Latency of 1–Message Fast Partially Synchronous Leader-Based Consensus: a Complete Characterization

ELENA BERARDINI, Department of Mathematics and Computer Science, Eindhoven University of Technology, the Netherlands

MATTHIEU RAMBAUD, Telecom Paris, Institut Polytechnique de Paris, France

This paper considers protocols for Consensus with External Validity tolerant to malicious corruptions, also known as Multi-valued Validated Byzantine Agreement (MVBA). We consider the model of [DLS88], where the network is assumed to become synchronous after some unknown finite time denoted GST, and the maximum corruption tolerance, of t players out of $3t + 1$. Since the latency of deterministic consensus is linear in the number of players, many works studied under which conditions it is possible to guarantee an output within 2 messages delay. Noticeably, many protocols exist since [DGV05, EPFL] which guarantee simultaneously:

(Slower Fast Track in 2 messages) if all players are honest and the network synchronous, players output within 2 messages delay;

(Normal–Case latency of 3 rounds) if some publicly designated player is honest and the network synchronous, then all players output within 3 rounds.

Since 3 rounds is the optimal Normal–Case latency for partially synchronous consensus, these protocols are said to have *Graceful Degradation*. We thus ask if it is possible to enable instead a Fast Track in only 1 message delay, as soon as all players are honest and have the same input, and still have Normal–Case termination in 3 rounds, i.e., graceful degradation. This question was much investigated in the crash-fault setting, e.g., [Lam06a, DISC], [DS06, DSN], [CS06, PRDC], where the optimal Normal–Case latency is of 2 rounds instead of 3.

We answer by the negative, by showing that for $t \geq 2$, any partially Synchronous Consensus, be it either with with External Validity or Weak Unanimity, if it has a Fast Track in 1 round, then it cannot have Normal–Case latency in 3 rounds.

We prove tightness of $t \geq 2$ by exhibiting a matching upper bound for $t = 1$ of independent interest, i.e., a consensus with Fast Track in 1 round when all four players are honest with the same input, and Normal–Case latency in 3 rounds, the latter noticeably unconditioned on a leader.

We prove tightness of the lower bound of 3 rounds by observing that it is possible to compile any consensus with strong unanimity, into one having a *Fast Track* in 1 message delay, in executions where the network is synchronous and all players are honest and have the same input. Applying this compilation to an existing protocol [Abr+18, Podc], we match a Normal–Case latency of 4 rounds.

1 MODEL AND RESULTS

1.1 Model

1.1.1 Malicious Corruptions. We consider a set $\mathscr{P} = (P_1, \ldots, P_n)$ of $n = 3t + 1$ players, which are polynomial deterministic machines. Up to t of them are corrupted by a polynomial machine denoted the *Environment* $\mathcal{E}$, they share all their state with it and behave as instructed by it. The remaining players are denoted *Honest*.

1.1.2 Digital Signatures. We consider the model known as the *authenticated* setting [LSP82; DS83]. We formalize it as the ideal signing functionality, denoted $\mathcal{F}_{CERT}$ in [Can04, Figure 2]. Any player can submit to $\mathcal{F}_{CERT}$ any bitstring of its choice, then is delivered a *signature* on this bitstring. Signatures can be copied and sent by message. Any player can query $\mathcal{F}_{CERT}$ to *verify* if some bitstring carrying a signature, was indeed submitted to $\mathcal{F}_{CERT}$ by the purported signer. This implies, in particular, that $\mathcal{E}$ cannot forge a signature on a bitstring, which would be certified by $\mathcal{F}_{CERT}$ as having been submitted by a honest player, if this did not happen.

1.1.3 Partially Synchronous Network. Players are linked by pairwise secure channels. *Secure* means that, in addition to be authenticated, the content of messages sent by honest players to honest players cannot be read by $\mathcal{E}$. We consider the Partially Synchronous model of [DLS88, §2.2 3], in which players have access to a global clock and proceed by rounds of communication of fixed duration Δ. Namely, players send their messages simultaneously at the beginning of a round, then wait for the round duration Δ before sending any other message, even if they received new messages in-between. Messages sent can be arbitrarily delayed by $\mathcal{E}$. In addition, in every infinite execution, $\mathcal{E}$ must set a finite time, denoted GST, such that synchrony holds after GST, i.e., all messages are delivered within Δ. Importantly, this includes the messages sent before GST, i.e., no message is lost, which makes our lower bound stronger. The actual value of GST is a priori not revealed to players. We notice that some consensus protocols [Yin+19] allow players in some executions, denoted "optimistically responsive", to output at the actual speed of the network. Thus, the model which we consider, with time-outs of Δ, makes our lower bound comparatively stronger. By convention the starting time of the $(r+1)^{\text{th}}$ round is denoted $t = r$. Thus, GST $= 0$ denotes that synchrony holds from the beginning, in which case all messages sent at the beginning of any round are delivered by the end of the round.

1.1.4 Consensus with External Validity (CEV), a.k.a. MVBA, Normal Case Latency and 1 Message Fast Track. Following [AMS19; Guo+22], we consider as given by the model a public deterministic efficiently computable predicate $\mathsf{ExtValid} : \{0,1\}^* \to \{\mathsf{true}, \mathsf{false}\}$, denoted *External Validity Predicate*. We denote *Valid Values* the set of strings $\mathscr{V} := \{v \in \{0,1\}^*$, such that $\mathsf{ExtValid}(v) = \mathsf{true}\}$. Since $\mathcal{E}$ is only polynomial, we enrich the model with what we denote as a *Forgery Oracle*, noted $\mathscr{O}$, which is a computationally unlimited entity which, on every query of $\mathcal{E}$, samples a valid value a random, i.e., some $v \in \mathscr{V}$, and returns it to $\mathcal{E}$. In the following definition, the terminology of Consensus with External Validity is also known as *Multi-valued Validated Byzantine Agreement* (MVBA) [CKPS01; AMS19; Guo+22]. We prefer the terminology of consensus [DLS88], since the one of Byzantine Agreement often denotes broadcast, which is however unsolvable under partial synchrony.

Definition 1. [CEV] Under the previous model of corruptions, network and leader designation, we denote as Partially Synchronous Consensus with External Validity (CEV) a deterministic protocol that has the following properties, for every possible external validity predicate $\mathsf{ExtValid}$ and Environment $\mathcal{E}$. Every honest player P_i may be initialized by $\mathcal{E}$ with an input value v_i, and may at some point irrevocably output a value.

(Consistency) no two honest players output different values;

(External Validity) if a player outputs v, then $\mathsf{ExtValid}(v) = \mathsf{true}$, i.e., $v \in \mathscr{V}$;

(Termination) in every infinite execution in which every honest player received a valid input value, then they all output.

What makes the problem of CEV not trivial, is that valid values may be computationally untractable to forge by players, e.g., in the typical use-case where $\mathsf{ExtValid}$ would check the signature of some external client. Thus, this rules-out trivial protocols, such as requiring players to brute-force the smallest valid value then output it. In our impossibility proof, we will leverage hardness to forge valid values in Lemma 11. *Termination* could be enlarged to any execution in which players receive a valid value at some point, possibly attached to some message, but formalizing this would require a black-box data structure for valid values. We notice that we did not consider the additional power given to $\mathcal{E}$ in the original definition of External Validity [CKPS01], in which it could set itself which values are valid or not, thus making our lower bound stronger.

Leader Designation and Normal–Case Latency. We assume that the Environment gives to the players the public identity of a specific player $L \in \mathscr{P}$ denoted the *Leader*.

Definition 2. A Consensus with External Validity, in the sense of Definition 1, has *Normal–case Latency in R rounds* if: in every execution where the leader is honest *and* GST = 0, then all honest players output by the end of the R^{th} round.

***Fast Track in* 1 *message*.**

Definition 3. A Consensus with External Validity, in the sense of Definition 1, is said to have a *Fast Track in* 1 *Message* if: in every execution where *all* $n = 3t + 1$ players are honest *and* all have the same input v, *and* $GST = 0$, then all the players output v at the end of the 1^{st} round.

1.2 Main Results

1.2.1 Main Impossibility.

Main Theorem 4. *Any deterministic Consensus with External Validity for $n \geq 7$ players in the sense of Definition 1, if it has a* Fast Track *in* 1 *message, then it cannot have* Normal–Case Latency *of* 3 *rounds.*

Our proof will actually use the Fast Track condition on only one specific value, which thus makes the impossibility stronger.

1.2.2 A Matching Upper-Bound for $n = 4$ and $t = 1$, Furthermore Without Leader. In §4 we prove:

Theorem 5. *There exists a deterministic protocol for Consensus with External Validity for $n = 4$ players in the sense of Definition 1, Fast Track in* 1 *message, and, if $GST = 0$, then all players output by the end of the 3^{rd} round.*

We describe and prove the protocol in §4. Apart from removing the leader condition, the protocol furthermore withstands the following modifications to the model considered for the lower bound Main Thm 4, which all make the result stronger.

- it withstands the model denoted “basic round” in [DLS88, §3], where $\mathcal{E}$ has the additional power to completely suppress any message sent before GST [We notice that in this model, then [DLS88, §2.4] observe that players unavoidably send infinitely many messages in some executions, even for protocols guaranteeing output within fixed latency after GST. Anticipating on the protocol, these scenarios will however not be explicit since entirely captured by $\mathsf{CEV}_{\mathsf{cN}}$.];
- it has Strong Unanimity;
- thus, by application of the compiler Lemma 6, players output in the Fast Track the actual speed of the network, i.e., it has *Optimistic Responsiveness* [PS18; Abr+20];
- the protocol holds in the model where furthermore $\mathcal{E}$ which can read in clear the content of messages sent between honest players, i.e., when downgrading secure channels into authenticated ones;
- it holds in the model [CKPS01] where $\mathcal{E}$ has the power to set itself which values are valid or not;
- the Fast Track is guaranteed even in settings where players would have input values enclosing different data vouching for their validity, e.g., signatures from different accredited entities or different proofs of works, as long as the content of values is the same. This follows straight from the model of [CKPS01], where $\mathcal{E}$ issues *certificates of validity.*
[However, even in the idealized model of signatures, we could not consider $\mathcal{E}$ unlimited, due to the artefact denoted *Requiring consistency in signature verification* [Can04, p10]. Namely, an unlimited $\mathcal{E}$ could have brute-forced verification of every polynomial length signatures of on every possible message, thereby preventing their later use by honest players.]

1.3 Consequences and Tightness of the Bounds

The purpose of remaining paragraphs §1.3, §1.4 and §1.5 is to state variants of our results under various related notions. We provide reminders for these notions, which are: Weak/Strong Unanimity, Partially Synchronous Broadcast, Slower Fast Track in 2 messages instead of 1, relaxed corruption tolerance for Fast Tracks, star-shaped communication patterns, changes of leaders. Thus this may also help to understand related works, surveyed in section §2.

