12-2019

Quantum consensus

Jorden SEET
Singapore Management University, jorden.seet.2016@sis.smu.edu.sg

Paul GRIFFIN
Singapore Management University, paulgriffin@smu.edu.sg

Quantum Consensus

Jorden Seet
School of Information Systems
Singapore Management University
Singapore, Singapore
jorden.seet.2016@sis.smu.edu.sg

Paul Griffin
School of Information Systems
Singapore Management University
Singapore, Singapore
paulgriffin@smu.edu.sg

***Abstract*— In this paper, we propose a novel consensus mechanism utilizing the quantum properties of qubits. This move from classical computing to quantum computing is shown to theoretically enhance the scalability and speed of distributed consensus as well as improve security and be a potential solution for the problem of blockchain interoperability. Using this method may circumvent the common problem known as the Blockchain Trilemma, enhancing scalability and speed without sacrificing de-centralization or byzantine fault tolerance [1]. Consensus speed and scalability is shown by removing the need for multicast responses and exploiting quantum properties to ensure that only a single multicast is required. We also leverage work done on the E91 quantum key distribution protocol [2] to securely transmit values and prevent a man-in-the-middle attack or system disturbance, enhancing confidentiality and integrity of transmitted information. Distributed ledger interoperability is explored by proposing a system to achieve a verifiable bridge for private transactions between a small private network and its corresponding consortium network. A proof-of-concept using IBM's Qskit is shown from which initial results appear to show a strong sensitivity to non-consensus which could be useful in many applications. The present practical feasibility of the above is shown and future work is explored.**

Keywords*— *Quantum Computation, Distributed Consensus, Distributed Ledger Technology, Blockchain, Byzantine Fault Tolerance

I. Introduction

Quantum computing has been theoretically proven to be able to solve various computational problems more efficiently than classical computers with the most significant example being the problem of Prime Factorization. The best published asymptotic running time in classical computers uses the General Number Field Sieve algorithm, which solves the problem in exponential time → O(e^((64/9)^1/3(log n)^1/3(log log n)^2/3)) for a bit number n [3]. Using quantum computing, we can utilise Shors algorithm to solve the same problem in polynomial time → O(nk) for some number n and k > 0 [4].

Distributed ledgers are a special type of distributed database that is able to achieve byzantine fault tolerant consensus. This allows various benefits such as enhanced data reconciliation across nodes and tamper-resistance of data. Classical permissioned distributed ledgers achieve consensus slowly due in part to the messaging system utilised has a 3-phase commit protocol (3PC), which achieves consensus in squared ($O(n^2)$) time-complexity [5]. Currently proposed quantum Byzantine agreements present consensus resolution of up to O(1) time-complexity, with varying pros and cons [6]. In this paper, we describe how we could utilise various aspects of quantum computing to accelerate the determination of agreement, or disagreement, of proposed states between participants of a distributed network.

Scalability in classical distributed ledgers is also a well-documented problem. Many BFT (Byzantine Fault Tolerant) and state-machine replication protocols are challenged as the number of nodes grows since the number of messages required to achieve consensus increases quadratically [7]. Even in schemes which follow a gossip protocol, such as the Bitcoins consensus mechanism, scalability has proven to be an issue with a confirmation throughput of 7 transactions/sec [8].

Security is also critical to blockchains. Significant work has been done on the quantum key distribution protocol E91 to securely transmit values and prevent a man-in-the-middle attacks or system disturbance, enhancing confidentiality and integrity of transmitted information. [3]

Another well documented problem for distributed ledgers is interoperability between different blockchains in general and in particular between heterogeneous blockchain platforms. One of the most significant challenges lies in private verifiability. Many entities wish to leverage the benefits of distributed ledgers, such as data reconciliation and tamper resistance, but must keep their data private due to confidentiality agreements and laws. Data reconciliation requires the verifiability of transactions, which if obfuscated can be solved with zero knowledge proofs. However, zero knowledge proofs currently have drawbacks which are discussed below.

II. System Design

A. Main Components

A distributed ledger comprises of several main components:

1. Nodes that run code that:
   a. Provides discovery and communication protocols
   b. Checks consensus (called mining in Ethereum)
   c. A stores of the agreed data
2. Networks between the nodes

In this paper we do not consider the discovery and communication protocols. Also, as we intend to only consider current available technology we do not consider storing data in qubits. We focus on the networks and the consensus algorithm only.

B. Quantum Networking with Quantum Computing

Key to achieving consensus in distributed ledgers is the messaging system between nodes. In our system, we propose the usage of a linear optical quantum system to connect our distributed ledger as has already been used for quantum key distribution. By using the qubits encoded on the optical

quantum system we can utilise quantum computation on those qubits as well as quantum communication. The probability wave functions of the photons will be utilised in calculating consensus.

C. The Network Communication System

In order to consider a realistic model of inter-node communication, three types of interconnections are described: Classical-Classical, Classical-Quantum and Quantum-Quantum.

1) Quantum-Quantum

The inter-quantum computer communication is for executing the consensus mechanism. This involves the sending of qubits from one quantum computer to another in the form of photons through linear optics.

2) Classical-Quantum

Each quantum computer is considered to be connected to a classical computer for storage and retrieval of the agreed data. This requires a means of communication and should be a secure channel to prevent any attacks such as a man-in-the-middle attacks.

