Convex Consensus with Asynchronous Fallback

Andrei Constantinescu
ETH Zürich, Switzerland

Diana Ghinea
ETH Zürich, Switzerland

Roger Wattenhofer
ETH Zürich, Switzerland

Floris Westermann
ETH Zürich, Switzerland

Abstract

Convex Consensus (CC) allows a set of parties to agree on a value v inside the convex hull of their inputs with respect to a predefined abstract convexity notion, even in the presence of byzantine parties. In this work, we focus on achieving CC in the best-of-both-worlds paradigm, i.e., simultaneously tolerating at most t_s corruptions if communication is synchronous, and at most $t_a \leq t_s$ corruptions if it is asynchronous. Our protocol is randomized, which is a requirement under asynchrony, and we prove that it achieves optimal resilience. In the process, we introduce communication primitives tailored to the network-agnostic model. These are a deterministic primitive allowing parties to obtain intersecting views (*Gather*), and a randomized primitive leading to identical views (*Agreement on a Core-Set*). Our primitives provide stronger guarantees than previous counterparts, making them of independent interest.

2012 ACM Subject Classification Theory of computation → Cryptographic protocols

Keywords and phrases convex consensus, network-agnostic protocols, agreement on a core-set

Digital Object Identifier 10.4230/LIPIcs.DISC.2024.15

Related Version *Full Version*: `https://ia.cr/2023/1364`

Acknowledgements We thank Julian Loss and the anonymous reviewers for their useful suggestions.

1 Introduction

Arranging a meeting place for a group of n people in a city is a common problem, as determining a location that is convenient and accessible for everyone can be challenging. While locations can be determined by their geographic coordinates, we need to prevent agreement on the coordinates of a restricted area, e.g., some private property. Hence, it may be more realistic to represent the city as a graph, with streets modeled as edges and intersections as vertices. Participants are initially in different locations (i.e., vertices), and they want to agree on a vertex for the meeting point using pair-wise communication channels. Finding such a meeting point, while also considering that some of the participants may choose not to follow the protocol, describes the Convex Consensus problem (CC).

The CC problem serves as a unifying framework for various agreement problems that deal with different input spaces. Such input spaces may be continuous, such as $\mathbb{R}^D$, or discrete, such as graphs and even lattices. Essentially, CC assumes a publicly available input space V (this could be the set of locations) equipped with a convexity notion $\mathcal{C}$ (roughly meant to formalize which meeting points are convenient with respect to the participants' inputs). For example, in the case of $\mathbb{R}^D$, the standard "straight-line" convexity notion can be considered. In contrast, convexity notions for graphs may be defined in various ways: for example, *geodesic convexity*, defined over shortest paths between vertices, or *monophonic convexity*,

defined over minimal/chordless paths. For a given convexity notion, CC is concerned with enabling parties to agree on a value in the convex hull of their inputs. This should be achieved even if up to t of the parties are corrupted (byzantine) and may exhibit malicious behavior.

A natural question to ask is *"For which values of t can CC be achieved?"*. Prior work provides an almost complete answer for *the synchronous model*, i.e., where the parties' clocks are synchronized and messages get delivered within a known amount of time Δ. In this model, the solvability of CC depends on the structure of the input space: concretely, on the space's Helly number ω (e.g., $D+1$ for $\mathbb{R}^D$ with straight-line convexity). CC can be solved in the synchronous model if $t < n/\omega$, and this condition is also necessary for convex geometries (a restricted class of convexity spaces) and for $\mathbb{R}^D$ with straight-line convexity [33, 37].

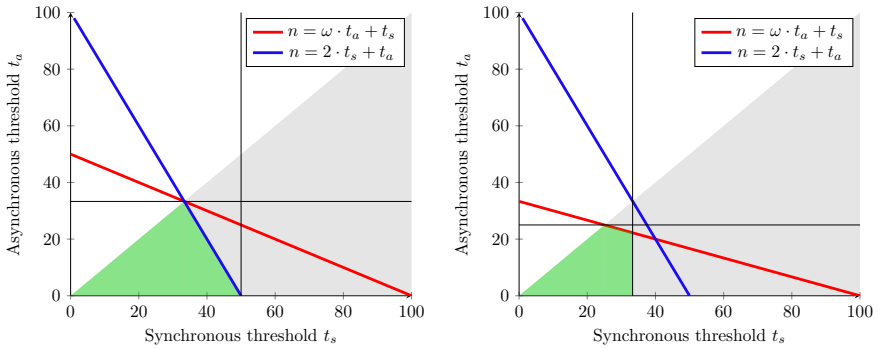
One may argue that the synchronous model's assumptions are too strong: in practice, the maximum delay Δ will often be violated during times of increased network load or outages. A well-established alternative is the *asynchronous model*. This only assumes that parties' messages get delivered eventually, leading to protocols that are highly robust to adverse network conditions. The solvability of CC in this model has only been partly characterized so far: for convex geometries and $\mathbb{R}^D$ with straight-line convexity, the condition $t < n/(\omega+1)$ is necessary [33, 37]. This is, however, only known to be sufficient in the asynchronous model for a relaxed version of CC which allows parties to agree up to some error (Approximate Agreement, AA), and only on particular input spaces. We highlight that the asynchronous model comes with an intrinsic limitation: even if the condition $t < n/(\omega+1)$ were to be proven sufficient for achieving asynchronous CC in all convexity spaces, there is a gap between this threshold and the $t < n/\omega$ threshold sufficing for synchronous networks. That is, an asynchronous CC protocol (which would have to be randomized, due to [18]) would achieve its guarantees regardless of the network conditions, but at the expense of tolerating a lower number of corruptions in comparison to synchronous alternatives.

This is where a third model steps in: *the network-agnostic model*, introduced by Blum, Katz, and Loss [9], which aims to combine the advantages of both established models. This model has gained significant popularity in recent years, and has covered problems such as Byzantine Agreement [9, 15], AA on real and multidimensional values [20, 21], State-Machine Replication [10] and Multi-Party Computation [6, 11, 15]. Concretely, given two thresholds $t_a \leq t_s$, a network-agnostic protocol should tolerate t_s corruptions if the network is synchronous and t_a if it is asynchronous, without knowing which of the two happens to be the case. We add that such a network-agnostic protocol with $t_a = 0$ still provides superior guarantees in comparison to a synchronous counterpart: it maintains its properties even if the synchrony assumptions fail provided that no party is corrupted.

Our work will primarily investigate the necessary and sufficient conditions for achieving CC for arbitrary convexity spaces in the *network-agnostic model*. We provide a complete characterization, showing that the condition $n > \max(\omega \cdot t_s, \omega \cdot t_a + t_s, 2 \cdot t_s + t_a)$ is necessary and sufficient in any convexity space with $\omega > 1$. This condition, illustrated in Figure 1, allows for a trade-off between the two expected optimal resilience bounds of the pure models (which we additionally prove to be tight leading up to our main result). We add that $\omega = 1$ refers to convexity spaces where some $v \in V$ is contained in all non-empty convex sets, allowing for trivial protocols (parties may simply output v).

1.1 Our contributions

Impossibility results. We first prove that the condition $n > \max(\omega \cdot t_s, \omega \cdot t_a + t_s, 2 \cdot t_s + t_a)$ is necessary. We additionally generalize the aforementioned lower bounds from convex geometries and $\mathbb{R}^D$ to all convexity spaces, so that $t < n/\omega$ is required for the synchronous case and $t < n/(\omega+1)$ for the asynchronous one. For these proofs, we define *adversarial families*, which will allow us to derive general scenario-based arguments [32].



(a) $\omega = 2$. **(b)** $\omega = 3$.

Figure 1 Our results on the feasibility of achieving CC resilient against t_s corruptions if the network is synchronous and $t_a \leq t_s$ corruptions if it is asynchronous. For a fixed value of $n = 100$, the two plots depict in green the set of pairs (t_s, t_a) for which a protocol exists as percentages of n: the condition $n > \max(\omega \cdot t_s, \omega \cdot t_a + t_s, 2 \cdot t_s + t_a)$. The two black lines correspond to the point-wise optimal resilience thresholds $n > \omega \cdot t_s$ and $n > (\omega + 1) \cdot t_a$ required in the synchronous and asynchronous models respectively. The condition $n > \max(\omega \cdot t_s, \omega \cdot t_a + t_s, 2 \cdot t_s + t_a)$ can be understood as $n > 2 \cdot t_s + t_a$ for $\omega = 2$ and $n > \max(\omega \cdot t_s, \omega \cdot t_a + t_s)$ for $\omega \geq 3$. The two cases are depicted above for $\omega = 2$ and $\omega = 3$.

Feasibility results. Afterwards, we show that the condition $n > \max(\omega \cdot t_s, \omega \cdot t_a + t_s, 2 \cdot t_s + t_a)$ is also sufficient: we give a protocol achieving CC when this condition holds. Together with our impossibility results, this completes the landscape of feasibility for the purely synchronous, purely asynchronous, and network-agnostic models. Our protocol assumes cryptographic setup, namely digital signatures. Note that this is necessary to tolerate $t_s < n/\omega$ corruptions in the synchronous model for $\omega = 2$, a fact which can be easily inferred from various impossibility results (e.g., [23]). When $\omega \geq 3$, however, signatures are no longer required (since $t_s < n/3$ holds in that case), and we will briefly explain how they can be removed. We also note that our protocol is randomized (which is needed [18]), but randomization is restricted to Byzantine Agreement subprotocols [15]. The use of signatures is similarly constrained to Reliable Broadcast and Byzantine Agreement subprotocols [15, 30].

Network-agnostic communication primitives. The core of our CC protocol is a novel network-agnostic implementation of Agreement on a Core-Set (ACS) [8], which may be of independent interest. In essence, ACS allows parties to distribute their inputs and obtain identical views. Our ACS protocol provides stronger guarantees than previous network-agnostic variants [4, 10, 11]. These stronger guarantees will be crucial for achieving CC: when the network is synchronous, we enable the parties to obtain a common view that includes all the honest parties' inputs. Obtaining these properties for ACS when $n > 2 \cdot t_s + t_a$ requires us to diverge from the outline of previous ACS constructions and to instead provide a novel implementation relying on *Gather* (GTHR) [2, 13], a second primitive that we adapt to the network-agnostic model. Roughly, GTHR enables parties to obtain intersecting views.

Prior works corrections. We need to note that our findings seemingly contradict an impossibility result of [33] for general convexity spaces, which depends on the Carathéodory number of the convexity space and not on its Helly number (in general, there is no relationship between the two). Upon closer inspection, we exhibit an error in the proof of [33], meaning that the correct bound is in terms of ω, and not the Carathéodory number.

As a secondary contribution, we identify a core issue in the asynchronous AA protocol for chordal graphs with monophonic convexity of [33]. In the full version of our paper, we describe this issue in detail, and we provide an alternative solution in the network-agnostic model. The alternative protocol is obtained by adapting the AA protocol on cycle-free (chordal) semilattices (i.e., a particular case of chordal graphs) of the same paper [33], while incorporating insights from the protocol of [5] achieving *wait-free* AA on chordal graphs.

1.2 Related work

CC and AA in the pure synchronous and pure asynchronous models. To the best of our knowledge, the problem of agreeing on a value in the honest inputs' convex hull was introduced for AA on $\mathbb{R}$ [16] (where $\omega = 2$). When considering this problem in the synchronous model, $t < n/3$ is tight when no cryptographic setup is allowed [16], and $t < n/2 = n/\omega$ is tight with cryptographic setup [20, 25]. In the asynchronous model, the optimal resilience threshold for AA is $t < n/3$ and was proven in [1]. The more general setting of $\mathbb{R}^D$ with straight-line convexity (where $\omega = D + 1$) was first considered in [27, 37] (see also the journal version [28]). Here, the bound $t < n/(D + 2)$ is necessary and sufficient for asynchronous AA. Along with the multidimensional variant of AA, Vaidya and Garg [37] have introduced the CC problem on $\mathbb{R}^D$ and showed that the condition $t < n/(D + 1)$ is tight for achieving CC in the synchronous model. Tseng and Vaidya [36] later presented an asynchronous variant of CC resilient against crash failures with incorrect inputs, also on $\mathbb{R}^D$, where parties agree on a polytope in the convex hull of the honest parties' inputs as opposed to a single value.