Some of the results have easy proofs which we provide directly. Also, §1.3.1 is proven in a self-contained way to be a consequence of Main Thm 4. Then, §1.3.3 is proven to be a consequence of the proof of Main Thm 4, in a way which does not require to understand the details of the latter. Thus these proofs may also serve as a warmup.

1.3.1 Corollary of Thm 4: Impossibility also Holds when Replacing: Fast Track in 1 Message*, by:* Strong Unanimity*.* A consensus protocol is said to have *Strong Unanimity*, if and only if in executions where all honest players have the same input value, then this must be the output value. Let us make the simple but possibly new observation that we have the following compiler:

Lemma 6. *Consider a protocol for Consensus External Validity in the sense of Definition 1, which has furthermore Strong Unanimity. Then, the following additional instructions enable to obtain a Fast Track in* 1 *message delay, furthermore at the actual speed of the network (Responsiveness), without increasing the latency to output in any execution:*

- *Players multicast their inputs in the first round, in the form of a report message with their signatures.*
- *Upon receiving $n = 3t + 1$ such reports for the* same *value v, a player* outputs *v.*

Proof. *Safety*: if some player fast outputs w, then it must have received $n = 3t + 1$ reports for this value w, which proves that all honest players *have* input w. Thus by Strong Unanimity, no honest player can output a value different from w. Liveness (optimistic output in one round) is obvious. ◻

By Lemma 6, we thus have that the impossibility Theorem 4 also holds when replacing the Fast Track assumption by the Strong Unanimity assumption.

1.3.2 Tightness of Thm 4 with Respect to Normal Case latency in 3 *Rounds.* Surprisingly, no existing work describes Byzantine leader–based consensus with a Fast Track in 1 message. *Applying* (the easy compiler of) Lemma 6, to the consensus [Abr+18, §6.1], which has Strong Unanimity and Normal–Case Latency in 4 rounds, we obtain a consensus with a Fast Track and Normal–case Latency in 4 rounds.

1.3.3 Corollary of the Proof of Thm. 4: Impossibility also holds when Replacing: External Validity*, by:* Weak Unanimity*.* Weak Unanimity requires that, if all players are honest and have the same input value, then the output must be this value. The proof of Thm 4 links two executions in which all players are honest and have input 0, resp., 1. Thus, our impossibility also holds when replacing External Validity by Weak Unanimity.

1.4 Other (Easy) Impossibilities When Trading Assumptions for Others

1.4.1 (Easy) Impossibility of Partially Synchronous Broadcast with a Fast Track, even Without Normal–Case Latency. Partially-Synchronous Broadcast (p–sync BC) [ANRX21], without external validity, may be defined as the following modifications to Definition 1:

replace External Validity by: if $GST = 0$ *and* the Leader is honest, then, if a player outputs v, it must be that the Leader received input v;

replace Termination by: all players output in every infinite execution.

Proposition 7. *No p–sync BC exists which would have a Fast Track in* 1 *message, in the sense of Definition 2.*

Proof. Assume existence of such a p–sync BC for $n = 4$ players of which $t = 1$ may be corrupt. Consider an execution in which: P_1 and P_2 are honest with input 0; P_3 is the Leader, is honest and has input 1, P_4 is corrupt and remains completely silent, and messages are delivered at least between P_1, P_2 and P_3. Then, P_1 and P_2 must ultimately output 1, without waiting to hear from P_4.

But P_1 and P_2 cannot tell apart from an execution in which P_3 is actually corrupt and behaves specifically towards P_4 as if having input 0, all messages to P_4 are delivered in the first round, thus P_4, upon hearing from all players, decides 0 in the 1^{st} round, then is cut from the network. □

1.4.2 (Easy) Impossiblity of a 1*-Message Fast Track Tolerating Corruptions.* We make the simple but possibly new observation that, for a Fast Track in 1 message, then it is impossible to relax the condition: all $3t + 1$ players are honest, by: any $3t$ out of $3t + 1$ players, including the leader, are honest. The statement is general since it holds without requiring External Validity nor Weak Unanimity, nor any Normal Case latency. This thus contrasts with the tight corruption bound for Slower Fast Tracks in 2 rounds, which can be guaranteed as soon as any $3t$ out of $3t + 1$ players, including the leader, are honest. This bound comes as a particular case of the unpublished results of [DGV05], as recalled in §2.2.

Proposition 8. *No consensus under partial synchrony, in the sense of [DLS88, §2.4] can guarantee that, if* $3t$ *players out of* $3t + 1$ *are honest and have the same input value, then players output this value at the end of the* 1^{st} *round.*

Proof. Otherwise, for four players P_1, P_2, P_3, P_4, assume an execution in which P_2 is the leader and is corrupt, P_1 and P_3 have input 0 and P_4 has input 1. P_2 behaves honestly towards P_1 as if having input 0, and honestly towards P_3 and P_4 as if having input 1. All messages are delivered in the 1^{st}round, except: the message from P_4 to P_1, and all messages from P_1. By the relaxed Fast Track condition, P_1 outputs 0 at the end of the 1^{st}round. Then P_1 is cut from the network. Thus, P_3 cannot safely output another value than 0.

Subsequently, P_4 cannot tell apart this execution, from one in which P_1 would have been dishonest and silent since the beginning. Thus P_4 must output, without waiting to hear from P_1.

However, P_4 cannot either tell apart this execution from one in which P_3 would have been corrupt and would have reported to P_1 having input 1, thereby triggering P_1 to output 1 at the end of the 1^{st}round. Thus, P_4 cannot safely output another value than 1. □

1.5 Further Variations, to be Detailed in a Future Version

1.5.1 Variant for the Subclass of Protocols with Linear Message Complexity. We claim the following variant of Main Theorem 4. Consider a CEV with a communication pattern which is star-shaped around the leader, as, e.g., in [Gol+19; Yin+19]. Then if is has a Fast Track in 2 messages, it has at least Normal Case Latency in 6 rounds. This impossibility extends to any family of CEV with a Fast Track in 2 messages, parametrized by n, such that the message complexity is linear in n.

1.5.2 Extension to After a Change of Leader. We claim that our techniques also apply to the following related setting. We consider a CEV in the sense of Definition 1, where we do not require anymore a fast track, but in which we enrich the leader designation abstraction such that a new leader may be designated after some time, and possibly itself replaced, etc. Typical such protocols are [CL99; Yin+19]. Then, we claim a (tight) latency of at least 4 rounds to output from the point when: synchrony holds, and a honest leader is designated, as long as the previous leader, possibly corrupted, had been in place for more than 3 rounds.

2 RELATED WORKS

In our setting of malicious corruptions, the only works known to us considering a Fast Track in 1 Message, as in Definition 3, are [Kur02; Zie06; FMR04; SR08; NIK10]. None of these previous works describes a consensus with a Fast Track in 1 message and Normal–Case latency of 4 rounds. We provide one in §1.3.2.

2.1 Slower Fast Tracks allow Optimal Normal Case Latency. Most existing works in the malicious setting consider Slower Fast Tracks, i.e., with output in 2 messages delay instead of 1 [DGV05; MA05; Kot+09; Cle+09; GQV08; GV10; PS18]. A slower Fast Track guarantees output within 2 messages delay if the network is synchronous and any $3t$ out of $3t + 1$ players, including the leader, are honest. We have the same under–optimality for [Gol+19] in the subclass of protocols with a star-shaped communication pattern around the leader. Namely, [Gol+19] has a slower Fast Track in 3 messages delay, instead of the optimal 2 in this subclass. Paying the price of such a slower Fast Track, in one more message delay, offers in return the feasability of reaching 3 rounds, which is the optimal Normal–Case Latency of partially–synchronous Consensus with External Validity with no Fast Track at all. A protocol combining a slower Fast Track with such optimal Normal–Case latency is achieved in [DGV05], they name this property *Graceful Degradation*. Their technique consists in performing *in parallel* a Fast track and a Slow track. This technique shows up in [GV10], in [Gol+19] in the subclass of star-shaped protocols, and in [Abr+20] in the synchronous setting. Let us outline it, which we observe precludes Strong Unanimity, since the leader imposes its input to players. The leader sends its input v to all players; players acknowledge reception in a signed message that they send to all players; a fast output of v is triggered by the reception of sufficiently many such signed acknowledgements, typically from all $3t + 1$ players ($3t$, with the optimization discussed in §2.2). Such a large set of acknowledging players is denoted as a *fast quorum*. Even if a fast quorum is not met, if $GST = 0$ and the leader honest, honest players are *still guaranteed* to collect a *slow quorum* of $2t + 1$ acknowledgements for the value proposed in the Fast Track. Thus they are exactly in the same situation as at the end of the 2^{nd}round of, e.g., [CL99], thus can output in the 3^{rd}round. In the case where $GST > 0$, then a leader proposing again a value has to collect reports from a quorum of players in order to show that no other value was output in the slower Fast Track. Some safety and liveness issues in some published protocols with slower Fast Tracks are discussed in [Abr+17; SK19; SKD19].

2.2 Slower Fast Tracks can be Allowed Under One More Corruption. In our setting of maximal corruption tolerance, i.e., $n = 3t + 1$ players, it was observed in [DGV05] that it is possible to guarantee a slower Fast Track with one corruption, instead of none. Namely, they safely achieve that all players output within 2 messages delay, as soon: as any $3t$ out of $3t + 1$ players, including the leader, are honest, and $GST = 0$. This result is stated in the Theorem of their §1.2, case 2., with authentication, particularized to $Q = 0$ and $F = M$. It is proven p46-47 under the name DGV_{Alg4}. Their technique consists in, upon detection of an equivocation of the leader in the slower fast track, to not take into account its subsequent messages.

[DGV05] provide lower bounds on the number of corruptions for slower Fast Tracks, under the most general parameters so-far: leader internal or external to the set of players, trade-offs between crash and malicious corruptions, trade-offs between overall corruption tolerance and corruptions tolerated for the slower Fast Track. They tightly match all these lower bounds.

In the crash fault-setting, [Lam06b] gives tight bounds on the number of corruptions under which a Fast Track in 1 message can be guaranteed.

