are seen at $\langle i, 2\rangle$ but i has seen one failure in each of the first two rounds, process i decides in $\textsc{Opt}_0$ but does not decide in $P0_{\mathrm{opt}}$. As a result, we have

Corollary 1. *The protocol $P0_{\mathrm{opt}}$ presented in [17] is **not** unbeatable.*

Neiger and Bazzi in [23] extend the results in [17], to the case of $\mathtt{V} = \{0, \ldots, d\}$ for $d > 1$. We remark that $\textsc{Opt}_0$ can readily be extended to cover the case in which $\mathtt{V}$ contains $\{0, \ldots, d\}$ for $d > 1$. The rule for 0 is unchanged, and if no hidden path exists a process can decide on the minimal value it has seen. Thus, a process decides v when it knows $\exists v$ and that correct processes will never see a smaller value. We call this protocol $\textsc{Opt}_{\min}$. In Section 4, we show how to extend $\textsc{Opt}_{\min}$ to the general case of k-set consensus.

3.1 Majority Consensus

Can we obtain other unbeatable consensus protocols? Clearly, the symmetric protocol $\textsc{Opt}_1$, obtained from $\textsc{Opt}_0$ by reversing the roles of 0 and 1, is unbeatable and neither dominates, nor is dominated by, $\textsc{Opt}_0$. Of course, $\textsc{Opt}_0$ and $\textsc{Opt}_1$ are extremely biased, each deciding on its favourite value if at all possible, even if only one process has it as an initial value. One may argue that it is natural, and may be preferable in many applications, to seek a more balanced solution, in which minority values are not favoured. Fix $n > 0$ and define the fact "$\mathsf{Maj} = 0$" to be true if more than $n/2$ initial values are 0, while "$\mathsf{Maj} = 1$" is true if at least $n/2$ values are 1. Finally, for a node $\langle i, m\rangle$, we define $Maj\langle i, m\rangle \triangleq 0$ if more than half of the processes whose initial value is known to i at time m have initial value 0; $Maj\langle i, m\rangle \triangleq 1$ otherwise. Consider the following protocol:

Protocol $\textsc{Opt}_{\mathsf{Maj}}$ (for an undecided process i at time m):

```
if     $K_i(\mathsf{Maj} = 0)$                          then decide(0)
elseif $K_i(\mathsf{Maj} = 1)$                          then decide(1)
elseif no hidden path w.r.t. $\langle i, m\rangle$ exists   then decide($Maj\langle i, m\rangle$).
```

Theorem 3. *If $t > 0$, then $\textsc{Opt}_{\mathsf{Maj}}$ is an unbeatable consensus protocol in γ^{cr}.*

Thus, $\textsc{Opt}_{\mathsf{Maj}}$ is an unbeatable consensus protocol that satisfies a much stricter validity condition than consensus:

> **Majority Validity:** For $v \in 0, 1$, if more than half of the processes are both correct and have initial value v, then all processes that decide in r must decide v.

4 Unbeatable Set Consensus

In this section we present an unbeatable protocol, $\textsc{Opt}_{\min\text{-}k}$, for (non-uniform) k-set consensus. Recall that for k-set consensus the **Agreement** property of consensus is replaced with the weaker ***k*-Agreement** property: the correct processes decide at most k distinct values. Since the **Validity** property of consensus is still required, Lemma 1 applies for k-set consensus as well.

Our protocol $\textsc{Opt}_{\min\text{-}k}$ generalizes the unbeatable consensus protocol $\textsc{Opt}_{\min}$ in Section 3. In $\textsc{Opt}_{\min\text{-}k}$, a process i decides on a ***low*** value (i.e. a value in $\{0, \ldots, k-1\}$) as soon as possible, namely, the first time $K_i \exists v$ holds, and decides on a ***high*** value $w \in \mathtt{V} \setminus \{0, \ldots, k-1\}$, as soon as it knows that no k values smaller than w will be decided on. Recall that $\mathtt{V} = \{0, \ldots, d\}$ for some $d \geq k$.

Definition 1. *Let r be a run, let k be a natural number, let i be a process and let m be a time. We define the following notations, in which r is implicit.*

1. *$Vals\langle i, m\rangle \triangleq \{v : K_i \exists v$ holds at time $m\}$,*
2. *$Min\langle i, m\rangle \triangleq \min Vals\langle i, m\rangle$,*
3. *$Low\langle i, m\rangle \triangleq Vals\langle i, m\rangle \cap \{0, \ldots, k-1\}$, and*
4. *Process i is called **low** at time m if $Low\langle i, m\rangle \neq \emptyset$; Otherwise, we say that i is **high** at m.*

As already mentioned, in $\textsc{Opt}_{\text{min-}k}$ low nodes decide immediately. In order to formalize the decision rule for high nodes, we first formalize the notion of the amount of process-disjoint hidden paths with respect to a node.

Definition 2. *Let i be a process and let m be a time. We define the* ***hidden capacity*** *of $\langle i,m\rangle$ (in given run) to be the maximum number c such that for every time $\ell \leq m$, there exist c distinct processes $i_1^\ell,\ldots,i_c^\ell$ such that $\langle i_1^\ell,\ell\rangle$ is hidden from $\langle i,m\rangle$, for all $\ell \leq m$. The nodes i_b^ℓ are said to be* ***witnesses to the hidden capacity of*** *$\langle i,m\rangle$.*

Analogously to hidden paths, as illustrated in Fig. 1 in the Appendix B a hidden capacity of c indicates that as many as c unknown low values may exist in the system. (See Lemma 9 in the Appendix B). As with hidden paths, it is straightforward to compute whether the hidden capacity of a node $\langle i,m\rangle$ in a run with adversary α based on the communication graph $\mathcal{G}_\alpha$. The hidden capacity of $\langle i,m\rangle$ can also be very efficiently calculated from the hidden capacity of $\langle i,m-1\rangle$ using auxiliary data calculated during the calculation of the latter. Using these definitions, we phrase a protocol for k-set consensus.

Protocol $\textsc{Opt}_{\text{min-}k}$ (for an undecided process i at time m):

```
    if  $i$ is low or $i$ has hidden capacity $< k$   then  decide($Min\langle i,m\rangle$)
```

The main technical challenge in proving that $\textsc{Opt}_{\text{min-}k}$ is unbeatable is, roughly speaking, showing that e.g. in the scenario depicted in Fig. 1, each of the "hidden" processes at time $m=2$ decides on the unique low value known to it, in *any* protocol that dominates $\textsc{Opt}_{\text{min-}k}$. (See Lemma 10 in Appendix B). We conclude that if i is high, then it cannot decide without violating ***k*-Agreement**. (See Lemma 11 in Appendix B). We give two proofs for Lemma 10. The first is completely constructive, and devoid of any topological arguments (Appendix B), while the second is a topological one (Appendix B.1). To the best of our knowledge, this is the first result in this field to be given proofs of both kinds, and a comparative reading sheds light on the relationship between these two dissimilar approaches. Our topological proof reasons in a novel way about subcomplexes of the protocol complex; see Appendix B for details and a discussion.

The above analysis implies that no k-set consensus protocol can strictly dominate $\textsc{Opt}_{\text{min-}k}$. Thus, to prove that $\textsc{Opt}_{\text{min-}k}$ is unbeatable, it is enough to show that it indeed solves k-set consensus.

Lemma 4. *$\textsc{Opt}_{\text{min-}k}$ solves k-set consensus. Furthermore, all processes decide in $\textsc{Opt}_{\text{min-}k}$ by time $\lfloor \frac{f}{k} \rfloor + 1$ at the latest.*

The proof of Lemma 4 sheds light on an inductive epistemic definition of $\textsc{Opt}_{\text{min-}k}$, formalizing the intuitive discussion from the beginning of this section. Assume that the decision rules for all values $w < v$ have been defined. Define the decision rule for v as: i decides on v as soon as it knows that (a) v is valid and (b) no more than $k-1$ values $< v$ will ever be decided upon.

Theorem 4. *$\textsc{Opt}_{\text{min-}k}$ is an unbeatable k-set consensus protocol in γ^{cr}.*

5 Unbeatable Uniform Consensus

Under crash failures, a process generally does not know whether or not it is correct. Indeed, so long as it has not seen ***t*** other processes crash, the process may (for all it knows) crash in the future. As a result, $K_i\exists 0$—the rule for deciding 0 in $\textsc{Opt}_0$—is an inappropriate rule for deciding 0 in any uniform consensus protocol. This is because a process starting with 0 immediately decides 0 with this rule, and may immediately crash. If all other processes have 1, all other decisions can only be on 1. Of course, $K_i\exists 0$ is still a precondition for deciding 0, but it can be strengthened. Denote by $\exists\mathsf{correct}(v)$ the fact "some ***correct*** process knows $\exists v$". We can show the following:

Lemma 5. $K_i\exists\mathsf{correct}(v)$ *is a precondition for i deciding v in any protocol solving Uniform Consensus.*

Lemma 6. *Let $r \in R_P = R(P, \gamma^{\mathrm{cr}})$ and assume that i knows of* ***d*** *failures at (r, m). Then $(R_P, r, m) \models K_i\exists\mathsf{correct}(v)$ iff one of (a) $m > 0$, i is active at m and $(R_P, r, m-1) \models K_i\exists v$, or (b) $(R_P, r, m) \models K_i(K_j\exists v$ held at time $m-1)$ for at least $(\boldsymbol{t}-\boldsymbol{d})$ distinct processes j, holds.*

It is easy to check that at time $\boldsymbol{t}+1$ the fact $K_i\exists v$ holds exactly if at least one of (a) or (b) does; thus, starting at that time $K_i\exists v$ and $K_i\exists\mathsf{correct}(v)$ are equivalent. As in the case of consensus, we note that if by time $\boldsymbol{t}+1$ we do not have $K_i\exists 0$ (equivalently, $K_i\exists\mathsf{correct}(0)$), then we never will. We thus phrase the following *beatable* algorithm, analogous to P_0, for Uniform Consensus; in this protocol, $K_i\exists\mathsf{correct}(0)$ (the precondition for deciding 0 in uniform consensus) replaces $K_i\exists 0$ (the precondition in consensus) as the decision rule for 0. The decision rule for 1 remains the same. Note that $K_i\exists\mathsf{correct}(0)$ can be efficiently checked, by Lemma 6.

Protocol U-P_0 (for an undecided process i at time m):

```
if     $K_i\exists\mathsf{correct}(0)$   then decide(0)
elseif $m = \boldsymbol{t}+1$            then decide(1).
```

Following a similar line of reasoning that lead us to obtain OPT$_0$, we use Lemma 3 to obtain the following unbeatable uniform consensus protocol.

Protocol U-OPT$_0$ (for an undecided process i at time m):

```
if     $K_i\exists\mathsf{correct}(0)$                                          then decide(0)
elseif no hidden path w.r.t. $\langle i, m\rangle$ exists and $\neg K_i\exists 0$   then decide(1).
```

Theorem 5. U-OPT$_0$ *is an unbeatable* ***uniform*** *consensus protocol in γ^{cr}. Moreover,*

- *If $f \geq t-1$, then all decisions are made by time $f+1$ at the latest.*
- *Otherwise, all decisions are made by time $f+2$ at the latest.*

Hidden paths again play a central role. Indeed, as in the construction of OPT$_0$ from P_0, the construction of U-OPT$_0$ from U-P_0 involved some decisions on 1 being moved forward in time, by means of the last condition, checking the absence of a hidden path. (Decisions on 0 cannot be moved up, as they are taken as soon as the precondition for deciding 0 holds.)

Despite the similarity in the design and the structure of the two protocols, the proof of unbeatability for U-OPT$_0$ is much more subtle and technically challenging than that for OPT$_0$. This is in a sense since in a uniform consensus protocol dominating U-OPT$_0$ (unlike the case of a consensus protocol dominating OPT$_0$), gaining knowledge even of an initial value of 0 that is known by a nonfaulty process, no longer implies that some process has already decided on 0. As a result, the possibility of dominating U-OPT$_0$ by switching 0 decisions to 1 decisions needs to be explicitly rejected. This is done by employing reachability arguments essentially establishing the existence of the continual common knowledge conditions of [17] (see the proofs in Appendix C for details).

5.1 Uniform Set Consensus

We now consider *uniform* k-set consensus. We present a protocol called U-PROT$_{\text{min-}k}$ that generalizes U-OPT$_0$ to k values (i.e. for $k = 1$, it behaves exactly like U-OPT$_0$). While in the protocol OPT$_{\text{min-}k}$ (which is defined above for non-uniform consensus) an undecided process i decides on its minimal value if and only if at the time of the decision i is low or has hidden capacity $< k$, in U-PROT$_{\text{min-}k}$ an undecided process i decides on a value v if and only if v is the minimal value s.t. i knows that both a) v was at some stage the minimal value known to a process was low or had hidden capacity $< k$ and b) v will be known to all processes deciding strictly after i.

Protocol U-PROT$_{\text{min-}k}$ (for an undecided process i at time m):

```
if $\big(i$ is low or has hidden capacity $< k\big)$ and $K_i\exists\mathsf{correct}(\mathit{Min}\langle i,m\rangle)$    then $\mathsf{decide}_{\mathit{Min}\langle i,m\rangle}$
elseif $m>0$ and $\big(\langle i,m-1\rangle$ was low or had hidden capacity $< k\big)$    then $\mathsf{decide}_{\mathit{Min}\langle i,m-1\rangle}$
elseif $m=\lfloor\frac{t}{k}\rfloor+1$    then $\mathsf{decide}_{\mathit{Min}\langle i,m\rangle}$
```

As shown by Theorem 6, U-PROT$_{\text{min-}k}$ meets the worst-case lower bounds proven in [13, 1] for uniform k-set consensus (see Appendix D for the proof of Theorem 6).

Theorem 6. U-PROT$_{\text{min-}k}$ *solves* ***uniform*** *k-set consensus in γ^{cr}. Moreover,*

- *If $f=t-1\equiv\ 0 \bmod k$, then all decisions are made by time $\frac{f}{k}+1$ at the latest.*
- *Otherwise, all decisions are made by time $\min\{\lfloor\frac{t}{k}\rfloor+1,\lfloor\frac{f}{k}\rfloor+2\}$ at the latest.*

We emphasize that U-PROT$_{\text{min-}k}$ strictly dominates all existing uniform k-set consensus protocols in the literature [4, 12, 14, 24]. As in the case of our unbeatable protocols for consensus, uniform consensus, and (nonuniform) k-set consensus, the dependence of our protocols on hidden capacity and hidden paths rather than on the number of failures seen often yields much faster stopping times. Thus, in runs r with the property that every correct process discovers exactly k new failures in each of the first $\left\lfloor\frac{f}{k}\right\rfloor$ rounds, all protocols in [4, 12, 14, 24] will decide in more than $\left\lfloor\frac{f}{k}\right\rfloor$ rounds. In contrast, for many of these runs the protocol U-PROT$_{\text{min-}k}$ may be able to decide in as few as 2 rounds. At this point, however, we have been unable to resolve the following

Open Question: *Is* U-PROT$_{\text{min-}k}$ *an unbeatable solution to uniform k-set consensus in γ^{cr}?*

6 Discussion

Unbeatability is a natural optimality criterion for distributed protocols. It formalizes the intuition that a given protocol cannot be strictly improved upon, which is significantly stronger than saying that it is worst-case optimal. When an all-case optimal solution exists, as for simultaneous consensus, an unbeatable protocol will be all-case optimal. We have presented the first unbeatable protocols for consensus, uniform consensus and k-set consensus. In addition, we suggested a protocol for uniform k-set consensus, that strictly dominates all known protocols for the problem.

