Asynchronous reference frame agreement in a quantum network

Tanvirul Islam[1, 2, 3, *] and Stephanie Wehner[2, 3, †]

[1]*School of Computing, National University of Singapore, 13 Computing Drive, 117417 Singapore*
[2]*Centre for Quantum Technologies, National University of Singapore, 3 Science Drive 2, 117543 Singapore*
[3]*Qutech, Delft University of Technology, Lorentzweg 1, 2628 CJ Delft*

An efficient implementation of many multiparty protocols for quantum networks requires that all the nodes in the network share a common reference frame. To establish such a reference frame from scratch is especially challenging in an asynchronous network where network links might have arbitrary delays and the nodes do not share synchronised clocks. In this work, we study the problem of establishing a common reference frame in an asynchronous network of n nodes of which at most t are affected by arbitrary unknown error, and the identities of the faulty nodes are not known. We present a protocol that allows all the correctly functioning nodes to agree on a common reference frame as long as the network graph is complete and not more than $t < n/4$ nodes are faulty. As the protocol is asynchronous, it can be used to synchronise clocks over a network. Also, the protocol has the appealing property that it allows any existing two-node protocol for asynchronous reference frame agreement to be lifted to a robust protocol for a quantum network.

PACS numbers:

I. INTRODUCTION

To use quantum cryptography on a global scale one must first have a functioning quantum internet [1]. Recently this necessity has inspired a lot of effort in the research and development of satellite [2–6], and ground based [7–9] quantum networks. The possible applications of such networks are not restricted to only cryptography. A fully general quantum network will allow us to perform general distributed quantum computing [10–12].

In this work, we study problems related to initialisation and construction of quantum networks. More specifically we study how well n nodes in an asynchronous quantum network can agree on a reference frame in the presence of at most t arbitrarily faulty nodes among them. By asynchronous network we mean in this setting we do not require the nodes to share a clock to begin with, and the channel delays can be arbitrarily in each use. In fact, an asynchronous protocol only assumes any message sent from a correct node to a correct node will eventually reach the destination, without imposing any bound on the channel delay. This assumption captures the most general reference frame agreement problem in a quantum network because during the initialisation of the network the pairwise channel delays might be unknown, clocks might not be synchronised and spatial reference frames might be unaligned.

In a quantum channel the qubits are encoded in some physical degree of freedom. For example, polarisation direction of photon is often used to encode qubits. Which requires the sender and receiver to agree on some set of orthonormal directions as their common spatial reference frame. Another example is the time-bin qubits were both of the parties requires synchronised clocks. That is they must have a pre-agreed temporal reference frame. So far these reference frame agreement problems are studied in a bipartite setting [13–19] with the exception of [20] where spatial direction are agreed on in a *synchronised* network of n nodes. Specially in [20] it is assumed that the network is synchronous that is all the nodes of the network have a shared clock and all the link delays have known upper bound. The bipartite reference frame agreement problem have been studied extensively. For example, many of the known results are discussed in the review article [21]. But agreeing on a reference frame in an asynchronous network of n nodes remained open.

One advantage of having asynchronous reference frame agreement protocol for a network with certain number of faulty nodes is that once a spatial reference frame is established, then new robust protocols can potentially be built on top of it to perform network-wide clock synchronisation. This is a task important by itself with various application in security, navigation and finance etc. [22]. The primary difficulty of executing any protocol in an asynchronous network comes from the fact that in presence of incorrect, that is, arbitrarily faulty nodes it is impossible to decide for an correct receiver whether a message is not arriving because the sender is faulty and not sending anything at all, or if the sender is correct but the channel is taking a very long time to transfer the message. It is nontrivial to decide how long to wait for a message before moving on to the next step of a protocol.

Another difficulty is that unlike in classical information theory where information can be represented in bits, a reference frame can only be transferred from scratch by exchanging systems which have an inherent sense of direction. Examples of such systems are spin qubits and photon polarisation qubits. The receiver can only extract direction information from these systems by performing tomography on them. While preparing the direction any node P_i will know the description of the direction as a

*tanvir@locc.la
†steph@locc.la

vector v_i in its local frame. Once the quantum system carrying that direction arrives to a receiver P_j, the receiver constructs a representation of the direction in it's own local frame as v_j. This inevitably introduces some error even in correct transmissions. That is, depending on the precision of the instruments one can only expect to have $d(v_i, v_j) \leq \delta$ for some $\delta > 0$. But this distance metric does not make sense as it is, because v_i and v_j are vector representations in two different local frames. So we must redefine our distance metric $d(.,.)$ where distance is computed by converting both vectors in the frame of the first argument. As a result $d(v_i, v_j)$ remains a valid distance measure even though P_i and P_j do not know each other's local frame. This computation of distance between two vectors of different reference frames is only done in the analysis of the protocol and not by the nodes while playing the protocol. Any distance computed by a node inside a protocol is only between vectors for which it has a representation in its local frame. The inherent imperfection of message transmission must be accounted for by any reference frame agreement protocol. We capture this in the definition as,

Definition 1. For $\eta > 0$, a protocol in an asynchronous network of n nodes is a *η-asynchronous reference frame agreement protocol* if it satisfies the following conditions.

Termination. Every correct node P_i eventually terminates and outputs a direction v_i.

Correctness. If correct node P_i outputs v_i and correct node P_j outputs v_j then $d(v_i, v_j) \leq \eta$.

But we have to achieve these termination and correctness condition in the presence of incorrect or faulty nodes. As it is unknown which nodes are faulty this resembles the Byzantine fault tolerance model [23] studied in from classical distributed computing. For quantum networks our assumptions are,

- The pairwise channels are *public*. That is, the messages are not secret. As a result an adversary can see the content of a message between two correct nodes and adapt its strategy accordingly.
- The pairwise channels are authenticated. That is, if a correct node sends a message to another correct node the message cannot be altered by the adversary. But there can be channel noise and message might be slightly changed within a bound.
- The pairwise channel delays might be controlled by and faulty nodes. Not only the faulty nodes can see the content of each pairwise channel in the network they can control the channel delays, even between the correct nodes.
- If an honest node sends a message to another honest node the message eventually reaches the receiver. That is, even though the delay is controlled by the some adversary they cannot put infinite delay on a message between two correct nodes. But the delay is unbounded.
- The dishonest nodes might have correlated error, or might cooperate with each other to thwart the protocol. To create protocol which tolerates the worst kind of faults we also assume that the faulty nodes can cooperate with each other and have a global strategy to fault the protocol. This is a realistic assumption because some nodes might get controlled by an adversary.

II. RESULTS

In this work we design a protocol that can take any two party reference frame agreement protocol and lift it up to a fault tolerant multiparty reference frame agreement protocol. More specifically, we design the first protocol A-Agree which allows n nodes in fully connected asynchronous network to agree on a reference frame in the presence of $t < n/4$ faulty nodes. The result can be summarised in the following theorem.

Theorem 1. *In a complete network of n nodes that are pairwise connected by public quantum channels, if a bipartite δ-estimate direction protocol that uses m qubits to achieve success probability $q_{\text{succ}} \geq 1 - e^{-\Omega(m\delta^2)}$ is used, then protocol A-Agree is a 34δ-asynchronous reference frame agreement protocol with success probability at least $1 - e^{-\Omega(m\delta^2 - \log n)}$, that can tolerate up to $t < n/4$ faulty nodes.*

The problem of both synchronous an asynchronous agreement on classical bits in the presence of arbitrary faulty node is extensively studied in classical literature as Byzantine agreement problem [23]. But we should emphasise that, any classical protocol cannot be used in our problem because firstly, unlike classical network, any communication of direction among correct nodes in a quantum network will have inherent noises. As a result any classical protocol would see all the correct nodes as faulty nodes. And the protocol will fail. Secondly, one cannot use the classical protocol directly because one cannot represent a reference frame in classical bits [24]. But classical literature can still inform us on important questions like, how to achieve constant expected time, how to handle asynchronicity, etc. Some of the approaches of our protocol regarding these questions are influenced by [25]. We also use the interactive consistency protocol by Ben-Or et. [26] as a subroutine.

We design our protocol in two steps. First, we design a protocol AR-Cast for asynchronous broadcast, which we later use as a subroutine in our final asynchronous agreement protocol A-Agree.

A. Asynchronous broadcast

As the name suggests using this protocol a sender node can send some message to all the other nodes in an asynchronous network. At first sight a naive protocol of just sending the message to all other nodes one by one seems to be a valid protocol. But this naive protocol do not work if the sender intentionally sends different message

to different nodes, which can easily happen in networks with faulty nodes. To guard from it all the other nodes must communicate between each other to make sure they are receiving the same message, or a close approximation to it. But, as we have at most t faulty nodes, this verification also becomes tricky. The whole thing becomes more challenging because the network is not synchronous. As a result a receiver who is waiting for a message, cannot be certain whether to keep waiting (because the message might be taking a long time in the channel) or move on (because the sending node might be faulty and not sending the message at all). Our protocol takes care of all these challenges.

Formally the protocol is defined as,

Definition 2. For $\eta > 0$, $\zeta > 0$, a protocol which is initiated by a sender node P_s, in an asynchronous network of n nodes, is called a (η, ζ)-*asynchronous reference frame broadcast protocol* if it satisfies the following conditions.

Termination.

1. If the sender is correct then every correct node eventually completes the protocol.
2. If any correct node completes the protocol, then all the correct nodes eventually complete the protocol.