2.3 The Crash Fault Setting is what motivated this work, since Fast Tracks in 1 Message are much more investigated in this setting than under malicious corruptions. The topic was initiated by [PS98; Bra+01]. The issue of degradation of the Normal Case Latency was observed by Lamport in Fast Paxos [Lam06a]: "If collisions are too frequent [here: if not all players have the same input], then classic Paxos might be better than Fast Paxos." Guerroui-Raynal [GR04, §5.5.3] then Dobre-Suri [DS06, §5], and independently Charron-Bost and Schiper [CS06], asked if a Fast Track *unavoidably* harms latency in the normal case. All of them answered that degradation *can* be circumvented. More precisely, they enable a Fast Track in 1 message, at least when all n players are honest and the network synchronous, and still reach an optimal Normal-Case latency of 2 rounds, which is the optimal in the crash fault setting [KR03]. Our lower bound shows that, by contrast, this degradation of latency cannot be circumvented in the partially synchronous Byzantine setting. Our one-more-round lower bound is also reminiscent of the $t + 2$ lower bound in the context of Indulgent consensus [DG02], i.e., based on unreliable failure detectors, and withstanding asynchrony.

2.4 Early decision in the Unauthenticated Deterministic Synchronous setting, without leader, for Broadcast. [DRS82, §9] asked if Byzantine *broadcast* in which players output fast, i.e., in $\leq f + 2$ rounds in favorable executions where there are $f < t$ corrupted players, can always output within the tight general $t + 1$ number of rounds with optimal $t < n/3$ resiliency. The problem was solved in [BGP92], then in [AD15] with optimal communication complexity and corruption tolerance in the subclass of protocols without digital signatures nor leaders. Notice that, in this unauthenticated leaderless setting, [DL13] show a strict *communication* cost for earlier output, even in the crash fault setting (cubic instead of quadratic).

Randomized algorithms without leader. Cohen [Coh+19] showed that under synchrony, then 3 rounds is the minimum to have a non negligible probability that any player outputs, which is matched by the baseline of [CM16] provided a common coin. The question is open whether this degrades over *partial* synchrony: for instance recall that in the *deterministic* setting, partial synchrony is shown in [DG02] (PODC'02) to require *one more* round before output. The question also holds for the other measurement of latency, which is the *expected* number of rounds before output. For instance in the partially synchronous [Abr+18, §6] (PODC'19 long version), which uses leader sortition using a trusted setup, the expected number of rounds before output is more than 5 "phases", so 20 rounds.

3 PROOF OF MAIN THEOREM 4

3.1 Formalism

An *execution* of a protocol is a sequence of rounds that starts from time $t = 0$ at which $\mathcal{E}$ initializes the players, gives the honest ones their inputs, corrupts up to t of players, and makes public the identity of a leader. Notice that by determinism, the internal state of a player in a round boils down to: its input, the leader, and all messages received so far, timestamped by the round in which they were received. We assume without loss of generality a protocol which instructs players to send their full internal state to all other players, at the beginning of every round. Then, to emulate any actual protocol, we simply have receivers discard the information that they are not supposed to process. We prove Main Theorem 4 for $n = 7$, then it generalizes to any n by standard methods [GR04; LSP82].

We say that two (partial) executions $\mathscr{E}$ and $\mathscr{E}'$ up to some finite time t are *equivalent* from the point of view of some *honest* player P (resp. some group of honest players S_h), and we write $\mathscr{E} \overset{P}{\sim}_t \mathscr{E}'$ (resp. $\mathscr{E} \overset{S_h}{\sim}_t \mathscr{E}'$), if and only if P (resp.

any player in S_h) outputs in *at least one* of the two executions at time t, and the state of P (resp. of any player in S_h) at time t is the same in both executions – that is, both executions are *identical* for P (resp. for every player in S_h) up to t. Notice that by determinism, this implies that P (or any player in S_h) outputs in *both* executions, at time t, the same values in both executions, as formalized in Lemma 3.1. We say that two executions $\mathscr{E}$ and $\mathscr{E}'$ are *linked*, which we denote by $\mathscr{E} \sim \mathscr{E}'$, if and only if there is a chain of equivalent executions from $\mathscr{E}$ to $\mathscr{E}'$. It is then a straightforward consequence of Consistency that, if some player outputs in $\mathscr{E}$, then no player in $\mathscr{E}'$ can output a different value.

Lemma 3.1. *Suppose there is a honest player P and two executions $\mathscr{E}$ and $\mathscr{E}'$ such that $\mathscr{E} \overset{P}{\sim}_t \mathscr{E}'$. Suppose further that P outputs the value $v_{\mathscr{E}}$ at time t in $\mathscr{E}$. Then P outputs in $\mathscr{E}'$ the same value $v_{\mathscr{E}}$ at time t.*

Proof. Since at time t, the player P has the same state in both $\mathscr{E}$ and $\mathscr{E}'$, by determinism of the protocol, it must take the same actions in both executions at time t. In particular, it must output, and output the same value, in both executions. □

We furthermore assume that valid values are hard to forge, in the precise sense given in §3.5, and that there exists at least 2 distincts valid values.

3.2 Main goal

We follow the classical strategy consisting in exhibiting two executions in which players output different values, then prove that they are linked, and thus the values should instead be the same. The two executions which we exhibit are denoted $\mathscr{E}^{(0,\text{fast})}$ and $\mathscr{E}^{(1)}$, and are described below. Input values are denoted for simplicity 0 and 1, more detailed assumptions and formalism are provided in §3.5.

$\mathscr{E}^{(0,\text{fast})}$ **:** GST = 0 , all players are honest and are given input 0 by $\mathcal{E}$. Thus by the Fast Track condition, they all output 0 by the end of the 1^{st} round.

$\mathscr{E}^{(1)}$ **:** GST = 0, all players are honest and are given input 1 by $\mathcal{E}$. Since valid values are hard to forge, we conclude by Lemma 11 in §3.5 that players cannot output another value than 1.

3.3 First step

Before we expose our strategy, we need to go to another starting point, from $\mathscr{E}^{(0,\text{fast})}$ in two steps, as follows.

$\mathscr{E}^{(0,\text{iso})}$ **:** GST = 4, only the leader, which is P_4, is corrupt. All players have input 0 and all messages sent are timely delivered, except all the messages sent by P_7, which are all delayed to GST. In that sense, P_7 is isolated, which accounts for the "iso". P_4 behaves honestly as if it were honest and had input 1, with the following exception: the message that P_4 delivers to P_7 in the 1^{st} round is the same as in $\mathscr{E}^{(0,\text{fast})}$, i.e., compatible with being honest and having input 0. Thus, executions $\mathscr{E}^{(0,\text{fast})}$ and $\mathscr{E}^{(0,\text{iso})}$ are equivalent for P_7 by the end of the 1^{st} round, when it outputs.

$\mathscr{E}^{(\text{sil})}$ **:** GST = 0, only P_7 is corrupt, and does not send any message, i.e., it is "silent". The leader P_4 is now honest, and has input 1. Since the leader is honest, all players output by the end of the 3^{rd} round. The executions $\mathscr{E}^{(0,\text{iso})}$ and $\mathscr{E}^{(\text{sil})}$ are equivalent for all players $\{P_1, P_2, P_3, P_4, P_5\}$ by the end of round 3, when they output.

3.4 Structure of the Proof

Our goal is to find a link from $\mathscr{E}^{(\text{sil})}$ to an execution in which all players are honest, including P_7, with P_7 having input 1. Then, by repeating the strategy to every other player, we will be able to ultimately link to $\mathscr{E}^{(1)}$. This is the overall strategy of [DS83], let us recall their approach.

Definition 9. We denote a corrupt player P as r–hidden if

- until, and including in, round $r-2$, P acts like a honest player, *i.e.* he sends messages that agree with its full state;
- until, and including in round $r-1$, P sends a subset of messages $\mathscr{M}' \subset \mathscr{M}$ that it would have been meant to sent if it were honest;
- from round r, P is forever silent, *i.e.* does not send any message.

A 1–hidden player is thus a player silent from the beginning, *e.g.* as P_7 in $\mathscr{E}^{(\mathrm{sil})}$.

3.4.1 Main Technical Tool. We assume that for $r \in \{2, 3, 4\}$ we have the following tool at our disposal for any player, *excepted* the leader P_4:

(T.r) consider one execution with GST $= 0$ in which only one player P is corrupt, r–hidden, and who in particular sends only a subset $\mathscr{M}' \subset \mathscr{M}$ of the messages that it is intended to send in round $r-1$. Then, we can link this execution with an execution with GST $= 0$ in which P is corrected by one more message m in round $r-1$, *i.e.* $\mathscr{M}' \longleftarrow \{\mathscr{M}' \cup \{m\}\}$.

We use this result for $r \in \{2, 3, 4\}$ in what follows. They correspond to Lemma 14, Lemma 13, and Lemma 12, respectively, proved in Section 3.6 and 3.7.

3.4.2 Claim. From the previous tool we can deduce the following

Claim: one can link any two executions $\mathscr{E}$ and $\mathscr{E}'$ such that

(i) in $\mathscr{E}$, P is 0–hidden, *i.e.* completely silent, and all other players honest;
(ii) in $\mathscr{E}'$, we have that all players are honest;
(iii) the inputs of all players $\mathscr{P} \backslash \{P\}$ are unchanged in $\mathscr{E}$ and $\mathscr{E}'$;
(iv) we can choose arbitrarily the input value v of P in $\mathscr{E}'$.

The proof of the Claim is discussed below in §3.4.4.

3.4.3 Concluding the Proof by using the Claim. Applying the *Claim* to P_6, it follows that one can link the execution $\mathscr{E}^{(\mathrm{sil})}$ to one in which all players are honest and with inputs $\{0, 0, 0, 1, 0, 0, 1\}$. Then, choosing another player in $\mathscr{P} \backslash \{P_4 \cup P_7\}$ with input 1, *e.g.* P_1, one can apply the Claim *backwards* to further link the execution with another execution in which P_1 is completely silent, then flip its input, and apply the Claim forward, to obtain an execution with GST $= 0$ in which all players are honest, with inputs $\{1, 0, 0, 1, 0, 0, 1\}$. Repeating the process for each remaining player $\{P_2, P_3, P_5, P_6\}$, we achieve $\{1, 1, 1, 1, 1, 1, 1\}$

3.4.4 Proof of the Claim. Before we start, let us notice that, in the synchronous context of [DS83, Thm. 2], the Main Technical Tool is proven all at once with the Claim, by backwards induction on r. However, they can start their induction at a fixed time, which is $t+1$ in their context, when their players are guaranteed to output. This guarantee however does not hold in our context, we do not have any fixed delay to output when the leader is corrupt.

Thus, instead, we prove directly the Claim from the tools above, as follows.

Starting from a completely silent player P, add one by one messages sent in round 1 using (T.2), all of them being compatible with having input v. Then add one by one messages sent in round 2 using (T.3), add one by one messages sent in round 3 using Lemma (T.1). At this point P is honest, with input v.

For the same reason, since we cannot either apply any backward induction as in [DS83] to prove the tool (T.r) for $r \in \{2, 3, 4\}$, we need to construct all of them by hand.

