J. Cryptology (2005) 18: 191–217
DOI: 10.1007/s00145-005-0308-x

Journal of CRYPTOLOGY

Byzantine Agreement Given Partial Broadcast*

Jeffrey Considine
Computer Science Department,
Boston University,
Boston, MA 02215, U.S.A.
jconsidi@cs.bu.edu

Matthias Fitzi
Department of Computer Science,
University of Århus,
8000 Aarhus C, Denmark
fitzi@daimi.au.dk

Matthew Franklin
Department of Computer Science,
University of California,
Davis, CA 95616, U.S.A.
franklin@cs.ucdavis.edu

Leonid A. Levin
Computer Science Department,
Boston University,
Boston, MA 02215, U.S.A.
lnd@cs.bu.edu

Ueli Maurer
Department of Computer Science,
ETH Zurich,
CH-8092 Zurich, Switzerland
maurer@inf.ethz.ch

David Metcalf
Computer Science Department,
Boston University,
Boston, MA 02215, U.S.A.
metcalf@cs.bu.edu

Communicated by Ran Canetti

Received 4 March 2003
Online publication 20 May 2005

Abstract. This paper considers unconditionally secure protocols for reliable broadcast among a set of n players, where up to t of the players can be corrupted by a (Byzantine) adversary but the remaining $h = n - t$ players remain honest. In the standard model with a complete, synchronous network of bilateral authenticated communication channels among the players, broadcast is achievable if and only if $2n/h < 3$.

We show that, by extending this model by the existence of partial broadcast channels among subsets of b players, global broadcast can be achieved if and only if the number h of honest players satisfies $2n/h < b+1$. Achievability is demonstrated by protocols with communication and computation complexities polynomial in the size of the network, i.e., in the number of partial broadcast channels. A respective characterization for the related consensus problem is also given.

Key words. Broadcast, Byzantine agreement, unconditional security.

* Preliminary versions of the results presented in this article were reported in [25], [9], [19], [10], and [20]. Leonid A. Levin was supported by NSF Grants CCR 9820934, 0311411, Matthias Fitzi was partly supported by the Packard Foundation, Matthew Franklin was supported by the Packard Foundation and NSF, and Ueli Maurer was partly supported by the Swiss National Science Foundation.

1. Introduction

A fundamental problem in fault-tolerant distributed computing is to achieve consistency of the involved parties' views, even if some of the parties (also called players) deviate from the protocol in an arbitrary manner. A core primitive for achieving global consistency is broadcast, i.e., a mechanism or protocol allowing one player, the sender, to send a value consistently to all other players such that, even in case of malicious behavior by the sender and/or some of the other players, all honest players receive the same value.

The standard model considered in fault-tolerant distributed computing is that every pair of players can communicate over a bilateral authenticated channel. In this model, authenticated channels are simply assumed to exist. In practice, they can be implemented using cryptographic techniques. Such techniques assume an initial set-up phase such as the establishment of a public-key infrastructure, or sharing pairwise secret keys.

The problem of implementing broadcast in the standard model [32] is a classical problem in distributed computing. The seminal result of Lamport et al. [32] is that broadcast can be implemented if and only if less than a third of all the players misbehave.

1.1. *Motivation*

In this paper we propose to investigate a new research direction by assuming, as part of the model, more powerful primitives than authenticated channels, i.e., primitives that guarantee some degree of consistency among the players. The additional primitive we consider is probably the simplest one that can serve as an extension of the standard model, namely channels that guarantee consistency among b participants when one of them sends a value to the others.

Our motivation for considering such enhanced models is twofold. First, the generic reduction of complex tasks to simple ones is a useful tool for proving whether or not a task is achievable under given conditions, only requiring a construction for the simple task in order to prove the achievability of the complex one, and only requiring to show the impossibility of the complex task in order to prove the simple one to be impossible.

Second, for unconditional multi-party computation[1] among n players, the achievability of broadcast is a limiting factor. As $2n/h < 3$ is the lower bound for multi-party computation when broadcast is not available, broadcast allows for $n/h < 2$. When additionally assuming oblivious transfer, non-robust multi-party computation is still achievable in the presence of any number of corrupted players. As broadcast is typically the only assumed primitive that involves all n players (in contrast to other commonly assumed primitives such as pairwise channels or oblivious transfer), it is a natural question to ask whether global broadcast is necessary for multi-party computation beyond $2n/h < 3$ or, alternatively, what resilience can be achieved for multi-party computation when only assuming primitives of constant size.

[1] Refer to Section 6.4 for an informal definition of multi-party computation as well as a short overview of previous results.

1.2. *Models and Definitions*

Byzantine agreement refers to the general problem of having a set $P = \{p_1, \ldots, p_n\}$ of n players agree on a value v from some finite domain $\mathcal{D}$ where some of the players may be corrupted. There are two main variations of Byzantine agreement, *broadcast* and *consensus*. The goal of broadcast (or the Byzantine generals problem) is to have some designated player p_s, called the sender, consistently send an input value (or message) x_s to all other players. The goal of consensus, where every player p_i starts with an input value x_i of his own, is to make all honest (non-corrupted) players decide on a common output value such that, if all honest players hold the same input value v, this common output value is v.

1.2.1. *Communication*

The players in P are connected via a complete, synchronous network of pairwise authenticated channels. A *pairwise authenticated channel* between two players p_i and p_j is a bilateral communication channel that guarantees that only the two respective players can send messages on the channel, i.e., excluding any third party from accessing it in any other way than possibly reading the communication between the two players. In particular, we assume that communication via an authenticated channel cannot be blocked by a third party. *Synchronicity* means that all players share common, synchronized clock cycles. In such a clock cycle, each player first receives a finite (possibly empty) set of messages from the other players, followed by a finite number (possibly zero) of local computation steps, and finally sends a finite (possibly empty) set of messages to the other players. Messages being sent during a clock cycle are guaranteed to have arrived at the beginning of the next cycle.

We refer to the communication model described so far in this section as the *classical model*, denoted by $\mathcal{M}_2$. In contrast, we introduce the *partial-broadcast model*, $\mathcal{M}_b$, below.

Definition 1 ($\mathcal{M}_b$). Model $\mathcal{M}_b$ extends the classical model by perfectly reliable synchronous broadcast channels among each b-tuple of players, i.e., authenticated broadcast channels (denoted BC_b) from p_{i_1} to players $p_{i_2}, \ldots, p_{i_b}$, for any selection of b distinct players from P. We assume all bilateral and BC_b-channels to be composable in parallel (or at least sequentially).

1.2.2. *Composability*

It has long been a common technique to construct complex protocols by combining subprotocols that achieve simpler tasks. When giving a security proof of such a construction, the fact that the subprotocols compose correctly is usually not made explicit because it is typically trivial in the context of the protocol itself. On the other hand, composability can become non-trivial when the whole context of the execution of the (sub-)protocols is not known in advance [6], [33]. We note that, in our modular construction, our subprotocols trivially compose with each other, and so do the final protocols.

1.2.3. *Adversary and Corruption*

The resilience of a protocol is characterized by the number t of players that may deviate from the protocol. We refer to such a player as being *corrupted* whereas a non-corrupted player is called *honest*. Alternatively, $h = n - t$ denotes the minimal number of players that are assumed to be honest. It helps to imagine a central adversary who can corrupt up to t players and make them cheat in an arbitrary, coordinated manner. We consider an *adaptive adversary* who can gradually corrupt arbitrary new players during the protocol, but at most t in total. Note, however, that our impossibility results are proven even with respect to the strictly weaker definition of a *non-adaptive* (or *static*) adversary that is assumed to preselect up to t of the players at the beginning of the protocol and not corrupt any further players during any later stage of the protocol.

1.2.4. *Security*

We demand our protocols to be *unconditionally secure*, i.e., we require that even a computationally unbounded adversary cannot make the protocol fail except for some negligible error probability. Our final broadcast protocol will even be *perfectly secure* (zero error probability). On the other hand, our impossibility result is given even with respect to an adversary that is bounded to polynomial-time computation.

1.2.5. *Setup Assumptions*

We assume that all players know the player set, the protocol, and the whole network topology, i.e., they know which players participate in the protocol and how they are connected by communication channels. Additionally, we assume that all players agree on a common point in time when the protocol is to be started.

The achievable resilience of Byzantine agreement depends on whether or not one assumes that a *public-key infrastructure* (*PKI*) is consistently set up among the players. Such a PKI would allow all messages to be signed and enable broadcast with arbitrary resilience and consensus for $n/h < 2$. In this paper we consider the case where no such PKI is set up among the players.

1.2.6. *Complexities*

We characterize the efficiency of the protocols in terms of the *computational complexity*, i.e., the local computational worst-case complexity of the honest players, the *bit complexity*, i.e., the total number of bits communicated by all honest players during the protocol in the worst case, and the *round complexity*, i.e., the maximal number of communication rounds for any honest player in the worst case. Our round complexity analyses are given under the assumption that the underlying channels are composable in parallel without any side-effects on each other.

1.2.7. *Broadcast*, *Consensus*, *and Proxcast*

Definition 2 (Broadcast). A protocol for the player set P, where player $p_s \in P$ (the *sender*) holds an input value $x_s \in \mathcal{D}$ and every player $p_i \in P$ computes an output value $y_i \in \mathcal{D}$, achieves *broadcast* (or is a *broadcast protocol*) if it satisfies the following conditions:

Consistency (or agreement): All honest players decide on the same output value, i.e., $y_i = y_j$ for all honest players p_i and p_j.
Validity: If the sender p_s is honest, then every honest player p_i decides on the sender's input value, i.e., $y_i = x_s$.

Definition 3 (Consensus). A protocol for the player set P, where every player $p_i \in P$ holds an input value $x_i \in \mathcal{D}$ and computes an output value $y_i \in \mathcal{D}$, achieves *consensus* if it satisfies the following conditions:

Consistency (or agreement): All honest players decide on the same output value, i.e., $y_i = y_j$ for all honest players p_i and p_j.
Validity (or persistency): If every honest player p_i holds the same input value $x_i = x$, then every honest player decides on it, i.e., $y_i = x$.

Note that, in contrast to broadcast, the consensus definition only makes sense if less than half of the players are corrupted. In this case, broadcast can easily be achieved using consensus and vice versa. Thus, we focus on broadcast in what follows, and generalize our results to consensus only at the very end. Furthermore, we mainly focus on binary broadcast (domain $\mathcal{D} = \{0, 1\}$) since broadcast for any finite domain $\mathcal{D}$ can be efficiently solved by $\lceil \log_2 |\mathcal{D}| \rceil$ invocations of its binary variant. A more efficient way to achieve this was given in [40] by Turpin and Coan.

We now introduce the primitive proxcast which serves as a fundamental building block for our protocol constructions. Proxcast was first defined in [38]. $\mathcal{P}_n^k$ is a broadcast-like primitive that achieves the validity property of broadcast. Additionally, it is guaranteed that the players' outputs are proximate in the sense that they do not deviate too strongly from each other. $\mathcal{P}_n^k$ is best introduced pictorially and by means of a binary input domain. See Fig. 1.

The sender sends a bit $x \in \{0, 1\}$. Each player p_i receives an output $\ell \in \{0, \ldots, k-1\}$. If the sender is honest then each honest player gets output $x \cdot (k-1)$. If the sender is corrupted then it is still guaranteed that there is a value m such that all honest players get an output $\ell \in \{m, m+1\}$. Alternatively, the output can be represented as a pair (y, g) with output bit y and grade value $g = 0, \ldots, \lfloor (k-1)/2 \rfloor$. If the sender is honest then each honest player gets bit $y = x$ and maximal grade $g = \lfloor (k-1)/2 \rfloor$. If the sender is corrupted then the honest players still receive adjacent grades $g \in \{z, z+1\}$.