We also note the works of [19, 32], which focus of achieving agreement *on an honest input*. This is a particular case of CC on a space with universe V where a subset's convex hull is the subset itself. Their necessary and sufficient conditions match the more general variants, as the Helly number of this convexity space is $\omega = \left|V\right|$: $t < n/\left|V\right|$ in the synchronous model [32], and $t < n/(\left|V\right| + 1)$ in the asynchronous one [19].

Nowak and Rybicki [33] generalized the problems of CC and AA to abstract convexity spaces. We partially answer an open question raised in [33] on whether there exists an input convexity space for which the optimal resilience threshold for AA depends on the Carathéodory number and not on the Helly number. Our tight conditions for CC imply that, at least for randomized protocols, the asynchronous resilience threshold is actually independent of the Carathéodory number and only depends on the Helly number instead. In addition, we identify a core issue in the deterministic algorithm of [33] for asynchronous AA on chordal graphs. A related line of work considers graph AA in the wait-free model (where $t < n$ of the parties involved may crash) and its variants, primarily focusing on characterizing the families of graphs on which wait-free AA can be achieved [3, 5, 14, 24, 26].

CC and AA in the network-agnostic model. The problem of AA on real values has also been considered in the network-agnostic model in [20], where the condition $n > 2 \cdot t_s + t_a$ has been proven necessary and sufficient. For the $\mathbb{R}^D$ variant of AA, the condition $n > (D+1) \cdot t_s + t_a$ has been proven to be sufficient [21], but whether this condition is also necessary is still an open problem. Our work will build upon and extend some of the techniques of [20, 21]. Concretely, our Gather protocol is obtained by making an adjustment to the network-agnostic Overlap All-to-All Broadcast primitive of [20, 21], while incorporating insights from asynchronous Gather protocols [2, 13]. In addition, we rely on similar insights on deriving *safe areas* in the honest parties' convex hulls to [21], and also of prior works in the asynchronous model such as [1, 27, 33, 37]. Previously known techniques and insights will be noted precisely in the following sections. As a summary, our paper will diverge from [20, 21] since: (i) we do not

assume a particular input space: we consider abstract convexity spaces, and this requires us to provide generalized variants of lower bounds and safe area calculations; (ii) we focus on CC as opposed to AA, and achieving exact agreement requires us to design a stronger communication primitive: ACS. We also need to note that our feasibility result defines the first protocol in the network-agnostic model achieving an optimal resilience trade-off with a non-linear boundary (see Figure 1b). This gives a hint (but not yet an answer) on the question regarding necessary conditions in multidimensional AA left open in [21].

ACS in the network-agnostic model. As previously mentioned, the term ACS has been present in network-agnostic literature, as a building block for State-Machine Replication [4,10] and Multi-Party Computation [11]. We highlight an important distinction between prior constructions and ours. First, prior ACS variants would only provide as output *a set of values*, while our construction provides a common view defined as *a set of value-party pairs*. Second, when running in the synchronous model, the ACS protocols of [4, 10, 11] only need to ensure that *pre-agreement* is maintained: if all honest parties hold input v, the output set is $\{v\}$. For our CC protocol, the following properties will be crucial: (i) parties agree on the output set if the network is synchronous and t_s of the parties are corrupted (even without pre-agreement); (ii) roughly, each value's multiplicity is reflected in the output set (hence why the output set consists of value-party pairs); and, most importantly, (iii) if the network is synchronous, all honest values are guaranteed to be included (with multiplicities) in the parties' common view. Our ACS implementation will hence focus on this stronger definition, requiring us to diverge from the outline of previous ACS constructions for $n > 2 \cdot t_s + t_a$.

The concurrent work of [22], addressing Atomic Broadcast, also proposes a network-agnostic ACS implementation. While their protocol also relies on Gather and appears to achieve agreement regardless of the type of network, our ACS protocol is strictly stronger, as the protocol proposed by [22] provides the parties with a single value as output, and this value may be proposed by a corrupted party. This would prevent CC, but is sufficient for achieving Atomic Broadcast as in [22], since parties' values are *justified*.

While our ACS definition is stronger than previous network-agnostic variants, the protocol of [4] remains the state-of-art in terms of efficiency. The ACS protocol of [4] achieves an expected communication complexity of $O(n^2 \cdot \ell + n^3 \cdot \kappa)$ bits, where κ is the security parameter, and parties' inputs are represented as ℓ-bit strings (assuming threshold signatures). Even with threshold signatures, our protocol would incur an expected communication complexity of $O(n^3 \cdot \ell + n^4 \cdot \kappa)$, where ℓ denotes the universe elements' size in bits.

2 Preliminaries

In the following, given a non-negative integer k, write $[k]$ for the set $\{1, 2, \ldots, k\}$.

Model. Consider n parties denoted by $P_1, P_2, \ldots, P_n$ running a protocol in a fully-connected network, where links model authenticated channels. A synchronous network ensures that the parties' clocks are perfectly synchronized and that each message is delivered within a publicly known amount of time Δ. If any of these two guarantees fails, then the network is asynchronous. We assume that the parties are not aware a priori of the type of network the protocol is running in. In addition, we assume an adaptive adversary that may corrupt at most t_s parties if the network is synchronous, and at most t_a parties if the network is asynchronous. Corrupted parties permanently become byzantine, meaning that they can deviate arbitrarily, even maliciously, from the protocol. Moreover, the adversary may control

the message delivery schedule, subject to the conditions of the network type. We will make use of a public key infrastructure (PKI), and a secure signature scheme. For simplicity, we assume that the signatures are perfectly unforgeable.

Abstract convexity spaces. Given a nonempty set V, also called the *universe*, an *abstract convexity space* on V is a family $\mathcal{C}$ of subsets of V such that $\varnothing, V \in \mathcal{C}$ and $\mathcal{C}$ is closed under arbitrary intersections: whenever $A, B \in \mathcal{C}$, it also holds that $A \cap B \in \mathcal{C}$ (and the infinite analogue). Sets in $\mathcal{C}$ are regarded as *convex sets*. For instance, when $V = \mathbb{R}^D$, one possible $\mathcal{C}$ consists of all sets satisfying the condition that the straight-line segment joining any two points in the set is also included in the set. Note that this yields the standard convexity notion on $\mathbb{R}^D$. However, this is not the only way to define a convexity space on $\mathbb{R}^D$ that is consistent with the definition's requirements; e.g., take $\mathcal{C}$ to be the family of "box" subsets of $\mathbb{R}^D$; i.e., subsets of the form $I_1 \times \ldots \times I_D$, where $(I_i)_{i\in[D]}$ are closed intervals of the real line.

A central notion is that of convex hulls. In particular, the *convex hull* of any (not necessarily convex) set $S \subseteq V$ is the intersection $\langle S \rangle$ of all convex sets $C \in \mathcal{C}$ containing S, which is indeed convex by closure under intersections. In $\mathbb{R}^D$ under straight-line convexity, hulls correspond to the usual notion of Euclidean convex hulls, while under "box"-convexity they correspond to so-called "bounding boxes"; i.e., take the box spanning the region between the infimum and the supremum along each axis. Note that the convex hull operator is idempotent, i.e., $\langle\langle S \rangle\rangle = \langle S \rangle$. Moreover, note that a set is convex if and only if $S = \langle S \rangle$.

The Helly Number ω of a Convexity Space. The following seminal result in convexity theory concerns $\mathbb{R}^D$ with straight-line convexity.

▶ **Theorem 1** (Helly's Theorem). *Consider a finite collection of convex sets in $\mathbb{R}^D$ with straight-line convexity. If every $D+1$ of them intersect, then all of them intersect.*

Helly's Theorem implies that, for instance, any finite collection of disks in $\mathbb{R}^2$ with triple-wise non-empty intersections has a non-empty intersection. Notice that the same would not hold if $D+1$ was replaced by D; e.g., one can draw three disks in $\mathbb{R}^2$ that pair-wise intersect but have no point common to all three. One might now wonder: "What about box convexity?" In that case, $D+1$ can be replaced by 2. For instance, this means that any finite collection of rectangles in $\mathbb{R}^2$ where any two intersect has a non-empty intersection, in contrast to disks. This number, which is $D+1$ for straight-line convexity and 2 for box convexity, is known as the Helly number ω of the convexity space.

More generally, the *Helly number* ω of a convexity space $\mathcal{C}$ is the smallest number h such that any finite collection of convex sets out of which any h intersect has a non-empty intersection. It is useful to think in terms of the contrapositive: any finite collection of convex sets that do not intersect has a subcollection consisting of (at most) h sets that do not intersect. As a result, ω is equivalently the size of the largest collection of convex sets with an empty intersection such that any of its proper subcollections have a non-empty intersection. Notationally, say that an *m-Helly family* for $\mathcal{C}$ is a collection of m convex sets $C_1, C_2, \ldots, C_m \in \mathcal{C}$ such that their intersection is the empty set, but the intersection of any $m-1$ of them is non-empty; i.e., $\cap_{j=1}^{m} C_j = \varnothing$ and $\cap_{j\neq i} C_j \neq \varnothing$ for any $i \in [m]$. The *Helly number* ω of $\mathcal{C}$ is then the largest number h such that there exists an h-Helly family for $\mathcal{C}$. We will mostly work with this latter definition of the Helly number. Note that for some spaces, there will exist arbitrarily large Helly families, in which case the Helly number is undefined.[1]

[1] We do not concern ourselves with this case in the statement of our main results, but note that our reasoning often still applies when ω is undefined, for instance when deriving impossibility results. For the rest of this work, we assume that the spaces we consider have a well-defined Helly number ω.

Convex agreement problems. A *convex agreement problem* is defined for a convexity space $\mathcal{C}$ over a universe V; e.g., $\mathbb{R}^D$ with straight-line convexity, or a graph $G = (V, E)$ with monophonic convexity. Each party P starts with an input $v_{\mathsf{in}}^P \in V$ and should produce an output v_{out}^P. Ideally, all outputs should match, and this common output should be in the convex hull of the inputs. However, one has to consider the presence of byzantine parties. Hence, an agreement problem is defined by a collection of properties taking this into account: a validity condition, an agreement condition, and a termination condition. Write V_{in} and V_{out} for the set of inputs v_{in}^P and respectively outputs v_{out}^P of the *honest* parties P.

A convex agreement problem has the following validity and agreement conditions:

Convex Validity: $V_{\mathsf{out}} \subseteq \langle V_{\mathsf{in}} \rangle$ (honest outputs are in the convex hull of honest inputs).
Exact Agreement: $|V_{\mathsf{out}}| = 1$ (honest parties obtain the same output).

Finally, let us discuss the termination requirements of the protocol. There are two flavors, one for deterministic protocols and one for randomized protocols, listed below:

Termination: all honest parties obtain outputs.
Probabilistic Termination: the probability that some honest party has not obtained output after T time units tends to 0 as $T \to \infty$.

We may then define CC as follows:

▶ **Definition 2** (Convex Consensus). *A protocol Π is a (t_s, t_a)-secure CC protocol if it achieves Probabilistic Termination, Convex Validity, and Exact Agreement when up to t_s parties are corrupted if it runs in a synchronous network, and when up to t_a parties are corrupted if running in an asynchronous network.*

3 Resilience Lower Bounds Using the Helly Number

In this section, we establish necessary conditions for achieving CC in the network-agnostic model. Concretely, we show that each of the following conditions is needed: $n > \omega \cdot t_s$, $n > 2 \cdot t_s + t_a$, and $n > \omega \cdot t_a + t_s$. Section 4 will show that these conditions are also sufficient.