3.5 Hard to Forge Valid Values Implies Weak Unanimity, in Some Sense

Definition 10 (Hard to Forge Valid Values). Consider a game involving a machine denoted $\mathscr{A}$, which can query arbitrarily many times the illimited Forgery Oracle $\mathscr{O}$. We say that $\mathscr{A}$ wins the game if, at some point, it outputs a value which is valid, and which was not returned by $\mathscr{O}$ in a previous response. We say that valid values are *Hard to Forge* if the ExtValid predicate is such that no polynomial machine $\mathscr{A}$ has nonnegligible advantage of winning.

An example is when ExtValid returns true only if the value carries a digital signature of which the private signing key is known to $\mathscr{O}$ (in particular, does not leak its signature key to anyone else). Indeed, the definition of Existential Unforgeability under Chosen-Message Attacks (EUF-CMA) implies Definition 10 of Hard, since in the EUF-CMA game the adversary can request valid signatures on any messages of its choice, and thus a fortiori random ones, before trying to forge a signature on a new message. A counter-example is when ExtValid checks validity of a signature of which $\mathcal{E}$ knows the private key.

All executions considered in the proof of Main Theorem 4 share the following common initial events: $\mathcal{E}$ queries once the input forgery oracle $\mathscr{O}$ and obtains a valid value, which we denote for simplicity 1. Notice that 1 is not the actual bitstring $\{1\}$, since otherwise it would be easy to forge. For technical reasons, in $\mathscr{E}^{(1)}$, $\mathcal{E}$ stops further querying $\mathscr{O}$.

Lemma 11. *Consider an execution of Consensus with External Validity (Def. 1). Assume that $\mathcal{E}$ queries $\mathscr{O}$ only once, and denote* 1 *the valid value received from $\mathscr{O}$. Assume that $\mathcal{E}$ initializes all players with input* 1 *and corrupts none of them. Then players cannot output another value than* 1*, except with negligible probability.*

Proof. Consider a Hardness game of $\mathcal{E}$ against $\mathscr{O}$, where $\mathcal{E}$ makes one single query, then initializes players with the response 1, as in the Lemma 11. Then $\mathcal{E}$ outputs the value output by players. If players would output a different value than 1, then the PPT machine $\mathscr{A}$, consisting in $\mathcal{E}$ together with the players, would win the Hardness game against $\mathscr{O}$. Thus this happens with negligible probability. □

All other executions in the proof of Main Theorem 4 also share the following subsequent events: $\mathcal{E}$ queries repeatedly $\mathscr{O}$ until it obtains a valid value which is different from 1, and which we denote for simplicity 0, then stops querying $\mathscr{O}$. Let us give a mainstream example captured by these assumptions. $\mathscr{O}$, which we denote as Mick, generates a signature key pair. ExtValid returns true on a bitstring if it comes appended with a signature which is valid with respect to the public verification key of Mick. $\mathcal{E}$ queries Mick and obtains 1, equal to the signed bitstring "stay", then queries again Mick and obtains 0, equal to the signed bitstring "go".

3.6 Easy Tools: Lemmas 12 and 13

Lemma 12. *Consider $n = 4 = 3 + 1$ players executing the protocol, and an execution $\mathscr{H}$ in which* GST $= 0$*, one player is corrupt, e.g. P_4, and such that P_4 has the initial state of a honest player, and has honest behavior in the first* 2 *rounds. Then, in the* 3^{rd} *round, P_4 sends a strict subset $\mathscr{M}' \subset \mathscr{M}$ of the $n = 4$ messages that it would have been meant to send in the* 3^{rd} *round if it were honest. Consider any message $m \in \mathscr{M} \backslash \mathscr{M}'$ not sent by P_4, e.g. of which the recipient should have been P_1. Then, the execution $\mathscr{H}$ is equivalent to the execution $\mathscr{H}'$ which differs from $\mathscr{H}$ by the fact that P_4 furthermore sends m in the* 3^{rd} *round.*

Proof. We consider the most difficult case, which is when P_1 is the leader. In $\mathscr{H}$ the leader is honest and GST $= 0$, thus all honest players output by the end of the 3^{rd} round. Let us consider the honest player P_2. He has same state at the end of the 3^{rd} round in both $\mathscr{H}$ and $\mathscr{H}'$, thus it also outputs in $\mathscr{H}'$, and outputs the same value. □

Lemma 13. *Consider* $7 = 5 + 2$ *players executing the protocol, and an execution* $\mathscr{G}$ *in which* GST $= 0$*, one single player, e.g.,* P_7*, is corrupt, and such that* P_7 *has the initial state of a honest player, and has honest behavior in the* 1^{st} *round; then in the* 2^{nd} *round* P_7 *sends a strict subset* $\mathscr{M}' \subset \mathscr{M}$ *of the* $n = 7$ *messages that it would have been meant to send in the* 2^{nd} *round if it were honest. Then it is silent in the* 3^{rd} *round. Consider any message* $m \in \mathscr{M} \backslash \mathscr{M}'$ *not sent by* P_7*, e.g. of which the recipient should have been* P_6*. Then, the execution* $\mathscr{G}$ *is equivalent to the execution* $\mathscr{G}'$ *which differs from* $\mathscr{G}$ *by the fact that* P_7 *furthermore sends* m *in the* 2^{rd} *round.*

Proof. We consider the most difficult case, which is when P_6 is the leader. In $\mathscr{G}$ the leader is honest and GST $= 0$, thus all players output by the end of the 3^{rd} round.

$\mathscr{G}_1$: GST $= 0$. P_6 is now also corrupt. P_7 now sends m to P_6 in the 2^{rd} round. Then, in the 3^{rd} round, P_6 behaves honestly towards all players except P_1, to which it sends a message corresponding to its state as if it had not received m. Both executions are equivalent from the point of view of P_1 at the end of the 3^{rd} round, who thus still outputs. Also, we have P_2 who outputs in $\mathscr{G}_1$ since from its perspective the leader P_6 is honest.

$\mathscr{G}'$: GST $= 0$ and the leader is honest. Thus all players output at the end of the 3^{rd} round. $\mathscr{G}'$ differs only from $\mathscr{G}_1$ by the leader sending also a honest message to P_1. In particular, $\mathscr{G}_1$ and $\mathscr{G}'$ are equivalent for P_2 at the end of the 3^{rd} round. □

3.7 Main technical tool: Lemma 14

Lemma 14. *Consider* $7 = 5 + 2$ *players executing the protocol, and an execution* $\mathscr{E}_1$ *in which GST*$= 0$*, one single player, e.g.* P_1*, is corrupt, and such that* P_1 *has the initial state of a honest player. In the* 1^{st} *round,* P_1 *sends a strict subset* $\mathscr{M}' \subset \mathscr{M}$ *of the* $n = 7$ *messages that it would have been meant to send in the* 1^{st} *round if it were honest. Then it is silent forever from the* 2^{rd} *round. Consider any message* $m \in \mathscr{M} \backslash \mathscr{M}'$ *not sent by* P_1 *in the* 1^{st} *round, e.g. of which the recipient should have been* P_4*. Then, the execution* $\mathscr{E}_1$ *is equivalent to the execution* $\mathscr{E}_7$ *which differs from* $\mathscr{E}_1$ *by the fact that* P_1 *furthermore sends* m *in the* 1^{st} *round.*

Proof. We consider the most difficult case, which is when P_4 is the leader. Throughout the executions the leader will remain P_4. In $\mathscr{E}_1$ the leader is honest and GST $= 0$, thus all players output by the end of the 3^{rd} round.

$\mathscr{E}_2$: GST $= 0$. Both P_1 and P_2 are corrupt. P_1 has same behavior, P_2 has honest behavior, except that in the 3^{rd} round it does not send the message to P_7 that it was intended to. Since the leader is honest, all players output by the end of the 3^{rd}round. $\mathscr{E}_2$ is equivalent to $\mathscr{E}_1$ for honest players $\{P_3, P_4, P_5, P_6\}$ at the end of the 3^{rd}round.

$\mathscr{E}_3$: GST=$T_{BQ} + 1$, where T_{BQ} will be defined from execution $\mathscr{E}_4$. P_3 and P_4 are corrupt. All messages of P_1 in $\mathscr{M} \backslash \{\mathscr{M}' \cup \{m\}\}$ are delayed to the 4^{th} round. All messages of P_7 sent from the 3^{rd}round are delayed to GST. All messages of P_2 are timely delivered, except the one sent to P_7.

P_4 has honest behavior in the 1^{st}round. Then in the 2^{nd}, P_4 delivers honest messages to $\{P_1, P_2, P_3\}$, in particular, compatible with its reception of m. Denote rec the content of these messages, which we can assume to be identical, which stands for “Received”. In the 2^{nd}, P_4 delivers messages to $\{P_3, P_5, P_6, P_7\}$ which are compatible with its state as if it had not received m. Denote nr the content of these messages, which we can assume to be identical, which stands for “Non Received”. In particular, P_3 is provided with both rec and nr. In the 3^{rd}round, P_4 delivers a message to P_7 only, compatible with nr, and is otherwise silent forever.

P_3 behaves honestly in the first two rounds. Then in the 3^{rd}round it equivocates, in that it delivers to P_7 a message compatible with reception of rec, whereas it delivers to all other players a message compatible with reception of nr.

Denote $BQ \coloneqq \{P_1, P_2, P_3, P_5, P_6\}$, which stands for "backup quorum". All players in BQ execute the protocol honestly after the 3^{rd}round. In particular, P_3 acts is if having received rec. All messages sent by BQ after the 4^{th} round are timely delivered.

$\mathscr{E}_4$: P_4 and P_5 are corrupt. GST $= T_{BQ} + 1$. P_7 has honest behavior in the two first rounds, then goes silent forever from the 3^{rd}round. P_4 behaves honestly in the 1^{st}round, then in the 2^{nd}it sends rec to $\{P_1, P_2, P_3\}$, and sends nr to $\{P_5, P_6\}$. Honest players, *i.e.* BQ, output at a time which we denote as T_{BQ}. Set GST $= BQ + 1$. The execution is equivalent to $\mathscr{E}_3$ for all honest players in BQ, *i.e.* $\{P_1, P_2, P_5, P_6\}$.

$\mathscr{E}_5$: GST $= T_{BQ} + 1$. P_4 and P_5 are corrupt. All messages of P_7 sent from the 3^{rd}round are delayed to GST. P_4 behaves honestly in the 1^{st}round, then in the 2^{nd}it sends nr to P_5, and rec to all players. In particular, P_5 is provided with both rec and nr.

P_5 behaves honestly in the first two rounds. Then in the 3^{rd}round it equivocates, in that it delivers to P_7 a message compatible with reception of rec, whereas it delivers to all other players a message compatible with reception of nr.

All players in BQ execute the protocol honestly after the 3^{rd}round. In particular, P_5 acts as if having received rec. All messages sent by BQ after the 4^{th} round are timely delivered.