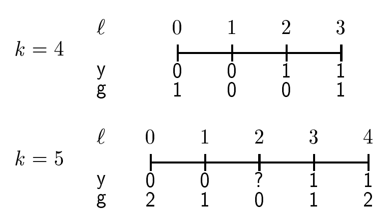

Fig. 1. $\mathcal{P}_n^4$ and $\mathcal{P}_n^5$ over binary input domain.

If any honest player gets a high enough grade g then it is guaranteed that all honest players hold the same output bit y—as can be verified, this is the case for grades $g > k \bmod 2$.[2] According to requirements we use the two different representations interchangeably.

Definition 4 (Proxcast). Let $k > 0$ be an integer. A protocol among player set P where player $p_s \in P$ (the *sender*) holds an input value $x_s \in \mathcal{D}$ and every player $p_i \in P$ finally decides on an output value $y_i \in \mathcal{D}$ and a grade $g_i \in \{0, \ldots, \lfloor (k-1)/2 \rfloor\}$ achieves *k-proxcast* ($\mathcal{P}_n^k$, for short) if it satisfies the following conditions:

Validity: If the sender is honest with input x_s then every honest player p_i computes $y_i = x_s$ and $g_i = \lfloor (k-1)/2 \rfloor$.

Consistency: There is a value $g \in \{1, \ldots, \lfloor (k-1)/2 \rfloor\}$ such that every honest player p_i decides on either $g_i = g - 1$ or $g_i = g$. If some honest player p_i computes $g_i > k \bmod 2$ then all honest players p_j compute the same value $y_j = y_i$.

Alternatively, if $\mathcal{D} = \{0, 1\}$, we say that a player with values $y_i \in \{0, 1\}$ and $g_i \in \{0, \ldots, \lfloor (k-1)/2 \rfloor\}$ *decides* on *level* $\ell_i = y_i \cdot (\lceil (k-1)/2 \rceil + g_i) + (1 - y_i) \cdot (\lfloor (k-1)/2 \rfloor - g_i)$.[3] The validity and consistency conditions then transform into

Validity′: If the sender is honest with input x_s then every honest player p_i computes $\ell_i = x_s \cdot (k-1)$.

Consistency′: There is a level $\ell \in \{0, \ldots, k-2\}$ such that every honest player p_i computes $\ell_i \in \{\ell, \ell + 1\}$.

Well known special cases of proxcast are multi-send ($k = 2$), crusader agreement [12] ($k = 3$), and graded broadcast [17] ($k = 5$). We denote an invocation of $\mathcal{P}_n^k$ with sender p_s and input x_s by $\mathcal{P}_n^k(P, p_s, x_s)$. Note the following trivial fact about proxcast.

Proposition 1. *$\mathcal{P}_n^k$ implies $\mathcal{P}_n^{k'}$ for any $k' < k$. $\mathcal{P}_n^k$ for any finite domain $\mathcal{D}$ can be efficiently achieved by binary $\mathcal{P}_n^k$.*

Proof. $\mathcal{P}_n^{k'}$ can be easily achieved by invoking $\mathcal{P}_n^k$ and merging $k - k' + 1$ adjacent output values together.

Let a protocol for binary $\mathcal{P}_n^k$ be given, i.e., $x \in \{0, 1\}$ and $g \in \{0, \ldots, \lfloor (b-1)/2 \rfloor\}$. Multi-valued $\mathcal{P}_n^k$ with a given domain $\mathcal{D}$, $x \in \mathcal{D}$, can be achieved by running an instance of binary $\mathcal{P}_n^k$ with respect to every single bit in the binary representation of x. The recipients then decide on the value y being composed of all the bits received during these invocations plus on the minimal grade ever received during the binary invocations. □

Since proxcast (broadcast) for any finite input domain efficiently reduces to binary proxcast (broadcast, respectively) we restrict ourself to the binary case in what follows.

[2] For odd k, $g = 1$ is not sufficient since the "middle level" $\ell = (k-1)/2$ cannot be uniquely associated with a particular output bit y.

[3] Which maps the possible pairs (y_i, g_i) to values $\ell_i \in \{0, \ldots, k-1\}$ according to Fig. 1.

1.2.8. *Protocol Notation*

Protocols are understood to be specified with respect to a player set $S \subseteq P = \{p_1, \ldots, p_n\}$. Each player $p_i \in S$ runs the same program, using as the input (if there is one) his own input, say x_i. The local variable names indicate the index i of the player p_i performing the instruction. For instance,

$$\text{Protocol } \mathsf{Broadcast}(S, p_1, x_1)$$

refers to a protocol for broadcast among the player set S where player p_1 holds input x_1 and the other players hold no input. Some of the instructions are indicated as being only for a specific player, e.g., the sender:

$$\texttt{if } i = 1 \texttt{ then } \mathsf{SendToAll}(v_1) \texttt{ fi; } \mathsf{Receive}(w_i)$$

means that player p_1 sends the value stored in (his local) variable v_1 to all players in S and that each player p_i (including p_1) assigns the received value to his local variable w_i. At the end of a protocol, each player outputs a value, usually stored in the local variable y_i, written `return` y_i.

The domain of the values is usually specified implicitly. For simplicity, it is not explicitly stated how to handle received values (from a corrupted player) outside the domain. Such a value can be assumed to be replaced by some default value, either an arbitrary value in the domain or a special extra symbol $\perp$.

1.3. *Previous Work*

The Byzantine agreement problem was introduced by Lamport et al. [32]. For the standard model $\mathcal{M}_2$ they presented a broadcast protocol among n players that is secure for $2n/h < 3$. As proven in [32], [31], and [18], this bound is tight, i.e., no protocol can tolerate $2n/h \geq 3$, not even if the adversary is computationally bounded. The first efficient (i.e., polynomial-time) broadcast protocol was given in [15] by Dolev and Strong, followed by a variety of alternative protocols with different interesting properties [14], [39], [1], [17], [5], [8], [28].

The extension of the standard communication model by partial broadcast was already considered in [27], [26], and [41] in the context of secure point-to-point communication over an incomplete network, a problem initially studied by Dolev et al. [13] for the standard communication model. In [27] Franklin and Yung show how to achieve private point-to-point communication in the presence of a passive adversary, given partial broadcast but not necessarily pairwise communication channels among the players. Secure point-to-point communication over partial-broadcast networks in the presence of an active adversary was considered by Franklin and Wright [26] and Wang and Desmedt [41].

1.4. *Result and Sources*

Theorem 1. *In Model $\mathcal{M}_b$, global broadcast among $n > b$ players is achievable if and only if $2n/h < b + 1$. If $b = O(1)$ or $n - b = O(1)$ then broadcast is achievable with message and computation complexities polynomial in n. In all other cases, our protocols are still polynomial in the size $\binom{n}{b}$ of the network.*

The special case of $b = 3$ was introduced and fully treated in [25]. In [9] $2n/h < b + 1$ was shown to be a lower bound for the case of general b. There, a protocol matching this bound for integers n/h was given. That protocol additionally assures agreement whenever the sender is honest regardless of the number of corrupted recipients; requiring this extra property, the protocol is optimal even for fractional n/h. Protocols matching the lower bound $2n/h < b+1$ for fractional n/h were independently developed in [19] and [10]. Protocols that are polynomial in the size of the network were given in [20].

1.5. *Outline*

We first give our proof of the lower bound in Section 2. The proof is obtained by using ideas of Fischer et al. who in [18] introduced a standard technique in order to prove the impossibility of Byzantine agreement in standard scenarios. We also use a simulation argument from [32] for this purpose.

We then describe two different protocols with respect to the optimal bound $2n/h < b + 1$. Since both protocols are built on b-proxcast, $\mathcal{P}_n^b$ (as given in Definition 4), we first show how to implement that primitive efficiently in Section 3.

In Section 4 we present our first protocol which extends the recursive construction in [32] known under the name "information gathering (IG)" [1]. This protocol is less complicated than the second one but generally superpolynomial in the size $\binom{n}{b}$ of the network. IG among n players is implicitly based on two-threshold broadcast among less than n players, a generalization of broadcast that achieves validity and consistency with respect to different thresholds [24].

In Section 5 we present our second construction. The resulting protocol's complexities are polynomial in the size $\binom{n}{b}$ of the network. The protocol is obtained along the lines of the protocols in [16] and [34] where a PKI is assumed to be set up among the players with respect to a (pseudo-)signature scheme. We demonstrate that k-proxcast (with sufficiently large k) is powerful enough to replace a PKI with respective signatures in the protocols of [16] and [34], thus yielding a protocol for our model without the need for a PKI or signatures. We also show how to transform $\mathcal{P}_n^b$ into $\mathcal{P}_n^k$ efficiently for the required k.

Final remarks and the extension of the results to consensus are given in Section 6.

2. Lower Bound

We prove that, in Model $\mathcal{M}_b$, secure global broadcast among $n > b$ players is impossible if $2n/h \geq b + 1$. We first prove the inexistence of a protocol for $n = b + 1$ and $h = 2$ by generalizing the proof idea in [18] for the impossibility of broadcast among n players in the standard model with respect to $2n/h < 3$. Actually, this yields a stronger result, namely that such a protocol cannot exist even for a weaker adversary whose choice of which players he must leave uncorrupted is restricted to two consecutive players. The final impossibility result for general n will then be derived from this special case along the lines of a similar generalization in [32].

2.1. *Impossibility for $n = b + 1$ and $h = 2$*

Our aim is to show that, for each possible protocol among $b + 1$ players, there is an admissible adversary that can make the protocol fail with some non-negligible probability by corrupting at most $b - 1$ of the players. For this, we assume any potential broadcast protocol Ψ to be given and consider it in two different contexts, distributed systems Σ and Σ' (see Fig. 2 for the special case $b = 3$).

System Σ is the original setting among the $b+1$ players where the adversary corrupts $b - 1$ of them. By assumption, the protocol Ψ achieves broadcast in this system.

In system Σ' no adversary is present, i.e., all players follow the protocol correctly. However, the players are arranged in a different way. In particular, Σ' is a distributed system built of $2b + 2$ players—the $b + 1$ original ones together with one identical copy of each of them. Still, protocol Ψ can be run in this extended system—meaning that all $2b + 2$ players run their respective local codes and communicate with the players they are connected to.

We show that, for certain pairs of players, their joint views in protocol Ψ are indistinguishable with respect to the different systems Σ and Σ'. That is, such a pair of players cannot tell whether they are involved in system Σ or Σ'. This implies that system Σ' (i.e., the rearrangement of the players) simulates an admissible adversary in the original system Σ with respect to several pairs of players simultaneously.

Since we assume the protocol to be secure in the presence of $h = 2$ honest players, the validity and consistency conditions of broadcast must thus be satisfied for each one of these pairs even in system Σ'. However, we will be able to conclude that it is impossible to achieve these conditions simultaneously with respect to all involved pairs—hence showing that the assumed protocol cannot be secure in the original system Σ.