We begin by showing that the conditions $n > \omega \cdot t$ and $n > (\omega + 1) \cdot t$ are necessary in the synchronous and resp. asynchronous model, where ω is the Helly number of the convexity space. These already imply that $n > \omega \cdot t_s$ and $n > (\omega + 1) \cdot t_a$ are required in the in the network-agnostic model. Afterward, we move towards conditions that are only required in the network-agnostic model. We note that a previously-known resilience bound [33, Theorem 13] given in terms of the Carathéodory number of the space is incompatible with our results: in general, there is no relation between the Carathéodory number and the Helly number. This bound turns out to be incorrect (detailed discussion in Appendix A).

Before formally showing our lower bounds, we introduce the notion of *adversarial families*, which will enable us to give general scenario-based arguments [32]. Roughly, these are families of pairwise-disjoint sets such that if the honest parties start with inputs from these sets, then Convex Validity forces them to output values from these sets, breaking Exact Agreement. The formal definition follows; see Figure 2 for an example.

▶ **Definition 3.** *Consider a convexity space $\mathcal{C}$ with Helly number ω defined on a universe V. Consider a family $\mathcal{A} = \{A_1, \ldots, A_m\}$ consisting of m non-empty pairwise-disjoint convex sets $A_i \in \mathcal{C}$ and write $A = \cup \mathcal{A}$. Then, $\mathcal{A}$ is* m-adversarial *if $A_i = \cap_{\ell \neq i} \langle A \setminus A_\ell \rangle$ for all $i \in [m]$.*[2]

[2] This definition requires $m > 1$ to avoid taking the intersection of an empty collection of sets. However,

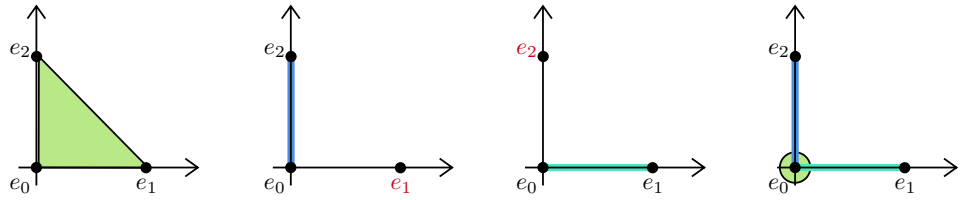


Figure 2 Consider $\mathbb{R}^2$ with straight-line convexity and the vectors $e_0, e_1, e_2 = (0,0), (1,0), (0,1)$, illustrated in the first figure. Define $A_i = \{e_i\}$ for $i = 0, 1, 2$ to be singleton sets for the previous (and hence convex sets). Then, $\mathcal{A} = \{A_0, A_1, A_2\}$ is a 3-adversarial family. To see why, first note that by definition $A = \{e_0, e_1, e_2\}$. By symmetry, it suffices to check the condition for $i = 0$; i.e., to show that $A_0 = \langle A \setminus A_1 \rangle \cap \langle A \setminus A_2 \rangle$. Simplifying, this amounts to $A_0 = \langle \{e_0, e_2\} \rangle \cap \langle \{e_0, e_1\} \rangle$. The second figure illustrates $\langle \{e_0, e_2\} \rangle$, the third illustrates $\langle \{e_0, e_1\} \rangle$, and the forth $\langle \{e_0, e_2\} \rangle \cap \langle \{e_0, e_1\} \rangle$, which is precisely A_0, as required. The illustrations come from [21].

Note that the pairwise-disjoint condition is equivalent to $\cap_{\ell=1}^{m} \langle A \setminus A_\ell \rangle = \varnothing$.[3] We add that, in the example of Figure 2, sets A_i are singletons, but requiring this would strictly decrease the power of adversarial sets in general. The following technical lemma, and the two following it, will be the main tools used to get impossibility results. The techniques used in its proof, supplied in the full version of our paper, are similar in spirit to the proofs for $\mathbb{R}^D$ in [28].

▶ **Lemma 4.** *Let $\mathcal{A} = \{A_1, \ldots, A_m\}$ be an m-adversarial family for convexity space $\mathcal{C}$. Assume $n \geq m$ and that, moreover, $n \leq m \cdot t$ if the network is synchronous and $n \leq (m+1) \cdot t$ if the network is asynchronous. Then, any n-party protocol satisfying Convex Validity and (Probabilistic) Termination has a terminating execution where there are honest parties $P_1, \ldots, P_m$ such that the output v^i_{out} of party P_i satisfies $v^i_{out} \in A_i$.*

The following two technical lemmas give similar guarantees, but in the network-agnostic model. The proof of the first is similar to that for $\mathbb{R}$ in [20], while that of the second is an extension of the asynchronous part of Lemma 4. The proofs are included in the full version of our paper.

▶ **Lemma 5.** *Assume a convexity space $\mathcal{C}$ admitting a 2-adversarial family $\mathcal{A} = \{A_1, A_2\}$. Assume $2 \leq n \leq 2 \cdot t_s + t_a$. Let Π denote an arbitrary protocol achieving Convex Validity and (Probabilistic) Termination for at most t_s corruptions when the network is synchronous and at most t_a corruptions when it is asynchronous. Then, Π has a terminating execution where the outputs v^1_{out} and v^2_{out} of two honest parties satisfy $v^1_{out} \in A_1$ and $v^2_{out} \in A_2$.*

▶ **Lemma 6.** *Let $\mathcal{A} = \{A_1, \ldots, A_m\}$ be an m-adversarial family for convexity space $\mathcal{C}$. Assume that $m \leq n \leq m \cdot t_a + t_s$. Then, any n-party protocol satisfying Convex Validity and (Probabilistic) Termination for at most t_s corruptions when the network is synchronous and at most t_a corruptions when the network is asynchronous has a terminating execution where there are honest parties $P_1, \ldots, P_m$ such that the output v^i_{out} of party P_i satisfies $v^i_{out} \in A_i$.*

We now show a relationship between adversarial families and Helly families.

▶ **Lemma 7.** *Consider a convexity space $\mathcal{C}$, then an m-adversarial family exists if and only if an m-Helly family exists. Hence, the size of the largest adversarial family for a convexity space equals its Helly number ω.*

---

for $m = 1$ all our results will hold if we assume that $\mathcal{A} = \{A\}$ is 1-adversarial for any convex set $A \neq \varnothing$. We will not discuss this technicality further and henceforth assume that $m \geq 1$ is well-defined.

[3] To see this, note that for $i \neq j$ we have $A_i \cap A_j = (\cap_{\ell \neq i} \langle A \setminus A_\ell \rangle) \cap (\cap_{\ell \neq j} \langle A \setminus A_\ell \rangle) = \cap_{\ell=1}^{m} \langle A \setminus A_\ell \rangle$.

Proof. First, consider an adversarial family $\mathcal{A} = \{A_1, \ldots, A_m\}$ for $\mathcal{C}$ and as usual write $A = \cup\mathcal{A}$. The family of sets $\langle A \setminus A_i\rangle_{i\in[m]}$ do not intersect, but any $m-1$ of them do, since for any i we assumed that $A_i = \cap_{\ell\neq i}\langle A \setminus A_\ell\rangle$ is non-empty, so it is an m-Helly family. Conversely, consider an m-Helly family; i.e., convex sets $C_1, \ldots, C_m \in \mathcal{C}$ that do not intersect, but any $m-1$ of them do. Define the family of non-empty convex sets $\mathcal{A} = \{A_1, \ldots, A_m\}$ where $A_i = \cap_{\ell\neq i} C_\ell$. Notice that for $i \neq j$ we have $A_i \cap A_j = \cap_{\ell\in[m]} C_\ell = \varnothing$, so the sets are pairwise disjoint. To show that $\mathcal{A}$ is an m-adversarial family, it remains to show that for all i it holds that $A_i = \cap_{\ell\neq i}\langle A \setminus A_\ell\rangle$. To see this, note that $A \setminus A_\ell = \cup\{A_1, \ldots, A_{\ell-1}, A_{\ell+1}, \ldots, A_m\}$ and that $A_{\ell'} \subseteq C_\ell$ for all $\ell' \neq \ell$, so $A \setminus A_\ell \subseteq C_\ell$. Since C_ℓ is convex, this means that $\langle A \setminus A_\ell\rangle \subseteq \langle C_\ell\rangle = C_\ell$. As a result, $\cap_{\ell\neq i}\langle A \setminus A_\ell\rangle \subseteq \cap_{\ell\neq i} C_\ell = A_i$. To also show that $A_i \subseteq \cap_{\ell\neq i}\langle A \setminus A_\ell\rangle$ just notice that $A_i \subseteq A \setminus A_\ell \subseteq \langle A \setminus A_\ell\rangle$ for all $\ell \neq i$. ◀

Note that a more restrictive definition of adversarial families where all the sets are singletons would not suffice to prove the previous, as in some spaces no singletons are convex.

To access the full power of Lemmas 4 and 6, which require n to be at least the size of the adversarial family, we would like that adversarial families of a certain size imply the existence of adversarial families of all smaller sizes. We show this in the following lemma.

▶ **Lemma 8.** *Given a convexity space, if there exists an m-Helly family, then there exist m'-Helly families for any $1 \leq m' < m$. The same holds if "Helly" is replaced by "adversarial."*

Proof. It suffices to consider $m' = m - 1$. If $C_1, \ldots, C_m$ is an m-Helly family, one can check that $C_1, \ldots, C_{m-2}, (C_{m-1} \cap C_m)$ is an $(m-1)$-Helly family. For the latter, apply Lemma 7. ◀

We now leverage Lemmas 4, 7 and 8 to get the following result, generalizing those in [33] by removing the strong requirement of a convex geometry.

▶ **Theorem 9.** *Consider a convexity space $\mathcal{C}$ with Helly number ω. Assume $n \leq \omega \cdot t$ if the network is synchronous and $n \leq (\omega + 1) \cdot t$ if the network is asynchronous. Then, there is no n-party protocol satisfying Convex Validity and (Probabilistic) Termination such that the set of outputs of the honest parties is guaranteed to have size at most* $\min(n, \omega) - 1$.

Proof. Write $m = \min(n, \omega)$. By Lemma 7, there is an ω-adversarial family for $\mathcal{C}$. Since $m \leq \omega$, using Lemma 8, let $\mathcal{A} = \{A_1, \ldots, A_m\}$ be an m-adversarial family for $\mathcal{C}$. Consider a protocol Π satisfying Convex Validity and Termination. By Lemma 4, there is a terminating execution of Π where the set of honest outputs contains $\{a_1, \ldots, a_m\}$ where $a_i \in A_i$. As sets in $\mathcal{A}$ are pairwise disjoint, this set has cardinality m, implying the conclusion. ◀

By leveraging Lemmas 5, 6, 7 and 8, we similarly get the following result. The proof is included in the full version of our paper.

▶ **Theorem 10.** *Consider a convexity space $\mathcal{C}$ with Helly number $\omega \geq 2$. Assume $2 \leq n \leq 2 \cdot t_s + t_a$ or $2 \leq n \leq \omega \cdot t_a + t_s$. Then, no n-party protocol satisfying Convex Validity, (Probabilistic) Termination, and Exact Agreement can simultaneously tolerate at most t_s corruptions when the network is synchronous and at most t_a when the network is asynchronous.*

4 Achieving Optimal-Resilience Convex Consensus

We now describe a construction achieving CC in the network-agnostic model that matches our previous resilience lower bounds. Concretely, we focus on proving the following theorem.