Consistency. If one correct node P_k outputs a direction v_k then all pairs of correct nodes P_i and P_j eventually output directions v_i, v_j where $d(v_i, v_j) \leq \eta$.

Correctness. If P_s is correct and broadcasts a direction u and if a correct node P_i outputs v_i then $d(u, v_i) \leq \zeta$.

We emphasize that the Termination condition of *asynchronous reference frame* ***broadcast*** is much weaker than the Termination condition of *asynchronous reference frame* ***agreement*** because in the broadcast protocol we do not require that the correct nodes complete the protocol if the sender is faulty.

We achieve this by our protocol AR-Cast. The following theorem summarises its properties.

Theorem 2. *In a complete network of n nodes that are pairwise connected by public quantum channels, if a bipartite δ-estimate direction protocol is used that succeeds with probability $q_{\text{succ}} \geq (1 - e^{-\Omega(n\delta^2)})$ then protocol AR-Cast is a 42δ-*asynchronous reference frame agreement protocol*, with success probability at least $1 - e^{-\Omega(m\delta^2 - \log n)}$ that can tolerate up to $t < n/4$ faulty nodes.*

The protocol AR-Cast works roughly as follows. It has several epochs. Which can be thought of as different state of the node running the protocol. In Epoch 0 the sender sends its intended direction to all as a [init] type message. In Epoch 1 all the nodes waits until they receive an [init] from sender or sufficient number of confirmation from other nodes that they have received some directions and proceeding to next epoch. This way, even if some correct node never receives an [init] message if the other correct nodes are advancing through the protocol this node in Epoch 1 will not stay behind waiting. In Epoch 2 the correct nodes who have decides upon a direction notifies the other nodes about its decision by sending ready_1 or ready_2 type messages to all. All these previous epochs make sure that all the correct nodes eventually arrives in Epoch 3 and outputs a direction which satisfies theorem 2. The protocol, and its analysis can be found in the supplemental material.

B. Asynchronous agreement

Our main protocol A-Agree uses AR-Cast as a subroutine and allows the correct nodes in an asynchronous network to agree on a reference frame.

This protocol also have several epochs (or states). In Epoch 0 of protocol A-Agree all the nodes proposes a direction which represents their local frame. They broadcast this direction using AR-Cast. All the correct nodes waits for at least $(3t + 1)$ such broadcasts to be complete. Then they enter Epoch 1. Since, there are $(3t + 1)$ correct nodes they will eventually arrive at Epoch 1. In this step all the correct nodes creates a bit string of length n where j'th bit represents if the j'th AR-Cast have been completed successfully in Epoch 0. Then all the nodes sends this bit string to all by playing Asynchronous-IC. The Asynchronous-IC is a classical protocol given by Ben-Or and El-Yaniv in [26]. After this they enters Epoch 2. In this Epoch every node has the same set of bit strings. They now look for the lowest inter k such that at least $(t + 1)$ bit strings has a 1 in that position. If they have completed that k'th AR-Cast they output their direction received from that broad cast. If they k'th AR-Cast is not complete for him they wait until it completes and then output. The election of k ensures that at least one correct node have completed the k'th AR-Cast so by Consistency of asynchronous reference frame broadcast all the honest nodes will eventually complete the k'th AR-Cast. This ensure that the A-Agree eventually completes. As, there is no loop in this protocol and all the subroutines run in constant expected time [28]. The A-Agree is also a constant expected time protocol.

III. CONCLUSION

In this work we have presented the first asynchronous reference frame agreement protocol. The synchronous protocol for spatial reference frame agreement presented in [20] can tolerate up to $t < n/3$ faulty nodes. Whereas, the asynchronous protocol we have presented tolerates only $t < n/4$ faulty nodes. Even though we pay this extra price in fault tolerance, an asynchronous protocol is a fully general reference frame agreement protocol. Because it can be used to synchronised clocks [27], which is an important problem in its own right. There are classical protocols for asynchronous agreement on bits which achieve $t < n/3$ in constant expected time, it remains open to see if this bound can be achieved by reference frame agreement protocols for a quantum network.

[1] H. J. Kimble, Nature **453**, 1023 (2008).
[2] M. Aspelmeyer, T. Jennewein, M. Pfennigbauer, W. Leeb, and A. Zeilinger, IEEE J. Sel. Topics Quantum Electron. **9**, 1541 (2003).
[3] C. Bonato, A. Tomaello, V. D. Deppo, G. Naletto, and P. Villoresi, New J. Phys. **11**, 045017 (2009).
[4] C.-Z. Peng, T. Yang, X.-H. Bao, J. Zhang, X.-M. Jin, F.-Y. Feng, B. Yang, J. Yang, J. Yin, Q. Zhang, et al., Phys. Rev. Lett. **94**, 150501 (2005).
[5] C. Bonato, M. Aspelmeyer, T. Jennewein, C. Pernechele, P. Villoresi, and A. Zeilinger, Opt. Express **14**, 10050 (2006).
[6] J. M. P. Armengol, B. Furch, C. J. de Matos, O. Minster, L. Cacciapuoti, M. Pfennigbauer, M. Aspelmeyer, T. Jennewein, R. Ursin, T. Schmitt-Manderbach, et al., Acta Astronaut. **63**, 165 (2008).
[7] M. Sasaki, M. Fujiwara, H. Ishizuka, W. Klaus, K. Wakui, M. Takeoka, S. Miki, T. Yamashita, Z. Wang, A. Tanaka, et al., Opt. Express **19**, 10387 (2011).
[8] J. I. Cirac, P. Zoller, H. J. Kimble, and H. Mabuchi, Phys. Rev. Lett. **78**, 3221 (1997).
[9] C. Elliott, New J. Phys. **4**, 46 (2002).
[10] R. Beals, S. Brierley, O. Gray, A. W. Harrow, S. Kutin, N. Linden, D. Shepherd, and M. Stather, Proc. R. Soc. A **469** (2013).
[11] Y. Li and S. C. Benjamin, New Journal of Physics **14**, 093008 (2012), 1204.0443.
[12] S. Barz, E. Kashefi, A. Broadbent, J. F. Fitzsimons, A. Zeilinger, and P. Walther, Science **335**, 303 (2012).
[13] S. Massar and S. Popescu, Phys. Rev. Lett. **74**, 1259 (1995).
[14] A. Peres and P. F. Scudo, Phys. Rev. Lett. **87**, 167901 (2001).
[15] E. Bagan, M. Baig, R. Muñoz-Tapia, and A. Rodriguez, Phys. Rev. A **69**, 010304 (2004).
[16] G. Chiribella and G. M. D'Ariano, J. Math. Phys. **45**, 4435 (2004).
[17] E. Bagan and R. Muñoz-Tapia, Int. J. Quantum Inf. **4**, 5 (2006).
[18] V. Giovannetti, S. Lloyd, and L. Maccone, Phys. Rev. Lett. **96**, 010401 (2006).
[19] M. Skotiniotis and G. Gour, New J. Phys. **14**, 073022 (2012).
[20] T. Islam, L. Magnin, B. Sorg, and S. Wehner, New J. Phys. **16**, 063040 (2014).
[21] S. D. Bartlett, T. Rudolph, and R. W. Spekkens, Rev. Mod. Phys. **79**, 555 (2007).
[22] P. Komar, E. M. Kessler, M. Bishof, L. Jiang, A. S. Sorensen, J. Ye, and M. D. Lukin, Nat Phys **advance online publication** (2014).
[23] L. Lamport, R. Shostak, and M. Pease, ACM T. Prog. Lang. Sys. **4**, 382 (1982).
[24] A. Peres and P. F. Scudo, Phys. Rev. Lett. **86**, 4160 (2001).
[25] R. Canetti and T. Rabin, in *Proc. ACM STOC'93* (ACM, 1993), pp. 42–51, ISBN 0-89791-591-7.
[26] M. Ben-Or and R. El-Yaniv, DISTRIB COMPUT **16**, 249 (2003).
[27] I. L. Chuang, Phys. Rev. Lett. **85**, 2006 (2000).
[28] Here time means the number of steps executed by the node while running the protocol.

Full Paper: Asynchronous reference frame agreement in a quantum network

Tanvirul Islam[1, 2, 3, *] and Stephanie Wehner[2, 3, †]

[1] *School of Computing, National University of Singapore, 13 Computing Drive, 117417 Singapore*
[2] *Centre for Quantum Technologies, National University of Singapore, 3 Science Drive 2, 117543 Singapore*
[3] *Qutech, Delft University of Technology, Lorentzweg 1, 2628 CJ Delft*

An efficient implementation of many multiparty protocols for quantum networks requires that all the nodes in the network share a common reference frame. To establish such a reference frame from scratch is especially challenging in an asynchronous network where network links might have arbitrary delays and the nodes do not share synchronised clocks. In this work, we study the problem of establishing a common reference frame in an asynchronous network of n nodes of which at most t are affected by arbitrary unknown error, and the identities of the faulty nodes are not known. We present a protocol that allows all the correctly functioning nodes to agree on a common reference frame as long as the network graph is complete and not more than $t < n/4$ nodes are faulty. As the protocol is asynchronous, it can be used to synchronise clocks over a network. Also, the protocol has the appealing property that it allows any existing two-node protocol for asynchronous reference frame agreement to be lifted to a robust protocol for a quantum network.