Our particular notion of unbeatability, due to Halpern, Moses and Waarts, is based on a natural and commonly accepted notion of domination among protocols [6, 12, 23, 24]. Indeed, the original early-stopping protocol P_0 was favoured because it improved on the earlier protocols, and our OPT$_0$ improves upon it. Nevertheless, our notion of unbeatability is just one criterion of this type. Alternative ways to compare runs of different protocols may make sense, depending on the application. One could, for example, compare runs in terms of the time at which the last correct process decides, rather than when each of the processes does. Let us call the corresponding notion ***last-decider unbeatability***.[3] We note that last-decider unbeatability neither implies, nor is implied by, the notion of unbeatability studied so far in this paper. Nevertheless, none of the protocols previously proposed in the literature for the problems we have studied are last-decider unbeatable. In Appendix E we show that all of our unbeatable protocols are also last-decider unbeatable:

Theorem 7. *The protocols* OPT$_0$, OPT$_{\mathsf{Maj}}$, OPT$_{\text{min-}k}$, *and* U-OPT$_0$ *are also last-decider unbeatable for consensus, majority consensus, k-set consensus and uniform consensus, respectively.*

In summary, this paper used a knowledge-based analysis to obtain the first ever *unbeatable* protocols for a range of agreement problems in the crash failure model. It identified and exposed hidden paths and hidden capacity as

[3] This notion was suggested to us by Michael Schapira; we thank him for the insight.

patterns that play an essential role in determining whether decisions can be taken. As a side effect, we were able to design an unbeatable protocol, $\textsc{Opt}_{\mathsf{Maj}}$, for ***majority consensus***, which provides more balanced decision behaviour than previously available in early stopping protocols.

For ease of exposition and analysis, all of our protocols were specified under the assumption of full-information message passing. In fact, they can all be implemented in such a way that any process sends any other process a total of $O(n \log n)$ bits throughout the protocol (see Lemma 20 in Appendix E). Thus, unbeatability is attainable at a modest price. Our study opens the way to many possible extensions. For one, we have left open the question of whether $\textsc{u-Prot}_{\text{min-}k}$ is unbeatable. But unbeatability can be sought in other models, and for other problems. Arguably, to be really good, a distributed protocol better be unbeatable!

References

[1] Dan Alistarh, Seth Gilbert, Rachid Guerraoui, and Corentin Travers. Of choices, failures and asynchrony: The many faces of set agreement. *Algorithmica*, 62(1-2):595–629, 2012.

[2] Armando Castañeda, Yannai A. Gonczarowski, and Yoram Moses. Brief announcement: Pareto-optimal solutions to consensus and set consensus. In *Proc. 32nd ACM Symp. on Principles of Distributed Computing*, pages 113–115, 2013.

[3] Bernadette Charron-Bost and André Schiper. Uniform consensus is harder than consensus. *J. Algorithms*, 51(1):15–37, 2004.

[4] Soma Chaudhuri, Maurice Herlihy, Nancy A. Lynch, and Mark R. Tuttle. Tight bounds for *k*-set agreement. *J. ACM*, 47(5):912–943, 2000.

[5] 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.

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

[7] 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.

[8] Danny Dolev. Beep protocols (personal communication).

[9] Partha Dutta, Rachid Guerraoui, and Bastian Pochon. Tight bounds on early local decisions inuniform consensus. In *DISC*, pages 264–278, 2003.

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

[11] R. Fagin, J. Y. Halpern, Y. Moses, and M. Y. Vardi. *Reasoning about Knowledge*. MIT Press, 2003.

[12] Eli Gafni, Rachid Guerraoui, and Bastian Pochon. The complexity of early deciding set agreement. *SIAM J. Comput.*, 40(1):63–78, 2011.

[13] Rachid Guerraoui, Maurice Herlihy, and Bastian Pochon. A topological treatment of early-deciding set-agreement. *Theor. Comput. Sci.*, 410(6-7):570–580, 2009.

[14] Rachid Guerraoui and Bastian Pochon. The complexity of early deciding set agreement: How can topology help? *Electr. Notes Theor. Comput. Sci.*, 230:71–78, 2009.

[15] Vassos Hadzilacos. On the relationship between the atomic commitment and consensus problems. In *Fault-Tolerant Distributed Computing*, pages 201–208, 1986.

[16] 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.

[17] Joseph Y. Halpern, Yoram Moses, and Orli Waarts. A characterization of eventual byzantine agreement. *SIAM J. Comput.*, 31(3):838–865, 2001.

[18] Maurice Herlihy, Yoram Moses, and Mark R. Tuttle. Transforming worst-case optimal solutions for simultaneous tasks into all-case optimal solutions. In *PODC*, pages 231–238, 2011.

[19] Maurice Herlihy, Sergio Rajsbaum, and Mark R. Tuttle. Unifying synchronous and asynchronous message-passing models. In *PODC*, pages 133–142, 1998.

[20] Idit Keidar and Sergio Rajsbaum. A simple proof of the uniform consensus synchronous lower bound. *Inf. Process. Lett.*, 85(1):47–52, 2003.

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

[22] Yoram Moses. *Knowledge and Distributed Coordination*. Morgan Claypool. in preparation.

[23] 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.

[24] Philippe Raipin Parvédy, Michel Raynal, and Corentin Travers. Early-stopping *k*-set agreement in synchronous systems prone to any number of process crashes. In *PaCT*, pages 49–58, 2005.

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

[26] Michel Raynal. Optimal early stopping uniform consensus in synchronous systems with process omission failures. In *In Proceedings of the Sixteenth Annual ACM Symposium on Parallelism in Algorithms and Architectures*, pages 302–310. ACM Press, 2004.

[27] Xianbing Wang, Yong Meng Teo, and Jiannong Cao. A bivalency proof of the lower bound for uniform consensus. *Inf. Process. Lett.*, 96(5):167–174, 2005.

A Proofs of Section 3 — Consensus

Lemma 1 follows from Theorem 1.

Proof of Lemma 2. Assume that $Q \preceq P_0$ solves consensus. We prove the claim for all processes i and adversaries α, by induction on the time m at which $K_i\exists 0$ first holds in $Q[\alpha]$ (and, equivalently, in $P_0[\alpha]$).

Base ($m = 0$): Since i decides at time 0 in $P_0[\alpha]$, it must decide at time 0 in $Q[\alpha]$ as well. At this point we have $K_i\exists 0$. Since process i knows only its initial value at time 0, it follows that i has initial value 0. Hence, $K_i\exists 1$ does *not* hold at 0. By **Validity**, i decides 0 in $Q[\alpha]$.

Inductive step ($m > 0$): Assume that the claim holds for all times $< m$. Recall that m is the first time at which $K_i\exists 0$ first holds. In an fip, this can only happen if i receives a message with 0 from some process j who was active at time $m - 1$. Thus, $K_j\exists 0$ holds at time $m - 1$, and by the induction hypothesis, j decides 0 when $K_j\exists 0$ first holds, which is no later than time $m - 1$ in $Q[\alpha]$. Observe that in γ^{cr}, if i receives a message from j in round m, then i cannot know that j is faulty at time m: An execution β in which the adversary does not crash j at all, and that otherwise agrees with α is both legal (initial values are in $\{0, 1\}$ and no more than t crash failures) and $Q[\beta]$ is indistinguishable to i from $Q[\alpha]$ at time m. Since Q satisfies **Agreement**, i cannot decide 1 at or before time m. Moreover, by **Validity**, $K_i\exists 0$ is a precondition for process i deciding 0, and so i cannot decide 0 before time m. Since Q dominates P_0, process i must decide by time m under $Q[\alpha]$, and it thus decides 0 at m. □

Proof of Lemma 3. If $K_i\exists 0$, then we immediately have $\neg K_i(\textsf{never-known}(\exists 0))$; the fact that the existence of a hidden path implies the possibility for a correct process to know $\exists 0$ is generalized by (and implied by) Lemma 9, and so its proof is omitted here. The second direction is generalized (and implied) by the proof of the k-Agreement property in Lemma 4, and so its proof is omitted here as well. □

Theorem 2 follows from Lemmas 2 and 3.

A.1 Proofs for Majority Consensus

The proof of Theorem 3 is based on two lemmas:

Lemma 7 (Decision at time 1)**.** *Assume that $n > 2$ and $t > 0$. Let $Q \preceq \textsc{Opt}_{\mathsf{Maj}}$ solve Consensus and let $r = Q[\alpha]$ be a run of Q. Let i be a process and let v be a value. If $K_i(\mathsf{Maj}\!=\!v)$ at $(r, 1)$, then Q makes i decide v before or at time 1 in r.*

Proof. By definition of $\textsc{Opt}_{\mathsf{Maj}}$, i decides in $\textsc{Opt}_{\mathsf{Maj}}[\alpha]$ by time 1, since $K_i(\mathsf{Maj}\!=\!v)$ holds at $(\textsc{Opt}_{\mathsf{Maj}}[\alpha], 1)$. As $Q \preceq \textsc{Opt}_{\mathsf{Maj}}$, we thus have that i must decide upon some value in $r = Q[\alpha]$ before or at time 1. Thus, it is enough to show that i cannot decide $1\!-\!v$ up to time 1 in r.

We prove the claim by induction on $n - |Z_i|$, where Z_i is defined to be the set of processes k with initial value v, s.t. $\langle k, 0\rangle$ is seen by $\langle i, 1\rangle$. As $K_i(\mathsf{Maj}\!=\!v)$ at $(r, 1)$, we have $|Z_i| \geq \frac{n}{2}$ and so $2 \leq |Z_i| \leq n$.

Base: $|Z_i| = n$. In this case, all initial values are v, and so by **Validity** i cannot decide $1\!-\!v$ in r.

Step: Let $2 \leq \ell < n$ and assume that the claim holds whenever $|Z_i| = \ell + 1$. Assume that $|Z_i| = \ell$. As $|Z_i| \geq 2$, there exists $j \in Z_i \setminus \{i\}$. We reason by cases.

I. If there exists a process k s.t. $\langle k, 0\rangle$ is hidden from $\langle i, 1\rangle$, then there exists a run r' of Q, s.t. *i)* $r'_i(1) = r_i(1)$, *ii)* neither i nor j fail in r', *iii)* k has initial value 0 in r', and *iv)* $Z_j = Z_i \cup \{k\}$ in r'. (Note that by definition, Z_i has the same value in both r and r'.) By the induction hypothesis (switching the roles of i and j), j decides v before or at time 1 at r', and therefore by **Agreement**, i cannot decide $1-v$ in r', and hence it does not decide $1-v$ up to time 1 in r.

II. If there exists a process $k \neq i$ with initial value $1-v$, s.t. $\langle k, 0\rangle$ is seen by $\langle i, 1\rangle$, then $k \notin \{i, j\}$. Hence, as $t > 0$, there exists a run r' of Q, s.t. *i)* $r'_i(1) = r_i(1)$, *ii)* neither i nor j fail in r', *iii)* $\langle k, 0\rangle$ is hidden from $\langle j, 1\rangle$ in r', and *iv)* $Z_j = Z_i$ in r'. (Once again, Z_i has the same value in both r and r'.) By Case I (switching the roles of i and j), j decides v before or at time 1 in r', and therefore by **Agreement**, i cannot decide $1-v$ in r', and hence it does not decide $1-v$ up to time 1 in r.

III. Otherwise, $\langle k, 0\rangle$ is seen by $\langle i, 1\rangle$ for all processes k, and k has initial value v for all processes $k \neq i$. As $|Z_i| < n$, we have that i has initial value $1-v$. Thus, there exists a run r' of Q, s.t. *i)* $r'_i(1) = r_i(1)$, *ii)* $f = 0$ in r', and *iii)* $Z_j = Z_i$ in r'. (Once again, Z_i has the same value in both r and r'.) As i has initial value $1-v$ in r' as well, by Case II (switching the roles of i and j), j decides v before or at time 1 in r', and therefore by **Agreement**, i cannot decide $1-v$ in r', and hence it does not decide $1-v$ up to time 1 in r, and the proof is complete. □

Lemma 8 (No Earlier Decisions). *Assume that $n > 2$ and $t > 0$. Let $Q \preceq \textsc{Opt}_{\mathsf{Maj}}$ solve Consensus and let r be a run of Q. Let i be a process and let m be a time, s.t. $\neg K_i(\mathsf{Maj} = 0)$ and $\neg K_i(\mathsf{Maj} = 1)$. If there exists a hidden path w.r.t. $\langle i, m\rangle$, then i does not decide at (r, m).*

Proof. Let $v \in \{0, 1\}$ be a value. We show that i does not decide v at (r, m).

We first consider the case in which $m = 0$. In this case, there exists a run r' of Q s.t. *i)* $r'_i(0) = r_i(0)$, *ii)* $\mathsf{Maj} = 1 - v$, and *iii)* $f = 0$. As $f = 0$ and $\mathsf{Maj} = 1 - v$ in r', we have $K_i(\mathsf{Maj} = 1 - v)$ at $(r', 1)$, and therefore, by Lemma 7, i decides $1-v$ before or at 1 in r'; therefore, i does not decide v at $(r', 0)$, and hence neither does it decide v at $(r, 0) = (r, m)$.

We turn to the case in which $m > 0$. As there exists a hidden path w.r.t. $\langle i, m\rangle$, for every $0 \leq \ell \leq m$ there exists a process b_ℓ s.t. $\langle b_\ell, \ell\rangle$ is hidden from $\langle i, m\rangle$. Thus, there exists a run r' of Q s.t. *i)* $r'_i(m) = r_i(m)$, *ii)* $\mathsf{Maj} = 1 - v$, *iii)* $\langle b_1, 1\rangle$ sees $\langle k, 0\rangle$ for all processes k (and therefore $K_{b_1}(\mathsf{Maj} = 1 - v)$ at 1, *iv)* $\langle b_\ell, \ell\rangle$ is seen by $\langle b_{\ell+1}, \ell + 1\rangle$ for every $1 \leq \ell < m$, and *v)* neither b_m nor i fail in r'. We show by induction that b_ℓ decides $1-v$ before or at ℓ in r', for every $1 \leq \ell \leq m$.