▶ **Theorem 11.** *If $t_a \leq t_s$ and $n > \max(\omega \cdot t_s, 2 \cdot t_s + t_a, \omega \cdot t_a + t_s)$, there is a protocol achieving (t_s, t_a)-secure CC assuming PKI. The protocol has expected round complexity $O(1)$. If ℓ denotes the universe elements' size in bits and κ is the security parameter, its expected communication complexity is $O(n^3 \cdot \ell + n^4 \cdot \kappa)$ bits. If threshold signatures are available, the expected communication complexity reduces to $O(n^3 \cdot \ell + n^3 \cdot \kappa)$ bits.*

To set up the intuition for our construction, we recall the outline of the synchronous protocol for $\mathbb{R}^D$ with straight-line convexity of [37]. The synchronous model offers powerful communication primitives (i.e., Synchronous Broadcast [17]), enabling the parties to distribute their inputs and obtain an *identical view* of the inputs. This view consists of a set of value-sender pairs, out of which $n - t_s$ correspond to honest parties. Then, the parties derive a *safe area* inside the honest inputs' convex hull by intersecting the convex hulls of all subsets of $n - t_s$ values received, as defined below. We extend the convex hull operator to value-sender sets straightforwardly: ignore party identities and take the convex hull of the values.

▶ **Definition 12** (Safe Area). *Let $\mathcal{M}$ denote a set of value-sender pairs. For a given k, $safe_k(\mathcal{M}) := \bigcap_{M \in restrict_k(\mathcal{M})} \langle M \rangle$, where $restrict_k(\mathcal{M}) := \{M \subseteq \mathcal{M} : |M| = |\mathcal{M}| - k\}$.*

Specifically, if parties received the (same) set $\mathcal{M}$ of $n - t_s + k$ value-sender pairs, they compute their safe area as $safe_k(\mathcal{M})$. We will later show (in a more general form) that, since $n > \omega \cdot t_s$, the safe area obtained is non-empty. Therefore, any value in the common safe area is valid. Hence, parties may output any such value chosen by some deterministic criterion.

Technical assumptions. Implementing such a protocol requires mild assumptions about $\mathcal{C}$: it should be possible to (i) store elements from V and to send them in messages; (ii) compute and intersect convex hulls; (iii) deterministically select a point from the safe area. *Only to express communication complexity bounds, we will also need that $|V| \leq 2^\ell$ for some ℓ.*

Identical views in asynchrony. Building towards our solution achieving network-agnostic guarantees, we first identify the challenges posed by translating the outline above to the purely asynchronous model (where $t_s = t_a$ and $n > (\omega + 1) \cdot t_a$). Given a primitive that provides the parties with an identical view of $n - t_a$ value-sender pairs, CC can be achieved in a similar manner: out of the set $\mathcal{M}$ of $n - t_a$ pairs agreed upon, at most t_a are corrupted. Then, honest parties derive the safe area $safe_{t_a}(\mathcal{M})$ inside their inputs' convex hull, and afterwards take a deterministic decision to obtain the same output.

Achieving the required identical view deterministically is impossible in the asynchronous model [18], but randomization allows for a simple solution by employing a primitive introduced in [7]. This primitive archives *Agreement on a Core-Set* (ACS) when up to $t_a < n/3$ of the parties involved are corrupted, which suffices for our case of $\omega \geq 2$. Roughly speaking, an ACS protocol assumes that each party holds a value meant to be distributed, and enables the parties to obtain the same set $\mathcal{M}$ of $n - t_a$ value-sender pairs. By utilizing the (randomized) ACS protocol presented in [8, Section 4], we achieve asynchronous CC with optimal resilience, in constant expected number of rounds, proving the lower bound $n > (\omega + 1) \cdot t_a$ to be tight.

Exploiting the advantages of synchrony. While the standard definition of ACS provides identical views when the network is asynchronous, the synchronous model still has a crucial advantage that needs to be used to achieve higher resilience. Namely, the key insight on why CC can be achieved up to $t_s < n/\omega$ corruptions in the synchronous model, while $t_a < n/(\omega + 1)$ is necessary in the asynchronous one, is that the former ensures all honest values are delivered. In contrast, in the asynchronous setting, t_a corrupted parties may

replace t_a honest parties: the honest parties' messages get delayed for sufficiently long, while the t_a corrupted parties follow the protocol correctly, but with inputs of their choice. To match the condition $n > \max(\omega \cdot t_s, 2 \cdot t_s + t_a, \omega \cdot t_a + t_s)$ in the network-agnostic model, we hence need an additional property in a synchronous network: all honest values must be included in the output set. Consequently, we propose the following enriched definition:

▶ **Definition 13** (Agreement on a Core-Set). *Let* Π *be a protocol where every party* P *holds an input* v_P *and outputs a set of value-sender pairs* $\mathcal{M}_P$*. We consider the following properties:*
Validity*: Let* P *and* P' *be two honest parties. If* $(v', P') \in \mathcal{M}_P$*, then* $v' = v_{P'}$*.*
Consistency*: If* P *and* P' *are honest,* $(v, P'') \in \mathcal{M}_P$ *and* $(v', P'') \in \mathcal{M}_{P'}$*, then* $v = v'$*.*[4]
T-Output Size*: If an honest party* P *outputs* $\mathcal{M}_P$*, then* $\left|\mathcal{M}_P\right| \geq n - T$*.*
Honest Core*: If an honest party* P *outputs* $\mathcal{M}_P$*, then* $(v_{P'}, P') \in \mathcal{M}_P$ *for every honest* P'*.*
Then, we say that Π *is a* (t_s, t_a)*-secure* ACS *protocol if it achieves the following:*

- *Validity, Consistency, Exact Agreement, Honest Core, Probabilistic Termination when running in a synchronous network where at most* t_s *parties are corrupted;*
- *Validity, Consistency, Exact Agreement,* t_s*-Output Size,*[5] *Probabilistic Termination when running in an asynchronous network where at most* t_a *parties are corrupted.*

Section 5 describes a protocol Π_{ACS} realizing the theorem below (given PKI). We will also describe a protocol for $t_a \leq t_s < n/3$ without PKI, which is suitable for the case $\omega \geq 3$.

▶ **Theorem 14.** *If* $n > 2 \cdot t_s + t_a$ *and* $t_a \leq t_s$*, there is a* (t_s, t_a)*-secure* ACS *protocol* Π_{ACS} *(assuming PKI). The protocol has expected round complexity* $O(1)$*. If* ℓ *denotes the universe elements' size in bits and* κ *is the security parameter, its expected communication complexity is* $O(n^3 \cdot \ell + n^4 \cdot \kappa)$ *bits. If threshold signatures are available, the expected communication complexity reduces to bits.*

If parties distribute their values via Π_{ACS}, they agree on a set $\mathcal{M}$ of $n - t_s + k$ value-sender pairs. If the network is asynchronous, at most t_a of these values are corrupted. In contrast, if the network is synchronous, at most k of these values are corrupted due to Honest Core. To cover both cases, parties locally compute their safe areas as $S := safe_{\max(k,t_a)}(\mathcal{M})$ and deterministically decide on an output $v_{\mathsf{out}} \in S$. Note that, by definition, S is indeed inside the honest inputs' convex hull. In addition, since the input space has Helly number ω and $n > \max(\omega \cdot t_s, \omega \cdot t_a + t_s)$, the safe area can be shown to be non-empty, so such v_{out} can be chosen. The proof of this result, stated below, is contained in the full version of our paper.

▶ **Lemma 15.** *Assume* $n > \max(\omega \cdot t_s, \omega \cdot t_a + t_s)$*, and that* $\mathcal{M}$ *is a set of* $n - t_s + k$ *value-party pairs, where* $0 \leq k \leq t_s$*. Then,* $safe_{\max(k,t_a)}(\mathcal{M}) \neq \varnothing$*.*

We may now conclude the section by providing the formal code of our CC protocol.

Protocol Π_{CC}

Code for party P with input v_{in}

1: Join Π_{ACS} with input v_{in}. Upon obtaining output $\mathcal{M}$:
2: $k := \left|\mathcal{M}\right| - (n - t_s)$; $S := safe_{\max(k,t_a)}(\mathcal{M})$.
3: Choose $v_{\mathsf{out}} \in S$ according to a public, predetermined, deterministic rule. Output v_{out}.

[4] Note that this implies that each party appears at most once as a sender in $\mathcal{M}_P$.
[5] This is intentional: we do not require the stronger property of t_a-Output Size.

Proof of Theorem 11. Π_{ACS} provides the parties with the same set $\mathcal{M}$ of $n - t_s + k$ value-sender pairs, with $0 \leq k \leq t_s$. $\mathcal{M}$ contains at most $\max(k, t_a)$ values from byzantine parties: in a synchronous network, this holds due to Π_{ACS}'s Honest Core property. Hence, there is a subset $M_H \subseteq \mathcal{M}$ of size $|\mathcal{M}| - \max(k, t_a)$ only containing honest inputs. By definition, $M_H \in \mathit{restrict}_{\max(k,t_a)}(\mathcal{M})$, so $S \subseteq \langle M_H \rangle$, i.e., honest parties obtain a safe area S that is included in their inputs' convex hull. Lemma 15 ensures that S is non-empty, and therefore honest parties agree on the same value v_{out} in the honest inputs' convex hull. Consequently, Π_{CC} is (t_s, t_a)-secure CC protocol. ◀

5 Agreement on a Core-Set

In this section, we describe the protocol realizing Theorem 14. We first focus on the easier case $t_a \leq t_s < n/3$: we begin by describing the asynchronous protocol of [8] (as presented in [29]), which fulfills all properties outlined in Definition 13, except for Honest Core in a synchronous network. We adapt this protocol to satisfy Honest Core as well, obtaining (t_s, t_a)-secure ACS when $t_a \leq t_s < n/3$. Note that our CC lower bound of $n > \max(\omega \cdot t_s, 2 \cdot t_s + t_a, \omega \cdot t_a + t_s)$ implies $t_a \leq t_s < n/3$ for $\omega \geq 3$, so the adapted protocol suffices if the latter is true. For $\omega = 2$, however, the adapted protocol is insufficient. Consequently, we then move on to the case $n > 2 \cdot t_s + t_a$, for which we present a novel construction.

We will utilize Reliable Broadcast (rBC) and Byzantine Agreement (BA) as building blocks. We include their definitions in the network-agnostic model below.

▶ **Definition 16** (Reliable Broadcast). *Let Π denote a protocol where a designated party S (the sender) holds a value v_S, and every party P may a value output v_P. Consider the following properties:*
***Validity**: If S is honest, and an honest party outputs v_P, then $v_P = v_S$.*
***Consistency**: If P and P' are honest and output v_P and resp. $v_{P'}$, then $v_P = v_{P'}$.*
***Honest Termination**: If S is honest, all honest parties obtain outputs.*
***Conditional Termination**: If an honest party P outputs, all honest parties obtain outputs.*

We say that Π is a (t_s, t_a)-secure rBC protocol if it achieves Validity, Consistency, Honest Termination, and Conditional Termination up to t_s corruptions if it runs in a synchronous network, and up to t_a corruptions if it runs in an asynchronous network.

▶ **Definition 17** (Byzantine Agreement). *Let Π be a protocol where every party P holds a bit as input and may output a bit, and consider the following property:*
***Weak Validity**: If all honest parties hold input b, no honest party outputs $b' \neq b$.*

Then, Π is a (t_s, t_a)-secure BA protocol if it achieves Weak Validity, Exact Agreement, and Probabilistic Termination up to t_s corruptions if it runs in a synchronous network, and up to t_a corruptions if it runs in an asynchronous network.