PACS numbers:

I. INTRODUCTION

To use quantum cryptography on a global scale one must first have a functioning quantum internet [1]. Recently this necessity has inspired a lot of effort in the research and development of satellite [2–6], and ground based [7–9] quantum networks. The possible applications of such networks are not restricted to only cryptography. A fully general quantum network will allow us to perform general distributed quantum computing [10–12].

In this work, we study problems related to initialisation and construction of quantum networks. More specifically we study how well n nodes in an asynchronous quantum network can agree on a reference frame in the presence of at most t arbitrarily faulty nodes among them. By asynchronous network we mean in this setting we do not require the nodes to share a clock to begin with, and the channel delays can be arbitrarily in each use. In fact, an asynchronous protocol only assumes any message sent from a correct node to a correct node will eventually reach the destination, without imposing any bound on the channel delay. This assumption captures the most general reference frame agreement problem in a quantum network because during the initialisation of the network the pairwise channel delays might be unknown, clocks might not be synchronised and spatial reference frames might be unaligned.

In a quantum channel the qubits are encoded in some physical degree of freedom. For example, polarisation direction of photon is often used to encode qubits. Which requires the sender and receiver to agree on some set of orthonormal directions as their common spatial reference frame. Another example is the time-bin qubits were both of the parties requires synchronised clocks. That is they must have a pre-agreed temporal reference frame. So far these reference frame agreement problems are studied in a bipartite setting [13–19] with the exception of [20] where spatial direction are agreed on in a *synchronised* network of n nodes. Specially in [20] it is assumed that the network is synchronous that is all the nodes of the network have a shared clock and all the link delays have known upper bound. The bipartite reference frame agreement problem have been studied extensively. For example, many of the known results are discussed in the review article [21]. But agreeing on a reference frame in an asynchronous network of n nodes remained open.

One advantage of having asynchronous reference frame agreement protocol for a network with certain number of faulty nodes is that once a spatial reference frame is established, then new robust protocols can potentially be built on top of it to perform network-wide clock synchronisation. This is a task important by itself with various application in security, navigation and finance etc. [22]. The primary difficulty of executing any protocol in an asynchronous network comes from the fact that in presence of incorrect, that is, arbitrarily faulty nodes it is impossible to decide for an correct receiver whether a message is not arriving because the sender is faulty and not sending anything at all, or if the sender is correct but the channel is taking a very long time to transfer the message. It is nontrivial to decide how long to wait for a message before moving on to the next step of a protocol.

Another difficulty is that unlike in classical information theory where information can be represented in bits, a reference frame can only be transferred from scratch by exchanging systems which have an inherent sense of direction. Examples of such systems are spin qubits and photon polarisation qubits. The receiver can only extract direction information from these systems by performing tomography on them. While preparing the direction any node P_i will know the description of the direction as a

*tanvir@locc.la
†steph@locc.la

vector v_i in its local frame. Once the quantum system carrying that direction arrives to a receiver P_j, the receiver constructs a representation of the direction in it's own local frame as v_j. This inevitably introduces some error even in correct transmissions. That is, depending on the precision of the instruments one can only expect to have $d(v_i, v_j) \leq \delta$ for some $\delta > 0$. But this distance metric does not make sense as it is, because v_i and v_j are vector representations in two different local frames. So we must redefine our distance metric $d(.,.)$ where distance is computed by converting both vectors in the frame of the first argument. As a result $d(v_i, v_j)$ remains a valid distance measure even though P_i and P_j do not know each other's local frame. This computation of distance between two vectors of different reference frames is only done in the analysis of the protocol and not by the nodes while playing the protocol. Any distance computed by a node inside a protocol is only between vectors for which it has a representation in its local frame. The inherent imperfection of message transmission must be accounted for by any reference frame agreement protocol. We capture this in the definition as,

Definition 1. For $\eta > 0$, a protocol in an asynchronous network of n nodes is a *η-asynchronous reference frame agreement protocol* if it satisfies the following conditions.

Termination. Every correct node P_i eventually terminates and outputs a direction v_i.

Correctness. If correct node P_i outputs v_i and correct node P_j outputs v_j then $d(v_i, v_j) \leq \eta$.

But we have to achieve these termination and correctness condition in the presence of incorrect or faulty nodes. As it is unknown which nodes are faulty this resembles the Byzantine fault tolerance model [23] studied in from classical distributed computing. For quantum networks our assumptions are,

- The pairwise channels are *public*. That is, the messages are not secret. As a result an adversary can see the content of a message between two correct nodes and adapt its strategy accordingly.
- The pairwise channels are authenticated. That is, if a correct node sends a message to another correct node the message cannot be altered by the adversary. But there can be channel noise and message might be slightly changed within a bound.
- The pairwise channel delays might be controlled by and faulty nodes. Not only the faulty nodes can see the content of each pairwise channel in the network they can control the channel delays, even between the correct nodes.
- If an honest node sends a message to another honest node the message eventually reaches the receiver. That is, even though the delay is controlled by the some adversary they cannot put infinite delay on a message between two correct nodes. But the delay is unbounded.
- The dishonest nodes might have correlated error, or might cooperate with each other to thwart the protocol. To create protocol which tolerates the worst kind of faults we also assume that the faulty nodes can cooperate with each other and have a global strategy to fault the protocol. This is a realistic assumption because some nodes might get controlled by an adversary.

II. PRELIMINARIES

The problem of reference frame agreement over an asynchronous quantum network is necessarily multidisciplinary in nature. That is, it combines various concepts from quantum physics, information theory, cryptography and distributed computing. In this section we introduce several concepts from these fields that will be useful throughout this work.

A. Reference frame

1. Spatial reference frame

Spatial reference frame defines a co-ordinate system in space. For example in a Cartesian coordinate system, once the Cartesian frame $(\vec{\boldsymbol{x}}, \vec{\boldsymbol{y}}, \vec{\boldsymbol{z}})$ is specified any vector $v = \alpha\vec{\boldsymbol{x}} + \beta\vec{\boldsymbol{y}} + \gamma\vec{\boldsymbol{z}}$ can be represented as (α, β, γ) where α, β and γ are scalers. For two distant parties, who only have the knowledge of their own local frame, it becomes necessary to establish a shared reference frame before they can successfully communicate spatial information (like, location, orientation,... etc).

We only use quantum communications to send a direction between a sender and a receiver. As an example we use protocol 2ED, one of the simplest possible protocols as studied in [13]. Here a sender creates many identical qubits with their Bloch vector pointing to the intended direction and the receiver measures them with Pauli measurements. From the statistics of the measurement outcomes, the receiver then estimates the Bloch vector's direction closely with high success probability.

2. Temporal reference frame

Similar to spatial reference frames multiple parties might need to synchronise their clock rates and time differences. Once they have established it, we say that they share a *temporal reference frame* and they are synchronised in time. Any multiparty protocol or computation performed by systems that do not share a temporal reference frame are respectively called *asynchronous protocol* or *asynchronous computation*.

Protocol 1: 2ED

input : Sender, direction u
output : Receiver, direction v

Sender: 2ED-Send
- Prepare $3n$ qubits with direction u
- Send them to the receiver

Receiver: 2ED-Receive
- Receive $3n$ qubits from the sender
- Measure n qubits with σ_x and compute p_x, the frequency of getting outcome $+1$
- Similarly on the remaining qubits, compute p_y and p_z with measurements σ_y and σ_z on n qubits each
- Assign $x \leftarrow 2p_x - 1$, $y \leftarrow 2p_y - 1$, $z \leftarrow 2p_z - 1$; Assign $l \leftarrow \sqrt{x^2 + y^2 + z^2}$
- Output $v \leftarrow (x/l, y/l, z/l)$

B. Asynchronous communication

In an asynchronous network we assume the nodes do not share any synchronized clock. Also, the communication channel between each pair is such that a message takes arbitrary amount of time to propagate through it. The only promise is, if a message is transmitted from a correct node the message will eventually reach to the receiver.

1. The asynchronous message

In the absence of a synchronized clock, each message must have a 'begin' and 'end' tag. Also, depending on the particular application, a message might carry a [type] information. In our problem we don't have a shared reference frame. As a result, we cannot use the quantum channel to carry these [type] information. This requires us to have a parallel classical channel that uses some classical degree of freedom to carry bits.

Let's assume Alice and Bob are connected by an asynchronous classical channel with maximum delay d_c and a asynchronous quantum channel with maximum delay d_q. Using these two separate channels we want to create an asynchronous CQ-channel (classical quantum channel) that can send a message with both classical and quantum component in the absence of a shared reference frame. An example of such combined message is shown in Table I where each quantum message m_q is sandwiched between a classical 'begin' and 'end' tag and also accompanied by a classical type tag m_c. The symbol $\perp$ denotes quantum signals that can be ignored.

TABLE I: Channel primitive: **A message**

Step	Classical	Quantum
1	begin	$\perp$
2	m_c	m_q
3	end	$\perp$

Messages might come out of a channel in a different order than the were put in initially. But we assume inside one atomic message the order of each of the 3 steps remains unchanged.

2. Time complexity of asynchronous protocols

Note that in an asynchronous network a message can take arbitrary amount of time to travel from sender to receiver. Also in asynchronous protocol, the nodes might take arbitrary amount of time to take next step in a protocol. So, to compute the time complexity of an asynchronous protocol it is a standard practice to only count the number of steps executed by any node. Each of these steps might take an arbitrary amount of time.