Thus, $\mathscr{E}_5$ is equivalent to $\mathscr{E}_4$ for the honest players in BQ, *i.e.* $\{P_1, P_2, P_3, P_6\}$ at the time T_{BQ} when they output.

$\mathscr{E}_6$: GST $= 0$. P_1 and P_6 are corrupt. P_1 sends messages $\mathscr{M}' \cup \{m\}$ in the 1^{st}round then is otherwise silent. P_4 delivers only rec to P_5. P_6 behaves honestly, except that it does not send its message to P_7 in the 3^{rd}round. Since the leader is honest, all players output in the 3^{rd}round.

Thus, $\mathscr{E}_6$ is equivalent to $\mathscr{E}_5$ for P_7 at the end of the 3^{rd}round, by when they have output.

$\mathscr{E}_7$: GST $= 0$. Only P_1 is dishonest, as in the statement of Lemma 14, *i.e.* it sends only messages $\mathscr{M}' \cup \{m\}$ in the 1^{st}round then is silent forever. $\mathscr{E}_7$ is equivalent to $\mathscr{E}_6$ for $\{P_2, P_3, P_4, P_5\}$ at the end of the 3^{rd}round, by when they have output.

□

4 PROOF OF THEOREM 5

4.1 Data structures and Terminology

We consider the model of §**1.2.2**. We consider in addition a special value denoted $\perp \notin \mathscr{V}$. By convention $\perp$ is never valid. We abuse notations, in that we denote non-valid values as $\perp$, and players in the protocol consider non-valid values as $\perp$.

4.1.1 Protocol Overview.

In the first three rounds of the protocol, every player essentially forwards to all players, including to itself, the set of all messages received in the previous round, appended with its signature on the whole. For the sake of clarity and message-efficiency, we do not instruct players to send messages in some cases, to be detailed. These cases are when where they received messages from 3 or less issuers in the previous round, of which one of them was openly corrupt. Indeed this implies that $GST > 0$ and thus that no output within a specific delay (3 rounds) is required.

Explicitely: each player sends to all a message denoted *Report* in the 1^{st}round, which by definition consist in its input, appended with its signature; sends to all a message denoted *Forward* in the 2^{nd}round, which by definition consist in the set of reports received in the 1^{st}round (including the one sent to itself), with its signature on the whole; and sends to all a message denoted *Set of Forwards* (SoF) in the 3^{rd}round, which by definition consists in the set of forwards received in the 2^{nd}round (including the one sent itself), with its signature on the whole.

	P_1	P_2	P_3	P_4	P_5	P_6	P_7	GST	
$\mathcal{E}_1$	C	nr	nr	nr	nr	nr	nr	0	equivalent for P_3, P_4, P_5, P_6 at t=3
$\mathcal{E}_2$	C	C	nr	nr	nr	nr	nr	0	equivalent for P_7 at $t = 3$
$\mathcal{E}_3$	rec	rec	C	C	nr	nr	nr	$T_{BQ} + 1$	equivalent for P_1, P_2, P_5, P_6 at $t = T_{BQ}$
$\mathcal{E}_4$	rec	rec	rec	C	nr	nr	C	$T_{BQ} + 1$	equivalent for P_1, P_2, P_3, P_6 at $t = T_{BQ}$
$\mathcal{E}_5$	rec	rec	rec	C	C	rec	rec	$T_{BQ} + 1$	equivalent for P_7 at t=3
$\mathcal{E}_6$	C	rec	rec	rec	rec	C	rec	0	equivalent for P_2, P_3, P_4, P_5 at $t = 3$
$\mathcal{E}_7$	C	rec	rec	rec	rec	rec	rec	0	

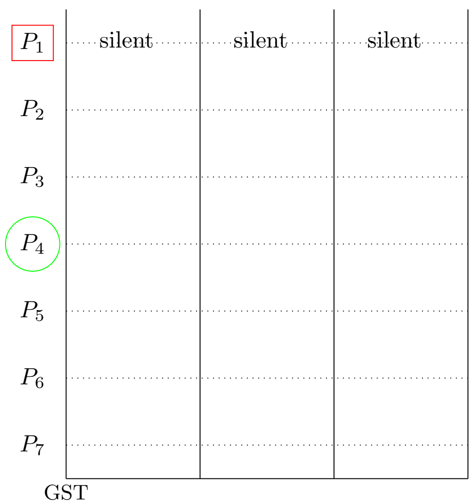
Fig. 1. C stands for Corrupt, $t = 3$ is the end of the 3^{rd} round. We write rec as soon as the player has been forwarded the message m, even if being forwarded conflicting declarations from P_4 not to have received m. This happens, *e.g.* for honest players in BQ after $t = 4$, in the case where they would share together what they received from P_4 in previous rounds before they output.

A set $\mathcal{Q}$ of 3 players is denoted as a *Quorum*. We then give a name to the specific SoFs satisfying the property which could be roughly phrased as: being compatible with an execution in which GST would be 0 and some quorum $\mathcal{Q}$ would be the honest players. We then say that such SoF is a *Forwardings from the Honest-Looking Quorum* $\mathcal{Q}$ (Fhlq). Three identical Fhlqs issued by the three players in $\mathcal{Q}$ constitute a *Decision Certificate.* It triggers a *Decision* for a value, as soon as at the end of the 3^{rd} round, in the executions denoted "normal", i.e., where GST=0. The decided value is determined from 3 or 4 values extracted from the Reports, forwarded by members of $\mathcal{Q}$, contained in the Fhlq*s*, as specified in §4.1.5.

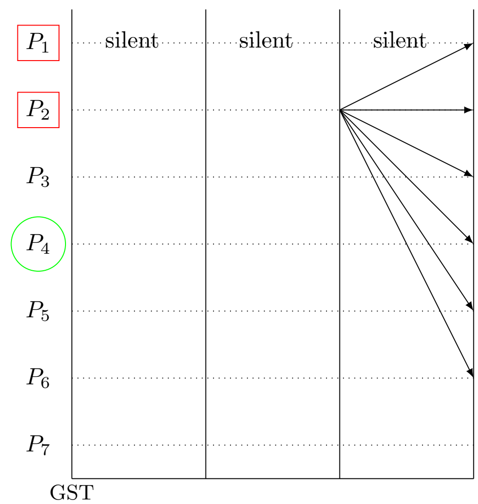
From the beginning of the 4^{th} *round,* players do the following additional tasks. They enable to catch up players which would not have output by then, due to possibly $GST > 0$. Each player P repeatedly sends, to all players, a message denoted "Testimony", in which it includes the most recent message sent in the 3 first rounds. Namely, it includes in the Testimony: a Decision certificate, if it sent any; else: a Fhlq, if it sent any; else: simply includes the report of its input.

Each player P, upon receiving testimonies from any three players, checks if the possibly many enclosed Fhlqs vouch for different values. If the case, then we will show that it must be the case that the testimonies contain contradictory signed messages from some given player Q, thereby constituting a *Proof of Corruption* of Q.

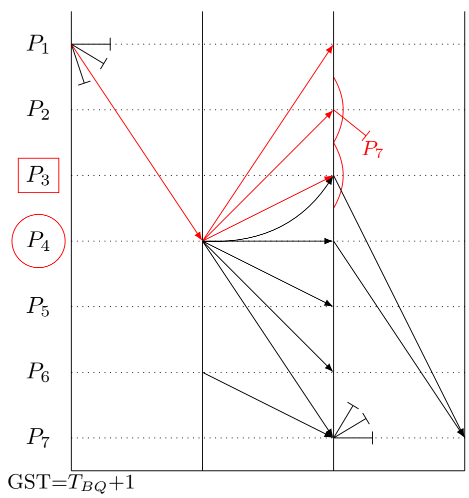
In this case, if Q was one of the issuers of the three testimonies received, then P waits for the testimony of the fourth player. **In such a situation, since $t = 1$, P is guaranteed not to wait indefinitely.** Indeed we must have that the fourth player is honest. Recall instead that, in the proof of impossibility, in the execution $\mathscr{E}_4$, despite players $\{P_1, P_2, P_3, P_5, P_6\}$ observe conflicting Fhlq, constituting a Proof of Corruption of P_4, they cannot tell if P_7 is corrupt and thus silent forever, or, if they will hear from it after GST. But our proof shows that players cannot safely decide



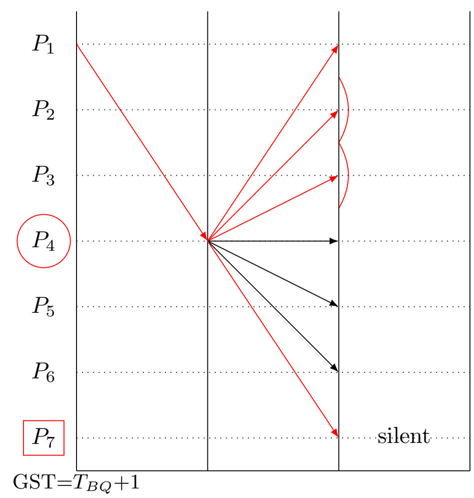
(a) Execution $\mathscr{E}_1$



(b) Execution $\mathscr{E}_2$



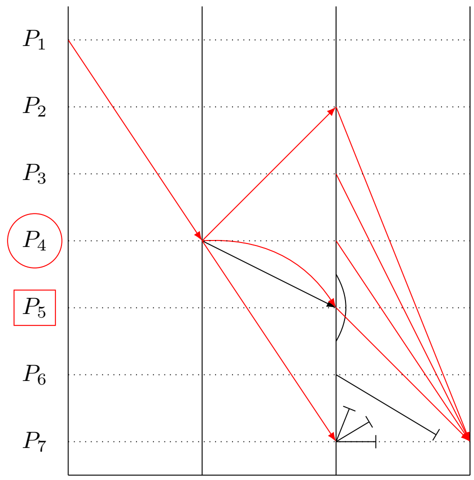
(c) Execution $\mathscr{E}_3$



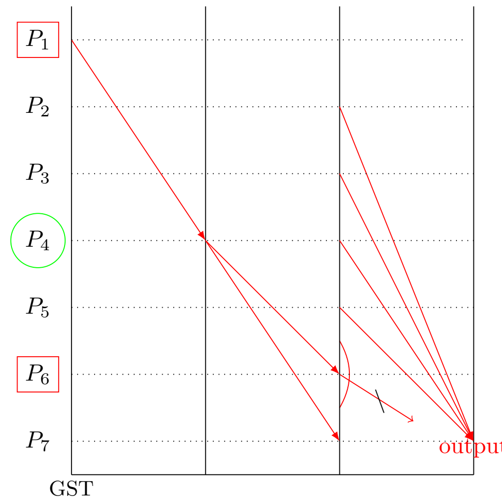
(d) Execution $\mathscr{E}_4$

without hearing from P_7, since P_7 could possibly be honest and possibly have decided, either based on one Fhlq with nr only (as in $\mathscr{E}_2$), or, based on a Fhlq with rec only (as in $\mathscr{E}_6$).