The asynchronous ACS protocol of [8]. We describe the protocol of [8], following the variant presented in [29]. We denote this protocol by $\Pi_{a\mathsf{ACS}}$. We highlight once again that $\Pi_{a\mathsf{ACS}}$ is designed for the purely asynchronous model: it assumes a single threshold $t < n/3$ and seeks properties that hold under asynchrony with at most t corrupted parties. To use this protocol in the hybrid model with $t_a \leq t_s < n/3$, one can set $t := t_s$. When this is done, $\Pi_{a\mathsf{ACS}}$ achieves all properties of being a (t_s, t_a)-secure ACS protocol as per Definition 13 (in fact, even (t_s, t_s)-secure), with the exception of Honest Core under synchrony.

In $\Pi_{a\mathsf{ACS}}$, every party first distributes its input value via rBC. Concretely, this is done using Bracha's protocol [12], denoted by $\Pi_{a\mathsf{rBC}}$, which achieves (t, t)-secure rBC for $t < n/3$. Due to asynchronous communication, $\Pi_{a\mathsf{rBC}}$ only guarantees that parties receive a value if

the sender is honest. As a result, at least $n - t$ values eventually get delivered to all parties, but these values might still be received at vastly different times, leading to inconsistent views if parties were to output the first $n - t$ values they received.

Then, to decide on a common output set, the parties will use BA to agree on which values they received and should be included in the output set. We utilize the protocol $\Pi_{a\mathsf{BA}}$ of Mostefaoui et al. [31], which achieves (t, t)-secure BA when $t < n/3$. There will be n parallel invocations of $\Pi_{a\mathsf{BA}}$ – one for each party. When a party P receives a value v from P' via $\Pi_{ar\mathsf{BC}}$, it joins the $\Pi_{a\mathsf{BA}}$ invocation corresponding to P' with input 1. Semantically, if the $\Pi_{a\mathsf{BA}}$ invocation of a party P' returns output 1, then the value distributed by P' via $\Pi_{ar\mathsf{BC}}$ should be included in the output set. Note that, when this happens, the Weak Validity property of $\Pi_{a\mathsf{BA}}$ ensures that at least one honest party P has joined the $\Pi_{a\mathsf{BA}}$ invocation for party P' with input 1. That is, P has received a value v from P', and $\Pi_{ar\mathsf{BC}}$'s Conditional Termination ensures that all honest parties eventually receive the same value v from P'. In simple terms, the value of P' is *worth waiting for*, and parties wait until they receive it before completing the protocol.

Eventually, at least $n - t$ invocations result in output 1 (suppose not, then, since at least $n - t$ honest values are eventually delivered to all honest parties, the honest parties will all join at least $n - t$ invocations of $\Pi_{a\mathsf{BA}}$ with input 1, guaranteeing that those invocations terminate with output 1). To complete the protocol, we still need that all $\Pi_{a\mathsf{BA}}$ invocations complete. Then, whenever some party P observes $n - t$ invocations completing with output 1, it should join all remaining invocations with input 0. Hence, once all honest parties join all $\Pi_{a\mathsf{BA}}$ invocations, all invocations are guaranteed to terminate. Upon observing that all $\Pi_{a\mathsf{BA}}$ invocations have terminated, each party P outputs the set of (at least $n - t$) value-sender pairs corresponding to the $\Pi_{a\mathsf{BA}}$ invocations that returned output 1 (after waiting to receive any outstanding values through $\Pi_{ar\mathsf{BC}}$). This way, the protocol ensures that all honest parties obtain an identical view, achieving all properties required by Definition 13 for asynchronous networks. Note that the Validity property follows immediately from the guarantees of $\Pi_{ar\mathsf{BC}}$.

Adjustments for the network-agnostic model. We now return to the network-agnostic model and adapt $\Pi_{a\mathsf{ACS}}$ to achieve our ACS definition for the parameter range $t_a \leq t_s < n/3$. As previously established, $\Pi_{a\mathsf{ACS}}$ already satisfies all properties required to be a (t_s, t_a)-secure ACS protocol by setting $t := t_s$, except for Honest Core in a synchronous network.

To see why the Honest Core property does not hold, consider a scenario where the network is synchronous, and the t_s corrupted parties follow the protocol correctly with inputs of their choice. All messages are delivered immediately, except for the messages sent by t_s of the honest parties: these are delivered exactly after Δ time. The remaining honest parties complete the protocol before time Δ, and the values of t_s honest parties are missing from the output set. To prevent this, we impose a waiting time to ensure that, if the network is synchronous, all honest parties' messages are received. Running $\Pi_{ar\mathsf{BC}}$ in the synchronous model guarantees additional properties, established in [21]: when an honest party sends a value via $\Pi_{ar\mathsf{BC}}$, all parties receive this output within $c_{ar\mathsf{BC}} \cdot \Delta$ time, where $c_{ar\mathsf{BC}} := 3$. Then, to achieve Honest Core, we impose a waiting period of at least $c_{ar\mathsf{BC}} \cdot \Delta$ time before allowing the parties to participate in $\Pi_{a\mathsf{BA}}$ invocations with input 0. This way, if the network is synchronous, all honest parties join every honest party's $\Pi_{a\mathsf{BA}}$ invocation with input 1, and these invocations return 1. Hence, all honest parties' values are included in the output set.

We include below the adapted $\Pi_{a\mathsf{ACS}}$, which achieves (t_s, t_a)-secure ACS for $t_a \leq t_s < n/3$. In fact, it achieves (t_s, t_s)-secure ACS: it is an asynchronous protocol with an added waiting period to ensure the Honest Core property under synchrony. The protocol incurs expected constant round complexity, and expected communication complexity $O(n^3 \cdot \ell)$, where ℓ denotes the universe elements' size. For the formal analysis, see the full version of our paper.

Adapted protocol $\Pi_{a\mathsf{ACS}}$ [8, 29]

Code for party P with input v ($\Pi_{ar\mathsf{BC}}$ and $\Pi_{a\mathsf{BA}}$ invocations use $t := t_s$)

1: $\tau_{\mathtt{start}} := \tau_{\mathtt{now}}$
2: Send v to every party via $\Pi_{ar\mathsf{BC}}$.
3: When receiving a value v from P' via $\Pi_{ar\mathsf{BC}}$:
4: If $\tau_{\mathtt{now}} \leq \tau_{\mathtt{start}} + c_{ar\mathsf{BC}} \cdot \Delta$ or less than $n - t_s$ invocations of $\Pi_{ar\mathsf{BC}}$ returned 1:
5: Join the invocation of $\Pi_{a\mathsf{BA}}$ for P' with input 1.
6: When $\tau_{\mathtt{now}} > \tau_{\mathtt{start}} + c_{ar\mathsf{BC}} \cdot \Delta$ and at least $n - t_s$ of the $\Pi_{a\mathsf{BA}}$ invocations returned 1:
7: Join the remaining $\Pi_{a\mathsf{BA}}$ invocations with input 0.
8: When all $\Pi_{a\mathsf{BA}}$ invocations have terminated:
9: $\mathcal{P}$:= parties whose corresponding $\Pi_{a\mathsf{BA}}$ invocations have terminated with output 1.
10: When all invocations of $\Pi_{ar\mathsf{BC}}$ having senders in $\mathcal{P}$ have terminated:
11: $\mathcal{M}$:= the set of pairs (v', P'), where $P' \in \mathcal{P}$ and v' is the value P' sent via $\Pi_{ar\mathsf{BC}}$.
12: Output $\mathcal{M}$.

Achieving ACS when $n > 2 \cdot t_s + t_a$. Finally, we describe our solution for the general case. We utilize building blocks designed specifically for this setting: the (t_s, t_a)-secure rBC protocol Π_{rBC} of Momose and Ren [30], and the (t_s, t_a)-secure BA protocol Π_{BA} of Deligios, Hirt and Liu-Zhang [15, Corollary 2]. We add that these protocols assume and need PKI.

While the $t_a \leq t_s < n/3$ setting enabled a solution based on tweaks to previously known protocols, the $n > 2 \cdot t_s + t_a$ case introduces different challenges. In particular, one detail we omitted when presenting $\Pi_{a\mathsf{ACS}}$ concerns protocol composability. Namely, Definition 17 of BA protocols assumes that honest parties join the protocol simultaneously in a synchronous network. In the outline of [8] and in $\Pi_{a\mathsf{ACS}}$, this is, in fact, not the case. However, when $t_s = t_a$, this assumption is not strictly required because the asynchronous guarantees step in whenever honest parties are unable to join simultaneously. In contrast, when $t_s > t_a$, (t_s, t_a)-secure BA does not have to provide any guarantees in a synchronous network with t_s corruptions if honest parties are unable to join simultaneously. This is especially problematic when $t_s \geq n/3$ because (t_s, t_s)-secure BA protocols do not exist. The Conditional Termination property of Π_{rBC} is too weak to ensure that honest parties are ready to join the BA invocations simultaneously when the network is synchronous – a challenge that we need to overcome.

Our goal is then to allow the parties to join each invocation of Π_{BA} at the same time when the network is synchronous – this refers both to invocations where parties join with input 1, and to invocations where parties join with input 0. We will do so by enabling the parties to decide their input bit for each invocation of Π_{BA} independently of the other invocations' outcomes. On top of this property, we still need to guarantee t_s-Output Size when the network is asynchronous, which amounts to ensuring that at least $n - t_s$ invocations of Π_{BA} return 1. Moreover, when the network is synchronous, the Π_{BA} invocations of honest parties have to output 1 to ensure the Honest Core property.

To enable the honest parties to safely decide each input bit for the Π_{BA} invocations independently, realizing the previous desiderata, we introduce a network-agnostic version of a primitive known as *Gather* (GTHR) [2,13]. This is a slightly weaker, but deterministic variant of ACS: GTHR relaxes Exact Agreement by only requiring that honest parties' output sets have at least $n - t_s$ values in common. Our definition of GTHR, provided below, additionally requires the previous Honest Core property to hold under synchrony. Moreover, we require that honest parties obtain outputs simultaneously if the network is synchronous.

▶ **Definition 18** (Gather). *Let Π be a protocol where every party P holds an input v_P and may output a set of value-sender pairs $\mathcal{M}_P$. We consider the following properties, additionally to those in Definition 13.*

***T*-Common Core**: *If all honest parties obtain outputs, then* $\left|\bigcap_{P \text{ honest}} \mathcal{M}_P\right| \geq n - T$.
Simultaneous Termination: *The honest parties obtain outputs simultaneously.*

Then, we say that Π *is a* (t_s, t_a)*-secure* GTHR *protocol if it achieves:*

- *Validity, Consistency, Honest Core and Simultaneous Termination when running in a synchronous setting where at most* t_s *of the parties involved are corrupted;*
- *Validity, Consistency,* t_s*-Common Core and Termination when running in an asynchronous setting where at most* t_a *of the parties involved are corrupted.*

In Appendix B, we provide a construction achieving our GTHR definition, as stated below. This is obtained by adding one step to the network-agnostic Overlap All-to-All Broadcast protocol of [20], while using insights from asynchronous variants of GTHR [2].

▶ **Theorem 19.** *If* $n > 2 \cdot t_s + t_a$ *and* $t_a \leq t_s$*, there is a* (t_s, t_a)*-secure* GTHR *protocol* Π_{GTHR} *(assuming PKI). The protocol has round complexity* $O(1)$*. If* ℓ *denotes the universe elements' size in bits and* κ *is the security parameter, it achieves a communication complexity of* $O(n^3 \cdot \ell + n^4 \cdot \kappa)$ *bits (can be reduced to* $O(n^3 \cdot \ell + n^3 \cdot \kappa)$ *with threshold signatures).*

Then, our ACS protocol proceeds as follows: the parties distribute their inputs using the Π_{GTHR} protocol realizing Theorem 19. When obtaining outputs $\mathcal{M}_{\mathsf{GTHR}}$ (potentially different for different parties), each party P joins the n invocations of Π_{BA}. P inputs 1 in the invocation for P' if $\mathcal{M}_{\mathsf{GTHR}}$ contains some value from P' and 0 otherwise. Note that, if the network is synchronous, Π_{GTHR} provides Simultaneous Termination, hence honest parties join Π_{BA} simultaneously, and therefore the guarantees of Π_{BA} now hold. Since the value-sender pairs $\mathcal{M}_{\mathsf{GTHR}}$ obtained by the honest parties intersect in $n - t_s$ pairs, at least $n - t_s$ of the Π_{BA} invocations result in output 1. In addition, if the network is synchronous, Π_{GTHR}'s Honest Core ensures that every invocation corresponding to an honest party returns 1.