3) Classical-Classical

For the quantum computers to know when to execute the consensus mechanism, each classical computer would have to feed the latest data to their respective quantum computers within the same time epoch. This requires a communication protocol between classical computers to update their interpretation of the network state at the same time. This could simply be done using a Gossip protocol on the classical networking layer [9] which is generally very efficient, with many-to-all communications shown to achieve an average time-complexity of O(n/log n) [10].

Both Classical-Classical and Classical-Quantum communication spaces could be eliminated if and when quantum computers are developed to the extent that it is practically feasible to store and process data within the quantum computer itself.

D. System Model

Referring to Fig. 1, we shall discuss the movement of a singular photonic qubit for our base case. Note that the saved state for each classical computer is empty as at this snapshot of time and consensus has not yet been reached. After each quantum computer has encoded their information into photonic qubits (or qubit in our base case), the quantum computer (QC) will create entangled duplicates of each qubit via the Hadamard gate and send the duplicates to the other known participants in the network. Due to the unique property of entanglement, this multicast only needs to occur once for both parties to synchronise their states, as opposed to a request-response style of multicasting used by classical BFT algorithms. This property alone halves the time needed to determine consensus.

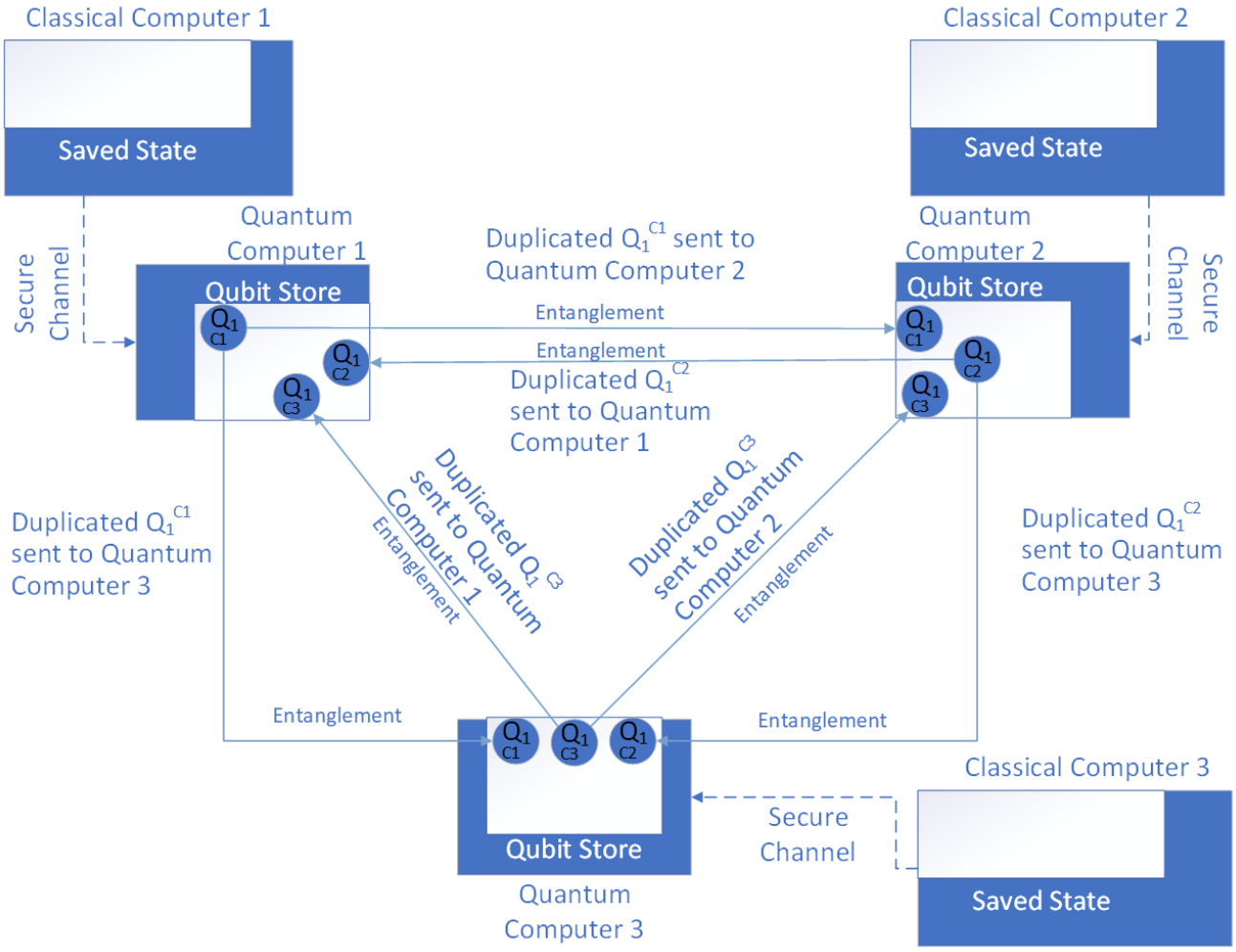

Fig. 1. System model.

III. Consensus Mechanism

Each QC will have their own qubits and the qubits sent from the other nodes. Here, we propose a method of achieving consensus by aggregating the wave functions of these qubits.