3. Asynchronous interactive consistency

Our protocol uses the solution to the following interactive consistency problem which was first proposed by Pease, Shostak and Lamport [24].

Definition 2 (The Interactive Consistency Problem)**.** Consider a complete network of n nodes in which communication lines are private. Among the n nodes up to t might be faulty. Let $P_1, P_2, \ldots, P_n$ denote the nodes. Suppose that each node P_i has some private value of information $V_i \in |V| \geq 2$. The question is whether it is possible to device a protocol that, given $n, t \geq 0$, will allow each correct nodes to compute a vector of values with an element for each of the n processors, such that:

1. All the correct nodes compute exactly the same vector;
2. The element of this vector corresponding to a given correct node is the private value of that node.

For an asynchronous network, Ben-Or and El-Yaniv [25] gives a protocol Asynchronous-IC which solves this problem for $t < n/3$ in constant expected time. We use this protocol as a subroutine.

III. RESULTS

In this paper we give a protocol that can take any two party reference frame agreement protocol and lift it up to a fault tolerant multiparty reference frame agreement protocol. More specifically, we present the first protocol A-Agree which allows n nodes in fully connected asynchronous network to agree on a reference frame in the presence of $t < n/4$ faulty nodes. The result can be summarised in the following theorem.

Theorem 1. *In a complete network of n nodes that are pairwise connected by public quantum channels, if a bipartite δ-estimate direction protocol that uses m qubits to*

achieve success probability $q_{\text{succ}} \geq 1 - e^{-\Omega(m\delta^2)}$ *is used, then protocol A-Agree is a* 34δ*-asynchronous reference frame agreement protocol with success probability at least* $1 - e^{-\Omega(m\delta^2 - \log n)}$*, that can tolerate up to* $t < n/4$ *faulty nodes.*

The problem of both synchronous an asynchronous agreement on classical bits in the presence of arbitrary faulty node is extensively studied in classical literature as Byzantine agreement problem [23]. But we should emphasise that, any classical protocol cannot be used in our problem because firstly, unlike classical network, any communication of direction among correct nodes in a quantum network will have inherent noises. As a result any classical protocol would see all the correct nodes as faulty nodes. And the protocol will fail. Secondly, one cannot use the classical protocol directly because one cannot represent a reference frame in classical bits [26]. But classical literature can still inform us on important questions like, how to achieve constant expected time, how to handle asynchronicity, etc. Some of the approaches of our protocol regarding these questions are influenced [27]. We also use the interactive consistency protocol by Ben-Or et. [25] as a subroutine.

To give the protocols we first need to define some notations.

$w_i[j]$ represents a vector received by node P_i from node P_j. Which is represented with respects to P_i's local reference frame.

In our protocol any message between two nodes that carries a direction also base a [type] tag. We call the direction vector is of type [type].

Next we fix a notation for a cluster of vectors of certain types where the cluster has a certain cluster centre and a cluster parameter. We write it as $C_i^{\delta}([\text{types}], w_c)$. This means the cluster with cluster centre $w_c$is computed and stored in node P_i, has a cluster parameter δ and contains only the types of vector stated in the [types]. Here [types] is a comma separated list of [type]s. The cluster parameter δ denotes that for all $u, v \in C_i^{\delta}([\text{types}], w_c)$ their distance $d(u, v) \leq \delta$.

For example, $C_i^{\delta}([\text{ready}_1,\text{ready}_2], v_c)$ denotes a cluster with any of ready_1 or ready_2 type directions with cluster center v_c such that $\forall u, v \in C_i^{\delta}([\text{ready}_1,\text{ready}_2], v_c)$, $d(u, v) \leq \delta$.

$P(C_i^{\delta}([\text{type}], w_c))$ is the set of all the nodes P_j such that, $w_i[j] \in C_i^{\delta}([\text{type}], w_c)$.

Now we give our protocol in two steps. First, we give a protocol for asynchronous broadcast, which we later use as a subroutine in our asynchronous agreement protocol.

A. Asynchronous broadcast

As the name suggests using this protocol a sender node can send some message to all the other nodes in an asynchronous network. At first sight a naive protocol of just sending the message to all other nodes one by one seems to be a valid protocol. But this naive protocol do not work if the sender intentionally sends different message to different nodes, which can easily happen in networks with faulty nodes. To guard from it all the other nodes must communicate between each other to make sure they are receiving the same message, or a close approximation to it. But, as we have at most t faulty nodes, this verification also becomes tricky. The whole thing becomes more challenging because the network is not synchronous. As a result a receiver who is waiting for a message, cannot be certain whether to keep waiting (because the message might be taking a long time in the channel) or move on (the sending node might be faulty and not sending the message at all). Our protocol takes care of all these challenges.

Formally the protocol is defined as,

Definition 3. For $\eta > 0$, $\zeta > 0$, a protocol which is initiated by a sender node P_s, in an asynchronous network of n nodes, is called a (η, ζ)-*asynchronous reference frame broadcast protocol* if it satisfies the following conditions.

Termination.

1. If the sender is correct then every correct node eventually completes the protocol.
2. If any correct node completes the protocol, then all the correct nodes eventually complete the protocol.

Consistency. If one correct node P_k outputs a direction v_k then all pairs of correct nodes P_i and P_j eventually output directions v_i, v_j where $d(v_i, v_j) \leq \eta$.

Correctness. If P_s is correct and broadcasts a direction u and if a correct node P_i outputs v_i then $d(u, v_i) \leq \zeta$.

We emphasize that the Termination condition of *asynchronous reference frame* ***broadcast*** is much weaker than the Termination condition of *asynchronous reference frame* ***agreement*** because in the broadcast protocol we do not require that the correct nodes complete the protocol if the sender is faulty.

We achieve this by our protocol AR-Cast. The following theorem summarises its properties.

Theorem 2. *In a complete network of* n *nodes that are pairwise connected by public quantum channels, if a bipartite* δ*-estimate direction protocol is used that succeeds with probability* $q_{\text{succ}} \geq (1 - e^{-\Omega(n\delta^2)})$ *then protocol AR-Cast is a* 42δ-asynchronous reference frame agreement protocol*, with success probability at least* $1 - e^{-\Omega(m\delta^2 - \log n)}$ *that can tolerate up to* $t < n/4$ *faulty nodes.*

The protocol 2 AR-Cast works roughly as follows. In Epoch 0 the sender sends its intended direction to all as a [init] type message. In Epoch 1 all the nodes waits until they receive an [init] from sender or sufficient number of confirmation from other nodes that they have received

Protocol 2: AR-Cast

input : Sender inputs direction u
output : $\forall i$ P_i outputs direction v_i

Epoch 0: (Only Sender)
- **Send-to-all** (init, u).

Epoch 1: (Player P_i)
- Listen to init, echo, ready$_1$ and ready$_2$ type messages.
- **Wait until *Either*** *received one (init, u_i)* **Then**
  - **Send-to-all** (echo, u_i).
  - **Goto** Epoch 2.
- ***Or* until** *received a cluster of directions $C_i^{4\delta}([echo], w_c)$ of size at least $(n-2t)$* **And** *a cluster of directions $C_i^{10\delta}([ready_1, ready_2], v_c)$ of size at least $(t+1)$, so that, $d(w_c, v_c) \leq 10\delta$* **Then**
  - **Send-to-all** (ready$_2$, w_c).
  - **Goto** Epoch 3.

Epoch 2: (Player P_i)
- Listen to echo, ready$_1$ and ready$_2$ type messages.
- **Wait until *Either*** *there exists a cluster of directions $C_i^{4\delta}([echo], w_c)$ of size at least $(n-t)$* **Then**
  - **Send-to-all** (ready$_1$, w_c).
  - **Goto** Epoch 3.
- ***Or* until** *there exists a cluster of directions $C_i^{4\delta}([echo], w_c)$ of size at least $(n-2t)$* **And** *a cluster of directions $C_i^{10\delta}([ready_1, ready_2], v_c)$ of size at least $(t+1)$, so that, $d(w_c, v_c) \leq 10\delta$,* **Then**
  - **Send-to-all** (ready$_2$, w_c).
  - **Goto** Epoch 3.

Epoch 3: (Player P_i)
- **Wait until** *there exists a cluster of directions $C_i^{20\delta}([ready_1, ready_2], v_c)$ of size at least $(n-t)$* **Then**
  - Output v_c.
  - Halt

Protocol 3: A-Agree

input : $\forall i$, P_i inputs direction u_i
output : $\forall i$, P_i outputs direction v_i

Epoch 0: (Player P_i)
- Create a direction array w_i of size n.
- $\forall j$, initialize $w_i[j] \leftarrow \perp$.
- Run AR-Cast(u_i).
- `// everyone broadcasts their local input`
- Store received direction from P_j in $w_i[j]$.
- After receiving $(3t+1)$ such directions **Goto** Epoch 1.
- But still continue the incomplete AR-Casts in parallel.