Base: By Lemma 7, b_1 decides $1-v$ before or at 1 in r'.

Step: Let $1 < \ell \leq m$ and assume that $b_{\ell-1}$ decides $1-v$ before or at $\ell - 1$ in r'. As $\langle b_{\ell-1}, \ell - 1\rangle$ is seen by $\langle b_\ell, \ell\rangle$ in r', there exists a run $r'' = Q[\gamma]$ of Q, s.t. *i)* $r''_{b_\ell}(\ell) = r'_{b_\ell}(\ell)$, and *ii)* Neither $b_{\ell-1}$ nor b_ℓ fail in r''. As $\langle b_{\ell-1}, \ell - 1\rangle$ is seen by $\langle b_\ell, \ell\rangle$, and as $r''_{b_\ell}(\ell) = r'_{b_\ell}(\ell)$, $b_{\ell-1}$ decides $1-v$ before or at $\ell - 1$ in r'' as well. As neither $b_{\ell-1}$ nor b_ℓ fail in r'', by **Agreement** b_ℓ does not decide v before or at ℓ in r''. As $\langle b_1, 1\rangle$ is seen by $\langle b_\ell, \ell\rangle$ in r', we have $K_{b_\ell}(\mathsf{Maj} = 1 - v)$ at (r', ℓ), and therefore also at (r'', ℓ). Thus, b_ℓ decides in $(\textsc{Opt}_{\mathsf{Maj}}[\gamma], \ell)$, and therefore b_ℓ decides before or at ℓ in r'', and so it decides $1-v$ before or at ℓ in r'', and hence it also decides $1-v$ before or at ℓ in r', and the proof by induction is complete.

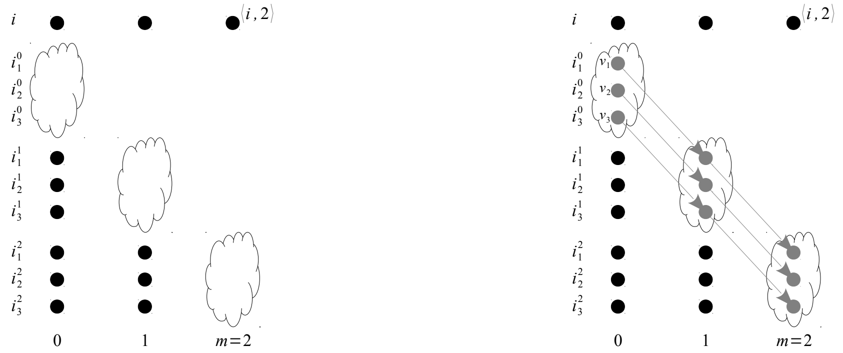
As we have shown, b_m decides $1-v$ in r'. As neither b_m nor i fail in r', by **Agreement** i does not decide v at (r', m), and therefore neither does it decide v at (r, m). □

We can now prove Theorem 3.

Proof of Theorem 3. **Agreement**, **Decision** and **Validity** are straightforward and left to the reader. If $n > 2$, then unbeatability follows from Lemma 8. If $n = 1$, then it is straightforward to verify that the single process always decides at time 0, and so $\textsc{Opt}_{\mathsf{Maj}}$ cannot be improved upon. Finally, if $n = 2$, then it is easy to check that $\textsc{Opt}_{\mathsf{Maj}}$ is equivalent to $\textsc{Opt}_1$, and so is unbeatable. □

We note that the condition $t > 0$ in Theorem 3 cannot be dropped if $n > 2$. Indeed, if $t = 0$ and $n > 2$, then both $\textsc{Opt}_0$ and $\textsc{Opt}_1$ (in which some decisions are made at time 0, and the rest — at time 1) dominate $\textsc{Opt}_{\mathsf{Maj}}$ (in which all decisions are made at time 1).

B Proofs of Section 4 — Set Consensus



(a) $\langle i, 2\rangle$ has hidden capacity 3.

(b) A run i considers at 2 to be possible, in which v_1, v_2, v_3 are held by distinct processes at 2.

Figure 1: A hidden capacity of $c{=}3$ at time $m{=}2$ indicates that any arbitrary c values unknown to i may exist in the system, each held by a distinct process.

Remark 1. *By definition, $Vals\langle i, m\rangle = \emptyset$ (and thus i is high) for all times $m < 0$, for all processes i in all runs.*

Remark 2. *The hidden capacity of i in r is (weakly) decreasing as a function of time.*

Lemma 9 (See Figure 1)**.** *For any fip, let r be a run, let i be a process and let m be a time such that i is active at time $m-1$. Let c be the hidden capacity of $\langle i, m\rangle$ and let i_b^ℓ, for all $\ell \le m$ and $b = 1, \ldots, c$, be as in Definition 2. For every c values $v_1, \ldots, v_c$ of $\mathtt{V}$, there exists a run r' of the protocol such that $r'_i(m) = r_i(m)$, and for all ℓ and b, (a) $v_b \in Vals\langle i_b^\ell, \ell\rangle$ (b) $Vals\langle i_b^\ell, \ell\rangle \setminus \{v_b\} \subseteq Vals\langle i, \ell\rangle$, and (c) $\langle i_b^\ell, \ell\rangle$ has hidden capacity $\ge c-1$ witnessed by $i_{b'}^{\ell'}$ for $b' \neq b$ and $\ell' \le \ell$.*

Proof. It is enough to define r' up to the end of round m. Let i^ℓ_b, for all $\ell \leq m$ and $b = 1, \ldots, c$, be as in Definition 2. We define r' to be the same as r, except for the following possible changes (possible, as they may or may not hold in r):

1. i^0_b is assigned the initial value b, for every b.
2. For every $0 \leq \ell < m$ and every b, the process i^ℓ_b fails at ℓ, at which it successfully sends a message only to $i^{\ell+1}_b$.
3. For every $0 < \ell \leq m$ and every b, the process i^ℓ_b receives, until time $\ell - 1$ inclusive, the exact same messages as in r. (By definition, $\langle i^\ell_b, \ell - 1\rangle$ is seen by $\langle i, m\rangle$ in r, and thus it indeed receives messages in r until time $\ell - 1$, inclusive.) At time ℓ, the process i^ℓ_b receives the exact same messages as i, and, in addition, a message from i and the aforementioned message from $i^{\ell-1}_b$.

It is straightforward to check, using backward induction on ℓ, that in r', each $\langle i^\ell_b, \ell\rangle$ is not seen up to time m by any process other than $i^{\ell'}_b$ for $\ell' > \ell$, and is thus hidden from $\langle i, m\rangle$ and from $i^{\ell'}_{b'}$ for all $b' \neq b$ and for all ℓ'. Thus, for all b and ℓ, $\langle i^\ell_b, \ell\rangle$ has hidden capacity $\geq c - 1$ witnessed by $i^{\ell'}_{b'}$ for $b' \neq b$ and $\ell' \leq \ell$.

We now show that none of the above changes alter the state of i at m. By definition, each $\langle i^\ell_b, \ell\rangle$ is hidden from $\langle i, m\rangle$ in r, and as explained above — in r' as well. We note that all modifications above affect a process i^ℓ_b only at or after time ℓ, and as this process at these times is not seen by $\langle i, m\rangle$ in either run, these modifications do not alter the state of i at m.

Let $b \in \{1, \ldots, c\}$. By definition of r', we have $\mathit{Vals}\langle i^0_b, 0\rangle = \{v_b\}$. Since for every $\ell > 0$, $\langle i^\ell_b, \ell\rangle$ receives a message from $\langle i^{\ell-1}_b, \ell - 1\rangle$, we have by induction that $v_b \in \mathit{Vals}\langle i^\ell_b, \ell\rangle$ for all ℓ.

We now complete the proof by showing by induction that for all ℓ, $\mathit{Vals}\langle i^\ell_b, \ell\rangle \subseteq \mathit{Vals}\langle i, \ell\rangle \cup \{v_b\}$.[4]

Base: $\mathit{Vals}\langle i^0_b, 0\rangle = \{v_b\} \subseteq \mathit{Vals}\langle i, 0\rangle \cup \{v_b\}$.

Step: Let $\ell > 0$. Let $v \in \mathit{Vals}\langle i^\ell_b, \ell\rangle$. If $v \in \mathit{Vals}\langle i^\ell_b, \ell - 1\rangle$, then $v \in \mathit{Vals}\langle i, \ell\rangle$, as v^ℓ_b is non-faulty at $\ell - 1$ and thus its message is received by $\langle i, \ell\rangle$. Otherwise, i^ℓ_b is informed that $\exists v$ by a message it receives at ℓ. By definition of r', a message received by $\langle i^\ell_b, \ell\rangle$ is exactly one of the following:

- A message received by $\langle i, \ell\rangle$. In this case, $v \in \mathit{Vals}\langle i, \ell\rangle$ as well.
- A message sent by $\langle i, \ell - 1\rangle$. In this case, we trivially have $v \in \langle i, \ell - 1\rangle \subseteq \mathit{Vals}\langle i, \ell\rangle$.
- A message sent by $i^{\ell-1}_b$. In this case, by the induction hypothesis,

$$v \in \mathit{Vals}\langle i^{\ell-1}_b, \ell - 1\rangle \subseteq \mathit{Vals}\langle i, \ell - 1\rangle \cup \{v_b\} \subseteq \mathit{Vals}\langle i, \ell\rangle \cup \{v_b\}.$$

Thus, the proof by induction, and thus the proof of the lemma, is complete. □

We now generalize Lemma 2 for k-set consensus. Lemma 10 performs this task.

Lemma 10. *Let P be a protocol solving k-set consensus. Assume that in P, every process i that is low at any*

[4] A similar argument to the one used below in fact further shows that for all $\ell > 0$ and for all b, $\mathit{Vals}\langle i^\ell_b, \ell\rangle = \mathit{Vals}\langle i, \ell\rangle \cup \{v_b\}$ in r' for all ℓ and b.

time m must decide by time m at the latest. Let i be a process and let m be a time. If the following conditions hold in a run r:

1. *i does not crash before m,*
2. *i is low at m for the first time,*
3. *$Low\langle i, m\rangle = \{v\}$ for some v (in particular, i has seen a single low value by time m),*
4. *$\langle i, m\rangle$ has hidden capacity $\geq k - 1$, and*
5. *there exist k distinct processes $j_1, \dots, j_k$ such that $\langle j_b, m - 1\rangle$ is high and $\langle j_b, m\rangle$ is hidden from $\langle i, m\rangle$, for all $b = 1, \dots, k$.*

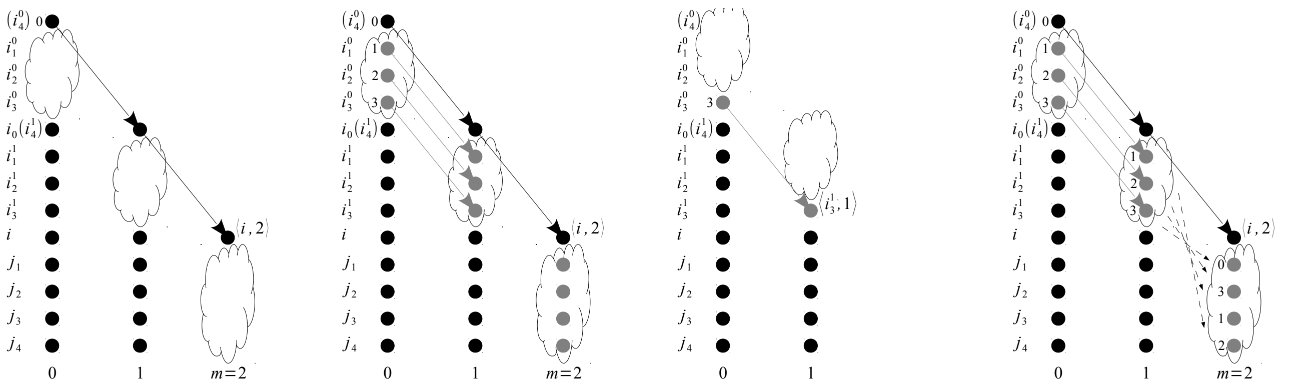
then i decides in P on its unique low value v at time m.

Remark 3. *The processes $j_1, \dots, j_k$ required by Condition 5 of Lemma 10 need not be disjoint from the processes $i^m_1, \dots, i^m_{k-1}$ required by Condition 4.*

Proof of Lemma 10. We prove the lemma by induction on m.

Base ($m = 0$): Since $K_i\exists v$ at time 0, the value v must be i's initial value, and thus $Vals\langle i, 0\rangle = \{v\}$. As $\langle i, m\rangle$ is low, i decides at 0. By the **Validity** property of P, it must decide on a value in $Vals\langle i, 0\rangle$, namely, on v.

Step ($m > 0$):



(a) r, as seen by $\langle i, 2\rangle$.

(b) The run r'.

(c) r', as seen by $\langle i^1_3, 1\rangle$. The induction hypothesis dictates i^1_3 decides 3 at 1.

(d) We aim to adjust the messages received by $j_1, \dots, j_4$ at 2 so that they collectively decide on all low values.

Figure 2: Lemma 10 induction step proof strategy (for $m = 2$, $k = 4$).

Let i^ℓ_b, for all $\ell \leq m$ and $b = 1, \dots, k-1$, be as in Definition 2. (See Figure 2(a).) Let r' be the run of P guaranteed to exist by Lemma 9, with respect to the values $\{0, \dots, k - 1\} \setminus \{v\}$. (See Figure 2(b).) As $j_1, \dots, j_k$ are seen by i up to time m, we assume w.l.o.g. that neither $j_1, \dots, j_k$ nor i ever fail in r'. We henceforth work in r'.

For readability, let us denote by i_w, for all $w \in \{0, \dots, k-1\} \setminus \{v\}$, the unique process among the $i^{m-1}_{b'}$ associated with the value w in the definition of r' by Lemma 9. Hence, $w \in Vals\langle i_w, m-1\rangle \cap \{0, \dots, k-1\} = Low\langle i_w, m-1\rangle$. By Condition 2, $\langle i, m - 1\rangle$ is high, and thus, by definition of i_w, $Low\langle i_w, m - 1\rangle = Vals\langle i_w, m - 1\rangle \cap$

$\{0,\ldots,k-1\} \subseteq (\mathit{Vals}\langle i,m-1\rangle \cap \{0,\ldots,k-1\}) \cup \{w\} = \mathit{Low}\langle i,m-1\rangle \cup \{w\} = \{w\}$. We conclude that $\mathit{Low}\langle i_w,m-1\rangle = \{w\}$.

As $\mathit{Low}\langle i,m\rangle \setminus \mathit{Low}\langle i,m-1\rangle = \{v\} \setminus \emptyset = \{v\}$, process i learned that $\exists v$ by a message it received at m. Let i_v denote the sender of this message. We thus trivially have that $v \in \mathit{Low}\langle i_v,m-1\rangle$. Furthermore, we have $\mathit{Low}\langle i_v,m-1\rangle \subseteq \mathit{Low}\langle i,m\rangle = \{v\}$, and thus $\mathit{Low}\langle i_v,m-1\rangle = \{v\}$.