A. Wave Function

In the following section we consider a single qubit model at first for simplicity and then move onto a more practical multi-qubit model.

1) Single-Qubit Model

We propose that consensus could be determined through the sum of each photonic qubits wave function Ψ.

To calculate the Ψ of each photonic qubit directly, we could measure the qubits from within the quantum computer. The key idea is that the wave function of each qubit should be similar to each other. Hence, each value should only contain information consistent with each other, with zero/minimal unique identifiers. Via the superposition principle, these wavelengths, if congruent, would be a scalar multiple nΨ, where n is the number of participants in the consensus mechanism. In that case, we can calculate the discrepancy between photonic qubits by this algorithm:

$$\left(n\Psi_{random} - \sum_{k=1}^{n} \Psi_k\right) \tag{1}$$

2) Algorithm Breakdown

Equation (1) seeks to determine the difference between the ideal system state wave function and the actual system state wave function.

The ideal system state wave function is determined by (2)

$$n\Psi_{\text{random}} \tag{2}$$

Ideally, all participating quantum computers (nodes) should have congruent data, hence random selection would return a result consistent with any other node's state wave function. This would be proven via the actual system state wave function (3):

$$\sum_{k=1}^{n} \Psi_k \tag{3}$$

If all nodes have the same state wave function then each node's state wave function would be a scalar multiple of the systems state wave function. The resultant wave function would thus be 0.

However, if any node has a different state wave function, then that nodes state wave function would not be a scalar multiple of the systems state wave function. Thus, the difference between the systems state wave function and the sum of all nodes state wave function would not be 0. The greater the discrepancy between any nodes state wave function and the control, the larger the difference would be. To be byzantine fault tolerant, the result of the algorithm should not fall below a certain threshold that has not yet been determined.

The consensus algorithm thus multiplies a randomly chosen wave function n times, with n being the number of participating nodes in the system. It then subtracts the sum of wave functions of each qubit in the system. In a perfectly coherent system with all wave functions being the same, the end result is expected to be 0 or very close to 0. As the system increases in incoherence, the end result is expected to grow larger.

3) Multi-Qubit Models

In practicality, single-qubit models are unlikely to be sufficient for information transfer. As of the time of writing, implementing 2-qubit gates is still considered to be a hard problem. This makes it hard for multi-qubit entanglement. One possible solution would be the One-Way Quantum Computation model, which exploits quantum correlations in cluster/graph states to pre-generate all the entanglements needed [11]. It then uses projective measurements and feed-forward schemes to implement multi-qubit quantum computation. Incidentally, a linear optics quantum computer based on this cluster state measurements has been proposed, which could be used for the one-way quantum computing model [12].

In the above consensus mechanism, the calculations for multi-qubit models remains largely the same. The main difference is that each in the previous model, we calculate the wave function from a single qubit from node k. In a multi-qubit model, we factor in that each node k may have multiple qubits. Hence, the formula is slightly revised to (4).

$$\left(\sum_{j=0}^{m}\left(n\Psi_{rj} - \sum_{k=0}^{n} \Psi_{kj}\right)\right) \tag{4}$$

Referring to (4) and Fig. 2, m is the total number of qubits each node has initially. In the Fig. 2, this refers to the top row of qubits for Quantum Computer 1, the middle row for Quantum Computer 2 and the centre row for Quantum Computer 3. As the data communicated across the nodes are expected to be the same, an IF condition can be applied in the consensus algorithm to ensure that the number of initial qubits per quantum computer is the same.

In equation (4), n refers to the number of nodes in the system. r refers to a random value within the number of nodes in the system.

Essentially, the algorithm operates similarly to the Single-Qubit model, except that instead of deriving the node's wave function from one qubit, we derive the wave function from the cumulative sum of all qubits belonging to that node.

In Qiskit, the code to achieve this would be as such:

 # deriving each qubit set's overall waveform for circuit in circuits: #for each quantum circuit

#retrieve wave function of qubit set

result = execute(circuit , backend=statevector backend).result()

#returns 2^n complex amplitudes

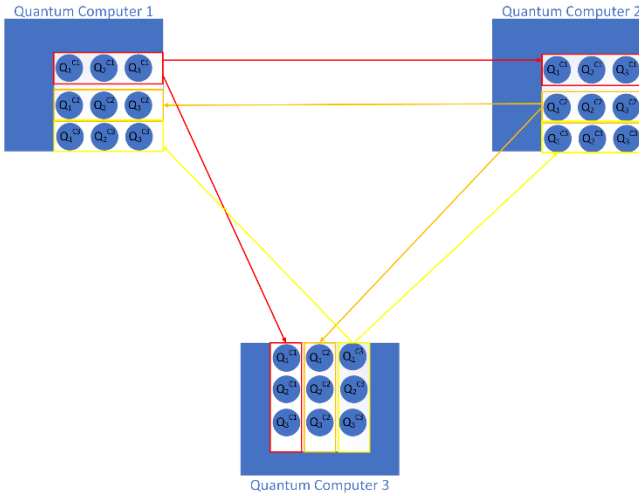

Fig. 2. Multi-qubit model

waveform = result.get statevector(circuit)

waveforms.append(waveform)

totalAmplitude = [] for waveform in waveforms:

totalSum = 0 for amplitude in waveform:

 # Deriving qubit set's waveform by summing up all qubit's # waveforms

totalSum += amplitude

#storing each qubit set's overall waveform totalAmplitude.append(totalSum)

 # applying consensus algorithm summedAmplitude = 0

totalCoherence = 0

for amplitude in totalAmplitude:

 # deriving actual overall system state summedAmplitude += amplitude

 # using quantum random number generators to randomly select a node randomWaveform = qrng.get random int32()%len(waveforms)

 # deriving ideal overall system state

controlWave function = len(waveforms)∗ totalAmplitude[randomWaveform]

for amplitude in totalAmplitude:

 # deriving difference between each qubit set's system state

 # and the ideal system state

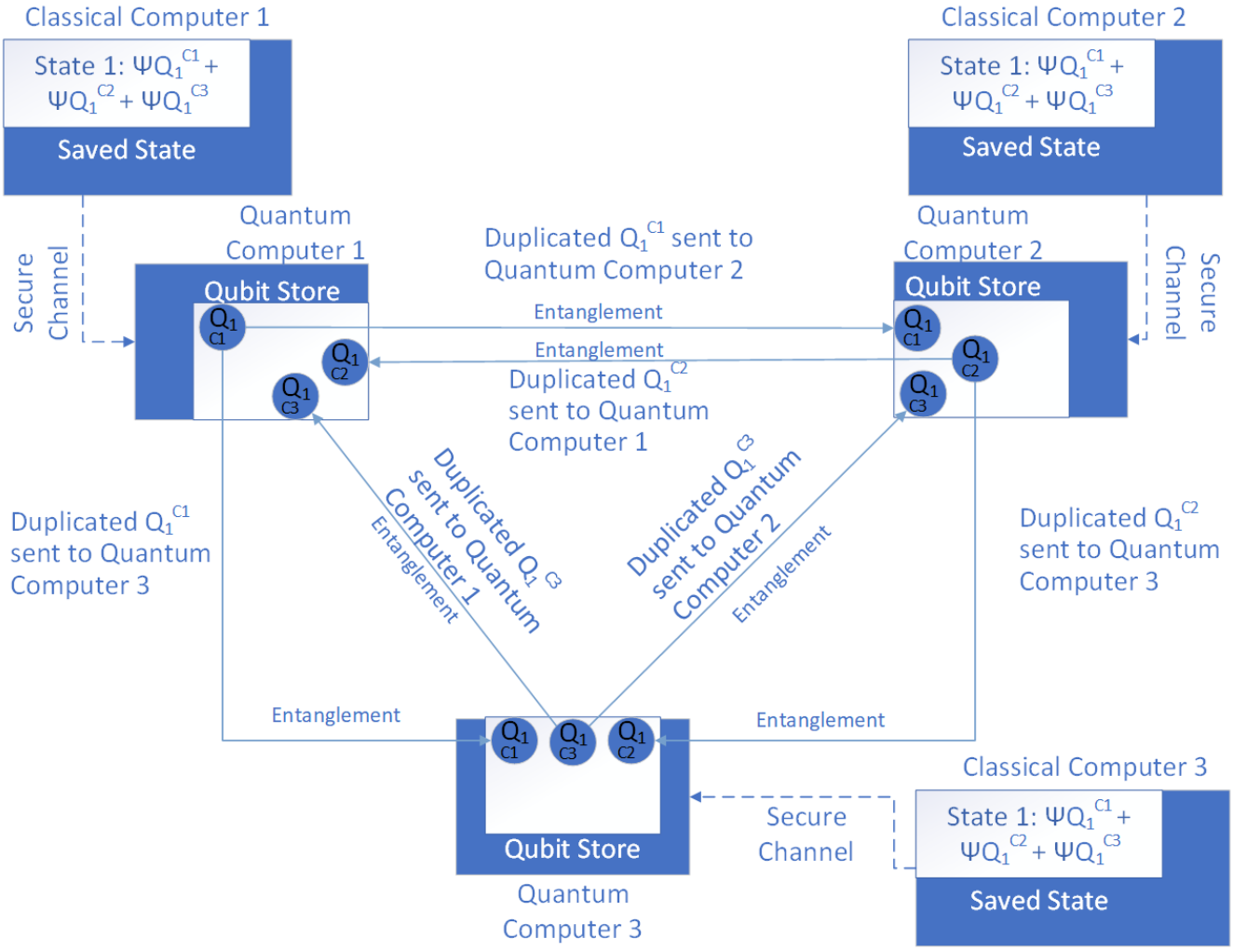

Fig. 3. Saved states.

coherence = controlWave function − summedAmplitude #accumulate all state differences

totalCoherence += coherence

#the closer the value is to 0, the more coherent the network state is

print(totalCoherence)

IV. Post-consensus

The result of a consensus mechanism can be summed up into two states: Tolerable or Intolerable.