Epoch 1: (Player P_i)
- Create a bit string a_i of size n.
- **for** $j \leftarrow 1$ **to** n **do**
  - **if** $w_i[j] \neq \perp$ **then**
    - Assign $a_i[j] \leftarrow 1$.
  - **else**
    - Assign $a_i[j] \leftarrow 0$.
- `//` a_i `records which A-Casts are completed so far by` P_i
- Run Asynchronous-IC(a_i).
- `// This step reports to all which A-Casts are successfully received by` P_i
- Store the output of Asynchronous-IC in vector b_i such that, element $b_i[j]$ is received from P_j.
- `// After this step every correct nodes know which A-Casts are reported to be complete by which node`
- **Wait until** *Asynchronous-IC completes* **Then**
  - **Goto** Epoch 2

Epoch 2: (Player P_i)
- Let k_i be the index of a column which has at least $(t+1)$ 1s in it. So that, for any other index l of column with (t+1) 1s $k < l$. `// After completion of Asynchronous-IC each row of` b_i `is a bit string of length` n. `That is` b_i `is essentially an` $n \times n$ `bit matrix.`
- **Wait until** *the A-Cast initiated by P_{k_i} completes* **Then**
  - Assign $v \leftarrow w_i[k_i]$.
  - Abort all incomplete A-Casts that are running since Epoch 0.
  - Output v.

some directions and proceeding to next epoch. This way, even if some correct node never receives an [init] message if the other correct nodes are advancing through the protocol this node in Epoch 1 will not stay behind waiting. In Epoch 2 the correct nodes who have decides upon a direction notifies the other nodes about its decision by sending ready$_1$ or ready$_2$ type messages to all. All these previous epochs make sure that all the correct nodes eventually arrives in Epoch 3 and outputs a direction which satisfies theorem 2.

B. Asynchronous agreement

Now we give our main protocol A-Agree which uses AR-Cast as a subroutine and allows the correct nodes in an asynchronous network to agree on a reference frame.

In Epoch 0 of protocol 3 A-Agree all the nodes proposes a direction which represents their local frame. They broadcast this direction using AR-Cast. All the correct nodes waits for at least $(3t+1)$ such broadcasts to be complete. Then they enter Epoch 1. Since, there are $(3t+1)$ correct nodes they will eventually arrive at Epoch 1. In this step all the correct nodes creates a bit string of length n where j'th bit represents if the j'th AR-Cast have been completed successfully in Epoch 0. Then all the nodes sends this bit string to all by playing Asynchronous-IC. After this they enters Epoch 2. In this Epoch every node has the same set of bit strings. They now look for the lowest inter k such that at least $(t+1)$ bit strings has a 1 in that position. If they have completed that k'th AR-Cast they output their direction received from that broad cast. If they k'th AR-Cast is not complete for him they wait until it completes and then output. The election of k ensures that at least one correct node

have completed the k'th AR-Cast so by Consistency of asynchronous reference frame broadcast all the honest nodes will eventually complete the k'th AR-Cast. This ensure that the A-Agree eventually completes. As, there is no loop in this protocol and all the subroutines run in constant expected time. The A-Agree is also a constant expected time protocol.

IV. CONCLUSION

In this work we have presented the first asynchronous reference frame agreement protocol. The synchronous protocol for spatial reference frame agreement presented in [20] can tolerate up to $t < n/3$ faulty nodes. Whereas, the asynchronous protocol we have presented tolerates only $t < n/4$ faulty nodes. Even though we pay this extra price in fault tolerance, an asynchronous protocol is a fully general reference frame agreement protocol. Because it can be used to synchronised clocks [28], which is an important problem in its own right. There are classical protocols for asynchronous agreement on bits which achieve $t < n/3$ in constant expected time, it remains open to see if this bound can be achieved by reference frame agreement protocols for a quantum network.

[1] H. J. Kimble, Nature **453**, 1023 (2008).
[2] M. Aspelmeyer, T. Jennewein, M. Pfennigbauer, W. Leeb, and A. Zeilinger, IEEE J. Sel. Topics Quantum Electron. **9**, 1541 (2003).
[3] C. Bonato, A. Tomaello, V. D. Deppo, G. Naletto, and P. Villoresi, New J. Phys. **11**, 045017 (2009).
[4] C.-Z. Peng, T. Yang, X.-H. Bao, J. Zhang, X.-M. Jin, F.-Y. Feng, B. Yang, J. Yang, J. Yin, Q. Zhang, et al., Phys. Rev. Lett. **94**, 150501 (2005).
[5] C. Bonato, M. Aspelmeyer, T. Jennewein, C. Pernechele, P. Villoresi, and A. Zeilinger, Opt. Express **14**, 10050 (2006).
[6] J. M. P. Armengol, B. Furch, C. J. de Matos, O. Minster, L. Cacciapuoti, M. Pfennigbauer, M. Aspelmeyer, T. Jennewein, R. Ursin, T. Schmitt-Manderbach, et al., Acta Astronaut. **63**, 165 (2008).
[7] M. Sasaki, M. Fujiwara, H. Ishizuka, W. Klaus, K. Wakui, M. Takeoka, S. Miki, T. Yamashita, Z. Wang, A. Tanaka, et al., Opt. Express **19**, 10387 (2011).
[8] J. I. Cirac, P. Zoller, H. J. Kimble, and H. Mabuchi, Phys. Rev. Lett. **78**, 3221 (1997).
[9] C. Elliott, New J. Phys. **4**, 46 (2002).
[10] R. Beals, S. Brierley, O. Gray, A. W. Harrow, S. Kutin, N. Linden, D. Shepherd, and M. Stather, Proc. R. Soc. A **469** (2013).
[11] Y. Li and S. C. Benjamin, New Journal of Physics **14**, 093008 (2012), 1204.0443.
[12] S. Barz, E. Kashefi, A. Broadbent, J. F. Fitzsimons, A. Zeilinger, and P. Walther, Science **335**, 303 (2012).
[13] S. Massar and S. Popescu, Phys. Rev. Lett. **74**, 1259 (1995).
[14] A. Peres and P. F. Scudo, Phys. Rev. Lett. **87**, 167901 (2001).
[15] E. Bagan, M. Baig, R. Muñoz-Tapia, and A. Rodriguez, Phys. Rev. A **69**, 010304 (2004).
[16] G. Chiribella and G. M. D'Ariano, J. Math. Phys. **45**, 4435 (2004).
[17] E. Bagan and R. Muñoz-Tapia, Int. J. Quantum Inf. **4**, 5 (2006).
[18] V. Giovannetti, S. Lloyd, and L. Maccone, Phys. Rev. Lett. **96**, 010401 (2006).
[19] M. Skotiniotis and G. Gour, New J. Phys. **14**, 073022 (2012).
[20] T. Islam, L. Magnin, B. Sorg, and S. Wehner, New J. Phys. **16**, 063040 (2014).
[21] S. D. Bartlett, T. Rudolph, and R. W. Spekkens, Rev. Mod. Phys. **79**, 555 (2007).
[22] P. Komar, E. M. Kessler, M. Bishof, L. Jiang, A. S. Sorensen, J. Ye, and M. D. Lukin, Nat Phys **advance online publication** (2014).
[23] L. Lamport, R. Shostak, and M. Pease, ACM T. Prog. Lang. Sys. **4**, 382 (1982).
[24] M. Pease, R. Shostak, and L. Lamport, J. ACM **27**, 228 (1980).
[25] M. Ben-Or and R. El-Yaniv, DISTRIB COMPUT **16**, 249 (2003).
[26] A. Peres and P. F. Scudo, Phys. Rev. Lett. **86**, 4160 (2001).
[27] R. Canetti and T. Rabin, in *Proc. ACM STOC'93* (ACM, 1993), pp. 42–51, ISBN 0-89791-591-7.
[28] I. L. Chuang, Phys. Rev. Lett. **85**, 2006 (2000).

V. APPENDIX

A. Asynchronous reference frame broadcast

To prove correctness of our AR-Cast we have to prove theorem 2 as restated here.

Theorem 2. *In a complete network of n nodes that are pairwise connected by public quantum channels, if a bipartite δ-estimate direction protocol is used that succeeds with probability $q_{\text{succ}} \geq (1 - e^{-\Omega(n\delta^2)})$ then protocol AR-Cast is a 42δ-asynchronous reference frame agreement protocol, with success probability at least $1 - e^{-\Omega(m\delta^2 - \log n)}$ that can tolerate up to $t < n/4$ faulty nodes.*

For this we observe several properties of Protocol 2 in the following lemmas. The first observation is that if two different correct nodes sends [ready$_1$] type message then the direction they send are close to each other with high probability.

Lemma 1. *For $t < n/4$, $\delta > 0, q_{\text{succ}} > 0$, if two correct nodes P_i and P_j send ([ready$_1$],u) and ([ready$_1$],v) respectively, then $d(u, v) \leq 10\delta$ with probability at least $q_{\text{succ}}^{n+n^2}$.*

Proof. In step 4 of Epoch 2 when a [ready$_1$] message is generated there are at most n init messages originated from the sender and at most n^2 echo messages generated by the other nodes. So, with probability at least $q_{\text{succ}}^{n+n^2}$ all the transmissions which are among correct nodes are successful. Conditioning on this we prove,

$$d(u, v) \leq 10\delta. \tag{1}$$

We show this in two steps. First, we show that there exists a common correct node P_k in $P(C_i^{4\delta}([\text{echo}], u))$ and $P(C_j^{4\delta}([\text{echo}], v))$. Then using the triangle inequality with the fact that the echo vector from P_k must be close to both of the cluster centers u and v, we derive inequality (1).