Technical details. Let $P = \{p_0, \dots, p_b\}$ be the $n = b + 1$ players with sender p_0 and let Ψ be a *protocol* among the players in P. Protocol Ψ specifies a *local program* ψ_i for each player p_i. Let the integer $i \in \{0, \dots, b\}$ be called the *type of player* p_i, uniquely defining the program ψ_i it is supposed to run. Our communication model suggests that each player p_i has *ports* with respect to each communication channel it shares with other players. Let p_i's *bilateral port of type* j denote the port it uses for its bilateral communication with player p_j. When necessary, we distinguish p_i's *bilateral read port of type* j (where it reads the messages received from player p_j) from its *bilateral write port of type* j (where it writes the messages to be sent to player p_j). Finally, let p_i's BC_b *port of type* j denote the port it uses for its communication via the BC_b channel it shares with the players in $P \backslash \{p_j\}$.

Reconnection of Players. The left part of Fig. 2 sketches how the players are connected with each other in the original setting for the special case of $b = 3$ (where the bilateral channels are represented by arrows and the BC_b channels are represented by shaded triangles). We refer to this distributed system as the *original system* Σ.

We now describe the *simulation system* Σ' which is sketched in the right part of the figure for the special case $b = 3$. For each player $p_i \in P$, let p_{i+n} be an identical copy of p_i. System Σ' consists of the $2n = 2(b+1)$ players $P' = \{p_0, \dots, p_{2b+1}\}$, all connected

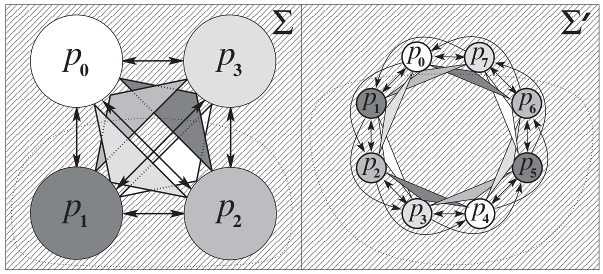

Fig. 2. Original system Σ and simulation system Σ' for the special case $b = 3$.

together as described further below. Let $\text{Type}(i) = i \bmod n$ denote the type of player $p_i \in P'$. There are hence two players of each of the $b + 1$ types and, in particular, two senders p_0 and p_n who take binary input x_0 and x_n, respectively.

In order to define the system Σ' exactly we need to specify, for each player p_i ($i \in \{0, \dots, 2n - 1\}$), with which other players its communication channels are connected. The $2n$ players $p_0, \dots, p_{2n-1}$ are arranged in a circular way, with the channels of each player arranged in a cyclically identical manner. It thus suffices to describe the channels of player p_0.

Each bilateral write port of p_0 of type $k = 1, \dots, b - 1$ is connected to the bilateral read port of type 0 of the specific player p_k as originally. Player p_0's bilateral write port of type b is connected to the bilateral read port of type 0 of the specific player p_{2n-1}. Each BC_b port of p_0 of type $k = 1, \dots, b$ is connected to the BC_b port of type k of the specific players $p_1, \dots, p_{k-1}$ and $p_{k+1+n}, \dots, p_{2n-1}$.

This way of connecting the players $p_i \in P'$ satisfies the following properties:

1. *Exclusive assignment of ports*. Each player p_i's bilateral write (read) port of type j is exclusively connected to the bilateral read (write) port of type $\text{Type}(i)$ of one player of type j. Furthermore, each player p_i's BC_b port of type j is exclusively connected to the BC_b ports of type j of $b - 1$ players of distinct types $k \notin \{\text{Type}(i), j\}$.

   Exclusive assignment of the bilateral ports immediately follows by cyclical symmetry of the construction. Furthermore, the connection rule for the BC_b channels guarantees that a player p_i's BC_b port of type j is assigned to a player p_k's port of type j if and only if player p_k's BC_b port of type j is assigned to player p_i's BC_b port of type j.[4]
2. *Mutual assignment of ports*. For each player pair $\{p_i, p_{(i+1) \bmod 2n}\}$ it holds that p_i's bilateral read (write) port of type Type(i+1) is connected to the write (read) port of the particular adjacent player $p_{(i+1) \bmod 2n}$. Furthermore, their BC_b ports of types $j \notin \{\text{Type}(i), \text{Type}(i + 1)\}$ are all mutually connected.

Exclusive and mutual assignment of ports (in Σ') now guarantees that any message sent (received) by player p_i via its bilateral port of type $\text{Type}(i + 1)$ is received (sent) by

[4] Note that the rule simply mutually groups together either all players $p_\ell \in P'$ such that $p_j < p_\ell < p_{j+n}$ or all players $p_\ell \in P'$ such that $p_\ell < p_j$ or $p_{j+n} < p_\ell$.

p_i's own adjacent player $p_{(i+1) \bmod 2n}$. The same holds for the mutual BC_b ports. Mutual assignment of ports additionally guarantees that any message sent on a BC_b channel of type $j \notin \{\text{Type}(i), \text{Type}(i+1)\}$ is either received by both adjacent players p_i and $p_{(i+1) \bmod 2n}$ or by none of them.

Identical joint views and contradiction. We now demonstrate that, for any pair $\{p_i, p_{(i+1) \bmod 2n}\}$ of adjacent players in system Σ', there is an admissible adversary for the original system Σ that achieves that the joint view of the players $p_{i \bmod n}$ and $p_{(i+1) \bmod n}$ is identical to the joint view of the players p_i and $p_{(i+1) \bmod 2n}$.

For this, the adversary corrupts the $b-1$ players in $P \setminus \{p_{i \bmod n}, p_{(i+1) \bmod n}\}$, simulates the virtual players in $P' \setminus \{p_i, p_{(i+1) \bmod 2n}\}$ of system Σ', and makes player $p_{(i-1) \bmod n}$ interact with the honest players like player $p_{(i-1) \bmod 2n}$ in Σ' and player $p_{(i+2) \bmod n}$ interact with the honest players like player $p_{(i+2) \bmod 2n}$ in Σ'.[5]

Since any two adjacent players p_i and $p_{(i+1) \bmod 2n}$ are consistently interconnected in Σ' (see the previous paragraph), this adversary strategy now guarantees that that the joint view of the players p_i and $p_{(i+1) \bmod 2n}$ is identical to the joint view of the players $p_{i \bmod n}$ and $p_{(i+1) \bmod n}$ in the original system Σ.

Lemma 2. *In model $\mathcal{M}_b$, broadcast among the $n = b+1$ players $P = \{p_0, \ldots, p_b\}$ is not achievable if, for any one pair $\{p_i, p_{(i+1) \bmod n}\} \subset P$, the adversary can corrupt the $b-1$ remaining players in $P \setminus \{p_i, p_{(i+1) \bmod n}\}$. In particular, the adversary can make the protocol fail with probability at least $1/n = 1/(b+1)$.*

Proof. We assume that, without loss of generality, the sender's program ψ_0 outputs its own input value. Now, consider the system Σ' being started with input $x_0 = 0$ for p_0 and input $x_n = 1$ for p_n. Let q_i, for $i = 0, \ldots, b$, be the probability (in system Σ') that players p_i and p_{i+1} output different values, i.e., $y_i \neq y_{i+1}$. Since $y_0 = 0$ and $y_n = 1$, we have

$$\sum_{i=0}^{b} q_i \geq 1. \tag{1}$$

Since for any pair of adjacent players in system Σ', their view is identical to their respective players' view in the original system Σ, the consistency condition of broadcast demands that $y_i = y_{i+1}$ holds for every $i = 1, \ldots, b$ also in system Σ'—in contradiction to (1). In particular, in the original system Σ, the following adversary strategy makes the protocol fail with a probability of at least $1/n$.

The adversary selects one of the n pairs $\{p_i, p_{(i+1) \bmod n}\} \subset P$ $(i = 0, \ldots, b)$ uniformly at random and corrupts the remaining players $(P \setminus \{p_i, p_{(i+1) \bmod n}\})$ by simulating the players in $\{p_0, \ldots, p_{i-1}, p_{i+2}, \ldots, p_{2n-1}\}$ of system Σ' towards the players p_i and $p_{(i+1) \bmod n}$. Thus, the probability that the honest players p_i and $p_{(i+1) \bmod n}$ disagree on

[5] This situation is depicted in Fig. 2 with respect to the player pair p_0 and p_3. On the left side, the corrupted players are encircled. On the right side, the players are encircled who are simulated by the adversary. In Σ, p_1 plays the role of player p_1 in Σ' and p_2 plays the role of player p_6 in Σ'.

their outputs is

$$\mathcal{P} \geq \frac{1}{n} \sum_{i=0}^{b} q_i \geq \frac{1}{n} = \frac{1}{b+1} \ , \tag{2}$$

and the lemma follows. □

2.2. *Impossibility for General* $2n/h \geq b+1$

We now give the impossibility proof for general n. We show that any protocol for general $n > b$ and $2n/h \geq b+1$ could be used in order to achieve broadcast among $b+1$ players where the adversary can corrupt at least $b-1$ consecutive players—which is impossible by Lemma 2.

Lemma 3. *Let* $|P| = n$ *and* $2n/h \geq b+1$. *It is possible to partition* P *into* $b+1$ *sets* $P_0 \dot\cup \cdots \dot\cup P_b = P$ *such that* $|\ P_i \cup P_{(i+1) \bmod (b+1)}\ | \geq h$ *holds for each* $i = 0, \ldots, b$.

Proof. Let $k = n \bmod (b+1)$ and $n = \lambda(b+1)+k$. The set P is partitioned into $b+1$ sets P_i of $\lceil n/(b+1) \rceil$ or $\lfloor n/(b+1) \rfloor$ elements in any possible way except for the following constraint: if $k \geq (b+1)/2$ then it is additionally assured that $|P_i| = \lfloor n/(b+1) \rfloor$ implies $|P_{(i+1) \bmod (b+1)}| = \lceil n/(b+1) \rceil$. The lemma follows by distinction of the following two cases.

$$k < \frac{b+1}{2} \quad \Rightarrow \quad \left|P_i \cup P_{(i+1) \bmod (b+1)}\right| \geq 2 \left\lfloor \frac{n}{b+1} \right\rfloor = \left\lfloor \frac{2n}{b+1} \right\rfloor \geq h \ , \quad \text{and}$$

$$k \geq \frac{b+1}{2} \quad \Rightarrow \quad \left|P_i \cup P_{(i+1) \bmod (b+1)}\right| \geq \left\lfloor \frac{n}{b+1} \right\rfloor + \left\lceil \frac{n}{b+1} \right\rceil \geq \left\lfloor \frac{2n}{b+1} \right\rfloor \geq h. \quad \square$$

Theorem 2. *In model* $\mathcal{M}_b$, *broadcast among* $n > b$ *players is not achievable if* $2n/h \geq b+1$. *In particular, the adversary can make the protocol fail with probability at least* $1/(b+1)$.