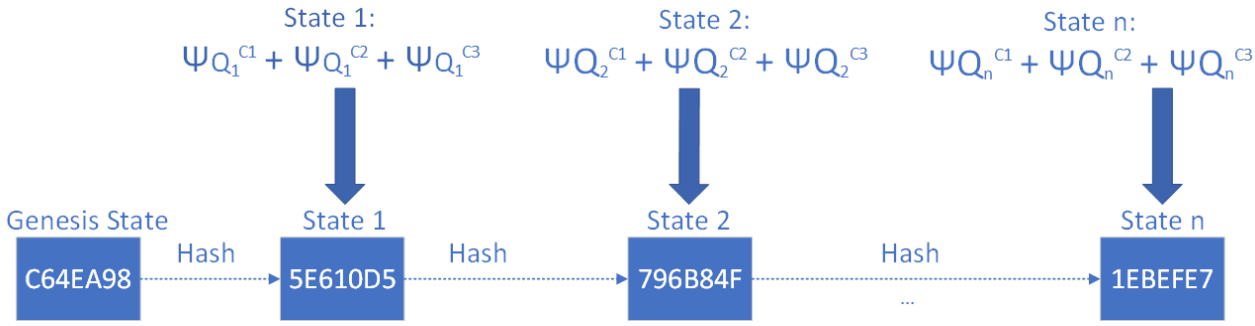

Fig. 4. Chain of states.

Tolerable means that the consensus mechanism has agreed that, with a result above the threshold, all nodes agree on a similar state of the network. In this case, the state will be represented in bits and recorded in a classical computer as a Saved State (Fig. 3). This allows the classical computer to have a track of states for auditing purposes, as well as allowing states to be chained to form an immutable, ordered series of hashes similar to how blocks are chained in a blockchain (Fig. 4).

Intolerable means that the consensus mechanism produces a result below the threshold, i.e. all nodes are unable to agree on a similar state of the network. In this case, the quantum computer will present a value to the classical computer that alerts the user of the discrepancy for error checking. The saved state on the classical computer would not be updated.

V. Benefits over the Classical model

A. Distributed Consensus Speed

Classical verification of state requires a ”request-response” style of communication. The verifier requests for the perspective of state held by a node, which responds with their perspective of state. In our described model, we leverage quantum entanglement to reduce the time complexity of verification by half. As the unique property of entangled particles is, as Einstein coined it, ”spooky action at a distance”, the verifier simply needs to receive the entangled qubit. Verification is then done locally, by comparing the wave functions of all qubits. In this case, entanglement of qubits mean that knowing the wave function of a duplicated qubit is equal to knowing the wave function of the original qubit.

Quantum entanglement is key in this case - the classical counterpart of this mechanism would be susceptible to man-in-the-middle attacks since the original state and sent state are not entangled, and thus can be manipulated in isolation. This would lead to incoherent perspectives of state across the network, which could lead to double-spending attacks before future iterations of the consensus mechanism reconciles the information [13]. With quantum entanglement, the original state and the sent state cannot be manipulated in isolation, and any attempt at manipulation would disturb the system [2]. Thus, consensus can be reached in half the time required as opposed the classical model.

1) Distributed Ledger Scalability

Due to reducing the time complexity of state verification by half, for any number of participating nodes, we are able to complete approximately twice the number of state verifications in a given amount of time.

TABLE I. The number of messages sent for classical and quantum communication

Consensus Type	*Number of nodes*	*Number of messages*
Classical Consensus	3	6
Quantum Consensus	3	3
Classical Consensus	10	90
Quantum Consensus	10	45
Classical Consensus	1000	999,000
Quantum Consensus	1000	499,500

Generally, the time complexity of state verification is proportional to the number of messages sent. In this case, we can see clearly that with state verification of the quantum model taking half the time of the classical model, we can thus increase verification throughput by approximately twice without sacrificing on decentralisation or fault tolerance, circumventing the trade-off known as the Blockchain Trilemma [1]. Thus, we conclude that the proposed model is linearly more scalable than the classical counterpart.

B. Distributed Ledger Interoperability

Classical distributed ledgers suffer for an inability to operate homogeneously. Here, we define homogeneity between distributed ledgers as the ability of distributed ledgers to communicate despite differences in data structures and operating mechanisms. The main contributors to this heterogeneity are a lack of common data formatting standards and the usage of differing privacy preserving protocols.

1) Lack of common data formatting standards

Similar to the issues faced by classical databases in distributed systems, differing naming conventions, data structures and value types can cause inconsistencies. Two general solutions have been proposed: a multi-database system and federated databases [14]. These solutions work because there is not much concern about privacy and these autonomous databases are usually operated by a single entity, or entities that have rights to the information. As such, access control is generally sufficient here, which would not be enough for our situation which necessitates communication with entities that may not have the right to certain information. Additionally, these solutions requires the use of a trusted intermediary. In the multi-database system this is done by software and the system administrator, whereas in the federated database heterogeneity must be manually resolved and integrated via a federation dictionary. This limits interoperability as there would be significant friction with each additional database integration. For distributed ledgers, this problem extends to the triple-entry accounting of shared resources [15]. Currently, the two leading methods of accounting are Bitcoins UTXO (Unspent Transaction Output) model [16] and Ethereum's Account-Based model [17]. While balances in an Account-Based model are updated linearly, UTXO records balances as the result of unspent transaction outputs, thus recording the transaction audit trail in a drastically different structure. Recording and reconciling new data, as well as cross-verification of historical data across distributed ledgers, can thus be a huge challenge.