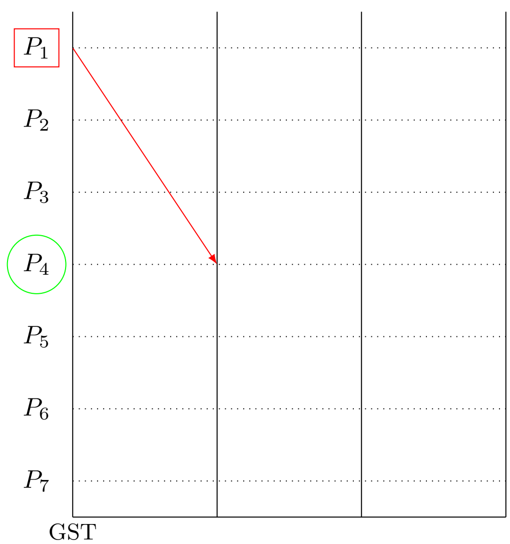
Backup Decision. P assembles the testimonies, and appends its signature on the whole, which forms what we denote as a *Collective Narrative*. Then, players reach consensus on one unique collective narrative, denoted cN. Either all Fhlq contained (possibly none) in cN vouch for the same value v, in which case players decide it. Otherwise, it means that



(e) Execution $\mathscr{E}_5$



(f) Execution $\mathscr{E}_6$



(g) Execution $\mathscr{E}_7$

Fig. 1. A half circle oriented to the right denotes sending a message to all which, unless further specification, are all timely delivered. However, when there is also a crossed arrow out of the circle, then it means that no message is sent to this specific player, *e.g.* in $\mathscr{E}_6$ player P_6 deliberately does not send a message to P_7 in the 3^{rd} round. Arrows with a perpendicular head denote messages delayed until GST, *e.g.* in $\mathscr{E}_3$, we have specifically that the message from P_2 to P_7 is delayed, although all other messages, depicted by the red half circle to the right from P_2, are timely delivered. The red color denotes messages following, and reporting, reception of m by P_4, *i.e.* rec. For instance, in $\mathscr{E}_3$, we have P_3 send rec messages to all, except to P_7, depicted by a black line, to which it sends a nr message. Let us be concrete, only for illustration. In all rec messages , since we assume wlog a full information protocol, we have in particular m which is attached. Messages of type rec are either sent directly to P_4 or indirectly, forwarded, as a declaration of P_4 reporting having received m, with the signed m enclosed in the forwarded report.

cN exhibits a proof of corruption of some player Q, and that cN contains testimonies from all three honest players. Players thus can trust each other's testimonies, thus can trivially extract from them a value which is safe to decide.

4.1.2 Choice Functions. We consider any public predetermined function $f : \mathscr{V}^3 \cup \mathscr{V}^4 \to \mathscr{V}$, denoted "(Strong Unanimity) Choice Function", satisfying that, for any $v \in \mathscr{V}$, we have $f(v_{i_1}, v_{i_2}, v_{i_3}) = v$ as soon as two inputs $v_{i_j} \in \mathscr{V}$ are equal to v, and such that $f(v_1, v_2, v_3, v_4) = v$ as soon as three inputs are equal to v. In particular, we have that if f is applied to 3 or 4 values reported by 3 or 4 players as their input, then, in the situation where all 3 honest players have the same input value $v \in \mathscr{V}$, then f outputs this value, even if one out of the (three or four) input values to f is issued by a malicious player. We abuse notations such that, if, e.g. $v_2 = \perp$, then we denote $f(v_1, v_2, v_3, v_4) := f(v_1, v_3, v_4)$.

We consider any public predetermined function $g : \mathscr{V}^3 \to \mathscr{V}$, denoted "(Weak Unanimity) Choice Function", satisfying that when all three inputs are equal, to some $v \in \mathscr{V}$, then g outputs v. In particular, we have that if g is applied to 3 values reported by the three honest players to be their input, then, in the situation where all 3 honest players have the same input value $v \in \mathscr{V}$, then g outputs this value. Notice that g could be set as f, we juste define it separately to highlight the weaker property required by g.

4.1.3 Auxilliary Consensus with External Validity $\mathsf{CEV}_{\mathsf{cN}}$. In order to reach consensus on one unique cN in the backup decision, players use an auxiliary protocol. Precisely, they use any protocol for consensus with external validity under partial synchrony, in the sense of Definition 1, e.g., [CL99]. The validity predicate of this consensus is specified to return true, if and only if, the input bitstring is a Collective Narrative signed by one of the players. We denote $\mathsf{CEV}_{\mathsf{cN}}$ this consensus protocol.

4.1.4 Normal Case: Types of Messages. For any $P \in \mathscr{P}$, we denote (report, P, v) *Report issued by* P a message consisting of: the string of characters "report", some valid value $v \in \mathscr{V}$, and appended with a signature of P on the whole.

For any $P \in \mathscr{P}$, we denote $(\mathsf{forward}, P, (\mathsf{r}_1, \mathsf{r}_2, \mathsf{r}_3, \mathsf{r}_4))$ *Forward issued by* P a message consisting of: the string of characters "forward", and four objects $(\mathsf{r}_i)_{i\in[4]}$, each of which is either $\perp$ or a report $(\mathsf{report}, P_i, v_i)$ from P_i; appended with a signature of P on the whole.

For any $P \in \mathscr{P}$, we denote $(\mathsf{sof}, P, (\mathsf{f}_1, \mathsf{f}_2, \mathsf{f}_3, \mathsf{f}_4))$ *Set of Forwards issued by* P a message consisting of: the string of characters "sof", and four objects $(\mathsf{f}_i)_{i\in[4]}$, each of which is either $\perp$ or a forward from P_i; appended with a signature of P on the whole.

4.1.5 Particular Case: (Set of) Forwardings from a Honest-Looking Quorum. Consider a SoF $\mathsf{s}_P = (\mathsf{sof}, P, (\mathsf{f}_1, \mathsf{f}_2, \mathsf{f}_3, \mathsf{f}_4))$ issued by some P, that furthermore satisfies the following. We require that there exists at least one quorum $\mathcal{Q}$, containing P, which we assume for simplicity of the presentation $\mathcal{Q} = \{P_1, P_2, P_3\}$, such that:

(i) $\forall i \in \{1, 2, 3\}$, f_i is a forward message from P_i, i.e., none of them is $\perp$;

(ii) $\forall Q \in \mathcal{Q}$, all three $\left(\mathsf{r}_Q^{(i)} \in \mathsf{f}_i\right)_{i\in\{1,2,3\}}$, are non-$\perp$ and equal (to a report from Q).

(iii) $\forall Q \in \mathcal{Q}$ we have furthermore that, if the fourth one $\mathsf{r}_Q^{(4)} \in \mathsf{f}_4$ is non-$\perp$ (which implies $\mathsf{f}_4 \neq \perp$), then it is also equal to all previous three ones $\left(\mathsf{r}_Q^{(i)} \in \mathsf{f}_i\right)_{i\in\{1,2,3\}}$.

When the case, we make the following notations. For each $Q \in \mathscr{Q}$, we denote v_Q the common value in all three (or four, in case (iii)) reports $\mathsf{r}_Q^{(i)} \in \mathsf{f}_i$ for $i \in \{1, 2, 3\}$ (and possibly 4), as above. We now define the notation v_4 ($P_4 \notin \mathcal{Q}$). If both the following hold:

(α) there is at least one forward f_i for $i \in \{1, 2, 3\}$ such that the $\mathsf{r}_4^{(i)} \in \mathsf{f}_i$ is not $\perp$, i.e., is a report from P_4;

(β) among the $\left(\mathsf{r}_4^{(i)}\right)_{i\in\{1,2,3\}}$, the ones which are not $\perp$, i.e., are reports of P_4, are all equal;

then we denote v_4 the unique value in $\mathscr{V}$ reported as input in all the $\big(\mathsf{r}_4^{(i)}\big)_{i\in\{1,2,3\}}$ that are not $\perp$. Else, we set $v_4 := \perp$. Thus by definition, we have $v_4 := \perp$ when either [not-(α)]: all $(\mathsf{r}_4^{(i)} \in \mathsf{f}_i)_{i\in\{1,2,3\}}$ are $\perp$ or [not-(β)]: there exists two indices $i \neq j$ out of $\{1,2,3\}$ such that $(\mathsf{r}_4^{(i)} \in \mathsf{f}_i)$ and $(\mathsf{r}_4^{(j)} \in \mathsf{f}_j)$ are both non-$\perp$ and not equal.

Importantly, in (β), we *do not put any condition* on the $\mathsf{r}_4^{(4)} \in \mathsf{f}_4$. Indeed, making v_4 dependent on the $\mathsf{r}_4^{(4)} \in \mathsf{f}_4$ would have enabled a malicious P_4 to provoke inconsistent Fhlq*s* between the members of $\mathcal{Q}$.

Definition 15 (Fhlq*).* Consider any quorum, which we assume $\mathcal{Q} = \{P_1, P_2, P_3\}$ without loss of generality. Consider any $P \in \mathcal{Q}$, and any set of forwards $\mathsf{s} = \big(\mathsf{sof}, P, (\mathsf{f}_1, \mathsf{f}_2, \mathsf{f}_3, \mathsf{f}_4)\big)$ which satisfies the above conditions (i), (ii) and (iii). We then say that s *constitutes a (Set of) Forwardings from Honest-Looking Quorum* $\mathcal{Q}$ (Fhlq) issued by P, or, equivalently that "Fhlq from $\mathcal{Q}$ can be extracted from s", and denote it as:

$$\mathsf{s} \longrightarrow \mathsf{Fhlq}[P](\mathsf{f}_1, \mathsf{f}_2, \mathsf{f}_3)(v_1, v_2, v_3, v_4)\,. \tag{1}$$

With the $(v_i)_{i\in[4]}$ defined as above. Finally, we say that the values (v_1, v_2, v_3, v_4) are *associated to* Fhlq, and that Fhlq *Vouches for* the value $f(v_1, v_2, v_3, v_4)$.

Notice that the same set of forwards $\mathsf{s}_P = \big(\mathsf{sof}, P, (\mathsf{f}_1, \mathsf{f}_2, \mathsf{f}_3, \mathsf{f}_4)\big)$ can possibly constitute a Forwardings from a Honest-Looking Quorum for both two different quorums $\mathcal{Q} \neq \mathcal{Q}'$. When the case, then Lemma 18 will guarantee that the four values $(v_i)_{i\in\{1,\ldots,4\}}$ are the same in both Fhlq.