Now, for the first step, let us denote A_i and A_j to be the set of nodes from which the vectors in $(C_i^{4\delta}([\text{echo}], u)$ have originated. And B_i and B_j to be the correct nodes in A_i and A_j respectively. Formally,

$$A_i = P(C_i^{4\delta}([\text{echo}], u)), \tag{2}$$
$$A_j = P(C_j^{4\delta}([\text{echo}], v)), \tag{3}$$
$$B_i = \{P_l : P_l \in A_i \text{ and } P_l \text{ is correct.}\}, \tag{4}$$
$$B_j = \{P_l : P_l \in A_j \text{ and } P_l \text{ is correct.}\}. \tag{5}$$

Note that at this step $|A_i| \geq n - t$ and $|A_j| \geq n - t$. We want to show that,

$$B_i \cap B_j \neq \emptyset. \tag{6}$$

We do this by contradiction: let us assume that,

$$B_i \cap B_j = \emptyset. \tag{7}$$

Note that,

$$|A_i| \geq n - t \tag{8}$$
$$\Rightarrow |A_i - B_i| + |B_i| \geq n - t, \tag{9}$$
$$\Rightarrow t + |B_i| \geq n - t, \tag{10}$$
$$\Rightarrow |B_i| \geq n - 2t, \tag{11}$$
$$\Rightarrow |B_i| > n - 2(n/4) = n/2. \tag{12}$$

Here, inequality (10) holds because at most t of the nodes are faulty. And inequality (12) holds because $t < n/4$.

Now,

$$|A_i \cup A_j| = |(A_i - B_i) \cup (A_j - B_j) \cup B_i \cup B_j|\,,$$
$$\geq |(A_j - B_j)| + |B_i| + |B_j|\,, \tag{13}$$
$$= |A_j| + |B_i|\,, \tag{14}$$
$$> (n - t) + n/2, \tag{15}$$
$$> n - n/4 + n/2 = 5n/4 \tag{16}$$

Here, inequality (14) uses inequality (7), inequality (15) follows from the definition from the size of A_j and inequality (12). And inequality (16) follows because, $t < n/4$. But this is a contradiction, because there are only n nodes in the network. So, we must have,

$$B_i \cap B_j \neq \emptyset. \tag{17}$$

So, there exists a common correct node $P_k \in B_i \cap B_j$ in $P(C_i^{4\delta}([\text{echo}], u))$ and $P(C_j^{4\delta}([\text{echo}], v))$. As P_k is correct, it must have sent the same echo type message to both P_i and P_j. So, using the triangle inequality we have,

$$d(w_i[k], w_j[k]) \leq d(w_i[k], u_k) + d(u_k, w_j[k]), \tag{18}$$
$$\leq \delta + \delta = 2\delta. \tag{19}$$

Now inequality (1) follows because,

$$d(u, v) \leq d(u, w_i[k]) + d(w_i[k], w_j[k]) + d(w_j[k], v), \tag{20}$$
$$\leq 4\delta + d(w_i[k], w_j[k]) + 4\delta, \tag{21}$$
$$\leq 4\delta + 2\delta + 4\delta = 10\delta. \tag{22}$$

Here, inequality (21) follows from the definitions of $C_i^{4\delta}([\text{echo}], u)$ and $C_j^{4\delta}([\text{echo}], v)$ and inequality (22) follows from inequality (19).

□

In lemma 1 we have shown the relation between two [ready$_1$] type directions from two different honest node. Now we show that if a correct node sends a [ready$_1$] and another honest node sends a [ready$_2$] type message then the directions they send are close with high probability.

Lemma 2. *For $t < n/4$, $\delta > 0, q_{\text{succ}} > 0$, if two correct nodes P_i and P_j send ([ready$_1$],u) and ([ready$_2$],v) accordingly, then $d(u, v) \leq 10\delta$ with probability at least $q_{\text{succ}}^{n+2n^2}$.*

Proof. When a [ready$_2$] message is generated there are at most n init, n^2 echo and in total n^2 [ready$_1$] or [ready$_2$] messages generated in the protocol. With probability at least $q_{\text{succ}}^{n+2n^2}$ all the transmissions which are among correct nodes are successful. Conditioning on this, we show that,

$$d(u,v) \leq 10\delta. \tag{23}$$

We do this in two steps, first we show that there is a common correct node P_k in $P(C_i^{4\delta}([\text{echo}],u))$ and $P(C_j^{4\delta}([\text{echo}],v))$. Then using the triangle inequality with the fact that both of the cluster centers u and v must be close to the echo direction sent from P_k we prove the inequality (23).

To see the first step, let us first define sets A_i, A_j, B_i and B_j by,

$$A_i = P(C_i^{4\delta}([\text{echo}],u)), \tag{24}$$
$$A_j = P(C_j^{4\delta}([\text{echo}],v)), \tag{25}$$
$$B_i = \{P_l : P_l \in A_i \text{ and } P_l \text{ is correct.}\}, \tag{26}$$
$$B_j = \{P_l : P_l \in A_j \text{ and } P_l \text{ is correct.}\}. \tag{27}$$

Note that here $|A_i| \geq n-t$ and $|A_j| \geq n-2t$. We want to show that,

$$B_i \cap B_j \neq \emptyset. \tag{28}$$

We do this by contradiction: let us assume that,

$$B_i \cap B_j = \emptyset. \tag{29}$$

Note that,

$$|A_i| \geq n-t \tag{30}$$
$$\Rightarrow |A_i - B_i| + |B_i| \geq n-t, \tag{31}$$
$$\Rightarrow t + |B_i| \geq n-t, \tag{32}$$
$$\Rightarrow |B_i| \geq n-2t, \tag{33}$$
$$\Rightarrow |B_i| > n-2(n/4) = n/2. \tag{34}$$

Here, inequality (32) holds because at most t of the nodes are faulty. And inequality (34) holds because $t < n/4$.

Now,

$$\begin{aligned} |A_i \cup A_j| &= |(A_i - B_i) \cup (A_j - B_j) \cup B_i \cup B_j|, \\ &\geq |(A_j - B_j)| + |B_i| + |B_j|, && (35)\\ &= |A_j| + |B_i|, && (36)\\ &> (n-2t) + n/2, && (37)\\ &> n - n/2 + n/2 = n && (38)\end{aligned}$$

Here, inequality (37) follows from the definition of A_j and inequality (34). And inequality (38) follows because, $t < n/4$. But this is a contradiction, because there are only n nodes in the network. So, we must have,

$$B_i \cap B_j \neq \emptyset. \tag{39}$$

So, there exists a common correct node P_k in $P(C_i^{4\delta}([\text{echo}],u))$ and $P(C_j^{4\delta}([\text{echo}],v))$. As P_k is correct, it mast have sent the same echo type message to both P_i and P_j. So, using the triangle inequality we have,

$$\begin{aligned} d(w_i[k],w_j[k]) &\leq d(w_i[k],u_k) + d(u_k,w_j[k]), && (40)\\ &\leq \delta + \delta = 2\delta. && (41)\end{aligned}$$

Now inequality (1) follows because,

$$\begin{aligned} d(u,v) &\leq d(u,w_i[k]) + d(w_i[k],w_j[k]) + d(w_j[k],v), && (42)\\ &\leq 4\delta + d(w_i[k],w_j[k]) + 4\delta, && (43)\\ &\leq 4\delta + 2\delta + 4\delta = 10\delta. && (44)\end{aligned}$$

Here, inequality (43) follows from the definitions of $C_i^{4\delta}([\text{echo}],u)$ and $C_j^{4\delta}([\text{echo}],v)$ and inequality (44) follows from inequality (41).

□

Now we show that all the correct nodes cannot send only [ready$_2$] type messages. That is if there exists a [ready$_2$] message sent from a correct node then there must pre-exist a [ready$_1$] message sent from another correct node.

Lemma 3. *For $t < n/4$, $\delta > 0, q_{\text{succ}} > 0$, if a correct node P_j sends ([ready$_2$],v) then,with probability at least $q_{\text{succ}}^{n+2n^2}$, there exists a correct node P_i which has sent ([ready$_1$],u) .*

Proof. When a [ready$_2$] message is generated there are at most n init, n^2 echo and in total n^2 [ready$_1$] or [ready$_2$] messages generated in the protocol. With probability at least $q_{\text{succ}}^{n+2n^2}$ all the transmissions which are among correct nodes are successful. In this case, just before making the decision to send a ([ready$_2$],v) message node P_j must have received at least (t+1) [ready$_1$] or [ready$_2$] messages from nodes in $P(C_i^{10\delta}([\text{ready}_1,\text{ready}_2]v_c))$. Of these, at least one node—let's call it P_k—is correct. If P_k has also sent a [ready$_2$] type message, we can find another correct node in its $P(C_k^{10\delta}([\text{ready}_1,\text{ready}_2]v_c))$ and so on. This way, eventually we will find a correct node who have sent a [ready$_1$] type message.

To see this, let us define a directed graph $G(V,E)$ with vertex set $V = \{P_i : P_i \text{ is correct}\}$, and

$$\begin{aligned} E = \{&(P_k,P_i) : P_k \text{ has sent ready}_2 \\ &\text{after receiving ready}_1 \text{ or ready}_2 \text{ from } P_i\}. \end{aligned} \tag{45}$$

One can convince oneself that G is a directed acyclic graph because any cycle in the graph would violate the cause and effect relation of the edge directions. Now if we look at the connected component of this graph containing P_j there must exist a node P_i in this component with no outgoing edges. Because V only contains correct nodes. This implies P_i is a correct node which has sent a [ready$_1$] type message ([ready$_1$],u). This completes the proof. □

Now the only thing remains is to show that two [ready$_2$] type directions sent from two correct nodes are close with high probability.