2) Differing privacy preserving protocols

One of the fundamental reasons why distributed ledgers remain heterogeneous is a need for privacy when running a distributed ledger network. We do not want to expose sensitive information to other networks, and thus the usage of differing hashing and encryption algorithms serves to maintain this privacy via obfuscation. A clever workaround design pattern called the Hashed time-lock contract enables different distributed ledgers to securely communicate via the usage of an escrow [18]. This, however, requires the two different distributed ledgers to utilise the same hashing algorithm so that the cryptographic proof is verifiable to both distributed ledgers. This also means that, as of the time of writing, there is no way to directly communicate between two distributed ledgers of different hashing algorithms.

C. Post-Quantum private verifiability

Practically, there is an incentive for heterogeneous distributed ledgers to communicate, especially in inter-entity collaborations. However, the key issue of privacy remains where entities may not want to expose certain sensitive information to collaborators and thus obfuscate their data. However, validators from collaborating entities may wish to verify the integrity of activity done with these obfuscated data, without knowing these contents of such data. As such, zero knowledge proofs have been used to great effect in these circumstances [19]. Currently, the three leading zero-knowledge protocols in use for distributed ledgers are zk-SNARKs, zk-STARKs and Bulletproofs. However, there are issues with implementing these protocols in a post-quantum setting. Zk-SNARKS [20] and Bulletproofs [21] utilise Elliptic Curve pairings, which are known to be vulnerable to quantum cryptographic attacks. Whilst zk-STARKs are quantum secure thanks to using both collision resistant hashing and a random oracle model, their proof size ranges from 45-200 kilobytes) [22] In contrast, zk-SNARKs has a proof size of approximately 200-300 bytes, while Bulletproofs has a proof size of around 1.3 kilobytes. As such, zk-STARKs is extremely computationally expensive, and would bottleneck many classical or quantum systems.

1) Quantum Solution to privacy-preserving

Leveraging on the proposed mechanism, we propose a solution for privacy-preserving interoperability that will allow interoperability on a need-to-know privacy basis. There are two key enablers to this design. The first is a method to generate second pre-image and collision resistant quantum wave functions to replace classical hashing algorithms. The second is quantum entanglement of qubits before and after applying the method above, between the private network and the consortium network.

Referring to Fig. 5 and Fig. 6, we assume that qubit set A (multiple qubits) contains information that is private to QC1, 2 and 3. In this case, it forms a private network where they go through the proposed consensus mechanism. If consensus is achieved, a quantum hashing function will then act on the qubit to produce a second pre-image and collision resistant quantum wave function, which we refer to as qubit set B. [23]. We can then apply the Hadamard and CNOT gates to create entangled duplicates of this wave function and disseminate it across the consortium network for consensus.

The described scheme allows us to ensure the integrity of transactions as the broadcasted qubits are entangled with the original qubits, whilst remaining private to those outside the private network. This means that one cannot manipulate qubits on the consortium network without interfering with the state of the qubits in the private network, and vice-versa. Such interference would cause discrepancies between the current system state and the latest saved state, and would be flagged.

Additionally, due to the E91 quantum key distribution protocol, we can ensure the provable integrity of communicated data. This allows us to circumvent Zero-knowledge proofs as there is now a method to verify the integrity of transactions whilst keeping it private.

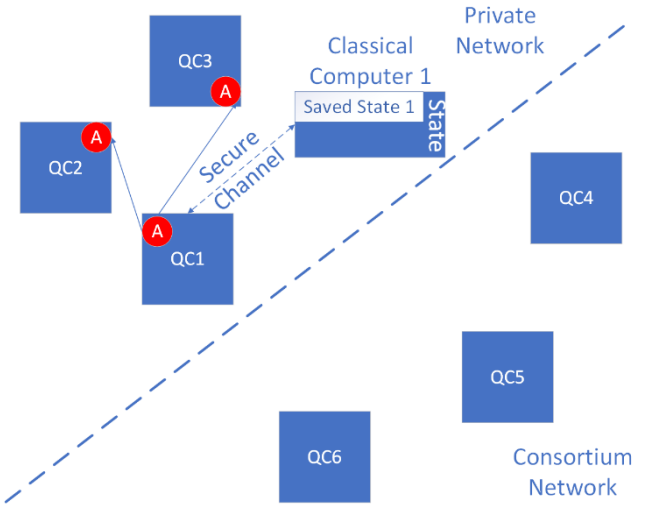

Fig. 5. Consensus within a private network.

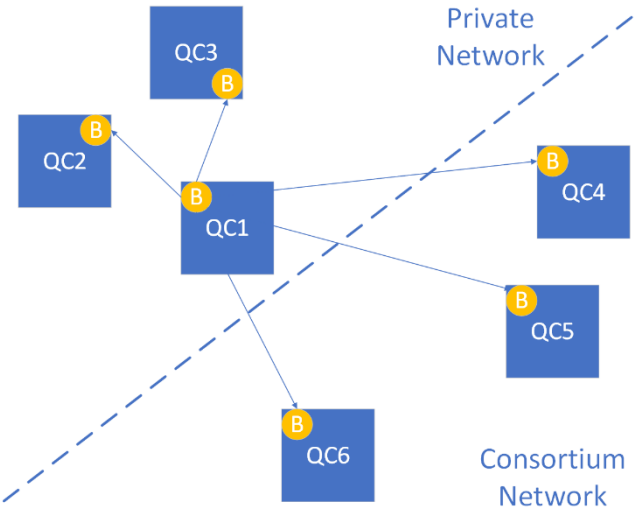

Fig. 6. Disseminating hashed wave functions to non-privy nodes.