An important and subtle caveat in the above is that obtaining output 1 in the Π_{BA} invocation for P' does not mean that all honest parties have received a value from P' via Π_{GTHR}. Although Π_{BA} ensures that at least one honest party P has received a value from P' via Π_{GTHR}, this may not be the case for all honest parties. To address this, we state an additional property provided by our implementation of Π_{GTHR}: if parties wait for sufficiently long even after obtaining outputs via Π_{GTHR}, they will (consistently) receive the missing values as well. This is because, internally to Π_{GTHR}, parties distribute their inputs via Π_{rBC}.

▶ **Observation 20.** *Let* P *and* P' *denote two honest parties, and let* $\mathcal{M}$ *and* $\mathcal{M}'$ *denote their internal message sets in* Π_{GTHR}*. If* $(v, P'') \in \mathcal{M}$*, then, eventually,* $(v, P'') \in \mathcal{M}'$ *as well.*

We may now present our ACS protocol. We defer the analysis to Appendix C.

Protocol Π_{ACS}

Code for party P with input v

1: Join Π_{GTHR} with input v. When receiving output $\mathcal{M}_{\mathsf{GTHR}}$ from Π_{GTHR}:
2: For each party P':
3: $b_{P'} := 1$ if $(v', P') \in \mathcal{M}_{\mathsf{GTHR}}$ for some v' and 0 otherwise.
4: Join the Π_{BA} invocation for P' with input $b_{P'}$.
5: When receiving v' from P' via Π_{rBC} [initiated in Π_{GTHR}], add (v', P') to $\mathcal{M}_{\mathsf{GTHR}}$.
6: When obtaining outputs in all invocations of Π_{BA}:
7: $\mathcal{P}$:= the set of parties whose Π_{BA} invocations returned output 1.
8: When $(v', P') \in \mathcal{M}_{\mathsf{GTHR}}$ for every $P' \in \mathcal{P}$:
9: Output $\mathcal{M}$:= the set of pairs $(v', P') \in \mathcal{M}_{\mathsf{GTHR}}$ with $P' \in \mathcal{P}$.

6 Conclusions

We investigated the necessary and sufficient conditions for achieving CC in the network-agnostic model, providing a necessary and sufficient condition for solvability. We have seen that, for any convexity space with Helly number ω, achieving CC, or, more precisely, Convex Hull Validity, (Probabilistic) Termination and Agreement on at most $\min(n, \omega) - 1$ values requires $n > \omega \cdot t$ in synchronous networks and $n > (\omega + 1) \cdot t$ in asynchronous networks. In the network-agnostic model, we have shown that $n > \max(\omega \cdot t_s, \omega \cdot t_a + t_s, 2 \cdot t_s + t_a)$ is necessary and sufficient for achieving CC. To this end, we provided a (t_s, t_a)-secure CC protocol Π_{CC} by making use of randomization, which can be seen to be necessary due to the FLP result [18], and assuming PKI only for the particular case $\omega = 2$ (where cryptographic setup is needed when $t_s \geq n/3$ [23]).

In the process, we proposed two communication primitives for the network-agnostic model, which may be of independent interest. These are variants of ACS and GTHR, which allow each party to distribute its input so that parties obtain consistent views on the original inputs. These stronger properties enabled us to ensure the synchronous resilience guarantees of CC in the network-agnostic model. With its stronger guarantees, our ACS protocol can simplify future works on network-agnostic secure Multi-Party Computation, where ACS protocols are often employed during the input-sharing part of the protocol (for instance, [11] uses a less general form of ACS).

References

1 Ittai Abraham, Yonatan Amit, and Danny Dolev. Optimal resilience asynchronous approximate agreement. In Teruo Higashino, editor, *Principles of Distributed Systems*, pages 229–239, Berlin, Heidelberg, 2005. Springer Berlin Heidelberg.

2 Ittai Abraham, Philipp Jovanovic, Mary Maller, Sarah Meiklejohn, Gilad Stern, and Alin Tomescu. Reaching consensus for asynchronous distributed key generation. In *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*, PODC'21, pages 363–373, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3465084.3467914.

3 Manuel Alcántara, Armando Castañeda, David Flores-Peñaloza, and Sergio Rajsbaum. The topology of look-compute-move robot wait-free algorithms with hard termination. *Distributed Computing*, 32(3):235–255, 2019. doi:10.1007/s00446-018-0345-3.

4 Andreea B. Alexandru, Erica Blum, Jonathan Katz, and Julian Loss. State machine replication under changing network conditions. In *Advances in Cryptology – ASIACRYPT 2022: 28th International Conference on the Theory and Application of Cryptology and Information Security, Taipei, Taiwan, December 5–9, 2022, Proceedings, Part I*, pages 681–710, Berlin, Heidelberg, 2023. Springer-Verlag. doi:10.1007/978-3-031-22963-3_23.

5 Dan Alistarh, Faith Ellen, and Joel Rybicki. Wait-free approximate agreement on graphs. In Tomasz Jurdziński and Stefan Schmid, editors, *Structural Information and Communication Complexity*, pages 87–105, Cham, 2021. Springer International Publishing. doi:10.1007/978-3-030-79527-6_6.

6 Ananya Appan, Anirudh Chandramouli, and Ashish Choudhury. Perfectly-secure synchronous mpc with asynchronous fallback guarantees. In *Proceedings of the 2022 ACM Symposium on Principles of Distributed Computing*, PODC'22, pages 92–102, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3519270.3538417.

7 Michael Ben-Or, Ran Canetti, and Oded Goldreich. Asynchronous secure computation. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing*, STOC '93, pages 52–61, New York, NY, USA, 1993. Association for Computing Machinery. doi:10.1145/167088.167109.

8 Michael Ben-Or, Boaz Kelmer, and Tal Rabin. Asynchronous secure computations with optimal resilience (extended abstract). In *Proceedings of the Thirteenth Annual ACM Symposium on Principles of Distributed Computing*, PODC '94, pages 183–192, New York, NY, USA, 1994. Association for Computing Machinery. doi:10.1145/197917.198088.

9 Erica Blum, Jonathan Katz, and Julian Loss. Synchronous consensus with optimal asynchronous fallback guarantees. In *Theory of Cryptography*, volume 11891 of *Lecture Notes in Computer Science*, pages 131–150, Cham, 2019. Springer International Publishing. doi:10.1007/978-3-030-36030-6_6.

10 Erica Blum, Jonathan Katz, and Julian Loss. Tardigrade: An atomic broadcast protocol for arbitrary network conditions. In Mehdi Tibouchi and Huaxiong Wang, editors, *ASIACRYPT 2021, Part II*, volume 13091 of *LNCS*, pages 547–572. Springer, Heidelberg, December 2021. doi:10.1007/978-3-030-92075-3_19.

11 Erica Blum, Chen-Da Liu-Zhang, and Julian Loss. Always have a backup plan: Fully secure synchronous mpc with asynchronous fallback. In Daniele Micciancio and Thomas Ristenpart, editors, *Advances in Cryptology – CRYPTO 2020*, pages 707–731, Cham, 2020. Springer International Publishing. doi:10.1007/978-3-030-56880-1_25.

12 Gabriel Bracha. Asynchronous byzantine agreement protocols. *Information and Computation*, 75(2):130–143, 1987. doi:10.1016/0890-5401(87)90054-X.

13 Ran Canetti and Tal Rabin. Fast asynchronous byzantine agreement with optimal resilience. In *Proceedings of the Twenty-Fifth Annual ACM Symposium on Theory of Computing*, STOC '93, pages 42–51, New York, NY, USA, 1993. Association for Computing Machinery. doi:10.1145/167088.167105.

14 Armando Castañeda, Sergio Rajsbaum, and Matthieu Roy. Convergence and covering on graphs for wait-free robots. *Journal of the Brazilian Computer Society*, 24(1):1, January 2018. doi:10.1186/s13173-017-0065-8.

15 Giovanni Deligios, Martin Hirt, and Chen-Da Liu-Zhang. Round-efficient byzantine agreement and multi-party computation with asynchronous fallback. In Kobbi Nissim and Brent Waters, editors, *Theory of Cryptography*, pages 623–653, Cham, 2021. Springer International Publishing. doi:10.1007/978-3-030-90459-3_21.

16 Danny Dolev, Nancy A. Lynch, Shlomit S. Pinter, Eugene W. Stark, and William E. Weihl. Reaching approximate agreement in the presence of faults. *J. ACM*, 33(3):499–516, May 1986. doi:10.1145/5925.5931.

17 Danny Dolev and H. Raymond Strong. Authenticated algorithms for byzantine agreement. *SIAM Journal on Computing*, 12(4):656–666, 1983. doi:10.1137/0212045.

18 Michael J Fischer, Nancy A Lynch, and Michael S Paterson. Impossibility of distributed consensus with one faulty process. *Journal of the ACM (JACM)*, 32(2):374–382, 1985. doi:10.1145/3149.214121.

19 Matthias Fitzi and Juan A. Garay. Efficient player-optimal protocols for strong and differential consensus. In Elizabeth Borowsky and Sergio Rajsbaum, editors, *22nd ACM PODC*, pages 211–220. ACM, July 2003. doi:10.1145/872035.872066.

20 Diana Ghinea, Chen-Da Liu-Zhang, and Roger Wattenhofer. Optimal synchronous approximate agreement with asynchronous fallback. In *Proceedings of the 2022 ACM Symposium on Principles of Distributed Computing*, PODC'22, pages 70–80, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3519270.3538442.

21 Diana Ghinea, Chen-Da Liu-Zhang, and Roger Wattenhofer. Multidimensional approximate agreement with asynchronous fallback. In *Proceedings of the 35th ACM Symposium on Parallelism in Algorithms and Architectures*, SPAA '23, pages 141–151, New York, NY, USA, 2023. Association for Computing Machinery. doi:10.1145/3558481.3591105.

22 Simon Holmgaard Kamp and Jesper Buus Nielsen. Byzantine agreement decomposed: Honest majority asynchronous atomic broadcast from reliable broadcast. Cryptology ePrint Archive, Paper 2023/1738, 2023. URL: https://eprint.iacr.org/2023/1738.

23 Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, 1982. doi:10.1145/357172.357176.

24 Jérémy Ledent. Brief announcement: Variants of approximate agreement on graphs and simplicial complexes. In *Proceedings of the 2021 ACM Symposium on Principles of Distributed Computing*, PODC'21, pages 427–430, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3465084.3467946.

25 Christoph Lenzen and Julian Loss. Optimal clock synchronization with signatures. In *Proceedings of the 2022 ACM Symposium on Principles of Distributed Computing*, PODC'22, pages 440–449, New York, NY, USA, 2022. Association for Computing Machinery. doi:10.1145/3519270.3538444.

26 Shihao Liu. The Impossibility of Approximate Agreement on a Larger Class of Graphs. In Eshcar Hillel, Roberto Palmieri, and Etienne Rivière, editors, *26th International Conference on Principles of Distributed Systems (OPODIS 2022)*, volume 253 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 22:1–22:20, Dagstuhl, Germany, 2023. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.OPODIS.2022.22.