Lemma 4. *For $t < n/4$, $\delta > 0, q_{\text{succ}} > 0$, if two nodes P_i and P_j sends ([ready$_2$],u) and ([ready$_2$],v) respectively, then $d(u, v) \leq 20\delta$ with probability at least $q_{\text{succ}}^{n+2n^2}$.*

Proof. When a [ready$_2$] message is generated there are at most n init, n^2 echo and in total n^2 [ready$_1$] or [ready$_2$] messages generated in the protocol. With probability at least $q_{\text{succ}}^{n+2n^2}$ all of these transmissions which are between honest players are successful. Conditioning on this, we show that, if correct P_i sends ([ready$_2$],u) then from Lemma 3 there exists a correct node P_k which has sent ([ready$_1$],w). From Lemma 2,

$$d(u, w) \leq 10\delta, \tag{46}$$

and

$$d(v, w) \leq 10\delta. \tag{47}$$

Using the triangle inequality with these we get,

$$d(u, v) \leq d(u, w) + d(w, v) \leq 10\delta + 10\delta = 20\delta. \tag{48}$$

□

Now we are ready to prove that our protocol 2 satisfies the first termination condition of definition 3.

Lemma 5 (Termination 1)**.** *For $t < n/4$, $\delta > 0, q_{\text{succ}} > 0$, if the sender P_k is honest then the protocol 2* AR-Cast *eventually terminates with probability at least $q_{\text{succ}}^{n+n^2}$.*

Proof. There are at most n init messages, n^2 echo messages and n^2 [ready$_1$] or [ready$_2$] type messages exchanged in the protocol. With probability at least $q_{\text{succ}}^{n+2n^2}$ all of these transmissions which are between honest players are successful. In this case, if the sender is correct all the correct nodes eventually receive init messages that are at most 2δ apart from each other and send echo message. So, all the received echo messages are at most 3δ apart from the received init direction of any correct node. Any node that has sent a [ready$_1$] type message will go to epoch 3. The faulty nodes cannot stop the init and echo messages from correct nodes but they can manipulate the delayes, so that some of the honest players send d [ready$_2$] type messages. Still after sending the [ready$_2$] these honest players will finally arrive at Epoch 3. From lemma 1 and lemma 2 we can see that for any correct P_i all the received [ready$_1$] and [ready$_2$] directions will be in $C_i^{16\delta}$([ready$_1$,ready$_2$], v_c). And because there are $(n-t)$ of them originating from the correct nodes the protocol 2 AR-Cast will eventually terminate. Note that, if the sender is faulty, the definition of *(η, ζ)-reference frame broadcast protocol* (Derinition 3)do not require any termination. □

Now we show that if one correct node outputs a direction, then all the correct nodes eventually output directions that are close to each other.

Lemma 6 (Consistency)**.** *For $t < n/4$, $\delta > 0, q_{\text{succ}} > 0$, in protocol* A-cast*, if an honest player P_k outputs v_k then all pair of correct nodes P_i, P_j eventually output v_i, v_j respectively such that, $d(v_i, v_j) \leq 42\delta$ with probability at least $q_{\text{succ}}^{n+n^2}$.*

Proof. When a [ready$_2$] message is generated there are at most n init, n^2 echo and in total n^2 [ready$_1$] or [ready$_2$] messages generated in the protocol. With probability at least $q_{\text{succ}}^{n+2n^2}$ all of these transmissions which are between honest players are successful. In this case, we prove,

$$d(v_i, v_j) \leq 42\delta, \tag{49}$$

by showing that the successful completion of P_k implies there are enough echo, [ready$_1$] and [ready$_2$] type messages generated by correct nodes so that all the other correct nodes eventually receive them and successfully terminate and each pair of their outputs satisfies inequality (49).

Now, if a correct node P_k outputs v_k then this implies it has received at least $(n-t)$ [ready$_1$] or [ready$_2$] messages from nodes in $P(C_k^{20\delta}([\text{ready}_1,\text{ready}_2], v_k))$, of which at least $(n-2t)$ are correct. Messages from these correct nodes eventually reach all the other correct nodes. Also, from lemma 3 there exists a correct node which has sent a [ready$_1$] message which implies all the correct nodes eventually receive at least $(n-2t)$ echo messages. That is, all the correct nodes waiting in Epoch 1 or Epoch 2 will satisfy the condition of sending a [ready$_2$] message and go to Epoch 3. Any correct node P_i, P_j waiting in Epoch 3 will eventually receive all the [ready$_1$] or [ready$_2$] messages sent from correct nodes in $P(C_i^{20\delta}([\text{ready}_1,\text{ready}_2], v_i))$ and $P(C_j^{20\delta}([\text{ready}_1,\text{ready}_2], v_j))$ accordingly, and output v_i, v_j accordingly.

Now we show that $P(C_i^{20\delta}([\text{ready}_1,\text{ready}_2], v_i))$ and $P(C_j^{20\delta}([\text{ready}_1,\text{ready}_2], v_j))$ have at least one common honest player, which implies the cluster centers are close.

To see this note that each of these clusters have at least $(n-2t) > n - 2(n/4) = n/2$ correct nodes. That is more than n correct nodes in total. But there are total n nodes in the networks. This implies at least some of the corrects nodes are common in both clusters. Let P_l be such a node.

Now using triangular inequality we have,

$$\begin{aligned} d(v_i, v_j) &\leq d(v_i, v_i[l]) + d(v_i[l], v_l) \\ &\quad + d(v_l, v_j[l]) + d(v_j[l], v_j), &(50)\\ &\leq 20\delta + \delta + \delta + 20\delta = 42\delta. &(51)\end{aligned}$$

□

Now the second termination condition.

Lemma 7 (Termination 2)**.** *For $t < n/4$, $\delta > 0, q_{\text{succ}} > 0$, if a correct node P_i completes the protocol then all the*

correct nodes complete the protocol with probability at least $q_{\text{succ}}^{n+2n^2}$.

Proof. This lemma is a corollary of lemma 6. Because lemma 6 ensures completion with probability at least $q_{\text{succ}}^{n+2n^2}$.

□

Now we are ready to prove that our protocol satisfies the correctness condition of definition 3.

Lemma 8 (Correctness)**.** *For $t < n/4$, $\delta > 0, q_{\text{succ}} > 0$, if a correct sender P_s sends (init,u) and a correct node P_i outputs v_i then $d(u, v_i) \leq 14\delta$ with probability $q_{\text{succ}}^{n+2n^2}$.*

Proof. There are at most n init messages, n^2 echo messages and n^2 [ready$_1$] or [ready$_2$] type messages exchanged in the protocol. With probability at least $q_{\text{succ}}^{n+2n^2}$ all of these transmissions which are between honest players are successful.

In this case we prove the lemma in three steps. First, we show that all the [ready$_1$] type directions sent from correct node are close to u. Secondly, we show that that all the [ready$_2$] type directions sent from the correct nodes are close to u. And finally, from these we conclude the proof.

For the first step, let's assume that correct node P_i has sent a ([ready$_1$], v_i) message in Epoch 2. So, it has received at least $(n-t)$ echo type messages, of which at least $(n-2t)$ are from correct nodes. Let's assume for some correct node P_j $w_i[j] \in C_i^{4\delta}(v_i)$. As P_j is correct, using the triangle inequality, we have,

$$d(u, w_i[j]) \leq d(u, u_j) + d(u_j, w_i[j]), \tag{52}$$
$$\leq \delta + \delta = 2\delta. \tag{53}$$

The diameter of the cluster $C_i^{4\delta}(v_i)$ is 4δ. So, we have, $d(v_i, w_i[j]) \leq 2\delta$. Using this and (53) with the triangle inequality, we have,

$$d(u, v_i) \leq d(u, w_i[j]) + d(w_i[j], v_i), \tag{54}$$
$$\leq 2\delta + 2\delta = 4\delta. \tag{55}$$

Now, for the second step, let's assume that the correct node P_l has sent a ([ready$_2$], v_l) message from Epoch 1 or Epoch 2. So, v_l is a cluster center of at least $(n-2t)$ echo type messages. Of which at least $(n-3t)$ are honest. So, a similar reasoning to the previous step shows,

$$d(u, v_l) \leq 4\delta. \tag{56}$$

Finally, as the sender is correct from lemma 5 we know, all the honest players eventually enters Epoch 3 and successfully completes the epoch.

Let's assume a correct node P_i has received a cluster of [ready$_1$] or [ready$_2$] type directions $C_i^{20\delta}([\text{ready}_1,\text{ready}_2], v_c)$ of size at least $(n-t)$. So, there is a correct node P_k for which $v_i[k] \in C_i^{20\delta}([\text{ready}_1,\text{ready}_2], v_c)$. Here, $C_i^{20\delta}([\text{ready}_1,\text{ready}_2], v_c)$ is a cluster of diameter 20δ. So, we have $d(v_i[k], v_c) \leq 10\delta$. Using the triangle inequality with this, and (55) and (56), we have,

$$(u, v_c) \leq d(u, w_i[k]) + d(w_i[k], v_c), \tag{57}$$
$$\leq 4\delta + 10\delta = 14\delta. \tag{58}$$

This concludes the proof.