Quantum entanglement is essential to this process. On a classical system, this can be replicated using hashes one can achieve classical consensus and produce a hash of the saved state to the consortium network. The issue lies with that we cannot prove that the hash submitted to the consortium network is the hash of the private transaction in the private network. In the proposed model, we can prove the connection between the private transaction and the hash submitted by measuring one qubit from both. We should observe that, similar to traditional proofs of Bell States, the value of one qubit should always be either identical or opposite to the other qubit. Thus, we can prove the connection without divulging confidential information in the private transaction.

D. Mitigating Partitioning Attacks

Some classical distributed ledgers are vulnerable to the partitioning attacks [24]. This is due to how distributed ledgers using probabilistic finality reconcile differences in state, where the most commonly accepted state would be deemed as the ”single source of truth”. Thus, by partitioning networks and creating multiple versions of the state, when these partitions re-join each other, only one of these partitions would have their state accepted as the ”single source of truth”. This would create opportunities for double-spending attacks to occur [13].

We assume that in the future, qubits would stabilise sufficiently to act as a means of storage, similar to how bits are utilised today. In this timeline, we describe how quantum entanglement can be used to mitigate data tampering via partitioning attacks.

1) Explaining the attack

In large distributed ledger networks, it is a common practice to limit the number of outgoing connections per node. This is mainly for performance purposes as maintaining connections with every other node increases the number of communication streams quadratically with each node added, thus utilising more time and computational resources. It is commonplace to see a gossip protocol used in such networks, where a limited number of outgoing connections communicate the state to other nodes, which in turn follows suit until all nodes in the network is updated with the latest state. For example, Bitcoin uses 8 outgoing connections [25].

In the network diagram (Fig. 7), we assume the system is configured to have two outgoing connections between nodes. In this snapshot of time, Quantum Computer 3 and 5 from the network are communicating with Quantum Computer 6 and 10 from the consortium network. What an attacker could do is partition the communicating nodes from the other nodes, as represented by the thin dashed lines in Fig. 7.

In classical systems, given that information of nodes from different partitions are isolated, an attacker could create different perspectives of state by executing a partitioning attack, then tamper with data from the largest partition. This could be done via source-code manipulation or rewriting the history of transactions with the attacker’s ideal version.

Referring to Fig. 7, the central partition could first merge with the partition on the left. This merged partition would follow the perspective of state shared by the central partition, given that it has 4 nodes as opposed the 3 from the left partition. Now the manipulated perspective of state would be shared with 7 nodes, and by iterative merging of smaller partitions, the manipulated perspective of state would be shared across all nodes.

2) Preventing the attack with Quantum Entanglement

The vulnerability in the classical model is due to the fact that data from nodes of different partitions are isolated. Manipulation could be done on one node without affecting the other, thus leading to differences in the perception of state. However, in the quantum model, manipulating a qubit from a node of one partition would still cause an observable disturbance between nodes who share entangled qubits, even if they are from different partitions. This allows the system to take action and prevent the manipulated perspective of state from being recorded as a saved state in the first place, thus preventing double spending attacks.

VI. Summary

We have shown that a quantum distributed ledger could enhance not just existing intra-network communication, but also for interoperability models. Speed and scalability are improved along with security. Therefore, there are vested benefits for further exploration of quantum distributed ledgers.

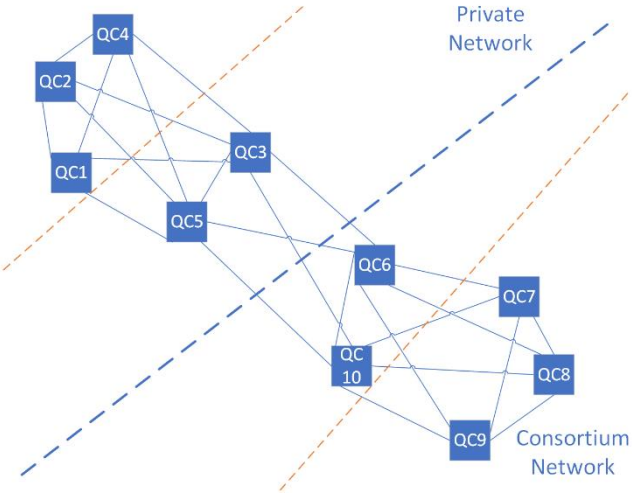

Fig. 7. Example network diagram of partitioned networks.

With the rapid growth in technology, it is possible that quantum computers could become commercialised, allowing corporates to utilise quantum algorithms for competitive advantages. An example is the Dynamic Portfolio Selection, the NP-Hard problem which quantum annealing could solve with two D-Wave chips [26]. In such an event, quantum-enabled distributed ledgers could be utilised in various ways such as know-your-customer (KYC) processing using data held across companies.

We also recognise that zero knowledge proofs, while extremely promising, are generally not quantum secure as of writing. Quantum secure zero knowledge proofs requires enormous cryptographic proof size and computing resources, which would likely throttle the communication network. As distributed ledgers are reliant on the communication network to achieve consensus, we find that existing solutions to be rather unfavourable.