Proof. Assume any broadcast protocol Ψ for $n > b$ players $Q = \{q_0, \ldots, q_{n-1}\}$ with sender q_0, secure for $2n/h \geq b+1$. With the help of protocol Ψ, the $b+1$ players $P = \{p_0, \ldots, p_b\}$ can achieve broadcast secure for any honest pair $\{p_i, p_{(i+1) \bmod (b+1)}\}$ as follows. The set Q is partitioned into $b+1$ sets $Q_0, \ldots, Q_b$ such that $q_0 \in Q_0$, and $|Q_i \cup Q_{(i+1) \bmod (b+1)}| \geq h$ for all $i = 0, \ldots, b$ which is possible by Lemma 3. The players in P can now achieve broadcast by having each player p_i simulate all players $q_j \in Q_i$ in an instance of protocol Ψ. There, the players in $Q_i \cup Q_{(i+1) \bmod (b+1)}$ for some $i = 0, \ldots, b$ are honest since at least one pair $\{p_i, p_{(i+1) \bmod (b+1)}\}$ of the simulating players is. Since $|Q_i \cup Q_{(i+1) \bmod (b+1)}| \geq h$ by construction, protocol Ψ achieves broadcast among the simulating players in P, as secure as with respect to the player set Q. Thus, by Lemma 2, protocol Ψ must have an error probability of at least $1/(b+1)$. □

Note that this impossibility result holds with respect to the stronger model where the players are connected by *secure* bilateral channels and where the adversary is static and limited to probabilistic polynomial computation.

3. Efficient *b*-Proxcast

Let $\Gamma := \lfloor (b-1)/2 \rfloor$ be the maximal possible grade in $\mathcal{P}_n^b$. $\mathcal{P}_n^b$ is achieved by having the sender p_s distribute his input value x_s by all $\binom{n-1}{b-1}$ different BC_b-channels including the sender (as a sender of the primitive). Depending on the consistency among the $\binom{n-2}{b-2}$ different BC_b-channels a recipient p_i is involved in, p_i decides on a value y_i and a grade g_i. Qualitatively speaking, player p_i decides on a higher grade g_i as more BC_b invocations involving p_i result in the same value y_i.

For example, assume $b = 6$, and let $y_i^{sijk\ell m}$ be the output value of the BC_6 instance among the players p_s, p_i, p_j, p_k, p_ℓ, and p_m, where p_s acts as the sender. If the sender p_s is honest then an honest player p_i receives the same value x_s in all instances of partial broadcast, i.e., "$y_i^{si****} \equiv x_s$." However, if such a player p_i sees "$y_i^{si****} \equiv x_s$" then the sender could still be corrupted, and another honest player p_j could have received the value $1 - x_s$ in an invocation where p_i does not participate, e.g., "$y_j^{sjcdef} = 1 - x_s$." However, honest player p_i seeing "$y_i^{si****} \equiv x_s$" implies that, for every honest player p_j, it holds that "$y_j^{sji***} \equiv x_s$." Furthermore, if p_j sees "$y_j^{sji***} \equiv x_s$" (but no honest player p_i sees "$y_i^{si****} \equiv x_s$") then it holds that every honest player p_k sees "$y_j^{skji**} \equiv x_s$; and so on. As a natural approach, the grades of the final proxcast directly relate to the maximal "number of asterisks" a player can infer. More precisely, in order to compute his grade g_i, a player p_i computes a minimal set of players $Z_i \subseteq (P \backslash \{p_s, p_i\})$ such that all invocations of BC_b involving the players in $\{p_s, p_i\} \cup Z_i$ resulted in output 0. For example, if there are players p_j and p_k such that "$y_i^{sijk**} \equiv 0$" but no p_c exists such that "$y_i^{sic***} \equiv 0$" then $Z_i = \{j, k\}$.

In step 4 of the protocol, let "min" denote any minimal set that satisfies the given condition and let ":⊆" denote the assignment of any set satisfying the respective condition.

Protocol 1. $\mathcal{P}_n^b(S, p_s, x_s)$

1. $\forall P_{b-2} \subseteq P \backslash \{p_s, p_i\},\ |P_{b-2}| = b-2$:
   $y_i^{P_{b-2}} := \mathsf{BC_b}(P_{b-2} \cup \{p_s, p_i\}, p_s, x_s)$ `fi`;
2. `if` $i = s$ `then` $y_i := x_s$; $g_i := \Gamma$; $\ell_i := y_i \cdot (b-1)$; return (y_i, g_i, ℓ_i) `fi`;
3. `if` $b = n$ `then` $y_i := y_i^{P \backslash \{p_s, p_i\}}$; $g_i := \Gamma$; $\ell_i := y_i \cdot (b-1)$; return (y_i, g_i, ℓ_i) `fi`;
4. `if` $\exists P_{b-2} : y_i^{P_{b-2}} = 0$ `then` $Z_i := \min(Z \subseteq P \backslash \{p_s, p_i\} | \forall P_{b-2} \supseteq Z :\ y_i^{P_{b-2}} = 0)$
   `else` $Z_i :\subseteq P \backslash \{p_s, p_i\}$ such that $|Z_i| = b-1$ `fi`; [0 never received]
5. `if` $|Z_i| < b/2$ `then` $y_i := 0$ `else` $y_i := 1$ `fi`;
   $g_i := \lfloor |(b-1)/2 - |Z_i|| \rfloor$; $\ell_i := |Z_i|$;
6. return (y_i, g_i, ℓ_i)

Lemma 4. *In model $\mathcal{M}_b$, Protocol* 1 *achieves $\mathcal{P}_n^b$.*

Proof. If $b = n$ then the lemma trivially holds. Thus we assume that $b < n$.

(*Validity′*) If the sender p_s is honest then every honest player p_i computes Z_i such that $\ell_i = |Z_i| = x_s \cdot (b-1)$.

(*Consistency*$'$) Consider an honest player p_i with a minimal set Z_i, i.e., such that for all players p_j it holds that $\ell_j = |Z_j| \geq |Z_i| = \ell_i$. If $|Z_i| \geq b-2$ then $|Z_j| \leq |Z_i|+1$ trivially follows. If $|Z_i| < b-2$ then $Z = Z_i \cup \{p_i\}$ satisfies that, for all $P_{b-2} \supseteq Z$, $y_j^{P_{b-2}} = 0$, and thus, that $|Z_j| \leq |Z| \leq |Z_i|+1$. Thus, consistency follows. □

Note that a minimal set Z_i can be efficiently (polynomial in the size of the communication network) computed in the case where $b \leq n/2$. However, in the general case, finding a minimal set Z_i calculates the witness for an $\mathcal{NP}$-complete problem and thus seems infeasible. Thus, in order to guarantee a computation complexity polynomial in the size $\binom{n}{b}$ of the communication network (and thus polynomial in n for $b = O(1)$ and $n - b = O(1)$), we have the players "approximate" such a minimal set by public discussion in the following way.

A player p_i with $Z_i = \emptyset$ (i.e., p_i received value 0 in every single BC_b invocation) can efficiently detect this fact. Thus, in a first round, we have every such player p_i distribute his set $Z_i = \emptyset$ to every other player. A player p_j (who has not computed Z_j yet) now accepts this statement if and only if $y_j^{sji*} \equiv 0$ by calculating $Z_j := \{p_i\}$ and distributing Z_j in a next round. A player p_k (who has not computed Z_k yet) now accepts p_j's statement if and only if $y_k^{skji*} \equiv 0$ by calculating $Z_k := Z_j \cup \{p_j\}$, and distributing Z_k in a next round; etc. This process is continued for $b-2$ rounds in total.

Although this process does not guarantee that the honest players p_i compute a minimal set Z_i it still guarantees that they compute an extremal set ($|Z_i| = 0$ if $x_s = 0$, and $|Z_i| = b-1$ if $x_s = 1$) when the sender is honest, and, that there is a player p_j such that each honest player p_k's set satisfies $|Z_k| \in \{|Z_j|, |Z_j|+1\}$.

The following protocol is to replace step 4 in Protocol 1. Note that step 5 below is necessary in order to guarantee that, in round z, p_i indeed composes a set Z_i of exact cardinality $z+1$ (in the textual description above this is not necessarily the case since the set obtained might contain p_i himself).

Protocol 2. $\mathsf{Approximate}_Z$

1. `if` $\nexists P_{b-2} : y_i^{P_{b-2}} = 1$ `then` $Z_i := \emptyset$ `else` $Z_i := \bot$ `fi`;
2. `for` $z = 0$ `to` $b-3$ `do`
3. `if` $Z_i \neq \bot \wedge |Z_i| = z$ `then` SendToAll(Z_i) `fi`; Receive$(Z_i^1, \ldots, Z_i^n)$;
4. `if` $Z_i = \bot \;\wedge\; (\exists Z_i^k, |Z_i^k| = z \wedge \forall P_{b-2} \supseteq Z_i^k \cup \{p_k\} : y_i^{P_{b-2}} = 0)$ `then`
5. $Z_i := Z_i^k \cup \{p_k\}$;
   `if` $p_i \in Z_i$ `then` pick arbitrary $p_\ell \notin Z_i \cup \{p_s, p_i\}$ and let $Z_i := (Z_i \backslash \{p_i\}) \cup p_\ell$
   `fi`;
6. `od`;
7. `if` $Z_i = \bot$ `then` $Z_i :\subseteq P \backslash \{p_s, p_i\}$ such that $|Z_i| = b-1$ `fi`;

Theorem 3. *In model* $\mathcal{M}_b$, *Protocol* 1 (*using Protocol* 2 *instead of step* 4) *achieves* $\mathcal{P}_n^b$. *The computation and communication complexities of the protocol are polynomial in the size* $\binom{n}{b}$ *of the network. In particular, the protocol is polynomial in the number of players if* $b = O(1)$ *or* $n - b = O(1)$.

Proof. If $b = n$ then the lemma trivially holds. Thus we assume that $b < n$.

(*Validity*$'$) Assume the sender p_s to be honest. If $x_s = 0$ then every honest player p_i immediately computes $Z_i := \emptyset$ in step 1 of Protocol 2, and thus $\ell_i = 0$. If $x_s = 1$ then there is no set P_{b-2} such that player p_i received $y_i^{P_{b-2}} = 0$ and p_i computes $\ell_i = |Z_i| = b - 1$.

(*Consistency*$'$) Consider an honest player p_i with a minimal set Z_i, i.e., such that for all players p_j it holds that $\ell_j = |Z_j| \geq |Z_i| = \ell_i$. If $|Z_i| \geq b - 2$ then $|Z_j| \leq |Z_i| + 1$ trivially follows. If $|Z_i| < b - 2$ then p_j either already computed Z_j with $|Z_j| = |Z_i|$ or accepts such a set Z_i by computing Z_j according to step 5 of Protocol 2 of exact cardinality $|Z_j| = |Z_i| + 1$, and $\ell_j = \ell_i + 1$.

(*Complexities*) Protocol 1 involves one communication round in step 1 and $b - 2$ communication rounds in step 4 and thus $R = b - 1$ rounds in total. The overall number of BC_b calls is $\binom{n-1}{b-1}$ and, additionally, in Protocol 2, each player sends at most one n-bit message to every other player. Thus, the bit complexity of Protocol 1 is $B = O(n^3 + \binom{n}{b})$. The computational complexity is dominated by the test in step 4 of Protocol 2 which is evidently polynomial in $\binom{n}{b}$. □

4. The Information-Gathering Protocol

We now present our information-gathering (IG) protocol for global broadcast in model $\mathcal{M}_b$ secure if $2n/h < b + 1$. Its complexities are generally superpolynomial in the size $\binom{n}{b}$ of the network. IG among n players is implicitly based on subprotocols for two-threshold broadcast [24].

Definition 5 (Two-Threshold Broadcast). A protocol among P where player $p_s \in P$ (called the *sender*) holds an input value $x_s \in \mathcal{D}$ and every player $p_i \in P$ finally decides on an output value $y_i \in \mathcal{D}$, and achieves *two-threshold broadcast* (*TTBC*, for short) with respect to thresholds t_v and t_c if it satisfies the following conditions:

- Validity: If the sender p_s and at most t_v players overall are corrupted then all honest players p_i decide on the sender's input value, $y_i = x_s$.
- Consistency: If at most t_c players are corrupted then all honest players decide on the same output value.