4.1.6 (Normal-Case) Decision Certificates.

Definition 16. Consider any quorum $\mathcal{Q}$, which we assume for simplicity $\mathcal{Q} = \{P_1, P_2, P_3\}$. Consider the data of sets of forwards $(\mathsf{SoF}_i)_{i\in\{1,2,3\}}$ issued by all three players in $\mathcal{Q}$, satisfying furthermore the following conditions:

(iv) each of them is furthermore a Forwardings from Honest-Looking Quorum $\mathcal{Q}$ (Fhlq) (Definition 15);

(v) to all those three Fhlq are associated the same values (v_1, v_2, v_3, v_4).

In particular, all three Fhlqs vouch for the same value $v = f(v_1, v_2, v_3, v_4)$. We say that the whole constitutes a *Decision Certificate* for v.

4.1.7 Backup Decision: Testimonies, (Honest-Looking) Collective Narrative and Proofs of Corruption. For any $P \in \mathscr{P}$, we denote $\big(\mathsf{testimony}, P, *\big)$ *Testimony issued by* P a message consisting of: the string of characters "testimony", and of an object $*$ which is either a report issued by P, a Fhlq issued by P, or a decision certificate; appended with a signature of P on the whole.

For any player $Q \in \mathscr{P}$, we denote as a *Proof of Corruption* of Q, the data of two different reports from Q or two different forwards from Q or two different Fhlqs issued by Q.

Definition 17 (cN*).* We denote as a *(Honest-Looking) Collective Narrative* (cN) the data of testimonies issued at least three players, such that, if they come from three players only, then *Either* no two Fhlq present in them (if any) vouch for different decision values, *Or* the three Testimonies together exhibit a proof of corruption for the fourth player not in cN.

4.2 Protocol

1^{st} round Every $P \in \mathscr{P}$, on input v, sends $\big(\mathsf{report}, P, v\big)$ to all players.

2^{nd} round Every $P \in \mathscr{P}$, if it received report messages from at least 3 distinct players in the 1^{st} round, sends $\big(\mathsf{forward}, P, (\mathsf{r}_1, \mathsf{r}_2, \mathsf{r}_3, \mathsf{r}_4)\big)$ to all, r_i is the report received from π or $\perp$ if no report was received from π.

3^{rd} round Every $P \in \mathscr{P}$, if there exists at least one quorum $\mathcal{Q} \ni P$ such that, assuming without loss of generality $\mathcal{Q} = \{1, 2, 3\}$: P received forward messages $(\mathsf{f}_i)_{i \in \{1,2,3\}}$ from all players in $\mathcal{Q}$ in the 1^{st} round, and these messages constitute a Fhlq for $\mathcal{Q}$, i.e. conditions (i), (ii) and (iii) are satisfied. Then P sends to all one unique message: $\big(\mathsf{sof}, P, (\mathsf{f}_1, \mathsf{f}_2, \mathsf{f}_3, \mathsf{f}_4)\big)$, where the $(\mathsf{f}_i)_{i \in \{1,2,3,4\}}$ are the Forwards received, or possibly $\mathsf{f}_4 = \bot$ if no forward was received from P_4.

End of 3^{rd}round Every $P \in \mathscr{P}$, upon receiving any three sets of forwards $(\mathsf{SoF}_{i_1}, \mathsf{SoF}_{i_2}, \mathsf{SoF}_{i_3})$ constituting a decision certificate for some value v, outputs v.

We now describe an additional set of instructions, which start from the beginning of the 4^{th}round. They enable players to output even if $GST > 0$.

Testimonies Every $P \in \mathscr{P}$ forms te $=\big(\mathsf{testimony}, P, *\big)$, where $*$ is: a decision certificate, if P received some in the Decision step; else: a set of forwards, if P sent some in the 3^{rd}round; else $* = r_P$ the report that P sent in the 1^{st}round. P repeatedly sends te to all in every round until instructed to stop, below.

Collect testimonies P waits until it receives enough testimonies to form a collective narrative, denoted cN_P, which it then inputs into CEV.

Backup Decision Upon receiving the output from CEV, denoted cN, P stops sending testimonies and decides as follows:

(a) *If* cN contains a decision certificate for some value v, then: decide v;

(b) *Else if* cN contains no proof of corruption *nor* any Fhlq, then: consider the set of reports contained in cN: there are at least three from distinct players, e.g., $\{1, 2, 3\}$. Denote (v_1, v_2, v_3, v_4) the input values which they report, where for the fourth index we set $v_4 := \bot$ if no report from P_4 is enclosed in cN. Output $f(v_1, v_2, v_3, v_4)$.

(c) *Else if* cN exhibits a proof of corruption for a player, e.g., P_4, then by definition of a cN it must contain reports from the 3 remaining players. Output $g(v_1, v_2, v_3)$

(d) *Else*, in particular it must be the case that cN contains at least a Fhlq. Then: choose any such Fhlq and decide the value v that it vouches for.

4.3 Proof of Termination, and of Normal case termination in 3 rounds

4.3.1 Normal case termination in 3 *rounds.* Consider the quorum of the honest players, wlog. $\mathcal{Q} = \{P_1, P_2, P_3\}$. If $GST = 0$, then at the end of round 3, each player receives $(\mathsf{SoF}_1, \mathsf{SoF}_2, \mathsf{SoF}_3)$, which all must be forwardings from honest looking $\mathcal{Q}$ (Fhlq), since $\mathcal{Q}$ is honest. All we have to show is the remaining condition (v), i.e., that all three are associated to the same set of values (v_1, v_2, v_3, v_4), as defined in Def 15. But this is guaranteed by the fact the the value v_4 taken in consideration in all three SoF*s*, does *not* depend on the possible forwarding(s), from P_4, as stressed by the remark below Definition 15. Thus, since all the forwardings from players in $\mathcal{Q}$ included in all three SoF*s* are identical, the sets of values (v_1, v_2, v_3, v_4) associated to all three of them are the same.

4.3.2 Termination in finite time after GST. All we have to show is that every player P is ensured to collect enough testimonies to form a cN. Indeed, if P receives only three testimonies, which exhibit a proof of corruption for some of the three senders, then, P is guaranteed to receive a testimony from the fourth player Q, since Q must be honest **since $t = 1$**.

4.4 Proof of Strong Unanimity

Throughout, strong unanimity is guaranteed by the application of the choice function f on at least three reported inputs. The only exception is in case (c), but there we are considering the inputs of three players which must necessarily be the honest ones, so application of the "weak unanimity" choice function g is enough.

4.5 Proof of Consistency

First we prove that two honest players deciding by the Normal Case rule, decide the same value. We then show that two honest players deciding by the Backup Decision rule, decide the same value. We finally show that two honest players, one deciding by the Normal Case rule and the other one by backup, decide the same value.

Consistency Between any two (Normal Case) Decision Certificates. Consider any two decision certificates. Each of them consists of three Fhlqs issued by distinct senders, each of them encapsulating one SoF. In particular, the six senders of these SoFs (with repetitions) intersect in at least one honest player, which has issued a *single* SoF. Thus, all what remains to show is:

Lemma 18. *From any* SoF*, it cannot possibly be extracted two* Fhlq*s vouching for conflicting values.*

Proof. Denote $\mathcal{Q}$ and $\mathcal{Q}'$ the quorums corresponding to Fhlq and Fhlq', and (v_1, v_2, v_3, v_4) and (v'_1, v'_2, v'_3, v'_4) the values associated to them. If $\mathcal{Q}' = \mathcal{Q}$, then by definition, since the underlying SoF is the same, the same four values must be associated. Thus in what follows we assume $\mathcal{Q} \neq \mathcal{Q}'$. Wlog. we assume that $\mathcal{Q} = \{P_1, P_2, P_3\}$ and $\mathcal{Q}' = \{P_1, P_2, P_4\}$ and that (at least) the player P_1 in their intersection is honest. Let us show that all four values appear identically in both Fhlqs.

- v_1*:* Since P_1 is honest, it issues only one report. Furthermore this report appears in both Fhlqs, since $P_1 \in \mathcal{Q} \cap \mathcal{Q}'$.

- v_2*:* Since P_1 is honest, it issues only one forward, in particular both fowards from P_1 corresponding to the two Fhlqs are the same, thus they contain the same report r_2 from P_2. Since P_2 is in both quorums, the value v_2 associated to both is thus non-$\perp$ in both (said otherwise, by condition (ii)). Thus the value v_2 enclosed in r_2, is associated to both Fhlqs.

- v_4*:* Since $P_1 \in \mathcal{Q}'$, the unique forward issued by P_1, included in the SoF, contains a report r_4 from P_4. In particular (α) is satisfied for P_4 with respect to $\mathcal{Q}$. Furthermore, since P_4 is in $\mathcal{Q}'$, we must have that v'_4 is equal to the value enclosed in this r_4. Combining the two facts above, the only possibility for $v_4 \neq v'_4$ would be (β) be not satisfied for $\mathcal{Q}$ (which would imply $v_4 = \perp$). Thus, we would have that the SoF contains one forward $\mathsf{f}^{(c)}$ containing a report $\mathsf{r}^{(c)}$ from P_4 different from r_4 (the (c) stands for "conflicting"). Thus, by (ii) or (iii) applied to $\mathcal{Q}'$ (depending on if the issuer of $\mathsf{f}^{(c)}$ is in $\mathcal{Q}'$ or not), the SoF cannot constitute a Fhlq for $\mathcal{Q}'$.

- v_3*:* The argument is symmetrical to the one for v_4. □

Consistency Between two Backup Decisions. Since the CEV returns the same cN for all players, all honest players are in the same case: (a), (b), (c) or (d). In the first three cases, the decision does not depend on free choice, so no two decisions can be different. It thus remains to show that in the last case (d), decision does not depend on the free choice of the Fhlq.

Lemma 19. *If some player is in the situation of the last item (d), in particular not in the situation of all previous items, in particular not (c), then no two* Fhlq*s contained in* cN *vouch for conflicting values.*

Proof. Let us prove the contraposite. Namely, consider two Fhlqs produced in the execution which vouch for conflicting values, then, we *Claim* that, together, they constitute a proof of corruption for some player. The *Claim* implies that, if we are not in the situation of (a) nor (b), but cN contains conflicting Fhlqs, then we must be in case (c), which concludes the Lemma.

Proof of the Claim: Denote $\mathcal{Q}$ and $\mathcal{Q}'$ the quorums corresponding to Fhlq and Fhlq', and (v_1, v_2, v_3, v_4) and (v_1', v_2', v_3', v_4') the values associated to them. Both quorums intersect at at least one honest player. Wlog we can assume this player to be P_1, and $\mathcal{Q} = \{P_1, P_2, P_3\}$.