Define $i_k^{m-1} \triangleq i_v$ and $v_k \triangleq v$. As $\mathit{Low}\langle i_k^{m-1},m-1\rangle = \{v\}$, for every $\ell < m-1$ there exists a process i_k^ℓ s.t. (a) $\langle i_k^\ell,\ell\rangle$ is seen by $\langle i_k^{\ell+1},\ell+1\rangle$ (and thus does not fail before ℓ) and (b) $v \in \mathit{Low}\langle i_k^\ell,\ell\rangle$ (and thus $\mathit{Low}\langle i_k^\ell,\ell\rangle = \{v\}$). (See Figure 2(b).) Let $w \in \{0,\ldots,k-1\} \setminus \{v\}$ and let $\ell < m$. As $\mathit{Low}\langle i_w,m-1\rangle = \{w\}$, and as $\mathit{Low}\langle i_k^\ell,\ell\rangle = \{v\} \neq \{w\}$, $\langle i_k^\ell,\ell\rangle$ is not seen by $\langle i_w,m-1\rangle$ and thus (as i_k^ℓ does not fail before ℓ), it is hidden from $\langle i_w,m-1\rangle$. Furthermore, as $\mathit{Low}\langle i_k^\ell,\ell\rangle = \{v\}$, it is distinct from all i_b^ℓ for $b < k$. Let now $w \in \{0,\ldots,k-1\}$. We conclude that $\langle i_w,m-1\rangle$ has hidden capacity $\geq k-1$ witnessed by i_b^ℓ for $\ell \leq m-1$ all for all b s.t. $v_b \neq w$. (See Figure 2(c).) Thus, by the induction hypothesis, i_w decides w by time $m-1$.

We now apply a sequence of consecutive possible changes to r' (possible, as they may or may not actually modify r'), numbered from k to 1. (See Figure 2(d).) For every $b = 1,\ldots,k$, change b possibly modifies only j_b, and only at times $\geq m$, and does not contradict the fact that i and all $j_1,\ldots,j_k$ never fail. Therefore, change b does not affect the state i or of $j_{b'}$'s up to time m, inclusive. Therefore, once change b is performed, the state of j_b at m is no longer affected by subsequent changes. As we show that following change b, j_b decides at m, and denote the value decided upon by v_b, we therefore have that the fact that j_b decides upon v_b at m at the latest continues to hold throughout the rest of the changes.

We now inductively describe the changes (recall that changes are performed starting with change k and concluding with change 1): Define $r^k \triangleq r'$. For every b, change b is applied to r^b to yield a run r^{b-1}. Let $b \in \{1,\ldots,k\}$ and assume that changes $k,\ldots,b+1$ were already performed, and that for each $b' > b$, we have that in $r^{b'-1}$ (and thus in r^b), $j_{b'}$ decides a low value $v_{b'}$ by m at the latest, such that $j_{b+1},\ldots,j_k$ are distinct of each other.

Change b: Let j_b never fail. Furthermore, let j_b receive at time m messages exactly from (a) $\{i_0,\ldots,i_{k-1}\} \setminus \{i_{v_{b+1}},\ldots,i_{v_k}\}$, (b) i, and (c) $j_1,\ldots,j_k$, except, of course, from j_b.

As i and $j_1,\ldots,j_k$ are all high at $m-1$, and as $\mathit{Low}\langle i_w,m-1\rangle = \{w\}$ for all w, we now have $\mathit{Low}\langle j_b,m\rangle = \{0,\ldots,k-1\} \setminus \{v_{b+1},\ldots,v_k\}$. In particular, as $b > 0$, $\langle j_k,m\rangle$ is low, and therefore must decide at m or before. We note that there exists a run s s.t. $s_{j_b}(m) = r_{j_b}^{b-1}(m)$, in which neither j_b, nor any of the processes from which it receives messages at m, ever fail. In this run, $j_{b+1},\ldots,j_k$ respectively decide on $v_{b+1},\ldots,v_k$, and $\{i_0,\ldots,i_{k-1}\} \setminus \{i_{v_{b+1}},\ldots,i_{v_k}\}$ decide on the rest of $\{0,\ldots,k-1\}$. Thus, by the ***k*-Agreement** property of P, j_b must decide in s on a value $v_b \in \{0,\ldots,k-1\}$. As $\mathit{Low}\langle j_b,m\rangle = \{0,\ldots,k-1\} \setminus \{v_{b+1},\ldots,v_k\}$, by the **Validity** property of P, we have that $v_b \neq \{v_{b+1},\ldots,v_k\}$. As $s_{j_b}(m) = r_{j_b}^{b-1}(m)$, j_b must decide on v_b in r^{b-1} as well and the proof by induction is complete.

By the above construction, $r_i^0(m) = r_i'(m) = r_i(m)$. Thus, it is enough to show that in r^0, i decides on v at m. We thus, henceforth, work in r^0. As in r, and thus also in r^0, $\langle i,m\rangle$ is low, i must decide by m at the latest. As all of $j_1,\ldots,j_k$ never fail, and furthermore, collectively decide on all of $\{0,\ldots,k-1\}$ (see Figure 2(d)), by the ***k*-Agreement** property of P, as i never fails, it must decide on a low value. By the **Validity** property, i must decide on a value known to it to exist. As $Low\langle i,m\rangle = \{v\}$ (in r, and thus also in r^0), we have that i decides v. As $v \notin Low\langle i,m-1\rangle$, by **Validity** we obtain that i does not decide before m and the proof is complete. □

Using Lemmas 9 and 10, we derive a necessary condition for deciding in $\textsc{Opt}_{\text{min-}k}$.

Lemma 11. *Let P be a protocol solving k-set consensus. Assume that in P, every process i that is low at any time m must decide by time m at the latest. Then no process decides in P as long as it is both high and has hidden capacity $\geq k$.*

Proof. Let r be a run of P, let i be a process and let m be a time s.t. $\langle i,m\rangle$ is high and has hidden capacity $\geq k$. Let i_b^ℓ, for all $\ell \leq m$ and $b = 1,\ldots,k$, be as in Definition 2. Let r' be the run of P guaranteed to exist by Lemma 9, with respect to the values $\{0,\ldots,k-1\}$, with i_b^ℓ associated with the value $b-1$ for all ℓ. As i_b^m, for all b, are seen by i up to time m, we assume w.l.o.g. that neither they nor i ever fail in r'. As $r'_i(m) = r_i(m)$, it is enough to show that i does not decide at m in r'. We thus, henceforth, work in r'.

Let $b \in \{0,\ldots,k-1\}$. By definition of r', $Low\langle i_b^m,m\rangle = Vals\langle i_b^m,m\rangle \cap \{0,\ldots,k-1\} \subseteq (Vals\langle i,m\rangle \cap \{0,\ldots,k-1\}) \cup \{b-1\} = Low\langle i,m\rangle \cup \{b-1\} = \{b-1\}$. As $b-1 \in Low\langle i_b^m,m\rangle$, we conclude that $Low\langle i_b^m,m\rangle = \{b-1\}$. If $m=0$, then we trivially have that i_b^m is low for the first time at m. Otherwise, as $\langle i,m\rangle$ is high, and as, by definition, $\langle i_b^m,m-1\rangle$ is seen by $\langle i,m\rangle$ (in r, and therefore in r'), we have that i_b^m is low at m for the first time as well. By definition of r', i_b^m has hidden capacity $\geq k-1$. By applying Lemma 10 with i and $\{i_{b'}^m\}_{b'\neq b}$ as $j_1,\ldots,j_k$, we thus obtain that i_b^m decides $b-1$ at m.

Thus, all of $\{0,\ldots,k-1\}$ are decided upon and so, by the ***k*-Agreement** property of P, i may not decide on any other value. As $\langle i,m\rangle$ is high, by the **Validity** property of P, i may not decide on any of $\{0,\ldots,k-1\}$ at m. Thus, i does not decide at m. □

Proof of Lemma 4. In some run of $\textsc{Opt}_{\text{min-}k}$, let i be a non-faulty process.

Decision: Let m be a time s.t. i has not decided until m, inclusive. Thus, $\langle i,m\rangle$ has hidden capacity $\geq k$. Let i_b^ℓ, for all $\ell \leq m$ and $b = 1,\ldots,k$, be as in Definition 2. By definition, i_b^ℓ, for every $\ell < m$ and $b = 1,\ldots,k$, fails at time ℓ. Thus, $k \cdot m \leq f$, where f is the number of failure in the current run. Thus, $m \leq \frac{f}{k}$, and therefore $m \leq \lfloor \frac{f}{k} \rfloor$. Therefore, i decides by time $\lfloor \frac{f}{k} \rfloor + 1$ at the latest.

Henceforth, let m be the decision time of i and let $v = Min\langle i,m\rangle$ be the value upon which i decides.

Validity: As $v = Min\langle i,m\rangle$, we have $v \in Vals\langle i,m\rangle$ and thus $K_i\exists v$ at m. Thus, $\exists v$.

***k*-Agreement**: It is enough to show that at most $k-1$ distinct values smaller than v are decided upon in the current run. Since i decides at m, $\langle i,m\rangle$ is either low or has hidden capacity $< k$. If $\langle i,m\rangle$ is low, then $v = Min\langle i,m\rangle \leq k-1$, and thus there do not exist more than $k-1$ distinct legal values smaller than v, let alone ones decided upon.

For the rest of this proof we assume, therefore, that $\langle i,m\rangle$ is high and has hidden capacity $< k$. As $\langle i,m\rangle$ does not have hidden capacity k, there exists $0 \leq \ell \leq m$ s.t. no more than $k-1$ processes at time ℓ are hidden from $\langle i,m\rangle$.

Let $w < v$ be a value decided upon by a non-faulty processor. Let j be this processor, and let m' be the time at which j decides on w. As $w < v$ and as $v = \mathit{Min}\langle i, m\rangle$, $\langle j, m'\rangle$ is not seen by $\langle i, m\rangle$. As j and i are both non-faulty, we conclude that $m' \geq m$, and thus $m' \geq \ell$. Let H be the set of all processes seen at ℓ by $\langle j, m'\rangle$. Since $m' \geq \ell$, We have $\mathit{Vals}\langle j, m'\rangle = \bigcup_{h \in H} \mathit{Vals}\langle h, \ell\rangle$. (Note that if $m' = \ell$, then $H = \{j\}$.) As $w = \mathit{Min}\langle j, m'\rangle$, we have $w = \mathit{Min}\langle h, \ell\rangle$ for some $h \in H$. As $w < v = \mathit{Min}\langle i, m\rangle$, we have $w \notin \mathit{Vals}\langle i, m\rangle$, and thus $\langle h, \ell\rangle$ is not seen by $\langle i, m\rangle$. As $\langle h, \ell\rangle$ is seen by $\langle j, m'\rangle$, h has not failed before ℓ, and thus $\langle h, \ell\rangle$ is hidden from $\langle i, m\rangle$. To conclude, we have shown that

$$w \in \big\{\mathit{Min}\langle h, \ell\rangle \mid \langle h, \ell\rangle \text{ is hidden from } \langle i, m\rangle\big\}.$$

As there are at most $k - 1$ processes hidden at ℓ from $\langle i, m\rangle$, we conclude that no more than $k - 1$ distinct values lower than v are decided upon by non-faulty processes, and the proof is complete. □

Theorem 4 follows from Lemmas 4 and 11.

B.1 A Combinatorial Topology Proof of Lemma 10

B.1.1 Basic Element of Combinatorial Topology

A *complex* is a finite set V and a collection of subsets $\mathcal{K}$ of V closed under containment. An element of V is called a *vertex* of $\mathcal{K}$, and a set in $\mathcal{K}$ is called a *simplex*. A (proper) subset of a simplex σ is called a *(proper) face*. The *dimension* $\dim \sigma$ is $|\sigma| - 1$. The dimension of a complex $\mathcal{K}$, $\dim \mathcal{K}$, is the maximal dimension of any of $\mathcal{K}$'s simplexes. A complex $\mathcal{K}$ is *pure* if all its simplexes have the same dimension.

For a simplex σ, let $\mathsf{Bd}\,\sigma$ denote the complex containing all proper faces of σ. If $\mathcal{K}$ and $\mathcal{L}$ are disjoint, their *join*, $\mathcal{K} * \mathcal{L}$, is the complex $\{\sigma \cup \tau : \sigma \in \mathcal{K} \wedge \tau \in \mathcal{L}\}$.

A *colouring* of a complex $\mathcal{K}$ is a map from the vertices of $\mathcal{K}$ to a set of *colours*. A simplex of $\mathcal{K}$ is *fully coloured* if its vertices are mapped to distinct colours.

Informally, a *subdivision* $\mathsf{Div}\,\sigma$ of σ is a complex constructed by subdividing each $\sigma' \subseteq \sigma$ into smaller simplexes. A subdivision $\mathsf{Div}\,\sigma$ maps each $\sigma' \subseteq \sigma$ to the pure complex $\mathsf{Div}\,\sigma'$ of dimension $\dim \sigma$ containing the simplexes that subdivide σ'. Thus, for all $\sigma', \sigma'' \subseteq \sigma$, $\mathsf{Div}\,\sigma' \cap \mathsf{Div}\,\sigma'' = \mathsf{Div}\,\sigma' \cap \sigma''$. For every vertex $v \in \mathsf{Div}\,\sigma$, its *carrier*, $\mathsf{Car}\,v$, is the face $\sigma' \subseteq \sigma$ of smallest dimension such that $v \in \mathsf{Div}\,\sigma'$.

The *barycentric* subdivision $\mathsf{Bary}\,\sigma$ of σ can be defined in many equivalent ways. Here we adopt the following combinatorial definition. $\mathsf{Bary}\,\sigma$ is defined inductively by dimension. For dimension 0, for every vertex v of σ, $\mathsf{Bary}\,v = v$. For dimension ℓ, $1 \leq \ell \leq \dim \sigma$, for every ℓ-face σ' of σ, for a new vertex $v = \sigma'$, $\mathsf{Bary}\,\sigma' = v * \mathsf{Bary}\,\mathsf{Bd}\,\sigma'$.

Let $\mathsf{Div}\,\sigma$ be a subdivision of σ. A *Sperner colouring* of $\mathsf{Div}\,\sigma$ is a colouring that maps every vertex $v \in \mathsf{Div}\,\sigma$ to a vertex in $\mathsf{Car}\,v$.

B.1.2 Proof of Lemma 10

Consider any run r. We proceed by induction on the time m.

For the base of the induction $m = 0$, if the four conditions holds for a process i at time 0, then it must be that i

starts in r with input v, and consequently $V\langle i,m\rangle = \{v\}$. Therefore, i decides v at time 0, since P satisfies the validity requirement of k-set consensus.