TTBC among a player set $S \subseteq P$ ($n = |S|$) with sender p_s and thresholds t_v and t_c ($t_v \geq t_c$) recursively works as follows. First, the sender p_s distributes his input value x_s to all players in S via an instance of $\mathcal{P}_n^b$. Then each player $p_i \in S \backslash \{p_s\}$ recursively redistributes the received value with an instance of TTBC among the $n' = n - 1$ remaining players ($S' := S \backslash \{p_s\}$) with respect to threshold $t_c' = t_c - 1$. Now, every player holds the same $n' = n - 1$ votes (one per remaining player) on what level the respective player received in the invocation of $\mathcal{P}_n^b$. The only difference between two players' views can now be that their initial levels received during $\mathcal{P}_n^b$ differ by one (consistency of $\mathcal{P}_n^b$). The decision rule finally manages to reunite respective adjacent views while still guaranteeing validity with respect to an honest sender. Note that the recursion works on reduced $n' = n - 1$ and $t_c' = t_c - 1$ but leaves t_v unchanged.

In the following protocol, let $h_v := n - t_v$ and $h_c := n - t_c$, and for any predicate Q, let $\bigwedge_{k=1}^{0} Q :=$ true. Note that the protocol is binary. Thus the recursion in step 3 does not only branch in order of n ($n-1$ subcalls) but also in order $\log b$ since $\ell_j \in \{0, \ldots, b-1\}$ must be processed bitwise.

Protocol 3. $\mathsf{TTBC}(S, p_s, x_s, t_v, t_c)$

1. `if` $n = b$ `then` $y_i := \mathsf{BC_b}(S, p_s, x_s)$ `else` $(y_i, g_i, \ell_i) := \mathcal{P}_n^b(P, p_s, x_s)$ `fi`;
2. `if` $i = s$ `then` $y_i := x_s$; return y_i `fi`
   `if` $t_c = 0 \vee b = n$ `then` return y_i `fi`;
3. $\forall p_j \in S \backslash \{p_s\}$: $\ell_i^j := \mathsf{TTBC}(S \backslash \{p_s\}, p_j, \ell_j, t_v, t_c - 1)$ `fi`;
4. $\forall \ell \in [0, b-1] : L_i[\ell] := |\{p_j \in S \backslash \{p_s\} | \ell_i^j = \ell\}|$;
5. `if` $\bigwedge_{k=1}^{\ell_i} (L_i[k-1] + L_i[k] \geq h_c) \;\wedge\; (L_i[0] \geq h_v - 1)$ `then`
6. $\quad y_i := 0$ `else` $y_i := 1$
7. `fi`; return y_i

Lemma 5. *Consider Protocol 3 in model $\mathcal{M}_b$. If $2t_v + (b-1)t_c < (b-1)n$ and $t_c \leq t_v$ then the protocol achieves TTBC with respect to thresholds t_v and t_c.*

Proof. The proof proceeds by backward induction over n. Thus, assume that Protocol 3 achieves TTBC among $n' = n - 1$ players whenever $2t_v' + (b-1)t_c' < (b-1)n'$, and hence achieves TTBC for the special case that $n' = n - 1$, $t_v' = t_v$, and $t_c' = t_c - 1$.

(*Validity*) Assume that the sender p_s is honest and that at most t_v players are corrupted. If $t_c = 0$ or $b = n$ then validity is trivially satisfied (step 2)—this case constitutes the induction base. Thus, assume that $t_c > 0$ and $b < n$, and, by induction, that the protocol achieves validity with respect to $n' = n - 1$, $t_v' = t_v$, and $t_c' = t_c - 1$.

Since honest p_s consistently distributes the same value x_s, every honest player p_j computes $\ell_j = x_s \cdot (b-1)$. By induction assumption, every honest player consistently receives this value ℓ_j by the at least $h_v - 1$ remaining honest players in $S \backslash \{p_s\}$ in step 3.

If $x_s = 0$ then every honest player p_i computes $\ell_i = 0$ and $L_i[0] \geq h_v - 1$, and thus $y_i = 0 = x_s$. If $x_s = 1$ then $\ell_i = b - 1$ and $L_i[b-1] \geq h_v - 1$. Thus, p_i computing $y_i = 0$ would imply that, additionally, $L_i[0] \geq h_v - 1$ and $L_i[k] + L_i[k+1] \geq h_c$ for $k = 0, \ldots, b-2$, and thus that at least $(2(h_v - 1) + (b-1)h_c)/2 > n - 1 = n'$ players participated in step 3. Thus p_i must compute $y_i = 1 = x_s$.

(*Consistency*) Assume that at most t_c players are corrupted. If $t_c = 0$ or $b = n$ then consistency is trivially satisfied according to step 2. If the sender p_s is honest then consistency follows from validity (proven above) since $t_v \geq t_c$.

Thus, assume that $t_c > 0$, $n > b$, the sender p_s is corrupted, and that, by induction, the protocol achieves TTBC with respect to $n' = n - 1$, $t_v' = t_v$, and $t_c' = t_c - 1$.

Since the sender is corrupted, only $t_c' = t_c - 1$ corrupted players remain in $S \backslash \{p_s\}$, and are involved in step 3. Hence, by induction, every invocation of the protocol in step 3 achieves consistency. Furthermore, since $t_v' \geq t_c'$, also validity is achieved, i.e., all invocations of the protocol in step 3 achieve broadcast. This implies that two honest players p_i and p_j compute exactly the same sets $L_i[0] = L_j[0] =: L[0], \ldots, L_i[b-1] = L_j[b-1] =: L[b-1]$.

Let p_i be an honest player with minimal ℓ-value, i.e., such that for all other honest players p_j: $\ell_i \leq \ell_j$. By the consistency property of $\mathcal{P}_n^b$, it holds that $\ell_j \in \{\ell_i, \ell_i + 1\}$.

We now show that all honest players p_j compute $y_j = y_i$. If $\ell_j = \ell_i$ then both players have exactly the same view and hence decide in the same way, $y_j = y_i$. Thus, assume that $\ell_j = \ell_i + 1$.

- If p_i computes $y_i = 0$ then $\bigwedge_{k=1}^{\ell_i}(L[k-1] + L[k] \geq h_c) \ \wedge \ (L[0] \geq h_v - 1)$, and by the consistency property of $\mathcal{P}_n^b$ it also holds that $L[\ell_i] + L[\ell_i + 1] \geq h_c$. Hence, $\bigwedge_{k=1}^{\ell_i+1}(L_{k-1} + L_k \geq h_c) \ \wedge \ (L_i[0] \geq h_v - 1)$ and p_j computes $y_j = 0 = y_i$.
- If p_i computes $y_i = 1$ then $\neg(\bigwedge_{k=1}^{\ell_i}(L[k-1] + L[k] \geq h_c) \ \wedge \ (L[0] \geq h_v - 1))$, and thus $\neg(\bigwedge_{k=1}^{\ell_i+1}(L[k-1] + L[k] \geq h_c) \ \wedge \ (L[0] \geq h_v - 1))$, and p_j computes $y_j = 1 = y_i$. □

Protocol 4. $\mathsf{Broadcast}(P, p_s, x_s)$

1. $y_i := \mathsf{TTBC}(P, p_s, x_s, n-h, n-h)$;
2. return y_i

Theorem 4. *In model $\mathcal{M}_b$, Protocol 4 achieves broadcast if $2n/h < b + 1$. Its round complexity is $R = \min(n-h, n-b) + 1$ and its bit complexity is polynomial in n for $n - b = O(1)$.*

Proof. Protocol 3 is invoked with parameters $t_v = t_c = n - h$. Since $2n/h < b + 1$, it holds that $2t_v + (b-1)t_c = (b+1)(n-h) = (b-1)n + (2n-(b+1)h) < (b-1)n$ and thus that Protocol 3 achieves TTBC. That Protocol 4 achieves broadcast now follows from Definition 5 and Lemma 5.

Furthermore, if Protocol 1 is run without the efficient approximation technique given in Protocol 2 then the round complexity is $R = \min(n-h, n-b) + 1$. Polynomial bit complexity for $n - b = O(1)$ follows from the efficiency of $\mathcal{P}_n^b$ and the fact that $R \leq n - b + 1 = O(1)$. □

5. The Protocol Along the Lines of Dolev–Strong

For any number t of corrupted players, the broadcast protocol of Dolev and Strong [16] can be based on any authentication scheme with transferability $k \geq t+1$, e.g., any digital signature scheme or the unconditional pseudo-signature scheme in [35]. The protocol then is as secure as the component authentication scheme.

In this section we first show that even the weaker assumption of $\mathcal{P}_n^{2(t+1)}$ (or $\mathcal{P}_n^{2(n-h+1)}$, respectively) is sufficient for broadcast, by slightly adapting the Dolev–Strong protocol to this different primitive. We then give an efficient construction for $\mathcal{P}_n^{2(n-h+1)}$ under the assumption that $2n/h < b + 1$, which can then be plugged into that broadcast protocol.

The stepwise construction of the final broadcast protocol is depicted in Fig. 3. First, BC_b is transformed into $\mathcal{P}_n^b$ with arbitrary resilience. How to achieve this was already shown in Section 3. Then $\mathcal{P}_n^b$ is iteratively transformed into $\mathcal{P}_n^{2(n-h+1)}$ which is possible if $2n/h < b + 1$. This step is demonstrated in Section 5.2. Finally, $\mathcal{P}_n^{2(n-h+1)}$ can be plugged into our modified Dolev–Strong protocol which we present in Section 5.1.

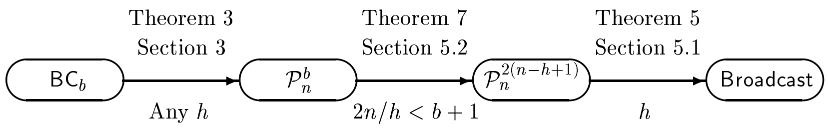

Fig. 3. Stepwise construction of our broadcast protocol along the lines of Dolev and Strong.

5.1. $\mathcal{P}_n^{2(n-h+1)}$ *Implies Broadcast*

We now show that (efficient) proxcast with parameter $k = 2(n-h+1)$ implies (efficient) broadcast secure if $2n/h < b+1$. For this, the Dolev–Strong protocol (with a small modification in [34]) is executed using proxcast instead of signatures. Every player $p_i \in P$ maintains a set A_i of accepted values that, at the end, is either $\emptyset$, $\{0\}$, $\{1\}$, or $\{0, 1\}$. Furthermore, every player p_i maintains two sets $S_i[0]$ and $S_i[1]$ that consist of elements in $\{1, \ldots, n\}$. For ease of exposition, we parameterize the following protocol by the number of corrupted players $t = n - h$ whereby $k = 2(n - h + 1)$ turns into $k = 2t + 2$.