□

Lemma 9. *If a two-node direction estimation protocol is used that transmits m qubits to δ approximation a direction which succeeds with probability $q_{\text{succ}} \geq (1 - e^{-\Omega(m\delta)})$ then with probability at least $q_{\text{succ}}^{n+2n^2} \geq 1 - e^{-\Omega(m\delta - \log n)}$, all the direction transmissions of init, echo, [ready$_1$] and [ready$_2$] type messages are successful.*

Proof. There are at most n init messages, n^2 echo messages and n^2 [ready$_1$] or [ready$_2$] type messages exchanged in the protocol. With probability at least $q_{\text{succ}}^{n+2n^2}$ all of these transmissions which are between honest players are successful. Now,

$$q_{\text{succ}}^{n+2n^2} \geq (1 - e^{-\Omega(m\delta^2)})^{n+2n^2}, \tag{59}$$
$$\geq 1 - (n + 2n^2)e^{-\Omega(m\delta^2)}, \tag{60}$$
$$\geq 1 - e^{-\Omega(m\delta^2 - \log n)} \tag{61}$$

Here inequality (60) follows using Bernoulli's inequality, which is, $(1+x)^e > 1 + rx$ for all real $x \geq -1$ and integer $r \geq 2$.

□

We can see that theorem 2 follows from lemma 5, 6, 7, 8 and 9.

B. Asynchronous interactive consistency

Our protocol uses the solution to the following interactive consistency problem as a subroutine which was first proposed by Pease, Shostak and Lamport [24].

Definition 4 (The Interactive Consistency Problem)**.** Consider a complete network of n nodes in which communication lines are private. Among the n nodes up to t might be faulty. Let $P_1, P_2, \ldots, P_n$ denote the nodes. Suppose that each node P_i has some private value of information $u_i \in |V| \geq 2$. The question is whether it is possible to devise a protocol that, given $n, t \geq 0$, will allow each correct nodes to compute a vector of values $\vec{v}_i \in V^n$ with an element for each of the n processors, such that:

1. All the correct nodes P_i compute exactly the same vector $\vec{v}$;

2. The element of this vector corresponding to a given correct node is the private value of that node. That is, $\forall i, \vec{v}[i] = u_i$

Note that, here V is a set of classical values. That is elements of V can be encoded in classical bits.

For an asynchronous network, Ben-Or and El-Yaniv [25] give a protocol Asynchronous-IC which solves this problem for $t < n/3$ in constant expected time. Here *time* means the number of steps performed by any correct node while playing the protocol. We use this protocol as a subroutine.

C. Asynchronous Agreement

So far we have presented an asynchronous broadcast protocol where a designated sender initiates the protocol with a direction. One major weakness of the protocol is that, if the sender is faulty then the protocol might never terminate. Because in this case the correct nodes cannot decide whether the sender is faulty and not responding, or correct but very slow. On the other hand, in an asynchronous reference frame agreement protocol the main goal is to allow the correct nodes to agree on some direction despite the presence of—up to a certain number of—unidentified faulty nodes in the network. This requires extra caution to make sure that the protocol eventually terminates. We show that our protocol 3 A-Agree successfully solves this problem by proving theorem 1. We restate the theorem here.

Theorem 1. *In a complete network of n nodes that are pairwise connected by public quantum channels, if a bipartite δ-estimate direction protocol that uses m qubits to achieve success probability $q_{\text{succ}} \geq 1 - e^{-\Omega(m\delta^2)}$ is used, then protocol A-Agree is a 34δ-asynchronous reference frame agreement protocol with success probability at least $1 - e^{-\Omega(m\delta^2 - \log n)}$, that can tolerate up to $t < n/4$ faulty nodes.*

There are three epochs in protocol 3 any correct node that successfully terminates must start at Epoch 0 and terminate at Epoch 3. At each Epoch the nodes inside it, and all the messages transmitted and received by the node while in that Epoch satisfies some invariances properties. We describe and prove these properties in the following lemmas. We first show that an honest node will eventually enter Epoch 1.

Lemma 10. *For $t < n/4$, all the correct nodes eventually enter Epoch 1 of A-Agreement with probability at least $q_{\text{succ}}^{n^2+2n^3} \geq 1 - e^{-\Omega(m\delta - \log n)}$.*

Proof. Each of the n nodes has initiated an A-Cast in Epoch 0. Each of the A-Cast as a success probability at least $q_{\text{succ}}^{n+2n^2}$. So, with probability at least $q_{\text{succ}}^{n^2+2n^3}$ all the A-Casts from correct senders are successful. From Lemma 9 this is at least $1 - e^{-\Omega(m\delta - \log n)}$.

As $t < n/4$, there are at least $(3t+1)$ correct nodes who initiates A-Cast as sender. According to Theorem 2 these $(3t+1)$ A-Casts will eventually terminate. So, every honest receiver will eventually receive at least $(3t+1)$ directions and go to Epoch 1 with probability at least $q_{\text{succ}}^{n^2+2n^3}$. □

Each correct node stores the output of the Asynchronous-IC protocol in an array b_i. Here b_i can be seen as a $n \times n$ matrix of bits where row j is received from node j. We can observe the following property of this matrix.

Lemma 11. *For $t < n/4$ and correct node P_i, after instruction 9 of Epoch 1 of A-Agreement, there exists a column in b_i with at least $(t+1)$ 1s in it.*

Proof. We show this by a counting argument. Note that a correct node arrives at Epoch 1 only after it have received at least $(3t+1)$ directions from other players. As a result after step 7 of Epoch 1 a_i contains at least $(3t+1)$ 1's. These a_i's become the rows of b_i after step 9. There are at most t faulty nodes. So, at least $(3t+1)$ rows of b_i are originated from correct nodes. Each of these rows must contain at least $(3t+1)$ 1's. So b_i has at least $(3t+1)^2$ 1s.

But if no column had at least $(t+1)$ 1s, then there would be at most $(4t+1) * t$ 1s in b_i. This contradicts the fact that b_i has at least $(3t+1)^2$ 1s. So, there must exist a column with at least $(t+1)$ 1s in it. □

We show that all the correct nodes selects the same column which has at least $t+1$ 1s in it.

Lemma 12. *After instruction 2 of Epoch 2 of A-Agreement, if correct node P_i has k_i and correct node P_j has k_j then $k_i = k_j$.*

Proof. After completion of protocol Asynchronous-IC in Epoch 1, all the correct nodes compute the same output vector. That is, $b_i = b_j$. Also from lemma 11 we know there exists a column in b_i with at least $(t+1)$ 1s. So, in Epoch 2 step 2 when correct node P_i and P_j selects k_i and k_j to be the chronologically smallest column index that has at least $(t+1)$ 1s. They select the same column. i.e., $k_i = k_j$. □

Now that every correct node agrees on a column k_i of b_i we can observe that.

Lemma 13. *If a correct node P_i selects k_i in instruction 2 of Epoch 2 then the A-Cast initiated by P_{k_i} must eventually complete successfully.*

Proof. We show this by showing that at least one correct node has completed the A-Cast initiated by P_{k_i}. Then the lemma follows from the termination condition of A-Cast.

Each row $b_i[j]$ represents P_i's knowledge of which A-Casts are successfully received by P_j. For example, if $b_i[j][l] = 1$, then it means node P_j has reported to P_i that it has completed the A-Cast initiated by node P_l in Epoch 0. If there are at least $(t+1)$ 1s in the k_ith column of b_i, it means that there are $(t+1)$ nodes who report that they have received the A-Cast initiated by

node P_{k_i} in Epoch 0. At least one of these is report is from a correct node. So, from the termination condition of A-Cast (Lemma 6) all the correct nodes eventually successfully complete the A-Cast by P_k. □

Now we are ready to prove **theorem 1.**

Proof. There are at most n AR-Cast initiated in Epoch 0 of which $(n-t)$ are by correct nodes. From lemma 9 each of these succeeds with probability $q_{\text{succ}}^{n+2n^2} \geq 1 - e^{-\Omega(m\delta - \log n)}$. So all the correct AR-Casts succeeds with,

$$q_{\text{succ}}^{n^2+2n^3} \geq (1 - e^{-\Omega(m\delta - \log n)})^n, \tag{62}$$

$$\geq (1 - e^{-\Omega(m\delta - \log n)}). \tag{63}$$

Here inequality (63) follows from Bernoulli's inequality. Conditioned on this we show,

a. Correctness. To prove consistency we show that if a correct node P_i outputs v_i and a correct node P_j outputs v_j then $d(v_i, v_j) \leq 42\delta$. From step 4 of Epoch 2 of A-Agreement we see that,

$$v_i = w_i[k_i], \tag{64}$$

$$v_j = w_j[k_j]. \tag{65}$$

From lemma 6 we know that for $t < n/4$,

$$d(w_i[k_i], w_j[k_j]) \leq 42\delta. \tag{66}$$

This with (64) and (65) gives,

$$d(v_i, v_j) \leq 42\delta. \tag{67}$$

b. Termination To prove termination we have to show that every correct node P_i terminates with an output direction v_i.

To prove this we show that P_i eventually completes all the Epochs of A-Agreement. From Lemma 10 we see that P_i must enter Epoch 1 from Epoch 0. All the steps in Epoch 1 are of constant expected time. So, an honest player will eventually complete them and go to Epoch 2. Only in step 3 of Epoch 2 P_i waits for completion of A-Cast from P_{k_i}. But from Lemma 13 we know that this A-Cast eventually successfully completes. All the other incomplete A-Casts are then aborted at Step 5 and the protocol terminates with output v_i.

□