Let us assume the claim holds until time $m-1$. We prove it holds at m. Let i be a process that satisfies the four conditions at time m.

Without loss of generality, let us assume $L\langle i,m\rangle = \{0\}$. Let i_0 be a process such that i receives a message from i_0 at time m and $L\langle i_0, m-1\rangle = \{0\}$. We have $\langle i,m\rangle$ has hidden capacity greater or equal than $k-1$, thus, Lemma 9 implies that there exist a run r' indistinguishable to $\langle i,m\rangle$ such that there are $k-1$ processes $i_1,\ldots,i_{k-1}$ such that for each i_x, $1 \leq x \leq k-1$, $\langle i_x, m-1\rangle$ hidden to $\langle i,m\rangle$ and $L\langle i_x, m-1\rangle = \{x\}$.

By induction hypothesis, every i_x, $0 \leq x \leq k-1$, decides at time $m-1$, at the latest, on its unique low value x. We assume, for the sake of contradiction, that i decides on a non-low value at time m (if i decides before, it necessarily decides on a non-low value). For simplicity, let us assume i decides on k.

By hypothesis, there are k processes, $j_1,\ldots,j_k$, (distinct from i and i_x) such for each $1 \leq y \leq k$, $L\langle j_y, m-1\rangle = \emptyset$. Note that $k \in H\langle j_y, m-1\rangle$, for every j_y.

Below, we only consider runs in which a subset of $i_0,\ldots,i_{k-1}$ crash in round m and every j_y receives at least one message from some i_x; all other process do not crash in round m. Thus, $L\langle j_y, m\rangle \neq \emptyset$, for every j_y, and consequently it decides at time m, at the latest.

We now define a subdivision, $\mathsf{Div}\,\sigma$, of a k-simplex $\sigma = \{0,\ldots,k\}$, and then define a map δ from the vertices $\mathsf{Div}\,\sigma$ to states of i, i_x or j_y at time m. The mapping δ will be defined in a way that the decisions of the processes induce a Sperner colouring on $\mathsf{Div}\,\sigma$. Finally, we argue that, for every simplex $\tau \in \mathsf{Div}\,\sigma$, all its vertices are mapped to distinct compatible process states in some execution. Therefore, by Sperner's Lemma, there must be a k-dimensional simplex in $\mathsf{Div}\,\sigma$ in which $k+1$ distinct values are decided by distinct processes, thus reaching a contradiction.

Lemma 12 (Sperner's Lemma)**.** *Let* $\mathsf{Div}\,\sigma$ *be a subdivision with a Sperners's colouring* ζ*. Then,* ζ *defines an odd number of fully coloured* $(\dim\sigma)$*-simplexes.*

We construct $\mathsf{Div}\,\sigma$ inductively by dimension. The construction is a simple variant of the well-known barycentric subdivision (see Figure 3 (left)).

For dimension 0, for every vertex $v \in \sigma$, we define $\mathsf{Div}\,v = v$; hence $\mathsf{Car}\,v = v$. For every 1-face (edge) σ' of σ, if $k \notin \sigma'$ or $\sigma' = \{0,k\}$, then $\mathsf{Div}\,\sigma' = \sigma'$; otherwise, for a new vertex $v = \sigma'$, $\mathsf{Div}\,\sigma' = v * \mathsf{Div}\,\mathsf{Bd}\,\sigma'$. Note that $\mathsf{Car}\,v = \sigma'$. For every x-face σ' of σ, $2 \leq x \leq k$, if $k \notin \sigma'$, then $\mathsf{Div}\,\sigma' = \sigma'$; otherwise, for a new vertex $v = \sigma'$, $\mathsf{Div}\,\sigma' = v * \mathsf{Div}\,\mathsf{Bd}\,\sigma'$. Again note that $\mathsf{Car}\,v = \sigma'$ (see Figure 3 (center)).

We now define the mapping δ and the Sperner colouring of $\mathsf{Div}\,\sigma$, which is induced by the decision function ζ of P.

For every vertex $v \in \sigma$, $\mathsf{Div}\,v = v$. If $v \neq k$, then $\delta(v)$ is the state $\langle i_v, m\rangle$ in which i_v sends and receives all its messages, i.e. i_v does not crash in round m; otherwise, $\delta(v) = \langle i,m\rangle$ in r'. Note that for all $v \in \sigma$, $\zeta(\delta(v)) = v$, by induction hypothesis and because we assume i decides on k.

For every, y-face σ' of σ, $1 \leq y \leq k$, if there is a vertex $v \in \mathsf{Div}\,\sigma'$ with $\mathsf{Car}\,v = \sigma'$, then $v = \sigma'$ and $k \in \sigma'$. For such a vertex, we define $\delta(v)$ to be the state $\langle j_y, m\rangle$ in which (a) j_y receives a message from i_w, for every

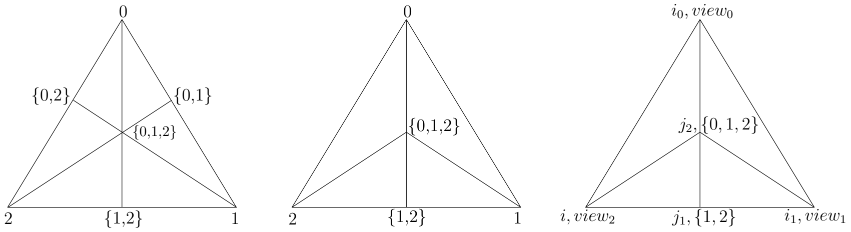


Figure 3: For dimension $k = 2$ and $\sigma = \{0, 1, 2\}$, the barycentric subdivision $\sigma = \{0, 1, 2\}$ appears at the left, while the subdivision $\mathsf{Div}\,\sigma$ appears at the center. In the subdivision at the right, the vertices are mapped to process states. For example, the triangle $\{\langle i_0, view_0\rangle, \langle i_1, view_1\rangle, \langle j_2, \{0, 1, 2\}\rangle\}$ corresponds to the execution in which i_0 and i_1 do not crash in round m, and hence j_1 receives 0 and 1, which are included in its view. Similarly, the triangle $\{\langle i_1, view_1\rangle, \langle j_1, \{1, 2\}\rangle, \langle j_2, \{0, 1, 2\}\rangle\}$ correspond to the execution in which i_1 does not crash in round m, while i_0 crashes and sends a message to j_2 and no message to j_1. The decisions of the processes induces an Sperner colouring: by assumption, i decides $k = 2$, and by induction hypothesis, i_0 and i_1 decided 0 and 1 at time $m - 1$; the rest of the processes have to decide at time m and they can only decide values in their views.

$w \in \sigma'$ (i_w may crash after sending a message to i_y), and (b) j_y does not receive any message from the i_x's whose subindexes do not appear in σ', namely, they crash in round m without sending a message to j_y. Observe that $L\langle i_y, m\rangle = \sigma' \setminus \{k\}$ and $H\langle i_y, m\rangle$ contains k and possible more high values distinct from k. Since P satisfies the validity requirement of k-set consensus, $\zeta(\delta(v))$ is any value in $V\langle i_y, m\rangle = L\langle i_y, m\rangle \cup H\langle i_y, m\rangle$. For now, we assume that if $\zeta(\delta(v)) \in H\langle i_y, m\rangle$, then $\zeta(\delta(v)) = k$, in other words, if i_y decides a high value, it decides on k; hence $\zeta(\delta(v)) \in \mathsf{Car}\, v$. Therefore, ζ defines a Sperner colouring for $\mathsf{Div}\,\sigma$. Later we explain that this assumption does not affect our argument below.

Consider a k-simplex $\tau \in \mathsf{Div}\,\sigma$. To show that δ maps the vertices of τ distinct process states, it is enough to see that for every $v \in \mathsf{Div}\,\sigma$, if $\dim \mathsf{Car}\, v = 0$, then $\delta(v)$ is a state of i or some i_x; and if $1 \leq \dim \mathsf{Car}\, v \leq k$, then $\delta(v)$ is a state of j_y, where $y = \dim \mathsf{Car}\, v$. And to show that δ map τ to states of an execution, note that if there is a $v \in \tau$ such that $\delta(v) = \langle i, m\rangle$, then the states in $\delta(\tau)$ correspond to an execution in which each j_y receives a subset of the messages from $i_0, \ldots, i_{k-1}$; otherwise, the states in $\delta(\tau)$ correspond to an execution in which some i_x's distinct from i_0 do not crash in round m (see Figure 3 (right)). Observe that in the second case, the state of i at time m in that execution is different from the state of i at time m in r', because in r' i only receives a message from i_0.

By Sperner's Lemma, there is at least one fully coloured k-simplex in $\mathsf{Div}\,\sigma$, and thus there is an execution of P in which $k + 1$ distinct values are decided at time m. A contradiction.

Finally, we assumed that if i_y decides a high value, it decides on k. Observe that if in $\mathsf{Div}\,\sigma$, we replace k with the actual decision of i_y, then, the number of distinct decision at the vertices of a simplex of $\mathsf{Div}\,\sigma$ can only increase. Thus, in any case, $\mathsf{Div}\,\sigma$ has a simplex with $k + 1$ distinct decisions. The lemma follows. □

B.2 A Discussion of Unbeatability and Connectivity

The topological proof of Lemma 10 is more than just a "trick" to prove the lemma. Actually, the proof shows what a topological analysis of unbeatable protocols is about.

In the protocol $\textsc{Opt}_{\text{min-}k}$, every process decides a low value as soon as possible, namely, at the very first time it knows there is low value. Therefore, if there is a k-set consensus protocol P that dominates $\textsc{Opt}_{\text{min-}k}$, then in P every process i that is low at any time m must decide by time m at the latest.

The key in the unbeatability proof for $\textsc{Opt}_{\text{min-}k}$ is Lemma 10, saying, intuitively, that in every k-set consensus protocol P with the property that at any time m in which a process i is low for the first time and has hidden capacity at least $k-1$, then i must decide on a low value. The topological proof essentially shows that i is forced to decide on a low value because the star complex, $\mathcal{K}$, of $\langle i, m\rangle$ in the protocol complex of P at time m, $\mathcal{P}_m$, is $(k-1)$-connected. Intuitively, $\mathcal{K}$ is the "part" of $\mathcal{P}_m$ containing all executions that are indistinguishable to $\langle i, m\rangle$. That $\mathcal{K}$ is $(k-1)$-connected is the reason the proof can map a subdivision of a k-simplexes to process states; indeed, the subdivision is mapped to a subcomplex of $\mathcal{K}$. It is well-known that $(k-1)$-connectivity precludes the existence of a decision function that maps process states to more than k values and avoids a simplex with $k+1$ distinct decisions at its vertices [19], namely, an execution with $k+1$ decided values. Therefore, i has no other choice than decide on a low value, because if it does not do it, its decision induces an Sperner colouring, which ultimately implies that the ***k*-Agreement** property is violated.

The previous discussion can be formalized in a lemma saying that if the hypothesis of Lemma 10 hold, then the star complex $\mathsf{St}(i, m, \mathcal{P}_m)$ is $(k-1)$-connected. Such a lemma can be proved using the techniques in [19].

It is worth noticing that in the previous analysis we only care about the connectivity of a proper subcomplex of the protocol complex, contrary to all known time complexity lower bound proofs [13, 19] for k-set consensus, which care about the connectivity of the whole protocol complex. There is no contradiction with this because these lower bounds proofs are about the time in which "all" processes can decide, which depends on the connectivity of the protocol complex in a given round. While unbeatability is a notion of optimality concerned with the time at which a "single" process can decide, which depends just on a subcomplex (the star complex of a given process state) of the protocol complex in a given round.

This analysis sheds light on the open question in [14] about how to extend previous topology techniques to deal with optimality of protocols. In summary, while all-decide lower bounds have to do with the whole protocol complex, optimal-single-decision lower bounds have to do with just subcomplexes of the protocol complex. Our topological proof of unbeatability here is the first proof that we are aware of that makes this distinction.

C Proofs of Section 5 — Uniform Consensus

We note that while the assumption $t<n$ simplifies presentation throughout the proofs below, the case $t=n$ can be analysed via similar tools.

Proof of Lemma 5. Let P be a uniform consensus protocol, and let r be a run of P such that $(R_P, r, m) \not\models K_i\exists\mathsf{correct}(v)$. Thus, there exists a run $r' \in P[\alpha']$ such that $r_i(m) = r'_i(m)$ and $(R_P, r', m) \not\models \exists\mathsf{correct}(v)$. Consider the adversary β that agrees with α' up to time m, and in which all active but faulty processes at (r', m) crash at time m without sending any messages. $\beta \in \gamma^{\text{cr}}$ because it has a legal input vector (identical to α'), and at most ***t*** crash failures, as it has the same set of faulty processes as $\alpha' \in \gamma^{\text{cr}}$. It follows that $r'' = P[\beta]$ is a run of P. Since β agrees with α' on the first m rounds, we have that $r''_i(m) = r'_i(m)$. Nonetheless, no correct process will ever know $\exists v$ in r'', and thus by **Validity** no correct process ever decides v in r''. By decision, all correct processes thus decide not on v. By **Uniform Agreement**, and as $t<n$ (i.e. there are correct processes), i cannot decide on v in r'', and thus, as $r''_i(m) = r'_i(m) = r_i(m)$, it cannot decide on v in r at m. □

Before moving on to prove Lemma 6. We first introduce some notation.

Definition 3. *For a node $\langle i, m\rangle$, we denote by $F\langle i, m\rangle \in \{0, \ldots, t\}$ the number of failures known to $\langle i, m\rangle$, i.e. the number of processes $j \neq i$ from which i does not receive a message at time m.*

We note that ***d***, as defined in Lemma 6, is precisely $F\langle i, m\rangle$.

Proof of Lemma 6. Sketch: It is straightforward to see that conditions (a) and (b) imply $K_i\exists\mathsf{correct}(v)$ (Condition (a): as $\langle i, m-1\rangle$ is seen at m by all correct processes; condition (b): as the number of distinct processes knowing $\exists 0$, *including i itself*, is greater than the maximum number of active processes that can yet fail). If neither condition holds, then i considers it possible that only incorrect processes know $\exists v$, and that they all immediately fail (i at time m before sending any messages, and the others — immediately after sending the last message seen by i), in which case no correct process would ever know $\exists v$. □

As with P_0 in the case of consensus, by analysing decisions in protocols dominating U-P_0, we show that no Uniform Consensus protocol can dominate U-OPT$_0$. Lemmas 14 and 15 give sufficient conditions for deciding 0 in any Uniform Consensus protocol dominating U-P_0. As mentioned above, the analysis is considerably subtler for Uniform Consensus, because the analogue of Lemma 2 is not true. Receiving a message with value 0 in a protocol dominating U-P_0 does not imply that the sender has decided 0.