VII. Future Work

To realise this model, more work is needed on determining a byzantine fault tolerant threshold for our quantum consensus mechanism which from first results appears to be highly sensitive to non-coherence. Actual implementation of the consensus model would require a quantum network of quantum computers for testing, which is unavailable to us as of time of writing. Finally, further research on quantum hashing functions is required to discover the optimum algorithm and implementation for various use cases.

References

[1] J. Abadi and M. K. Brunnermeier, Blockchain economics, Working Paper.

[2] A. K. Ekert, Quantum cryptography based on bells theorem, Physical Review Letters 67 (1991) 661.

[3] C. Pomerance, A tale of two sieves, Biscuits of Number Theory 85 (2008) 175.

[4] P. W. Shor, Polynomial-time algorithms for prime factorization and discrete logarithms on a quantum computer, SIAM review 41 (1999) 303–332.

[5] M. Castro, B. Liskov, et al., Practical byzantine fault tolerance, in OSDI, volume 99, pp. 173–186.

[6] M. Ben-Or and A. Hassidim, Fast quantum byzantine agreement, in Proceedings of the thirty-seventh annual ACM symposium on Theory of computing, ACM, pp. 481–485.

[7] M. Vukolic, The quest for scalable blockchain fabric: Proof-of-work vs. bft replication, in International Workshop on Open Problems in Network Security, Springer, pp. 112–125.

[8] K. Croman, C. Decker, I. Eyal, A. E. Gencer, A. Juels, A. Kosba, A. Miller, P. Saxena, E. Shi, E. G. Sirer, et al., On scaling decentralized blockchains, in International Conference on Financial Cryptography and Data Security, Springer, pp. 106–125.

[9] M. Jelasity, Gossip (2011).

[10] B. S. Chlebus, D. R. Kowalski and M. A. Rokicki, Average-time complexity of gossiping in radio networks, in Structural Information and Communication Complexity, 13th International Colloquium, SIROCCO 2006, Chester, UK, July 2-5, 2006, Proceedings, pp. 253–267.

[11] D. E. Browne and H. J. Briegel, One-way Quantum Computation - a tutorial introduction, arXiv e-prints (2006) quant–ph/0603226.

[12] D. E. Browne, T. Rudolph, Resource-efficient linear optical quantum computation, Phys. Rev. Lett. 95 (2005) 010501.

[13] Z. Peng and Y. Chen, All roads lead to rome: Many ways to double spend your cryptocurrency, arXiv:1811.06751

[14] W. Litwin, L. Mark and N. Roussopoulos, Interoperability of multiple autonomous databases, ACM Comput. Surv. 22 (1990) 267–293.

[15] Y. Ijiri, A framework for triple-entry bookkeeping, The Accounting Review Vol. LXI, No. 4 (1986).

[16] S. Nakamoto, Bitcoin: A peer-to-peer electronic cash system, Cryptography Mailing list at https://metzdowd.com. (2009).

[17] D. Wood, Ethereum: A secure decentralised generalised transaction ledger, EIP-150 REVISION (1e18248 - 2017-04-12)

[18] M. Herlihy, Atomic cross-chain swaps, in Proceedings of the 2018 ACM Symposium on Principles of Distributed Computing, PODC '18, ACM, New York, NY, USA, 2018, pp. 245–254.

[19] J.-J. Quisquater, M. Quisquater, M. Quisquater, M. Quisquater, L. Guil- lou, M. A. Guillou, G. Guillou, A. Guillou, G. Guillou and S. Guillou, How to explain zero-knowledge protocols to your children, (1989), 628-631. 10.1007/0-387-34805-0_60.

[20] N. Bitansky, R. Canetti, A. Chiesa and E. Tromer, From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again, in Proceedings of the 3rd Innovations in Theoretical Computer Science Conference, ITCS '12, ACM, New York, NY, USA, 2012, pp. 326–349.

[21] B. Bnz, J. Bootle, D. Boneh, A. Poelstra, P. Wuille and G. Maxwell, Bulletproofs: Short proofs for confidential transactions and more, in 2018 IEEE Symposium on Security and Privacy (SP), IEEE Computer Soci- ety, Los Alamitos, CA, USA, 2018.

[22] E. Ben-Sasson, I. Bentov, Y. Horesh and M. Riabzev, Scalable, transparent, and post-quantum secure computational integrity, IACR Cryptology ePrint Archive 2018 (2018) 46.

[23] Y.-G. Yang, P. Xu, R. Yang, Y.-H. Zhou and W.-M. Shi, Quantum hash function and its application to privacy amplification in quantum key distribution, pseudo-random number generation and image encryption, Scientific reports 6 (2016) 19788.

[24] A. Maria, Z. Aviv and V. Laurent, Hijacking bitcoin: Routing attacks on cryptocurrencies, in Security and Privacy (SP), 2017 IEEE Symposium on, IEEE.

[25] B. C. Team, Bitcoin core, retrieved from https://github.com/bitcoin/bitcoin/blob/master/doc/reduce-traffic.md, 29 August 2019.

[26] R. Orus, S. Mugel, E. Lizaso, Quantum computing for finance: overview and prospects, arXiv preprint arXiv:1807.03890 (2018).