27 Hammurabi Mendes and Maurice Herlihy. Multidimensional approximate agreement in byzantine asynchronous systems. In Dan Boneh, Tim Roughgarden, and Joan Feigenbaum, editors, *45th ACM STOC*, pages 391–400. ACM Press, June 2013. doi:10.1145/2488608.2488657.

28 Hammurabi Mendes, Maurice Herlihy, Nitin Vaidya, and Vijay K Garg. Multidimensional agreement in byzantine systems. *Distributed Computing*, 28(6):423–441, 2015. doi:10.1007/S00446-014-0240-5.

29 Andrew Miller, Yu Xia, Kyle Croman, Elaine Shi, and Dawn Song. The honey badger of BFT protocols. In Edgar R. Weippl, Stefan Katzenbeisser, Christopher Kruegel, Andrew C. Myers, and Shai Halevi, editors, *ACM CCS 2016*, pages 31–42. ACM Press, October 2016. doi:10.1145/2976749.2978399.

30 Atsuki Momose and Ling Ren. Multi-threshold byzantine fault tolerance. In *Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security*, CCS '21, pages 1686–1699, New York, NY, USA, 2021. Association for Computing Machinery. doi:10.1145/3460120.3484554.

31 Achour Mostefaoui, Hamouma Moumen, and Michel Raynal. Signature-free asynchronous byzantine consensus with $t < n/3$ and $o(n^2)$ messages. In *Proceedings of the 2014 ACM Symposium on Principles of Distributed Computing*, PODC '14, pages 2–9, New York, NY, USA, 2014. Association for Computing Machinery. doi:10.1145/2611462.2611468.

32 Gil Neiger. Distributed consensus revisited. *Information Processing Letters*, 49(4):195–201, 1994. doi:10.1016/0020-0190(94)90011-6.

33 Thomas Nowak and Joel Rybicki. Byzantine Approximate Agreement on Graphs. In Jukka Suomela, editor, *33rd International Symposium on Distributed Computing (DISC 2019)*, volume 146 of *Leibniz International Proceedings in Informatics (LIPIcs)*, pages 29:1–29:17, Dagstuhl, Germany, 2019. Schloss Dagstuhl – Leibniz-Zentrum für Informatik. doi:10.4230/LIPIcs.DISC.2019.29.

34 Thomas Nowak and Joel Rybicki. Byzantine approximate agreement on graphs, 2019. arXiv:1908.02743.

35 Gerard Sierksma. Caratheodory and helly-numbers of convex-product-structures. *Pacific Journal of Mathematics*, 61:275–282, 1975.

36 Lewis Tseng and Nitin H. Vaidya. Asynchronous convex hull consensus in the presence of crash faults. In Magnús M. Halldórsson and Shlomi Dolev, editors, *33rd ACM PODC*, pages 396–405. ACM, July 2014. doi:10.1145/2611462.2611470.

37 Nitin H. Vaidya and Vijay K. Garg. Byzantine vector consensus in complete graphs. In Panagiota Fatourou and Gadi Taubenfeld, editors, *32nd ACM PODC*, pages 65–73. ACM, July 2013. doi:10.1145/2484239.2484256.

Appendix

A Comparison with [33, Theorems 17 and 13]

In this section, we compare our impossibility results with the related [33, Theorems 17 and 13]. We find that our results generalize the aforementioned, with the exception of the first part of [33, Theorems 13], to which our findings are orthogonal. However, we exhibit an error in the proof of this part, rendering the result false in general.

To better understand these results, let us first introduce convex geometries. An abstract convexity space $\mathcal{C}$ on universe V is a *convex geometry* if it additionally satisfies that, for all convex sets $C \subsetneq V$, there exists $v \in V \setminus C$ such that $C \cup \{v\}$ is convex. This is a non-trivial requirement; e.g., $\mathbb{R}^D$ with straight-line convexity or box convexity is **not** a convex geometry. An example of a convex geometry is a chordal graph endowed with monophonic convexity. These notions are further discussed in the full version of our paper.

▶ **Theorem 21** ([33, Theorem 17]). *Let $\mathcal{C}$ be a convex geometry with Helly number ω. If the network is synchronous and $n \leq \omega \cdot t$, then no n-party protocol satisfies Convex Validity, Termination and Exact agreement.*

Contrasting this with Theorem 9, for the synchronous case our results generalize the previous by removing the strong requirement on $\mathcal{C}$ to be a convex geometry and by adding the fact that even agreement on at most $\min(n, \omega) - 1$ values is not possible.

Before continuing, we need to introduce a few more notions from convexity theory. One relevant notion will be that of *extreme* points. Namely, given a non-necessarily convex set $S \subseteq V$, the set $ex(S) = \{s \in S \mid \langle S \setminus s \rangle \subsetneq \langle S \rangle\}$ is the set of points in S any of whose removal would "shrink" the convex hull. Set S is called *free* if $\langle S \rangle = ex(S)$. Note that free sets are necessarily convex, as $\langle S \rangle = ex(S) \subseteq S \subseteq \langle S \rangle$, from which $\langle S \rangle = S$. Equivalently, S is free if and only if S is convex and $S = ex(S)$.

▶ **Theorem 22** ([33, Theorem 13]). *Let $\mathcal{C}$ be a convexity space with Helly number ω and Carathéodory number c. Assume the network is asynchronous and consider a protocol satisfying Convex Validity and termination, then:*

1. *If $n \leq (c+1) \cdot t$ there is an execution where the honest outputs do not form a free set in $\mathcal{C}$.*
2. *If $n \leq (\omega+1) \cdot t$ and $\mathcal{C}$ is a convex geometry, there is an execution where the set of honest outputs either has size at least ω or is not a free set in $\mathcal{C}$.*

Contrasting with Theorem 9, for the asynchronous case, our results generalize Part 2 of the above by once again removing the requirement on $\mathcal{C}$ to be a convex geometry and also by no longer requiring the clause "or is not a free set in $\mathcal{C}$." Our result also replaces ω by $\min(n, \omega)$, which we believe is also implicitly meant in the original result, as when $n < \omega$ the condition becomes vacuous, and a protocol where parties just output their own inputs satisfies Convex Validity and termination in some convex geometries.

Part 1 of Theorem 22, on the other hand, is orthogonal to our results. In our attempt to use adversarial families to potentially also recover Part 1, we have discovered an error in the proof of this part, making the result false in general. To display the error, we need to introduce one more notion: call a (not necessarily convex) subset $\mathcal{I} \subseteq V$ *irredundant* if there is a point $p \in \langle I \rangle$ such that the hull of no proper subset of $\mathcal{I}$ contains p. The Carathéodory number c of $\mathcal{C}$ is then the size of the largest such irredundant set $\mathcal{I}$. Now, the proof of Part 1 hinges on the following technical lemma:

▶ **Lemma 23** ([33, Lemma 15 of the full version [34]]). *Let $\mathcal{C}$ be a convexity space and A be an irredundant set such that $|A| > 1$. Then for any $a \in A$ and $y \in \langle A \rangle \setminus A$ there exists $b \in A \setminus \{a\}$ such that $y \notin \langle A \setminus \{b\} \rangle$.*

Note that we have added the condition "A is irredundant" missing from the original statement.[6] The error in the proof is towards the end where, using the original notation, it is stated that $y \notin \partial A = \langle A \rangle \setminus B \subseteq \langle A \rangle \setminus A$ implies that $y \notin \langle A \rangle \setminus A$, contradicting the hypothesis. However, in general, if some sets satisfy $S_1 \subseteq S_2$ and $y \notin S_1$ it does not follow that $y \notin S_2$. We next construct a convexity space where the lemma in fact fails for all irredundant sets A and all $a \in A$. First, introduce some auxiliary notation: given two convexity spaces $\mathcal{C}_1$ and $\mathcal{C}_2$ defined on universes V_1 and V_2 respectively, define $\mathcal{C}_1 \oplus \mathcal{C}_2$ to be the convexity space on universe $V_1 \times V_2$ such that $\mathcal{C}_1 \oplus \mathcal{C}_2 = \{C_1 \times C_2 \mid C_1 \in \mathcal{C}_1, C_2 \in \mathcal{C}_2\}$. For the construction, start with an arbitrary convexity space $\mathcal{C}$ on universe V and consider the convexity space $\mathcal{C}' = \mathcal{C} \oplus \{\varnothing, \{0,1\}\}$. To build intuition for $\mathcal{C}'$, notice that $\langle \{(v,i)\} \rangle = \{(v,0),(v,1)\}$ for any $v \in V$ and $i \in \{0,1\}$. Assume A is an irredundant set for $\mathcal{C}'$. Note that for no $v \in V$ does A contain both points $(v,0)$ and $(v,1)$, as otherwise it would be that $\langle A \rangle = \langle A \setminus \{(v,1)\} \rangle$, so A would not be irredundant. Consider any $a = (v,i) \in A$ and take $y = (v, 1-i) \in \langle A \rangle \setminus A$, then for any $b \in A \setminus \{a\}$ it holds that $a \in A \setminus \{b\}$, from which $\langle \{a\} \rangle = \{(v,0),(v,1)\} \subseteq \langle A \setminus \{b\} \rangle$, so $y \in \langle A \setminus \{b\} \rangle$, contradicting the statement of the lemma. Hence, we have constructed a space for which the lemma fails for any irredundant set A and any $a \in A$, indicating that any correct weakening of the lemma might sadly not be of much use in its current form.

We conclude by constructing a space whose Carathéodory number c is much larger than its Helly number ω, showing that our possibility results are not consistent with Part 2 of Theorem 22. To do so, we will use the fact [35, Theorems 2.1 and 3.2] that given convexity spaces $\mathcal{C}_1$ and $\mathcal{C}_2$ with Helly numbers ω_1, ω_2 and Carathéodory numbers c_1, c_2 the space $\mathcal{C}_1 \oplus \mathcal{C}_2$ has Helly number $\omega = \max\{\omega_1, \omega_2\}$ and Carathéodory number c satisfying $c_1 + c_2 - 2 \leq c \leq c_1 + c_2$. Consider the space $\mathcal{C} = \mathbb{R}^2$ with straight-line convexity, whose Helly and Carathéodory numbers are both 3. Then, the space $\mathcal{C}_k = \bigoplus_{\ell=1}^{k} \mathcal{C}$ has Helly number $\omega_k = 3$ and Carathéodory number $c_k \geq 3k - 2(k-1) = k + 2$. For this space, our possibility results imply that, when the network is asynchronous, convex consensus be solved assuming $n > 4 \cdot t$, while Part 1 of Theorem 22 would imply that it can not be solved for $n \leq (k+3) \cdot t$, which are incompatible statements for k large enough.

The keen-eyed reader might note that [33,34] require the universe to be finite, which is not the case for our counterexample. To also construct a counterexample with a finite universe, we will use the same technique, replacing $\mathbb{R}^2$ with straight-line convexity by a finite convexity space $\mathcal{X}$. The requirements for our technique to apply are mild since increasing k keeps the Helly number constant while strictly increasing the lower bound on the Carathéodory number, provided the Carathéodory number of $\mathcal{X}$ is at least 3. Hence, for k sufficiently large, the Carathéodory number of $\mathcal{C}_k = \bigoplus_{\ell=1}^{k} \mathcal{C} = \bigoplus_{\ell=1}^{k} \mathcal{X}$ will exceed its Helly number.[7] It remains to construct a finite $\mathcal{X}$ with Carathéodory number at least 3. This is not at all difficult: let the universe be four points $A, B, C, D \in \mathbb{R}^2$ such that A, B, C form an equilateral triangle and D is the center of the triangle, and the convexity notion be inherited from $\mathbb{R}^2$

[6] The proof of the lemma notes that if A is not irredundant the claim becomes vacuous, however, one can actually consider $\mathbb{R}^2$ with straight-line convexity, $A = \{(\pm 1, \pm 1)\}$ and $y = (0.5, 0.5)$, in which case $y \in \langle A \setminus \{a\} \rangle$ for any $a \in A$. This issue is however only minor since the lemma is only invoked in the proof of the subsequent [33, Lemma 16 of the full version [34]], where A is assumed to be irredundant.