Lemma 13 (No decision at time 0)**.** *Assume that $t > 0$. Let Q solve Uniform Consensus. No process decides at time 0 in any run of Q.*

Proof. As $t < n$, by Lemma 5 it is enough to show that $\neg K_i\exists v$ for every process i and $v \in \{0, 1\}$. As $0 < t$, and as $F\langle i, 0\rangle = 0$ for all processes i by definition, we have that by Lemma 6, the proof is complete. □

Lemma 14 (Decision at time 1)**.** *Let $Q \preceq$ U-P_0 solve Uniform Consensus and let $r = r[\alpha]$ be a run of Q. Let i be a process with initial value 0 in r s.t. i is active at time 1 in r. If either of the following hold in r, then $\langle i, 1\rangle$ decides 0 in r.*

1. *$t > 0$ and there exists a process $j \neq i$ with initial value 0 s.t. $\langle j, 0\rangle$ is seen by $\langle i, 1\rangle$.*
2. *$t > 1$ and $F\langle i, 1\rangle < t$.*

Proof. For both parts, we first note that by Lemma 6 and by definition of U-P_0, i decides 0 at (U-$P_0[\alpha], 1)$. As $Q \preceq$ U-P_0, we thus have that i must decide upon some value in r by time 1. By Lemma 13, i does not decide at $(r, 0)$. Thus, i must decide at $(r, 1)$.

We now show Part 1 by induction on $n - |Z_i^0|$, where Z_i^0 is defined to be the set of processes k with initial value 0, s.t. $\langle k, 0\rangle$ is seen by $\langle i, 1\rangle$. Note that by definition, $i, j \in Z_i^0$, and so $1 < |Z_i^0| \leq n$.

Base: $|Z_i^0| = n$. In this case, all initial values are 0, and so by **Validity** i decides 0 at $(r, 1)$.

Step: Let $1 < \ell < n$ and assume that Part 1 holds whenever $|Z_i^0| = \ell + 1$. Assume that $|Z_i^0| = \ell$. We reason by cases.

I. If there exists a process k s.t. $\langle k,0\rangle$ is hidden from $\langle i,1\rangle$, then there exists a run r' of Q, s.t. *i)* $r'_i(1)=r_i(1)$, *ii)* j is active at $(r',1)$, *iii)* k has initial value 0 in r', and *iv)* $Z^0_j = Z^0_i \cup \{k\}$ in r'. (Note that by definition, Z^0_i has the same value in both r and r'.) By the induction hypothesis (switching the roles of i and j), j decides 0 at $(r',1)$, and therefore by **Uniform Agreement**, i cannot decide 1 at $(r',1)$, and hence it does not decide 1 at $(r,1)$. Thus, i decides 0 at $(r,1)$.

II. Otherwise, $\langle k,0\rangle$ is seen by $\langle i,1\rangle$ for all processes k. As $|Z^0_i| < n$, there exists a process $k \notin Z^0_i$ (in particular, $k \notin \{i,j\}$). Hence, as $t>0$, there exists a run r' of Q, s.t. *i)* $r'_i(1)=r_i(1)$, *ii)* j is active at $(r',1)$, *iii)* $\langle k,0\rangle$ is hidden from $\langle j,1\rangle$ in r', and *iv)* $Z^0_j = Z^0_i$ in r'. (Once again, Z^0_i has the same value in both r and r'.) By Case I (switching the roles of i and j), j decides 0 at $(r',1)$, and therefore by **Uniform Agreement**, i cannot decide 1 at $(r',1)$, and hence it does not decide 1 at $(r,1)$. Thus, i decides 0 at $(r,1)$.

We move on to prove Part 2. If $\langle k,0\rangle$ is hidden from $\langle i,1\rangle$ for all processes $k \neq i$, then $\neg K_i \exists 1$ at $(r,1)$. Thus, by Lemma 5, i cannot decide 1 at $(r,1)$, and so must decide 0 at $(r,1)$. Otherwise, there exists a process $k \neq i$ s.t. $\langle k,0\rangle$ is seen by $\langle i,1\rangle$. As $n>t>1$, we have $n>2$ and so there exists a process $j \notin \{i,k\}$; if $F\langle i,1\rangle > 0$, then we pick j s.t. $\langle j,0\rangle$ is hidden from $\langle i,1\rangle$. Since $t > 1$ (for the case in which $F\langle i,1\rangle = 0$ and $\langle j,0\rangle$ is seen by $\langle i,1\rangle$) and since $t > F\langle i,1\rangle$ (for the case in which $\langle j,0\rangle$ is hidden from $\langle i,1\rangle$), there exists a run r' of Q,s.t. *i)* $r'_i(1)=r_i(1)$, *ii)* k never fails in r', *iii)* j fails at $(r',0)$ before sending any messages except perhaps to i, and *iv)* i fails at $(r',1)$, immediately after deciding but before sending any messages. Thus, there exists a run r'' of Q, s.t. *i)* $r''_k(m')=r'_k(m')$ for all m', *ii)* k never fails in r'', *iii)* i and j both have initial value 0 in r'', *iv)* j fails at $(r'',0)$ while successfully sending a message only to i (and therefore $j \in Z^0_i$ in r''), and *v)* i fails at $(r'',1)$, immediately after deciding but before sending out any messages. By Part 1, i decides 0 at $(r'',1)$, and therefore k can never decide 1 during r'', and therefore neither during r'. As k never fails during r', by **Decision** it must thus decide 0 at some point during r'. Therefore, by **Uniform Agreement**, i cannot decide 1 at $(r',1)$, and thus it does not decide 1 at $(r,1)$. Thus, i decides 0 at $(r,1)$. □

Lemma 15 (Decision at times later than 1)**.** *Let $Q \preceq$ U-P_0 solve Uniform Consensus, let $r=Q[\alpha]$ be a run of Q and let $m>0$. Let i be a process s.t. $K_i\exists 0$ holds at time m for the first time in r, s.t. $K_i\exists\mathsf{correct}(0)$ holds at time $m+1$ for the first time in r, and s.t. i is active at $(r,m+1)$. If either of the following hold in r, then i decides 0 at $(r,m+1)$.*

1. *All of the following hold.*
   - *$F\langle i,m+1\rangle < t$.*
   - *There exists a process z s.t. $K_z\exists 0$ holds at time $m-1$, s.t. $\langle z,m-1\rangle$ is seen by $\langle i,m\rangle$, but s.t. $\langle z,m\rangle$ is not seen by $\langle i,m+1\rangle$,*
   - *There exists a process $j \neq i$ s.t. $\langle j,m\rangle$ is seen by $\langle i,m+1\rangle$ and $\langle z,m-1\rangle$ is seen by $\langle j,m\rangle$.*
2. *$F\langle i,m+1\rangle < t-1$.*

Proof. We prove the lemma by induction on m, with the base and the step sharing the same proof (as will be seen below, the conceptual part of an induction base will be played, in a sense, by Lemma 14).

We prove both parts together, highlighting local differences in reasoning for the different parts as needed. For Part 2, we denote by z an arbitrary process s.t. $K_z\exists 0$ holds at time $m-1$ and s.t. $\langle z,m-1\rangle$ is seen by $\langle i,m\rangle$. (As $m>0$, such a process must exist for i to know $\exists 0$ at time m for the first time; nonetheless, unlike when proving

Part 1, it is not guaranteed when proving this part that $\langle z,m\rangle$ is not seen by $\langle i,m+1\rangle$.)

We first note that by Lemma 5 and by definition of $\text{U-}P_0$, i decides 0 at $(\text{U-}P_0[\alpha],m+1)$. As $Q \preceq \text{U-}P_0$, we thus have that i must decide upon some value in r by time $m+1$. By Lemma 5, the precondition for deciding 0 is not met by i at (r,m). Therefore, it is enough to show that i does not decide 1 before or at time $m+1$ in r in order to show that i decides 0 at $(r,m+1)$.

Let $Z_i^{z,m}$ be the set of processes k s.t. $\langle k,m\rangle$ is seen by $\langle i,m+1\rangle$ in r and s.t. $\langle z,m-1\rangle$ is seen by $\langle k,m\rangle$ in r. (By definition, $i \in Z_i^{z,m}$.) Let C_i be the set of all processes k s.t. $\langle k,m\rangle$ is either seen by, or hidden from $\langle i,m+1\rangle$ (i.e. the set of nodes that $\langle i,m+1\rangle$ does not know to be inactive at time m). Note that by definition, $Z_i^{z,m} \subseteq C_i$. We first consider the case in which $Z_i^{z,m} \supsetneq \{i\}$, and prove the m-induction step (for the given m) for this case by induction on $|C_i \setminus Z_i^{z,m}|$.

Base: $Z_i^{z,m} = C_i$. In this case, $\langle i,m+1\rangle$ does not know that z fails at time $m-1$. Thus, $z \in C_i$ and therefore $z \in Z_i^{z,m}$. It follows that $\langle z,m\rangle$ is seen by $\langle i,m+1\rangle$ and therefore the second condition of Part 1 does not hold. Thus, the condition of Part 2 holds: $F\langle i,m+1\rangle < t-1$. Furthermore, we thus have that z is active at time m. We now argue that z decides 0 at (r,m), which completes the proof of the base case, as by **Uniform Agreement** i can never decide 1 during r. We reason by cases; for both cases, note that since $\langle z,m\rangle$ is seen by $\langle i,m+1\rangle$, we have that $F\langle z,m\rangle \le F\langle i,m+1\rangle < t-1$.

- If $m=1$: As $K_z\exists 0$ at time $m-1=0$, z has initial value 0. As $F\langle z,m\rangle < t-1$, we have that $t>1$. By Part 2 of Lemma 14 (for $i=z$), we thus have that z decides 0 at $(r,1)=(r,m)$.
- Otherwise, $m>1$. In this case, as $\langle z,m-2\rangle$ is seen by $\langle i,m-1\rangle$, and as $K_i\exists 0$ holds at time m for the first time, we have that $K_z\exists 0$ holds at time $m-1$ for the first time. Similarly, as $\langle z,m-1\rangle$ is seen by $\langle i,m\rangle$, and as $K_i\exists\mathsf{correct}(0)$ does not hold at time m, we have that $K_z\exists\mathsf{correct}(0)$ does not hold at time $m-1$. By Part 2 of the m-induction hypothesis (for $i=z$), z decides 0 at (r,m).

Step: Let $\{i\} \subsetneq Z_i^{z,m} \subsetneq C_i$, and assume that the claim holds whenever $Z_i^{z,m}$ is of larger size. For Part 1, note that $j \in Z_i^{z,m}$, for j as defined in the conditions for that part; for Part 2, let $j \in Z_i^{z,m}$ be arbitrary. Analogously to the proof of the induction step in the proof of Part 1 of Lemma 14, we reason by cases. For the time being, assume that the conditions of Part 2 hold, i.e. that $F\langle i,m+1\rangle < t-1$.

I. If there exists a process $k \in C_i$ s.t. $\langle k,m\rangle$ is hidden from $\langle i,m+1\rangle$, then there exists a run r' of Q, s.t. *i)* $r'_i(m+1) = r_i(m+1)$, *ii)* j is active at $(r',m+1)$, *iii)* $\langle z,m-1\rangle$ is seen by $\langle k,m\rangle$ in r', and *iv)* $Z_j^{z,m} = Z_i^{z,m} \cup \{k\}$ and $C_j = C_i$ in r'. (Note that by definition, $Z_i^{z,m}$ and C_i have the same values in both r and r'.) We note that $F\langle j,m+1\rangle = F\langle i,m+1\rangle - 1$ in r', and that by definition $F\langle i,m+1\rangle$ is the same in both r and r'. By the inductive hypothesis for $Z_j^{z,m}$ (i.e., for j w.r.t. z at time m), j decides 0 at $(r',m+1)$, and therefore by **Uniform Agreement**, i cannot decide 1 in r', and therefore it cannot decide 1 before or at $m+1$ in r', and the proof is complete.
II. Otherwise, for each process $k \in C_i$, $\langle k,m\rangle$ is seen by $\langle i,m+1\rangle$. As $Z_i^{z,m} \subsetneq C_i$, there exists a process $k \ne i$ s.t. $\langle k,m\rangle$ is seen by $\langle i,m+1\rangle$ but s.t. $\langle z,m-1\rangle$ is hidden from $\langle k,m\rangle$ (thus $k \ne j$). Hence, and since $F\langle i,m+1\rangle < t$, there exists a run r' of Q, s.t. *i)* $r'_i(m+1) = r_i(m+1)$, *ii)* j is active at $(r',m+1)$, *iii)* $\langle k,m\rangle$ is hidden from $\langle j,m+1\rangle$ in r', and *iv)* $Z_j^{z,m} = Z_i^{z,m}$ and $C_j \supseteq C_i$ in r'. (Once again, $Z_i^{z,m}$ and C_i have the same

values in both r and r'.) We note that $F\langle j, m+1\rangle = F\langle i, m+1\rangle + 1$ in r', and that once more, by definition, $F\langle i, m+1\rangle$ is the same in both r and r'. By Case I (for $i = j$), and since Case I uses the inductive hypothesis for $Z_j^{z,m}$ with one less failure, we conclude that j decides 0 at $(r', m{+}1)$. Therefor, by **Uniform Agreement**, i cannot decide 1 at $(r', m{+}1)$, and thus it cannot decide 1 before or at $m+1$ in r, and the proof is complete.

To show that the $Z_i^{z,m}$-induction step also holds under the conditions of Part 1, we observe that since $\langle z, m\rangle$ is not seen by $\langle i, m{+}1\rangle$ in this case, the amount of invocations of Case II (which uses Case I with one additional known failure) before reaching the $Z_i^{z,m}$-induction base is strictly smaller than that of Case I (which uses the $Z_i^{z,m}$-induction hypothesis with one less known failure), and therefore the $Z_i^{z,m}$-induction base is reached with less known failures, i.e. with less than $t-1$ known failures, i.e. the conditions of Part 2 hold at that point.

Finally, we consider the case in which $Z_i^{z,m} = \{i\}$. As any j as in Part 1 satisfies $j \in Z_i^{z,m}$, we have that the conditions of Part 2 hold, i.e. $F\langle i, m{+}1\rangle < t{-}1$. Furthermore, in we have that $\langle z, m\rangle$ is not seen by $\langle i, m{+}1\rangle$ (otherwise, $z \in Z_i^{z,m}$). As $F\langle i, m{+}1\rangle < t{-}1 < n{-}2$, there exist two distinct processes $j, k \neq i$ that are not known to $\langle i, m{+}1\rangle$ to fail (and thus i, j, k, z are distinct). Thus, $\langle j, m\rangle$ and $\langle k, m\rangle$ are seen by $\langle i, m{+}1\rangle$.