Protocol 5. $\mathsf{Broadcast}(P, p_s, x_s)$

The whole protocol proceeds for $t + 1$ phases. In a first phase, p_s initiates an instance of $\mathcal{P}_n^{2t+2}(P, p_s, x_s)$ sending x_s, sends $\{s\}$ to every other player over the pairwise channels, computes $y_s := x_s$, and halts. During phases $r = 1, \ldots, t + 1$, every player p_i $(i \neq s)$ performs the following actions where, initially, each $A_i = \emptyset$:

- If any value $v \in \{0, 1\}$ has been newly added to the set of accepted values A_i during phase $r - 1$ then p_i initiates an instance of $\mathcal{P}_n^{2t+2-2r}(P, p_i, v)$ sending v, and sends $S_i[v] \cup \{i\}$ to everybody over the pairwise channels.
- Suppose (v, S) is received from any player p_j such that $v \in \{0, 1\}$ and the set S contains at least r distinct values m including s such that p_i received value v with grade $\geq t - r + 1$ from some instance of $\mathcal{P}_n^k$ initiated by p_m. Then v is added to A_i, and $S_i[v] := S$.

At the end of the protocol, every player p_i computes output $y_i = 1$ if $A_i = \{1\}$, and $y_i = 0$ otherwise.

Lemma 6. *If all instances $\mathcal{P}_n^k$ $(k \leq 2t + 2)$ execute correctly then, in the standard pairwise-channels model, Protocol 5 achieves broadcast for any number $t < n$ of corrupted players. Let R_0, B_0, and C_0 be the round, bit, and computational complexities of $\mathcal{P}_n^{2t+2}$. Then the respective complexities of Protocol 5 are $R \leq (t+1)R_0$, $B = O(ntB_0)$, and $C = Poly(nC_0)$.*

Proof. (*Validity*) Assume that the sender p_s is honest. Now, p_i accepts x_s after the first phase but never accepts the value $1 - x_s$ since p_s never initiates any instance of the form $\mathcal{P}_n^k(P, p_s, 1 - x_s)$. Hence every honest player p_i decides on $y_i = x_s$.

(*Consistency*) Assume players p_i and p_j to be honest. We show that p_i and p_j decide on the same value $y_i = y_j$ by showing that $A_i = A_j$ at the end of the protocol.

Consider any value $v \in A_i$. If p_i adds v to A_i for the first time during phase $r \in [1 \cdots t]$, then there are r distinct values of m (including s) in $S_i[v]$ such that p_i received v with grade $\geq t-r+1$ from some instance of $\mathcal{P}_n^k$ initiated by p_m. This implies that p_j received v with grade $\geq t-r$ from the same r instances of $\mathcal{P}_n^k$. Note that p_i will initiate an instance of $\mathcal{P}_n^{2t+2-2(r+1)}(P, p_i, v)$ in phase $r+1$, and p_j will receive this instance with maximum grade $t-r-1$. Also note that p_j will receive $(v, S_i[v] \cup \{i\})$ from p_i in phase $r+1$. This will cause p_j to accept v in phase $r+1$, if he has not already done so.

On the other hand, if p_i accepts v only during phase $t+1$ then some player sent him (v, S) with $t+1$ distinct values of m (including s) in S such that p_i received v with grade $\geq t-r+1$ from some instance of $\mathcal{P}_n^k$ initiated by p_m. One of those $t+1$ distinct values of m corresponds to an honest player who was convinced to accept v in an earlier phase, and then sent convincing information to all parties. Thus every honest player accepts v by the end of the protocol.

(*Complexities*) The round complexity of Protocol 5 is $R \leq (t+1)R_0$, its bit complexity is $B = O(ntB_0)$,[6] and its computational complexity is evidently polynomial in nC_0. □

Theorem 5. *If* $2n/h < b+1$ *then* $\mathcal{P}_n^{2(n-h+1)}$ *allows for efficient broadcast.*

Proof. The theorem follows from Lemma 6. □

5.2. *Transformation from* $\mathcal{P}_n^b$ *to* $\mathcal{P}_n^{2(n-h+1)}$

We now present an efficient transformation from $\mathcal{P}_n^b$ to $\mathcal{P}_n^{2(n-h+1)}$ for the case that $2n/h < b+1$. The transformation proceeds in a stepwise manner from $\mathcal{P}_n^k$ to $\mathcal{P}_n^{k+1}$. The basic step involves one invocation of $\mathcal{P}_n^k$ and n invocations of $\mathcal{P}_n^b$. Since the basic step involves $\mathcal{P}_n^k$ only once, the final reduction will be efficient.

5.2.1. *Transformation Idea*

In a first round, an instance of $\mathcal{P}_n^k$ is executed with the same sender as designated for the broadcast. In a second round, every player (including the original sender, for simplicity) distributes his result using an instance of $\mathcal{P}_n^b$. It is convenient to interpret the initial (binary) $\mathcal{P}_n^k$ with respect to the alternative definition where each player p_i receives a level $\ell_i \in \{0, \ldots, k-1\}$ and the second (non-binary) instances $\mathcal{P}_n^b$ with respect to the original definition where each player p_i receives a value $y_i \in \{0, \ldots, k-1\}$ and a grade $g_i \in \{0, \ldots, \lfloor (b-1)/2 \rfloor\}$.

Thus, in the final protocol, each player p_i receives an initial level $\ell_i \in \{0, \ldots, k-1\}$ and n further messages (one per player p_j) of the form (ℓ_i^j, g_i^j) where $g_i^j \in \{0, \ldots, \lfloor (b-1)/2 \rfloor\}$—expressing that player p_j claimed towards p_i to have received (as a result of $\mathcal{P}_n^k$) level ℓ_i^j, and that p_i received this claim "ℓ_j" from p_j with grade g_i^j. Based on this information, each player p_i finally decides on a new level $L_i \in \{0, \ldots, k\}$.

[6] We adopt the convention that not initiating $\mathcal{P}_n^{2t+2-2r}$ for any value $v \in \{0, 1\}$ is done by initiating $\mathcal{P}_n^{2t+2-2r}$ with value $v = \perp$. Thus, every player initiates a *proxcast* during every phase.

For simplicity, we first describe the initial transformation from $\mathcal{P}_n^b$ to $\mathcal{P}_n^{b+1}$, i.e., $k = b$. The following transformation steps then proceed in a very similar way.

5.2.2. *Decision Rule*

Consider the case $k = b$ and let $\Gamma = \lfloor (b-1)/2 \rfloor$ be the maximal possible grade in the second instances of proxcast. In order to guarantee validity, a player p_i with level $\ell_i = b - 1$ ($\ell_i = 0$) who received at least h values of the form (ℓ_i, Γ) must decide on $L_i = b$ ($L_i = 0$, respectively).

However, if the sender is corrupted and an honest player p_i still has this respective view then, in order to guarantee consistency, an honest player p_j with level $\ell_j = b - 2$ must change his level to $L_j \in \{b-1, b\}$. This, in turn, implies that an honest player p_m with level $\ell_m = b - 3$ must upgrade his level to $L_m \in \{b-2, b-1\}$ whenever an honest player p_j with this respective view exists, etc.

We now describe how a player p_i computes his final level L_i based on his local view, i.e., level ℓ_i and the n received pairs (ℓ_i^j, g_i^j). In order to do so, we define the distance between two pairs of the form (x, g) ($g \in \{0, \ldots, \Gamma\}$). Informally, the distance between two pairs simply characterizes how far they are apart in the "scale" of proxcast. Reconsider Fig. 1.

Definition 6. The *distance* between two levels ℓ_i and ℓ_j is $\mathcal{D}[\ell_i, \ell_j] = |\ell_i - \ell_j|$. Accordingly, the *distance* of two pairs (x_i, g_i) and (x_j, g_j) is

$$\mathcal{D}[(x_i, g_i), (x_j, g_j)] = \begin{cases} |g_i - g_j|, & \text{if} \quad x_i = x_j \ , \\ g_i + g_j, & \text{if} \quad x_i \neq x_j \ \wedge \ b \text{ odd}, \\ g_i + g_j + 1, & \text{if} \quad x_i \neq x_j \ \wedge \ b \text{ even}. \end{cases}$$

In these new terms, validity demands the following rule:

$$\left.\begin{array}{c} \ell_i = b - 1 \\ \wedge \exists S_{b-1} \subset P\colon\ |S_{b-1}| \geq h \ \wedge \ \forall j \in S_{b-1}\colon\ \mathcal{D}[(b-1, \Gamma), (\ell_i^j, g_i^j)] = 0 \end{array}\right\} \longrightarrow L_i := b.$$

Given any honest player p_i following the above rule, an honest player p_k with level $\ell_k = b - 2$ (which is possible in case the sender is corrupted) must also upgrade his level to $L_k = b - 1$ (or $L_k = b$) in order to guarantee consistency.

Thus, assume that an honest player p_i follows the above rule. By the consistency property of proxcast (second round), the h pairs $(\ell_i^j, g_i^j) = (b-1, \Gamma)$ must also be received by player p_k—as pairs of the form $(b-1, \Gamma)$ or $(b-1, \Gamma-1)$, i.e., p_k sees h pairs (ℓ_k^j, g_k^j) such that $\mathcal{D}[(b-1, \Gamma), (\ell_k^j, g_k^j)] \leq 1$. Furthermore, by consistency of proxcast (first round), every honest player must have sent level $b-1$ or $b-2$ during the second round of proxcast. Thus, by validity of proxcast (second round), player p_k must also have received h pairs of the form $(b-1, \Gamma)$ or $(b-2, \Gamma)$. Thus, consistency of $\mathcal{P}_n^{b+1}$ demands the following rule:

$$\left.\begin{array}{r} \ell_i = b - 2 \qquad\qquad\qquad\qquad\qquad \\ \wedge \exists S_{b-2} \subset P : |S_{b-2}| \geq h \wedge \forall j \in S_{b-2} : \mathcal{D}[(b-1, \Gamma), (\ell_i^j, g_i^j)] \leq 1 \\ \wedge \exists S_{b-1} \subset P : |S_{b-1}| \geq h \wedge \forall j \in S_{b-1} : \ell_i^j \in \{b-1, b-2\} \\ \wedge \mathcal{D}[(\ell_i^j, \Gamma), (\ell_i^j, g_i^j)] = 0 \end{array}\right\} \longrightarrow L_i \coloneqq b-1.$$

The rule now progresses further to

$$\left.\begin{array}{r} \ell_i = b-3 \\ \wedge\exists S_{b-3} \subset P : |S_{b-3}| \geq h \wedge \forall j \in S_{b-3} : \mathcal{D}[(b-1,\Gamma),(\ell_i^j, g_i^j)] \leq 2 \\ \wedge\exists S_{b-2} \subset P : |S_{b-2}| \geq h \wedge \forall j \in S_{b-2} : \ell_i^j \in \{b-1, b-2\} \\ \wedge\mathcal{D}[(\ell_i^j,\Gamma),(\ell_i^j, g_i^j)] \leq 1 \\ \wedge\exists S_{b-1} \subset P : |S_{b-1}| \geq h \wedge \forall j \in S_{b-1} : \ell_i^j \in \{b-2, b-3\} \\ \wedge\mathcal{D}[(\ell_i^j,\Gamma),(\ell_i^j, g_i^j)] = 0 \end{array}\right\} \longrightarrow L_i := b-2.$$

In order to guarantee consistency, this rule must now progress all the way to level $\ell_i = 1$. Note that a player p_i with level $\ell_i < b-1$ simply inherits the upgrade rule for $\ell_i + 1$ (weakened by a distance of 1) plus he gets one additional rule. Finally, we apply the following rule for $\ell_i = 0$:

$$\ell_i = 0 \ \wedge\ |\{j \mid (\ell_i^j, g_i^j) = (0,\Gamma)\}| < h \longrightarrow L_i := 1,$$

i.e., a player p_i with level $\ell_i = 0$ upgrades whenever there is not enough "support" for 0. In all other cases, a player p_i keeps his level, $L_i := \ell_i$.