[7] Note that for finite $\mathcal{X}$ the Helly number is always well-defined.

with straight-line convexity; i.e., $\langle\{A, B, C\}\rangle = \{A, B, C, D\}$. This space has a Carathéodory number of at least 3 because the set $\{A, B, C\}$ is irredundant (in fact exactly 3 because $\{A, B, C, D\}$ is not irredundant).

B Gather

In the following, we describe our protocol Π_{GTHR} realizing Theorem 19, restated below.

▶ **Theorem 19.** *If $n > 2 \cdot t_s + t_a$ and $t_a \leq t_s$, there is a (t_s, t_a)-secure GTHR protocol Π_{GTHR} (assuming PKI). The protocol has round complexity $O(1)$. If ℓ denotes the universe elements' size in bits and κ is the security parameter, it achieves a communication complexity of $O(n^3 \cdot \ell + n^4 \cdot \kappa)$ bits (can be reduced to $O(n^3 \cdot \ell + n^3 \cdot \kappa)$ with threshold signatures).*

As mentioned in Section 5, our protocol Π_{GTHR} adds one more step to the Overlap All-to-All Broadcast (oBC) protocol of [20], which we denote by Π_{oBC}. We need to highlight the difference between the definition of (t_s, t_a)-secure oBC presented in [20] and our definition of (t_s, t_a)-secure GTHR: oBC does not require t_s-Common Core. Instead, the oBC definition of [20] requires the weaker property t_s-Overlap:

***T*-Overlap**: If two honest parties P and P' terminate, then $\left|\mathcal{M}_P \cap \mathcal{M}_{P'}\right| \geq n - T$.

Until we reach the additional step, the protocols Π_{oBC} and Π_{GTHR} are identical. Π_{oBC} (and hence also Π_{GTHR}) heavily relies on the *witness technique* [1]. That is, in both protocols, parties send their values via Π_{rBC}. Then, they report to each other which values they received. When the values reported by a party P match the values received by a party P' via Π_{rBC}, P' marks P as a *witness*. In Π_{oBC}, parties are ready to terminate when (i) sufficient time has passed for honest values to be received via Π_{rBC} in a synchronous network, ensuring Honest Core; (ii) parties have gathered sufficient witnesses to ensure that every two honest parties P and P' have a common witness $P^\star$. This way, P and P' have received the same set of at least $n - t_s$ values reported by $P^\star$, and therefore the t_s-Overlap property holds. To achieve the superior guarantee t_s-Common Core required by GTHR in the asynchronous model, we will need a stronger termination condition as well.

Termination time in rBC. Before describing the protocol precisely, we need to include the rBC definition of [21], which makes the termination time explicit.

▶ **Definition 24** (Reliable Broadcast with explicit termination time)**.** *Let Π be a protocol where a designated party S (the sender) holds a value v_S, and every party P may output a value v_P. Consider the following properties:*
***Validity**: If S is honest, and an honest party outputs v_P, then $v_P = v_S$.*
***Consistency**: If P and P' are honest and output v_P and resp. $v_{P'}$, then $v_P = v_{P'}$.*
***c*-Honest Termination**: If S is honest, parties obtain outputs eventually. In addition, if the network is synchronous and the parties start executing the protocol at the same time τ, every honest party obtains output by time $\tau + c \cdot \Delta$.*
***c'*-Conditional Termination**: If an honest party P obtains output at time τ, then all honest parties obtain outputs eventually. In addition, if the network is synchronous and the honest parties start executing the protocol at the same time, then all honest parties obtain output by time $\tau + c' \cdot \Delta$.*

We say that Π is a (t_s, t_a, c, c')-secure Reliable Broadcast protocol if it achieves Validity, Consistency, c-Honest Termination, and c'-Conditional Termination even when t_s of the parties involved are corrupted if it runs in a synchronous network, and when up to t_a corruptions if it runs in an asynchronous network.

As mentioned in Section 5, we make use of the rBC protocol Π_{rBC} of Momose and Ren [30] described in the theorem below. The guarantees regarding explicit termination time follow from the analysis of [20].

▶ **Theorem 25** (Momose and Ren [30]). *Assume that $n > 2 \cdot t_s + t_a$ and $t_s \geq t_a$. Then, there is an n-party protocol achieving $(t_s, t_a, c_{rBC}, c'_{rBC})$-secure rBC (assuming PKI), where $c_{rBC} := 3$ and $c'_{rBC} := 1$. The protocol has round complexity $O(1)$. If ℓ denotes the universe elements' size in bits and κ is the security parameter, it achieves a communication complexity of $O(n^2 \cdot \ell + n^3 \cdot \kappa)$ bits. If, in addition, threshold signatures are available, the communication complexity reduces to $O(n^2 \cdot \ell + n^2 \cdot \kappa)$.*

Common steps of Π_{oBC} and Π_{GTHR}. We now describe the common steps of Π_{oBC} and Π_{GTHR} more precisely. Parties distribute their inputs via Π_{rBC}. When a party receives a value v from P via Π_{rBC}, it adds (v, P) to a set of value-party (or value-sender) pairs $\mathcal{M}$. Additionally, it adds P to a set W_0, representing *level-zero witnesses*. When at least $c_{\mathsf{rBC}} \cdot \Delta$ time has passed (meaning that, if the network is synchronous, every honest input was received), and when $\left|\mathcal{M}\right| \geq n - t_s$ (since at most t_s parties are corrupted), the parties reliably broadcast their set of level-zero witnesses W_0. Even after broadcasting W_0, parties may continue gathering level-zero witnesses. Then, if a party P receives a set of level-zero witnesses W'_0 from P' such that all values sent by parties in W'_0 were also received by P ($W'_0 \subseteq W_0$), P marks P' as a *level-one witness* by adding it to its set W_1.

Once each honest party gathers $n - t_s$ level-one witnesses, we have the guarantee that every pair of honest parties has a level-one witness in common. This means that every pair of honest parties has received $n - t_s$ common values via Π_{rBC}. This is the point where Π_{oBC} allows the parties to output the set of value-sender pairs obtained so far and terminate, as (t_s, t_a)-secure oBC is achieved.

Additional step in Π_{GTHR}. In contrast, to achieve the stronger property t_s-Common Core, our GTHR protocol continues: following the insights from the asynchronous GTHR protocol of [2], we obtain that, when $n - t_s$ honest parties hold sets W_1 of size $n - t_s$, then $t_s + 1$ honest parties have a common level-one *honest* witness $P^\star$. This will then enable us to achieve the t_s-Common Core property. Concretely, when party P gathers $n - t_s$ level-one witnesses, it sends its set W_1 to all the parties. P may continue marking parties as level-one witnesses even after sending its set W_1. When receiving a set W'_1 from some party P' such that $W'_1 \subseteq W_1$, P marks P' as a *level-two witness* by adding it to its set W_2. Once P collects $n - t_s$ level-two witnesses, it may output its set $\mathcal{M}$. This ensures that P has marked at least one of the parties that have reported sets W_1 containing $P^\star$ – and therefore P has marked $P^\star$ as a level-one witness as well. Hence, P has received the set $W_0^\star$ sent by $P^\star$, and therefore all the values sent by the parties in $W_0^\star$ have been included in P's set $\mathcal{M}$. This argument applies to every honest party, which ensures that the t_s-Common Core property holds.

We include the formal code of Π_{GTHR} below. Due to space constraints, we defer the analysis to the full version of our paper.

Protocol Π_{GTHR}

Code for party P with input v

$\tau_{\mathtt{start}} := \tau_{\mathtt{now}}$; $\mathcal{M} := \varnothing$; $W_0, W_1, W_2 := \varnothing$.
Send v to every party via Π_{rBC}.
Whenever receiving a value v' from P' via Π_{rBC}, add (v', P') to $\mathcal{M}$ and P' to W_0.
If $\tau_{\mathtt{now}} \geq \tau_{\mathtt{start}} + c_{\mathsf{rBC}} \cdot \Delta$ and $\left|W_0\right| \geq n - t_s$:
    Send W_0 to all parties via Π_{rBC}.
Whenever receiving W_0' from P' via Π_{rBC} such that $\left|W_0'\right| \geq n - t_s$:
    When $W_0' \subseteq W_0$, add P' to W_1.
When $\tau_{\mathtt{now}} \geq \tau_{\mathtt{start}} + 2c_{\mathsf{rBC}} \cdot \Delta$ and $\left|W_1\right| \geq n - t_s$:
    Send W_1 to all parties.
Whenever receiving W_1' from P' such that $\left|W_1'\right| \geq n - t_s$:
    When $W_1' \subseteq W_1$, add P' to W_2.
When $\tau_{\mathtt{now}} \geq \tau_{\mathtt{start}} + (2c_{\mathsf{rBC}} + c'_{\mathsf{rBC}}) \cdot \Delta$ and $\left|W_2\right| \geq n - t_s$:
    Output $\mathcal{M}$.

C Analysis of Π_{ACS}

We include the proof of Theorem 14, restated below.

▶ **Theorem 14.** *If $n > 2 \cdot t_s + t_a$ and $t_a \leq t_s$, there is a (t_s, t_a)-secure ACS protocol Π_{ACS} (assuming PKI). The protocol has expected round complexity $O(1)$. If ℓ denotes the universe elements' size in bits and κ is the security parameter, its expected communication complexity is $O(n^3 \cdot \ell + n^4 \cdot \kappa)$ bits. If threshold signatures are available, the expected communication complexity reduces to bits.*

Proof. In the following, we show that Π_{ACS} achieves (t_s, t_a)-secure ACS when $n > 2 \cdot t_s + t_a$.

First, the Validity and Consistency properties follow immediately from the Validity and Consistency properties of Π_{rBC} and Π_{GTHR}. Next, Π_{GTHR} ensures that all honest parties obtain sets $\mathcal{M}_{\mathsf{GTHR}}$ that intersect in at least $n - t_s$ values. In addition, if the network is synchronous, these sets contain all honest values, and are obtained simultaneously.

Therefore, if the network is synchronous, all parties join the Π_{BA} invocations simultaneously, hence all properties that Π_{BA} ensures when running in a synchronous network hold. It follows that all Π_{BA} invocations corresponding to honest parties result in output 1, and hence all honest values are included in the output sets $\mathcal{M}$, ensuring the Honest Core property. Moreover, parties agree on the same bit for every corrupted party. If the output bit for some corrupted party is 1, then at least one honest party has included this corrupted party's value in its set $\mathcal{M}_{\mathsf{GTHR}}$. By Observation 20, eventually, all honest parties receive this value as well. Therefore, all honest parties output the same set $\mathcal{M}$, hence Exact Agreement and Probabilistic Termination hold.

If the network is asynchronous, since Π_{GTHR} achieves Termination, all honest parties eventually join all Π_{BA} invocations and hence agree on a bit for each party. Π_{GTHR}'s t_s-Common Core property ensures that there exist at least $n - t_s$ invocations of Π_{BA} in which all honest parties input 1, and therefore output 1 in these invocations. For each invocation returning 1, the Weak Validity property ensures that at least one honest party P has input 1, meaning that P has received the corresponding value via Π_{GTHR}. Observation 20 then ensures that all parties eventually receive this value, and therefore all honest parties output the same set $\mathcal{M}$ of at least $n - t_s$ value-sender pairs, hence Exact Agreement, t_s-Output Size and Probabilistic Termination hold. ◀