By definition of j, k, there exists a run r' of Q, s.t. *i)* $r'_i(m{+}1) = r_i(m{+}1)$, *ii)* k never fails in r', *iii)* j fails at (r', m) before sending any messages, *iv)* i fails at $(r', m+1)$, immediately after deciding but before sending any messages, and *v)* the faulty processes in r' are those known by $\langle i, m\rangle$ to fail in r, and in addition i and j. We note that by definition, $F\langle i, m{+}1\rangle$ is the same in r and r', even though the number of failures in r' is $F\langle i, m{+}1\rangle + 2$. We notice that there exists a run r'' of Q, s.t. *i)* $r''_k(m') = r''_k(m')$ for all m', *ii)* k never fails in r'', *iii)* $\langle z, m-1\rangle$ is seen by both $\langle i, m\rangle$ and $\langle j, m\rangle$ in r'', *iv)* j fails at (r'', m) while successfully sending a message only to i (and therefore both $j \in Z_i^{z,m}$ and $F\langle i, m+1\rangle < t-1$ in r''), and *v)* i fails at $(r'', m+1)$, immediately after deciding but before sending out any messages. By the proof for the case in which $Z_i^{z,m} \supsetneq \{i\}$ ($j \in Z_i^{z,m}$), i decides 0 at $(r'', m{+}1)$, and therefore k can never decide 0 during r'', and therefore neither during r'. As k never fails during r', by **Decision** it must thus decide 0 at some point during r'. Therefore, by **Uniform Agreement**, i cannot decide 1 before or at $m{+}1$ in r', and thus it does not decide 1 before or at $m+1$ in r, and the proof is complete. □

Now that we have established when processes must decide 0 in any protocol dominating P_0, we can deduce when processes cannot decide in any such protocol.

Lemma 16 (No Earlier Decisions when $K_i\exists 0$)**.** *Let $Q \preceq$ U-P_0 solve Uniform Consensus, let r be a run of Q, let m be a time, and let i be a process. If at time m in r we have $K_i\exists 0$, but $\neg K_i\exists\mathsf{correct}(0)$, then i does not decide at (r, m).*

Proof. If $m{=}0$, then by Lemma 6 and since $\neg K_i\exists\mathsf{correct}(0)$ at $m{=}0$ (even though $K_i\exists 0$), we have $t{>}0$. Thus, by Lemma 13, i does not decide at (r, m). Assume henceforth, therefore, that $m{>}0$.

As $\neg K_i\exists\mathsf{correct}(0)$, we have that by Lemma 6, $\neg K_i\exists 0$ at time $m{-}1$. Thus, there exists a process z s.t. $K_z\exists 0$ at $m{-}1$, and $\langle z, m{-}1\rangle$ is seen by $\langle i, m\rangle$. In turn, by Lemma 6, we have that $F\langle i, m\rangle < t-1$. There exists a run r' of Q, s.t. *i)* $r'_i(m){=}r_i(m)$, and *ii)* the faulty processes in r' are those known by $\langle i, m\rangle$ to fail in r. We henceforth reason about r'. By definition of r', $F\langle i, m{+}1\rangle = F\langle i, m\rangle < t{-}1$ (by definition, the value of $F\langle i, m\rangle$ is the same in both r and r'). Thus, by Part 2 of Lemma 15, i decides 0 at $(r', m{+}1)$, and hence i does not decide at (r', m), and therefore neither does it decide at (r, m). □

Lemma 17 (No Earlier Decisions when $\neg K_i\exists 0$)**.** *Assume that $t>0$. Let $Q \preceq$ U-P_0 solve Uniform Consensus, let r be a run of Q, let m be a time, and let i be a process. If there exists a hidden path w.r.t. $\langle i,m\rangle$ in r, and if at time m in r we have $\neg K_i\exists 0$, then i does not decide at (r,m).*

Proof. As $\neg K_i\exists 0$ at time m, then by **Validity**, i does not decide 0 at (r,m). Thus, it is enough to show that i does not decide 1 at (r,m) in order to complete the proof. If $m=0$, then by Lemma 13, i does not decide 1 at (r,m) either. Assume henceforth, therefore, that $m>0$.

As there exists a hidden path w.r.t. $\langle i,m\rangle$, there exist processes $z,j \neq i$ s.t. $\langle z,m-1\rangle$ is hidden from $\langle i,m\rangle$ and s.t. $\langle j,m-1\rangle$ is seen by $\langle i,m\rangle$.

We first consider the case in which $F\langle i,m\rangle < t$. In this case, there exists a run $r' = Q[\beta]$ of Q, s.t. all of the following hold in r':

- $r'_i(m) = r_i(m)$.
- z is the unique process that knows $\exists 0$ at $m-1$, and knows so then for the first time, either having initial value 0 (if $m=1$) or (as explained in the Non-Uniform Consensus section) seeing only a single node that knows $\exists 0$ at $m-2$ (if $m>1$).
- z fails at $(r',m-1)$, successfully sending messages to all nodes except for i.
- The faulty processes in r' are those known by $\langle i,m\rangle$ to fail in r, and in addition i, which fails at time m without sending out any messages. In particular, j never fails.

We henceforth reason about r'. First, we note that $\langle j,m+1\rangle$ does not know that z fails at $m-1$ (as opposed to at m). As $\langle j,m\rangle$ sees $\langle z,m-1\rangle$, as $K_z\exists 0$ at $m-1$, and as j never fails, by Lemma 6 we have that $K_j\exists\mathsf{correct}(0)$ at $(r',m+1)$. Thus, j decides at (U-$P_0[\beta],m+1$), and so j must decide before or at $m+1$ in r'. As $r_i(m)=r'_i(m)$, then by **Uniform Agreement** it is enough to show that j does not decide 1 up to time $m+1$ in r' in order to complete the proof.

There exists a run r'' of Q, s.t. *i)* $r''_j(m+1) = r'_j(m+1)$, and *ii)* the only difference between r'' and r' up to time m is that in r'', z fails only at time m, after deciding but without sending a message to j. By **Uniform Agreement**, it is enough to show that z decides 0 at (r'',m) in order to complete the proof.

We henceforth reason about r''. As z does not know at m that neither z nor i fail, we have $F\langle z,m–1\rangle \leq F\langle z,m\rangle < t–1$. Thus, $t>1$. If $m=1$, we therefore have by Part 2 of Lemma 14 that z decides 0 at (r'',m). Otherwise, $m>1$. As $K_z\exists 0$ at $m–1$ for the first time, as $\langle z,m–1\rangle$ sees only one node at $m–1$ that knows $\exists 0$, and as $F\langle z,m\rangle < t–1$, by Lemma 6 we have $\neg K_z\exists\mathsf{correct}(0)$ at $m-1$. Thus, by Part 2 of Lemma 15 (for $i = z$), z decides 0 at (r'',m). Either way, the proof is complete.

We now consider the case in which $F\langle i,m\rangle = t$. There exists a run $r'=Q[\beta]$ of Q, s.t. all of the following hold:

- $r'_i(m) = r_i(m)$.
- All processes k s.t. $\langle k,m-1\rangle$ is hidden from $\langle i,m\rangle$ (including $k = z$) know $\exists 0$ at $(r',m-1)$, either having initial value 0 (if $m=1$) or all seeing only a single node that knows $\exists 0$ at $m-2$ (and which fails at time $m-2$ without being seen by $\langle i,m\rangle$) — denote this node by z'.
- All such processes fail at time $m-1$, successfully sending messages to all nodes except for i.
- The faulty processes failing in r' are those known by $\langle i,m\rangle$ to fail in r. In particular, there are t such processes.

We henceforth reason about r'. We note that as i never fails, $F\langle i,m{-}1\rangle \leq F\langle j,m\rangle$ (equality can actually be shown to hold here, but we do not need it). As the number of nodes at $m-1$ knowing $\exists 0$ that are seen by $\langle j,m\rangle$ equals $F\langle i,m\rangle - F\langle i,m-1\rangle \geq t - F\langle j,m\rangle$ (by the above remark, equality holds here as well), we have by Lemma 6 that $K_j\exists\mathsf{correct}(0)$ at m, and therefore j decides at $(\textsc{u-}P_0[\beta],m)$; thus, it must decide before or at m in r'. As $r_i(m)=r'_i(m)$, by **Uniform Agreement** it is enough to show that j does not decide 1 up to time m in r' in order to complete the proof.

We proceed with an argument similar in a sense to those of Part 1 of Lemma 14 and the inner induction in the proof of Lemma 15.

As $\langle z,m-1\rangle$ is seen by $\langle j,m\rangle$, there exists a run r'' of Q, s.t. *i)* $r''_j(m) = r'_j(m)$, and *ii)* the only difference between r'' and r' up to time m is that in r', z never fails, but rather i fails at $m{-}1$ after sending a message to j but without sending a message to z. We note that there are t processes failing throughout r''. We henceforth reason about r''. If $m{=}1$, then z has initial value 0 and if $m{>}1$, then $\langle z,m{-}1\rangle$ sees $\langle z',m{-}2\rangle$; either way, by Lemma 6, $K_z\exists\mathsf{correct}(0)$ at (r'',m) and therefore z must decide before or at time m. Thus, it is enough to show that z does not decide 1 up to time m in r'' in order to complete the proof.

As $\langle i,m-1\rangle$ is not seen by $\langle z,m\rangle$, there exists a run r''' of Q, s.t. *i)* $r'''_z(m){=}r''_z(m)$, and *ii)* the only difference between r''' and r'' up to time m is that in r''', $\langle i,m-1\rangle$ sees $\langle z',m{-}2\rangle$ (or, if $m = 1$, then the difference is that i has initial value 0); we note that $\langle i,m-1\rangle$ is still seen by $\langle j,m\rangle$. We note that there are t processes failing throughout r'''. Observe that the number of nodes at $m{-}1$ knowing $\exists 0$ that are seen by $\langle j,m\rangle$ in r''' is greater than in r'/r'' (between which j at m cannot distinguish), however $F\langle j,m\rangle$ remains the same between r'/r'' and r'''; thus, $K_j\exists\mathsf{correct}(0)$ at m in r''' as well, and therefore j must decide before or at time m in r'''. Thus, it is enough to show that j does not decide 1 up to time m in r''' in order to complete the proof. We henceforth reason about r'''.

As $\langle i,m-1\rangle$ is seen by $\langle j,m\rangle$, there exists a run r'''' of Q, s.t. *i)* $r''''_j(m) = r'''_j(m)$, and *ii)* the only difference between r'''' and r''' up to time m is that in r'''', i does not fail (and is thus seen by $\langle z,m\rangle$). We note that there are $t-1$ processes failing throughout r'''', and thus in particular $F\langle z,m\rangle < t$. If $m = 1$, then by Part 1 of Lemma 14 (for $i = z$ and $j = i$), z decides 0 in (r'''',m). Otherwise, i.e. if $m{>}1$, by Part 1 of Lemma 15 (for $i = z$, $z = z'$, and $j = i$), z decides 0 in (r'''',m). Either way, the proof is complete. □

From Lemmas 16 and 17, we deduce sufficient conditions for Unbeatability of Uniform Consensus protocols dominating $\textsc{u-}P_0$; these conditions also become necessary if it can be shown that there exists some Uniform Consensus protocol dominating $\textsc{u-}P_0$ that meets them, as we indeed show momentarily for $\textsc{u-Opt}_0$.

Corollary 1. *Assume that $0 < t < n$. A protocol $Q \preceq \textsc{u-}P_0$ that solves Uniform Consensus and in which a node $\langle i,m\rangle$ decides whenever any of the following hold at m, is a unbeatable Uniform Consensus protocol.*

- $K_i\exists\mathsf{correct}(0)$.
- *No hidden path w.r.t. $\langle i,m\rangle$ exists, and $\neg K_i\exists 0$.*

By Corollary 1, we have that if $\textsc{u-Opt}_0$ solves Uniform Consensus, then it does so in a unbeatable fashion.

Lemma 18. $\textsc{u-Opt}_0 \preceq \textsc{u-}P_0$

Proof. As explained above, at time $\boldsymbol{t}+1$ no hidden paths exist, and furthermore, $K_i\exists 0$ iff $K_i\exists\mathsf{correct}(0)$. □

Theorem 8. $\textsc{u-Opt}_0$ *solves Uniform Consensus in* γ^{cr}. *Furthermore,*

- *If* $f \geq t-1$, *then all decisions are made by time* $f+1$ *at the latest.*
- *Otherwise, all decisions are made by time* $f+2$ *at the latest.*

Proof. This is a special case of Theorem 6, for which a complete proof is given below. □

Theorem 5 follows from Corollary 1 and Theorem 8; in the boundary case of $t=0$ (which is not covered by Corollary 1), we note that $\textsc{u-Opt}_0$ and $\textsc{Opt}_0$ coincide, as do the problems of uniform consensus and consensus; hence $\textsc{u-Opt}_0$ is unbeatable, and Theorem 5 holds, in that case as well.

D Proofs of Section 5.1 — Uniform Set Consensus

Proof of Theorem 6. **Decision**: By definition of $\textsc{u-Prot}_{\text{min-}k}$, every process that is active at time $\left\lfloor\frac{t}{k}\right\rfloor+1$, and in particular every non-faulty process, decides by this time at the latest.

Before moving on to show **Validity** and **Uniform *k*-Agreement**, we first complete the analysis of stopping times. In some run of $\textsc{u-Prot}_{\text{min-}k}$, let i be a process and let m be a time s.t. i is active at m but has not decided until m, inclusive. Let $\tilde{m} \leq m$ be the latest time not later than m s.t. $\langle i,\tilde{m}\rangle$ has hidden capacity $\geq k$. By definition of $\textsc{u-Prot}_{\text{min-}k}$, as i is undecided at m, we have $\tilde{m} \geq m-1$.

As $\langle i,\tilde{m}\rangle$ has hidden capacity $\geq k$ at $\tilde{m}$, let i_b^ℓ, for all $0 \leq \ell \leq \tilde{m}$ and $b=1,\ldots,k$, be as in Definition 2. By definition, $\langle i_b^\ell,\ell\rangle$, for every $0 \leq \ell < \tilde{m}$ and $b=1,\ldots,k$, is hidden from $\langle i,\tilde{m}\rangle$. Thus, $k \cdot \tilde{m} \leq F\langle i,\tilde{m}\rangle \leq f$. therefore, $\tilde{m} \leq \frac{f}{k}$ and so $\tilde{m} \leq \left\lfloor\frac{f}{k}\right\rfloor$. Hence, as $m-1 \leq \tilde{m}$, we have $m \leq \tilde{m}+1 \leq \left\lfloor\frac{f}{k}\right\rfloor+1$. We thus have that every process that is active at time $\left\lfloor\frac{f}{k}\right\rfloor+2$, decides by this time at the latest.