This approach obviously guarantees validity and consistency as long as no honest player p_i holds $\ell_i = 0$. In order to prove consistency for $\ell_i = 0$, we finally show that any honest player p_j upgrading from $\ell_j = 1$ to $L_j = 2$ implies that $|\{j \mid (\ell_i^j, g_i^j) = (0,\Gamma)\}| < h$, and, thus, that p_i computes $L_i := 1$. We end up with the following rule:

Upgrade Rule

`if` $\ell_i > 0 \ \wedge\ \exists\, S_{\ell_i}, S_{\ell_i+1}, \ldots, S_{b-1} \subset P$:

$$\begin{array}{c} \forall S_\ell :\ |S_\ell| \geq h \\ \wedge\forall k, m :\ (S_k \cap S_m \neq \emptyset \Rightarrow |m-k| \leq 1) \\ \wedge\forall S_\ell \forall j \in S_\ell :\ \ell_i^j \in \{\ell, \ell+1\} \wedge \mathcal{D}[(\ell_i^j,\Gamma),(\ell_i^j, g_i^j)] \leq \ell - \ell_i \end{array}$$

`or` $\ell_i = 0 \wedge |\{j \mid \ell_i^j = 0 \wedge g_i^j = \Gamma\}| < h$
`then` $L_i := \ell_i + 1$ `else` $L_i := \ell_i$.

Note that two consecutive sets S_k and S_{k+1} are not necessarily disjoint whereas, in favor of our final analysis, we demand that two non-consecutive sets S_k and S_m ($m \notin \{k, k \pm 1\}$) are disjoint. Such sets S_k can be efficiently constructed in a way that guarantees the precondition of the upgrade rule exactly if it can be satisfied: this is achieved by assigning the sets S_k by increasing index k. This guarantees that "close" pairs are not wasted for too "distant" sets S_k that tolerate more relaxed conditions.

The upgrade rule is also depicted in Fig. 4 for the case where $\ell_i = 1$ and where p_i must upgrade to $L_i = 2$ (the sets T_0 and T_1 will be required later for a counting argument). Each possible output value $\ell \in \{0, \ldots, b-1\}$ of the initial instance of $\mathcal{P}_n^b$ is represented by a row. The columns represent distances from the pairs $(0,\Gamma), \ldots, (b-1,\Gamma)$. Thus the matrix position (x, y) stands for pairs (u, v) received during a secondary instance of $\mathcal{P}_n^b$ that satisfy $\mathcal{D}[(x,\Gamma),(u,v)] = y$. With respect to this example, the upgrade rule now demands that there are sets $S_1, \ldots, S_{b-1}$ that each contain at least h pairs matching the respective distance constraints.

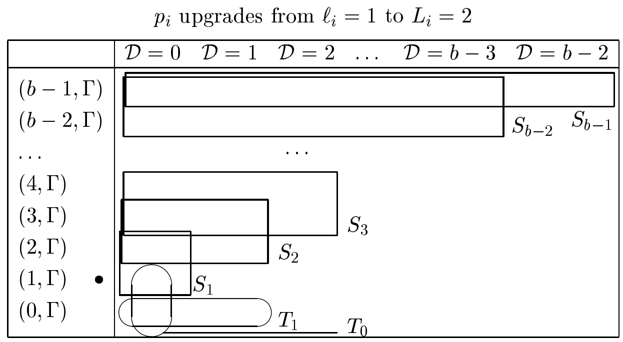

Fig. 4. Upgrade rule and consistency argument for the non-trivial case.

Lemma 7. *Based on* (*efficient*) $\mathcal{P}_n^b$, *the described protocol achieves* (*efficient*) $\mathcal{P}_n^{b+1}$.

Proof. (*Validity*′) Validity is trivially satisfied since p_i receives $\ell_i = x_s \cdot (b-1)$ from sender p_s and at least h pairs of the form (ℓ_i, Γ).

(*Consistency*′) The way the upgrade rule is designed, consistency is trivially satisfied as long as there is no honest player p_j with level $\ell_j = 0$. Thus, assume that such a player p_j exists and that there is at least one player p_i with $\ell_i \neq 0$ and thus, by consistency of the first proxcast, $\ell_i = 1$. We have to show that whenever p_i upgrades ($L_i = \ell_i + 1 = 2$), p_j also upgrades. In particular, we show that it cannot happen that p_i upgrades while p_j stays with his level $L_j = 0$. For this, we distinguish whether or not b is even. Player p_i upgrading to $L_i = 2$ implies the view depicted in Fig. 4.

Since all honest players p_j hold a level $\ell_j \in \{0, 1\}$, player p_i must hold a set T_0 of pairs $(0, \Gamma)$ or $(1, \Gamma)$ with $|T_0| \geq h$ and, since p_j holds at least h pairs $(0, \Gamma)$, p_i also holds a set T_1 of pairs $(\cdot, \cdot)$ such that $\mathcal{D}[(0, \Gamma), (\cdot, \cdot)] \leq 1$ (see Fig. 4). We distinguish whether or not b is even.

ODD b. Since $S_k \cap S_{k+2} = \emptyset$ there are $(b-1)/2$ distinct sets $S_1, S_3, \ldots, S_{b-2}$ of cardinalities at least h. All pairs $(\cdot, \cdot) \in S_1 \cup \ldots \cup S_{b-2}$, for some $\ell \neq 0$, satisfy $\mathcal{D}[(\ell, \Gamma), (\cdot, \cdot)] \leq b-3$, whereas the pairs $(\cdot, \cdot) \in T_1$ satisfy $\mathcal{D}[(0, \Gamma), (\cdot, \cdot)] \leq 1$. Thus, the sets $S_1, S_3, \ldots, S_{b-2}$ and T_1 are all pairwise distinct, and p_i must have received at least $((b+1)/2)h > n$ different pairs in contradiction to the fact that there are at most n players.

EVEN b. Since $S_k \cap S_{k+2} = \emptyset$ the sets $S_1, S_3, \ldots, S_{b-1}$ are pairwise distinct. All pairs $(\cdot, \cdot) \in S_1 \cup \ldots \cup S_{b-1}$, for some $\ell \neq 0$, satisfy $\mathcal{D}[(\ell, \Gamma), (\cdot, \cdot)] \leq b-2$. Thus the sets $T_0 \backslash S_1, S_1, \ldots, S_{b-1}$ are pairwise distinct, and p_i must have received at least $(b/2)h + |T_0 \backslash S_1|$ different pairs.

Furthermore, the sets $S_2, \ldots, S_{b-2}$ are also pairwise distinct. All pairs $(\cdot, \cdot) \in S_2 \cup \ldots \cup S_{b-2}$, for some $\ell \neq 0$, satisfy $\mathcal{D}[(\ell, \Gamma), (\cdot, \cdot)] \leq b-3$, whereas the pairs $(\cdot, \cdot) \in T_0 \cup T_1$ satisfy $\mathcal{D}[(0, \Gamma), (\cdot, \cdot)] \leq 1$. Thus the sets $T_1 \backslash T_0, T_0, S_2, \ldots, S_{b-2}$ are pairwise distinct, and p_i must have received at least $(b/2)h + |T_1 \backslash T_0|$ different pairs.

Hence, since $(T_0 \backslash S_1) \subseteq T_1$, p_i received at least $(b/2)h + \max(|T_0 \backslash S_1|, |T_1 \backslash T_0|) \geq ((b+1)/2)h > n$ pairs overall, contradicting the number n of involved players. □

5.2.3. *General Step from $\mathcal{P}_n^k$ to $\mathcal{P}_n^{k+1}$*

It can be easily seen that the described transformation from $\mathcal{P}_n^b$ to $\mathcal{P}_n^{b+1}$ directly generalizes to the general step from $\mathcal{P}_n^k$ to $\mathcal{P}_n^{k+1}$ ($k > b$). The counting argument basically stays the same whereas more sets S_m get involved which makes the counting even easier.

Theorem 6. *For any $k \geq b$, $\mathcal{P}_n^{k+1}$ can be efficiently achieved from one instance of $\mathcal{P}_n^k$ and n instances of $\mathcal{P}_n^b$.*

Proof. The theorem follows from the above text. □

5.2.4. *Complete Transformation from $\mathcal{P}_n^b$ to $\mathcal{P}_n^{2(n-h+1)}$*

Theorem 7. *If $2n/h < b+1$ then $\mathcal{P}_n^{2(n-h+1)}$ can be achieved from $\mathcal{P}_n^b$. Let R_0, B_0, and C_0 be the round, bit, and computational complexities of $\mathcal{P}_n^b$. Then the resulting protocol for $\mathcal{P}_n^{2(n-h+1)}$ has respective complexities $R = (2(n-h) - b + 3) \cdot R_0$, $B = O(n^2 \log n \cdot B_0)$, and $C = Poly(n \cdot C_0)$.*

Proof. The reduction presented in the previous two sections implies the theorem. $\mathcal{P}_n^k$ results from one invocation of binary $\mathcal{P}_n^{k-1}$ and n invocations of $\mathcal{P}_n^b$ with domain $\{0, \ldots, k-2\}$. Thus the given complexities follow. □

5.3. *The Final Broadcast Protocol*

By Theorems 3 and 7 and Lemma 6, we get the following complexities for the final broadcast protocol: round complexity $R = (n-h+1)(2(n-h) - b + 3)(b-1)$, bit complexity $B = O\left(n^4 \log n \left(n^3 + \binom{n}{b}\right)\right)$, and computational complexity $C = Poly\left(\binom{n}{b}\right)$. We conclude

Theorem 1. *In Model $\mathcal{M}_b$, global broadcast among $n > b$ players is achievable if and only if $2n/h < b+1$. If $b = O(1)$ or $n - b = O(1)$ then broadcast is achievable with computation and communication complexities polynomial in n. In all other cases, our protocols are still polynomial in the size $\binom{n}{b}$ of the network.*

Proof. The theorem now follows from Theorems 3, 7, 5, and 2. □

6. Remarks

6.1. *Pairwise Channels*

Initially, Model $\mathcal{M}_b$ was defined as an extension of Model $\mathcal{M}_2$ (Definition 1), i.e., in addition to partial-broadcast channels, we required pairwise channels. However, since our protocols do not involve any secrecy, every invocation of a pairwise channel can be simulated by the invocation of a partial-broadcast channel. Thus the assumption of pairwise communication channels is not required.

6.2. *Erroneous Partial Broadcast*

In the previous analyses of our protocols, we assumed the BC_b-channels to be perfectly reliable, i.e., to involve no error probability. The results naturally generalize to the case when the underlying BC_b-channels involve some error probability. In order to achieve an overall failure probability negligible in a security parameter k, a security parameter $\kappa = k + O(\log \binom{n}{b})$ for the BC_b-channels is sufficient—which is $\kappa = k + O(\log n)$ for the special case of $b = O(1)$ or $n - b = O(1)$.

6.3. *Consensus*

The given results immediately extend to the consensus variant of Byzantine agreement.

Theorem 8. *In Model $\mathcal{M}_b$, global consensus among $n > b$ players is achievable if and only if $2n/h < \min(b + 1, 4)$; with computation and communication complexities polynomial in n.*

Proof. Note that consensus implies broadcast for $n/h < 2$ and that consensus is impossible when $n/h \geq 2$. Thus impossibility beyond the stated bound follows.