First case: $\mathcal{Q} = \mathcal{Q}'$. Then, if we do not have $(v_1, v_2, v_3) = (v_1', v_2', v_3')$, it must be the case that the two Fhlq contain conflicting reports from some player in $\mathcal{Q}$, constituting a proof of corruption. Suppose now that $v_4 \neq v_4'$. The only possibility for not having a proof of corruption of P_4, is that condition (α) is not satisfied for either $\mathcal{Q}$ or $\mathcal{Q}'$. But this would imply that some player P_i in $\{P_2, P_3\}$ forwards a report from P_4 in Fhlq but not in Fhlq′, which constitutes a proof of corruption of P_i

Second case: $\mathcal{Q} \neq \mathcal{Q}'$, wlog. one can assume $\mathcal{Q}' = \{P_1, P_2, P_4\}$. Since P_1 is honest it issues a unique forward and, since $P_1 \in \mathcal{Q} \cap \mathcal{Q}'$, this forward contains reports from all four players. Thus, in particular, we are not in case (α) for neither $P_4 \notin \mathcal{Q}$ nor $P_3 \notin \mathcal{Q}'$. Thus, the only possibility for not having $(v_1, v_2, v_3, v_4) = (v_1', v_2', v_3', v_4')$ is the existence of conflicting reports from some player, hence a proof of corruption. □

Consistency between a Normal Case Decision and a Backup Decision. Suppose that we are in case (a), then by unicity of the Normal Case decision, proven above (essentially Lemma 18), the two decided values are the same.

Suppose that we are in case (b), then no Normal Case decision could have happened, since otherwise at least one honest player in the three issuers of testimonies would have reported sending a Fhlq.

Suppose that we are in case (c), then the cN contains testimonies from all three honest players. None of them could have decided in the normal case, since otherwise it would have exhibited a decision certificate in its testimony and thus we would have been in case (a).

Suppose that we are in case (d). Consider v a value vouched by one of the Fhlqs present in the cNs. No other normal decision could have happened for a value $v' \neq v$. Indeed, otherwise at least a Fhlq vouching for v' would have been reported in one of the three testimonies in cN. This would then contradict the unicity of v, proven in Lemma 19.

REFERENCES

[Abr+17] Ittai Abraham, Guy Gueta, Dahlia Malkhi, Lorenzo Alvisi, Ramakrishna Kotla, and Jean-Philippe Martin. "Revisiting Fast Practical Byzantine Fault Tolerance". In: *CoRR* abs/1712.01367 (2017). eprint: 1712.01367. URL: http://arxiv.org/abs/1712.01367.

[Abr+18] Ittai Abraham, T-H. Hubert Chan, Danny Dolev, Kartik Nayak, Rafael Pass, Ling Ren, and Elaine Shi. *Communication Complexity of Byzantine Agreement, Revisited (PODC'19 long version).* 2018. eprint: 1805.03391.

[Abr+20] Ittai Abraham, Kartik Nayak, Ling Ren, and Nibesh Shrestha. "On the Optimality of Optimistic Responsiveness". In: *CCS*. 2020.

[AD15] Ittai Abraham and Danny Dolev. "Byzantine Agreement with Optimal Early Stopping, Optimal Resilience and Polynomial Complexity". In: *STOC*. 2015.

[AMS19] Ittai Abraham, Dahlia Malkhi, and Alexander Spiegelman. "Asymptotically Optimal Validated Asynchronous Byzantine Agreement". In: *PODC*. 2019.

[ANRX21] Ittai Abraham, Kartik Nayak, Ling Ren, and Zhuolun Xiang. "Good-Case Latency of Byzantine Broadcast: A Complete Categorization". In: *PODC*. 2021.

[BGP92] Piotr Berman, Juan A. Garay, and Kenneth J. Perry. "Optimal early stopping in distributed consensus". In: *Distributed Algorithms*. Berlin, Heidelberg: Springer Berlin Heidelberg, 1992.

[Bra+01] Francisco Brasileiro, Fabíola Greve, Achour Mostefaoui, and Michel Raynal. "Consensus in One Communication Step". In: *Parallel Computing Technologies*. Ed. by Victor Malyshkin. Berlin, Heidelberg: Springer Berlin Heidelberg, 2001, pp. 42–50. ISBN: 978-3-540-44743-6.

[Can04] R. Canetti. "Universally composable signature, certification, and authentication". In: *Proceedings. 17th IEEE Computer Security Foundations Workshop, 2004.* 2004.

[CKPS01] Christian Cachin, Klaus Kursawe, Frank Petzold, and Victor Shoup. "Secure and Efficient Asynchronous Broadcast Protocols". In: *CRYPTO*. 2001.

[CL99] Miguel Castro and Barbara Liskov. "Practical Byzantine Fault Tolerance". In: *Proceedings of the Third Symposium on Operating Systems Design and Implementation.* OSDI '99. 1999.

[Cle+09] Allen Clement, Manos Kapritsos, Sangmin Lee, Yang Wang, Lorenzo Alvisi, Mike Dahlin, and Taylor Riché. "UpRight Cluster Services". In: *In Proc. of 22nd SOSP, Big Sky*. 2009.

[CM16] Jing Chen and Silvio Micali. *Algorand*. 2016. eprint: 1607.01341.

[Coh+19] Ran Cohen, Iftach Haitner, Nikolaos Makriyannis, Matan Orland, and Alex Samorodnitsky. "On the Round Complexity of Randomized Byzantine Agreement". In: *33rd International Symposium on Distributed Computing, DISC 2019, October 14-18, 2019, Budapest, Hungary*. 2019, 12:1–12:17.

[CS06] B. Charron-Bost and A. Schiper. "Improving Fast Paxos: being optimistic with no overhead". In: *12th Pacific Rim International Symposium on Dependable Computing (PRDC'06)*. 2006.

[DG02] Partha Dutta and Rachid Guerraoui. "The Inherent Price of Indulgence". In: *PODC*. 2002.

[DGV05] Partha Dutta, Rachid Guerraoui, and Marko Vukolić. *Best-case complexity of asynchronous Byzantine consensus*. Tech. rep. EPFL, 2005.

[DL13] Danny Dolev and Christoph Lenzen. "Early-deciding Consensus is Expensive". In: *PODC*. 2013.

[DLS88] Cynthia Dwork, Nancy Lynch, and Larry Stockmeyer. "Consensus in the Presence of Partial Synchrony". In: *J. ACM* (1988).

[DRS82] D. Dolev, R. Reischuk, and H. R. Strong. "'Eventual' is earlier than 'immediate'". In: *FOCS*. 1982.

[DS06] D. Dobre and Neeraj Suri. "One-step Consensus with Zero-Degradation". In: *DSN*. 2006.

[DS83] Danny Dolev and H. Strong. "Authenticated Algorithms for Byzantine Agreement". In: *SIAM J. Comput.* (1983).

[FMR04] R. Friedman, A. Mostefaoui, and M. Raynal. "Simple and efficient oracle-based consensus protocols for asynchronous Byzantine systems". In: *SRDS*. 2004.

[Gol+19] G. Golan Gueta, I. Abraham, S. Grossman, D. Malkhi, B. Pinkas, M. Reiter, D. Seredinschi, O. Tamir, and A. Tomescu. "SBFT: A Scalable and Decentralized Trust Infrastructure". In: *DSN*. 2019.

[GQV08] Rachid Guerraoui, Vivien Quema, and Marko Vukolić. "The next 700 BFT protocols". In: *EPFL Technical Report LPD-REPORT-2008-008* (2008).

[GR04] R. Guerraoui and M. Raynal. "The information structure of indulgent consensus". In: *IEEE Transactions on Computers* (2004).

[Guo+22] Bingyong Guo, Yuan Lu, Zhenliang Lu, Qiang Tang, Jing Xu, and Zhenfeng Zhang. "Speeding Dumbo: Pushing Asynchronous BFT Closer to Practice". In: *NDSS*. 2022.

[GV10] Rachid Guerraoui and Marko Vukolić. "Refined quorum systems". In: *Distributed Computing* (2010).

[Kot+09] Ramakrishna Kotla, Lorenzo Alvisi, Michael Dahlin, Allen Clement, and Edmund L. Wong. "Zyzzyva: Speculative Byzantine Fault Tolerance". In: *ACM Trans. Comput. Syst.* 27 (2009).

[KR03] Idit Keidar and Sergio Rajsbaum. "On the Cost of Fault-Tolerant Consensus When There Are No Faults - A Tutorial". In: *LADC*. 2003.

[Kur02] K. Kursawe. "Optimistic Byzantine agreement". In: *SRDS*. 2002.

[Lam06a] Leslie Lamport. "Fast Paxos". In: *Distributed Computing* (2006). ISSN: 1432-0452.

[Lam06b] Leslie Lamport. "Lower bounds for asynchronous consensus". In: *Distributed Computing* (2006).

[LSP82] Leslie Lamport, Robert Shostak, and Marshall Pease. "The Byzantine Generals Problem". In: *ACM Trans. Program. Lang. Syst.* (1982).

[MA05] J.-P. Martin and L. Alvisi. "Fast Byzantine consensus". In: *DSN*. 2005.

[NIK10] Nazreen Banu, T. Izumi, and Koichi Wada. "Doubly-expedited one-step Byzantine consensus". In: *DSN*. 2010.

[PS18] Rafael Pass and Elaine Shi. "Thunderella: Blockchains with Optimistic Instant Confirmation". In: *EUROCRYPT*. 2018.

[PS98] Fernando Pedone and André Schiper. "Optimistic Atomic Broadcast". In: *Distributed Computing*. Ed. by Shay Kutten. Berlin, Heidelberg: Springer Berlin Heidelberg, 1998.

[SK19] Nibesh Shrestha and Mohan Kumar. *Revisiting EZBFT: A Decentralized Byzantine Fault Tolerant Protocol with Speculation*. 2019. eprint: 1909.03990.

[SKD19] Nibesh Shrestha, Mohan Kumar, and SiSi Duan. "Revisiting hBFT: Speculative Byzantine Fault Tolerance with Minimum Cost". In: *CoRR* abs/1902.08505 (2019). URL: http://arxiv.org/abs/1902.08505.

[SR08] Yee Jiun Song and Robbert van Renesse. "Bosco: One-Step Byzantine Asynchronous Consensus". In: *DISC*. 2008.

[Yin+19] Maofan Yin, Dahlia Malkhi, Michael K. Reiter, Guy Golan Gueta, and Ittai Abraham. "HotStuff: BFT Consensus with Linearity and Responsiveness". In: *PODC*. 2019.

[Zie06] P. Zielinski. "Optimistically Terminating Consensus: All Asynchronous Consensus Protocols in One Framework". In: *2006 Fifth International Symposium on Parallel and Distributed Computing*. 2006.