Assume now that $m = \left\lfloor\frac{f}{k}\right\rfloor+1$ and that f is a multiple of k. (i is still a process that is active but undecided at m.) As f is a multiple of k, then $m=\frac{f}{k}+1$, and so $f=k\cdot(m-1)$. As $f=k\cdot(m-1) \leq k\cdot\tilde{m} \leq F\langle i,\tilde{m}\rangle \leq F\langle i,m\rangle \leq f$, we we have that both $\tilde{m}=m-1$ and $F\langle i,m\rangle = f$. As $\tilde{m}=m-1$, we have that i has hidden capacity $< k$ at $m > \tilde{m}$. As i is undecided at m, we thus have, by definition of $\textsc{u-Prot}_{\text{min-}k}$, that $\neg K_i\exists\mathsf{correct}(v)$ for $v \triangleq \mathit{Min}\langle i,m\rangle$. As by definition $K_i\exists v$ at m, we have by Lemma 6 that $K_i\exists v$ at m for the first time. Therefore, as $m > \tilde{m} \geq 0$, there exists a process j such that $K_j\exists v$ at $m-1$ and s.t. $\langle j,m-1\rangle$ is seen by $\langle i,m\rangle$. Thus, by Lemma 6 and since $\neg K_o\exists\mathsf{correct}(v)$, we have $F\langle i,m\rangle < t-1$, and so $f=F\langle i,m\rangle < t-1$.

We thus have that if $f=t-1$ and if this value is a multiple of k, then every process that is active at time $\left\lfloor\frac{f}{k}\right\rfloor+1$ decides by this time at the latest.

We move on to show **Validity** and **Uniform *k*-Agreement**. Henceforth, let i be a (possibly faulty) process that decides in some run of $\textsc{u-Prot}_{\text{min-}k}$, let m_i be the decision time of i, and let v be the value upon which i decides. Thus, there exists $m'_i \in \{m_i, m_i-1\}$ s.t. $\langle i,m'_i\rangle$ is low or has hidden capacity $< k$, and s.t. $v=\mathit{Min}\langle i,m'_i\rangle$. (To show this when $m_i = \left\lfloor\frac{t}{k}\right\rfloor+1$, we note that in this case $m_i > \left\lfloor\frac{f}{k}\right\rfloor$, and so, as shown in the stopping-time analysis above, this implies that $\langle i,m_i\rangle$ has hidden capacity $< k$.)

Validity: As $v=\mathit{Min}\langle i,m'_i\rangle$, we have $K_i\exists v$ at m'_i, and thus $\exists v$.

Uniform *k*-Agreement: It is enough to show that at most $k-1$ distinct values smaller than v are decided upon in the current run. If $\langle i, m'_i \rangle$ is low, then $v = \mathit{Min}\langle i, m'_i \rangle < k-1$, and thus there do not exist more than $k-1$ distinct legal values smaller than v, let alone ones decided upon. For the rest of this proof we assume, therefore, that $\langle i, m'_i \rangle$ is high, and so has hidden capacity $< k$.

Let $w < v$ be a value decided upon by some process. Let j be this process, and let m_j be the time at which j decides on w. Thus, $w = \mathit{Min}\langle i, m'_j \rangle$ for some $m'_j \in \{m_j, m_j - 1\}$ s.t. if $m'_j = m_j$, then either $K_j \exists \mathsf{correct}(w)$ at m_j, or $m_j = \left\lfloor \frac{t}{k} \right\rfloor + 1$.

We first show that $m'_j \geq m'_i$. If $m'_j = m_j$ and $m_j = \left\lfloor \frac{t}{k} \right\rfloor + 1$, then we immediately have $m'_j = \left\lfloor \frac{t}{k} \right\rfloor + 1 \geq m_i \geq m'_i$, as required. Otherwise, the analysis is somewhat more subtle. We first show that in this case, if i is active at $m'_j + 1$, then $K_i \exists w$ at $m'_j + 1$. We reason by cases, according to the value of m'_j.

- If $m'_j = m_j$, then $K_j \exists \mathsf{correct}(w)$ at m'_j, and thus there exists a process k that never fails, s.t. $K_k \exists w$ at m'_j. As k never fails, $\langle k, m'_j \rangle$ is seen by $\langle i, m'_j + 1 \rangle$, and thus $K_i \exists w$ at $m'_j + 1$, as required.
- Otherwise, $m'_j = m_j - 1$. As j is active at m_j, it does does not fail at $m'_j < m_j$, and therefore $\langle j, m'_j \rangle$ is seen by $\langle i, m'_j + 1 \rangle$. Thus, as $K_j \exists w$ at m'_j, we obtain that $K_i \exists w$ at $m'_j + 1$ in this case as well.

As $w < v$ and as $v = \mathit{Min}\langle i, m'_i \rangle$, we have $\neg K_i \exists w$ at m'_i. Thus, we obtain that $m'_i < m'_j + 1$, and therefore $m'_j \geq m'_i$ in this case as well, as required. We have thus shown that we always have $m'_j \geq m'_i$.

As $\langle i, m'_i \rangle$ does not have hidden capacity k, there exists $0 \leq \ell \leq m'_i$ s.t. no more than $k-1$ processes at time ℓ are hidden from $\langle i, m'_i \rangle$. As $m'_i \geq \ell$, we have $m'_j \geq m'_i \geq \ell$. Let H be the set of all processes seen at ℓ by $\langle j, m'_j \rangle$. (Note that if $m'_j = \ell$, then $H = \{j\}$.) Since $m'_j \geq \ell$, we have $\mathit{Vals}\langle j, m'_j \rangle = \bigcup_{h \in H} \mathit{Vals}\langle h, \ell \rangle$. Thus, $w = \mathit{Min}\langle j, m'_j \rangle = \min_{h \in H} \{\mathit{Min}\langle h, \ell \rangle\}$. Therefore, $w = \mathit{Min}\langle h, \ell \rangle$ for some $h \in H$. As $\neg K_i \exists w$ at m'_i, we thus have that $\langle h, \ell \rangle$ is not seen by $\langle i, m'_i \rangle$. As $\langle h, \ell \rangle$ is seen by $\langle j, m'_j \rangle$, h does not fail before ℓ, and thus $\langle h, \ell \rangle$ is hidden from $\langle i, m'_i \rangle$. To conclude, we have shown that

$$w \in \big\{ \mathit{Min}\langle h, \ell \rangle \mid \langle h, \ell \rangle \text{ is hidden from } \langle i, m'_i \rangle \big\}.$$

As there are at most $k-1$ processes hidden at ℓ from $\langle i, m'_i \rangle$, we conclude that no more than $k-1$ distinct values lower than v are decided upon, and the proof is complete. □

E Different Types of Unbeatability

We first formally define last-decider unbeatability.

Definition 4 (Last-Decider Domination and Unbeatability)**.**

- *A decision protocol Q **last-decider dominates** a protocol P in γ, denoted by $Q \overset{\textit{l.d.}}{\preceq}_\gamma P$ if, for all adversaries α, if i the last decision in $P[\alpha]$ is at time m_i, then all decisions in $Q[\alpha]$ are taken before or at m_i. Moreover, we say that Q **strictly last-decider dominates** P if $Q \overset{\textit{l.d.}}{\preceq}_\gamma P$ and $P \overset{\textit{l.d.}}{\not\preceq}_\gamma Q$. I.e., if for some $\alpha \in \gamma$ the last decision in $Q[\alpha]$ is* strictly before *the last decision in $P[\alpha]$.*

- *A protocol P is a* ***last-decider unbeatable*** *solution to a decision task S in a context γ if P solves S in γ and no protocol Q solving S in γ strictly last-decider dominates P.*

Remark 4.

- *If $Q \preceq_\gamma P$, then $Q \overset{\textit{\textbf{l.d.}}}{\preceq}_\gamma P$. (But not the other way around.)*
- *None of the above forms of strict domination implies the other.*
- *None of the above forms of unbeatability implies the other.*

Last-decider domination does not imply domination in the sense of the rest of this paper (on which our proofs is based). Nonetheless, the specific property of protocols dominating $\textsc{Opt}_0$, $\textsc{Opt}_{\mathsf{Maj}}$, $\textsc{Opt}_{\text{min-}k}$ and $\textsc{u-Opt}_0$, which we use to prove that these protocols are unbeatable, holds also for protocols that only last-decider dominate these protocols.

Lemma 19.

1. *Let $Q \overset{\textit{l.d.}}{\preceq} P_0$ satisfy* **Decision**. *If $K_i\exists 0$ at m in a run $r = Q[\alpha]$ of Q, then i decides in r no later than at m.*
2. *Let $Q \overset{\textit{l.d.}}{\preceq} \textsc{Opt}_{\mathsf{Maj}}$ satisfy Decision. If $K_i(\mathsf{Maj} = v)$ for $v \in \{0, 1\}$ at m in a run $r = Q[\alpha]$ of Q, then i decides in r no later than at m.*
3. *Let $Q \overset{\textit{l.d.}}{\preceq} \textsc{Opt}_{\text{min-}k}$ satisfy* **Decision**. *If i is low at m in a run $r = Q[\alpha]$ of Q, then i decides in r no later than at m.*
4. *Let $Q \overset{\textit{l.d.}}{\preceq} \textsc{u-}P_0$ satisfy* **Decision**. *If $K_i\exists\mathsf{correct}(0)$ at m in a run $r = Q[\alpha]$ of Q, then i decides in r no later than at m.*

The main idea in the proof of each of the parts of Lemma 19 is to show that i considers it possible that all other active processes also know the fact stated in that part, and so they must all decide by the current time in the corresponding run of the dominated protocol. Hence, the last decision decision in that run is made in the current time; thus, by last-decider domination, i must decide. The proofs for the first three parts are somewhat easier, as in each of these parts, any process at m who sees (at least) the nodes seen by $\langle i, m\rangle$ (or has the same initial value, if $m = 0$) also knows the relevant fact stated in that part. We demonstrate this by proving Part 1; the analogous proofs of Parts 2 and 3 are left to the reader.

Proof of Part 1 of Lemma 19. If $m = 0$, then there exists a run $r' = Q[\beta]$ of Q, s.t. *i)* $r'_i(0) = r_i(0)$, *ii)* in r' all initial values are 0, and *iii)* i never fails in r'. Hence, in $P_0[\beta]$ all decisions are taken at time $m = 0$, and therefore so is the last decision. Therefore, the last decision in r' must be taken at time 0. As i never fails in r', by **Decision** it must decide at some point during this run, and therefore must decide at 0 in r'. As $r_i(0) = r'_i(0)$, i decides at 0 in r as well, as required.

If $m > 0$, then there exists a process j s.t. $K_j\exists 0$ at $m - 1$ in r and $\langle j, m-1\rangle$ is seen by $\langle i, m\rangle$. Thus, there exists a run $r' = Q[\beta]$ of Q, s.t. *i)* $r'_i(m) = r_i(m)$, and *ii)* i and j never fail in r'. Thus, all processes that are active at m in r' see $\langle j, m-1\rangle$ in r' and therefore know $\exists 0$ in r'. Hence, in $P_0[\beta]$ all decisions are taken by time m, and therefore so is the last decision. Therefore, the last decision in r' must be taken no later than at time m. As i never fails in r', by **Decision** it must decide at some point during this run, and therefore must decide by m in r'. As $r_i(m) = r'_i(m)$, i decides by m in r as well, as required. □

As the proof of Part 4 is slightly more involved, we show it as well.

Proof of Part 4 of Lemma 19. If $m = 0$, then by Lemma 6, $t = 0$. There exists a run $r' = Q[\beta]$ of Q, s.t. *i)* $r'_i(0) = r_i(0)$, and *ii)* in r' all initial values are 0. Therefore, as $t = 0$, we have by Lemma 6 that all processes know $\exists\mathsf{correct}(0)$ at $m = 0$ in r'. Hence, in U-$P_0[\beta]$ all decisions are taken at time $m = 0$, and therefore so is the last decision. Therefore, the last decision in r' must be taken at time 0 as well. Since $t = 0$, i never fails in r', and so by **Decision** it must decide at some point during this run, and therefore must decide at 0 in r'. As $r_i(0) = r'_i(0)$, i decides at 0 in r as well, as required.

If $m > 0$, then there exists a process j s.t. $K_j\exists 0$ at $m-1$ in r and $\langle j, m-1\rangle$ is seen by $\langle i, m\rangle$ in r. Furthermore, as $t < n$, there exists a set of processes I s.t. *i)* $i, j \notin I$, *ii)* $|I| = t - F\langle i, m\rangle - 1$, and *iii)* $\langle k, m-1\rangle$ is seen by $\langle i, m\rangle$ for every $k \in I$. Thus, there exists a run $r' = Q[\beta]$ of Q, s.t. *i)* $r'_i(m) = r_i(m)$, *ii)* i and j never fail in r', *iii)* all of I fail in r' at $m-1$, successfully sending messages only to i, and *iv)* every process at $m-1$ in r' that is not seen by $\langle i, m\rangle$, is not seen by any other process at m as well. We henceforth reason about r'. Every process $k \neq j$ that is active at m sees $\langle j, m-1\rangle$ and furthermore satisfies $F\langle k, m\rangle \geq F\langle i, m\rangle + |I| = t - 1$. Thus, by Lemma 6, $K_k\exists\mathsf{correct}(0)$ at m, and thus k decides at (U-$P_0[\beta], m$). Additionally, as $K_j\exists 0$ at $m-1$, by Lemma 6 $K_j\exists\mathsf{correct}(0)$ at m, and thus j decides at (U-$P_0[\beta], m$). Hence, in U-$P_0[\beta]$ all decisions are taken by time m, and therefore so is the last decision. Therefore, the last decision in r' must be taken no later than at time m. As i never fails in r', by **Decision** it must decide at some point during this run, and therefore must decide by m in r'. As $r_i(m) = r'_i(m)$, i decides by m in r as well, as required. □

As explained above, Theorem 7 follows from Lemma 19, and from the proofs of Theorems 2 to 5.

Finally, we sketch the structure of communication-efficient implementations for the protocols proposed in the paper:

Lemma 20. *For each of the protocols* OPT$_0$, OPT$_{\mathsf{Maj}}$, OPT$_{\text{min-}k}$, U-OPT$_0$ *and* U-PROT$_{\text{min-}k}$ *there is a protocol with identical decision times for all adversaries, in which every process sends at most $O(n \log n)$ bits overall to each other process.*

Proof. (Sketch) Moses and Tuttle in [21] show how to implement full-information protocols in the crash failure model with linear-size messages. In our case, a further improvement is possible, since decisions in all of the protocols depend only on the identity of hidden nodes and on the vector of initial values. In a straightforward implementation, we can have a process i report "$\mathtt{value}(j) = v$" once for every j whose initial value it discovers, and "$\mathtt{failed\_at}(j) = \ell$" once where ℓ is the earliest failure round it knows for j. In addition, it should send an "`I'm_alive`" message in every round in which it has nothing to report. Process i can send at most one `value` message and two `failed_at` messages for every j. Since `I'm_alive` is a constant-size message sent fewer than n times, and since encoding j's ID requires $\log n$ bits, a process i sends a total of $O(n \log n)$ bits overall. □