Consensus for $n/h < 2$ can be efficiently simulated by broadcast among the same players. In the case of $b = 2$, the resulting protocol is directly polynomial in n. In order to be polynomial in n for the case of $b > 2$, we have to make sure that only polynomially many BC_b-channels are involved. However, since BC_3-channels are sufficient to achieve $n/h < 2$, we can simply use the construction for $b' = 3$ thereby simulating each BC_3-channel by a BC_b-channel involving the same three players. □

6.4. *Multi-Party Computation*

Previous work. Byzantine agreement is a special case of the more general problem of *multi-party computation* (*MPC*), initially defined by Yao [42], where the players want to evaluate distributedly some agreed function(s) on their inputs in a way preserving privacy of their inputs and correctness of the computed result.

Goldreich et al. [29] gave the first complete solution to the problem for Model $\mathcal{M}_2$: an efficient protocol that is computationally secure for $n/h < 2$—which is optimal (with respect to computational security).

Ben-Or et al. [4] and Chaum et al. [7] gave the first optimal solutions with respect to unconditional security for Model $\mathcal{M}_2$: efficient protocols unconditionally secure for $2n/h < 3$. Also this bound is tight.

For Model $\mathcal{M}_n$ (i.e., when additionally given broadcast channels), Beaver [2] and Rabin and Ben-Or [37] proposed efficient protocols that are unconditionally secure if $n/h < 2$. Also this bound is tight. A more efficient protocol for this model was given by Cramer et al. [11].

When additionally assuming oblivious transfer [36] besides broadcast, non-robust multi-party computation is achievable even in presence of any number of corrupted players [29], [3], [30].[7] Furthermore, when only demanding robustness in the case that

[7] Whereas the protocol in [29] may be completely unfair, the protocols in [3] and [30] guarantee that the adversary has practically no advantage over the honest players in obtaining information about the computation result.

no players are corrupted, $h = n$, then the same results as in [2], [37], [29], [3], and [30] can also be achieved without broadcast channels [21]–[23].

Implications. The results derived in this paper now imply that, instead of Model $\mathcal{M}_n$, Model $\mathcal{M}_3$ is sufficient in order to achieve the result in [2] and [37]. That is, broadcast among three players is sufficient for MPC unconditionally secure for $n/h < 2$. Furthermore, assuming oblivious transfer in Model $\mathcal{M}_b$ (instead of Model $\mathcal{M}_n$ as in [29], [3], and [30]) still allows for unconditionally secure MPC for $2n/h < b + 1$.

7. Conclusion

It was shown that broadcast among every subset of b players allows for global broadcast if and only if $2n/h < b + 1$ players are corrupted. Achievability was demonstrated by protocols whose communication and computation complexities are polynomial in the size $\binom{n}{b}$ of the network and, in particular, polynomial in n whenever $b = O(1)$ or $n - b = O(1)$.

Acknowledgments

We thank Ran Canetti and the anonymous referees for their helpful comments on this paper.

References

[1] A. Bar-Noy, D. Dolev, C. Dwork, and H. R. Strong. Shifting gears: changing algorithms on the fly to expedite Byzantine agreement. *Information and Computation*, 97(2):205–233, Apr. 1992.

[2] D. Beaver. Multiparty protocols tolerating half faulty processors. In *Advances in Cryptology*: *CRYPTO* '89, pp. 560–572. Volume 435 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1989.

[3] D. Beaver and S. Goldwasser. Multiparty computation with faulty majority. In *Proceedings of the* 30*th Annual IEEE Symposium on Foundations of Computer Science* (*FOCS* '89), pp. 468–473, 1989.

[4] M. Ben-Or, S. Goldwasser, and A. Wigderson. Completeness theorems for non-cryptographic fault-tolerant distributed computation. In *Proceedings of the* 20*th Annual ACM Symposium on Theory of Computing* (*STOC* '88), pp. 1–10. ACM Press, New York,1988.

[5] P. Berman, J. A. Garay, and K. J. Perry. Towards optimal distributed consensus (extended abstract). In *Proceedings of the* 30*th Annual IEEE Symposium on Foundations of Computer Science* (*FOCS* '89), pp. 410–415, 1989.

[6] R. Canetti. Security and composition of multiparty cryptographic protocols. *Journal of Cryptology*, 13(1):143–202, 2000.

[7] D. Chaum, C. Crépeau, and I. Damgård. Multiparty unconditionally secure protocols (extended abstract). In *Proceedings of the* 20*th Annual ACM Symposium on Theory of Computing* (*STOC* '88), pp. 11–19. ACM Press, New York, 1988.

[8] B. A. Coan and J. L. Welch. Modular construction of a Byzantine agreement protocol with optimal message bit complexity. *Information and Computation*, 97(1):61–85, Mar. 1992.

[9] J. Considine, L. A. Levin, and D. Metcalf. Byzantine agreement with bounded broadcast. http://arxiv.org/abs/cs.DC/0012024, version 1, 2000.

[10] Ibid., versions 3 and 4, 2003.

[11] R. Cramer, I. Damgård, S. Dziembowski, M. Hirt, and T. Rabin. Efficient multiparty computations secure against an adaptive adversary. In *Advances in Cryptology*: *EUROCRYPT* '99, pp. 311–326. Volume 1592 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1999.

[12] D. Dolev. The Byzantine generals strike again. *Journal of Algorithms*, 3(1):14–30, 1982.
[13] D. Dolev, C. Dwork, O. Waarts, and M. Yung. Perfectly secure message transmission. *Journal of the ACM*, 40(1):17–47, Jan. 1993.
[14] D. Dolev, M. J. Fischer, R. Fowler, N. A. Lynch, and H. R. Strong. An efficient algorithm for Byzantine agreement without authentication. *Information and Control*, 52(3):257–274, Mar. 1982.
[15] D. Dolev and H. R. Strong. Polynomial algorithms for multiple processor agreement. In *Proceedings of the* 14*th Annual ACM Symposium on Theory of Computing* (*STOC* '82), pp. 401–407. ACM Press, New York, 1982.
[16] D. Dolev and H. R. Strong. Authenticated algorithms for Byzantine agreement. *SIAM Journal on Computing*, 12(4):656–666, 1983.
[17] P. Feldman and S. Micali. An optimal probabilistic protocol for synchronous Byzantine agreement. *SIAM Journal on Computing*, 26(4):873–933, Aug. 1997.
[18] M. J. Fischer, N. A. Lynch, and M. Merritt. Easy impossibility proofs for distributed consensus problems. *Distributed Computing*, 1:26–39, 1986.
[19] M. Fitzi. Generalized Communication and Security Models in Byzantine Agreement. Ph.D. thesis, ETH Zurich, 2002.
[20] M. Fitzi and M. Franklin. Byzantine agreement in the partial-broadcast model. Manuscript, 2003.
[21] M. Fitzi, N. Gisin, and U. Maurer. Quantum solution to the Byzantine agreement problem. *Physical Review Letters*, 87(21):7901-1–7901-4, Nov. 2001.
[22] M. Fitzi, N. Gisin, U. Maurer, and O. von Rotz. Unconditional Byzantine agreement and multi-party computation secure against dishonest minorities from scratch. In *Advances in Cryptology*: *EUROCRYPT* '02, pp. 482–501. Volume 2332 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 2002.
[23] M. Fitzi, D. Gottesman, M. Hirt, T. Holenstein, and A. Smith. Detectable Byzantine agreement secure against faulty majorities. In *Proceedings of the* 21*st ACM Symposium on Principles of Distributed Computing* (*PODC* '02), pp. 118–126, 2002.
[24] M. Fitzi, M. Hirt, T. Holenstein, and J. Wullschleger. Two-threshold broadcast and detectable multi-party computation. In Eli Biham, editor, *Advances in Cryptology—EUROCRYPT* '03, pp. 51–67. Volume 2656 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, May 2003.
[25] M. Fitzi and U. Maurer. From partial consistency to global broadcast. In *Proceedings of the* 32*nd Annual ACM Symposium on Theory of Computing* (*STOC* '00), pp. 494–503. ACM Press, New York, 2000.
[26] M. Franklin and R. N. Wright. Secure communication in minimal connectivity models. *Journal of Cryptology*, 13(1):9–30, Winter 2000.
[27] M. Franklin and M. Yung. Secure hypergraphs: privacy from partial broadcast (extended abstract). In *Proceedings of the* 27*th Annual ACM Symposium on Theory of Computing* (*STOC* '95), pp. 36–44. ACM Press, New York, 1995. Also to appear in *SIAM Journal on Discrete Mathematics*.
[28] J. A. Garay and Y. Moses. Fully polynomial Byzantine agreement for $n > 3t$ processors in $t + 1$ rounds. *SIAM Journal on Computing*, 27(1):247–290, Feb. 1998.
[29] O. Goldreich, S. Micali, and A. Wigderson. How to play any mental game. In *Proceedings of the* 19*th Annual ACM Symposium on Theory of Computing* (*STOC* '87), pp. 218–229. ACM Press, New York, 1987.
[30] S. Goldwasser and L. Levin. Fair computation of general functions in presence of immoral majority. In *Advances in Cryptology*: *CRYPTO* '90, pp. 11–15. Volume 537 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1990.
[31] A. Karlin and A. C. Yao. Probabilistic lower bounds for Byzantine agreement and clock synchronization. Manuscript, 1984.
[32] L. Lamport, R. Shostak, and M. Pease. The Byzantine generals problem. *ACM Transactions on Programming Languages and Systems*, 4(3):382–401, July 1982.
[33] Y. Lindell, A. Lysyanskaya, and T. Rabin. On the composition of authenticated Byzantine agreement. In *Proceedings of the* 34*th Annual ACM Symposium on Theory of Computing* (*STOC* '02), pp. 514–523. ACM Press, New York, 2002.
[34] B. Pfitzmann and M. Waidner. Unconditional Byzantine agreement for any number of faulty processors. In *Proceedings of Symposium on Theoretical Aspects of Computer Science* (*STACS* '92), pp. 339–350. Volume 577 of Lecture Notes in Computer Science. Springer-Verlag, Berlin, 1992.
[35] B. Pfitzmann and M. Waidner. Information-theoretic pseudosignatures and Byzantine agreement for $t >= n/3$. Technical Report RZ 2882 (#90830), IBM Research, 1996.

[36] M. Rabin. How to exchange secrets by oblivious transfer. Technical Memo TR-81, Harvard Aiken Computation Laboratory, 1981.

[37] T. Rabin and M. Ben-Or. Verifiable secret sharing and multiparty protocols with honest majority. In *Proceedings of the* 21*st Annual ACM Symposium on Theory of Computing* (*STOC* '89), pp. 73–85. ACM Press, New York, 1989.

[38] O. von Rotz. Reduktion von informationstheoretisch sicheren Konsistenzprimitiven. Master's thesis, ETH Zürich, 2000.

[39] S. Toueg, K. J. Perry, and T. K. Srikanth. Fast distributed agreement. *SIAM Journal on Computing*, 16(3):445–457, June 1987.

[40] R. Turpin and B. A. Coan. Extending binary Byzantine agreement to multivalued Byzantine agreement. *Information Processing Letters*, 18(2):73–76, Feb. 1984.

[41] Y. Wang and Y. Desmedt. Secure communication in multicast channels: the answer to Franklin and Wright's question. *Journal of Cryptology*, 14(2):121–135, 2001.

[42] A. C. Yao. Protocols for secure computations. In *Proceedings of the* 23*rd Annual IEEE Symposium on Foundations of Computer Science* (*FOCS* '82), pp. 160–164, 1982.