Network stack and Protocol Design and Implementation of a Multipartite Entanglement-based Quantum Network

Arahant Ashok Kumar (aak700)

Abstract

I present a network stack and protocol design for multipartite entanglement-based Quantum Network. This is an attempt to arrive at a comprehensive network stack design with all the protocols included. There have been several incremental works regarding different aspects of this domain in the past 2 decades, and I aim to build on them.

I also back my design with an implementation and a simulation. I‘ve used Qutech’s SimulaQron for this purpose. It provides a fairly open and programmable framework to implement, which abstracts the nature of the underlying quantum technology.

I’ve incorporated a fairly polymathic design approach. I’ve retained the central concepts in classical networking - network stack, packet (datagram) design, layer protocols - and included several diverse design concepts from Operating Systems, Distributed Computing (HDFS), Programming Languages, Compiler design, making it a very effective and rich design.

Introduction

Throughout this research I follow two main papers: [1] and [2]. [1] gives a bottom-up, fairly realistic, algorithmic, low level stack and protocol design and implementation framework. It takes into account the quantum hardware strengths and limitations. [2], however, is theoretical, but gives an elaborate and well thought out design to build upon. This is top-down design approach. These two, therefore, give diverse perspectives and design considerations to build upon.

I’ve focused on designing a quantum network framework which supports multipartite entanglement between network nodes. Different entanglement types - bell pairs, multipartite (GHZ, W, etc.) - provide different benefits, but at added costs in terms of resource consumption. I use this differentiation to design a network architecture which uses specific entanglement types for specific applications. I’ve also designed a framework which allows dynamic conversion between entanglement types depending on currently available quantum resources and the protocols.

As mentioned earlier, I’ve retained some classical networking concepts: network stack, packet (datagram), layer protocols - and incorporated some external concepts from

Operating Systems, Distributed Computing (HDFS), Compiler design and Programming languages.

My design is broadly divided into two aspects: *Quantum Computing packages* and *Network Stack design*. Modularising the design in this sense leads to a more resuable, robust and an effective design. Every layer uses different Quantum Computing packages based on its responsibilities. Although, most of the Quantum operations will happen in the bottom 2 layers, as that made the most design sense. As outlined in [1], the functioning of the network will depend on these quantum operations, which depend on the underlying quantum computing hardware and its capabilities. Although the design will depend on quantum computer, it is fairly versatile which enables it to be configured it to different ecosystems as well.

In terms of network layers, my focus is on the lower layers in the network stack and in terms of routing, it's intra-domain networking. While I try to make use of both the approaches from [1] and [2], it seemed prudent to design lower layers first. Regarding the intra-domain networking: There are many organisations - academic and commercial - which are working on different aspects of quantum computing simultaneously. It would be logical to deduce that each will develop their own version of Quantum Autonomous Systems and network. Therefore, just as the first designers of the internet, the existing quantum network design must support such diversity in quantum computing.

However, within this boundary, things will be, more or less, homogeneous. As a result, it only seems prudent enough to design layers and protocols for intra-domain networking first and then address the inter-domain aspect later.

Quantum Computing concepts

Qubit

Qubit is the quantum equivalent of a classical bit, although completely different in nature. It does store one unit of quantum information, in terms of states. Information is stored as a linear combination of orthogonal bases. A single qubit, |Q>, is most commonly represented in the {|0>,|1>} and {|+>,|->} bases.

Entanglement

Mathematically, 2 qubits are entangled if their entangled state cannot be expressed as a tensor product of the 2 individual qubits. Measurement outcomes of measuring the two qubits in any of the Bell states in the standard bases are either perfectly correlated or perfectly anti-correlated. Entanglement with 2 qubits are bipartite (Bell pairs) and those with more than 2 qubits are multipartite (GHZ, W).

Importantly, a Bell state can be converted into another one by using local unitary transformations and classical communication (LOCC). Moreover, Bell states can also be used to develop several informational tasks such as superdense coding and quantum teleportation. In the presence of bit-flip noise, GHZ states allow correction of error. This way, it is possible to improve fidelity.

Heralded entanglement

In [1], they've devised a method to signal the participating nodes when an entanglement state (channel) has been successfully generated. This heralding signal is transmitted to the nodes. Such heralding allows long-distance quantum communication without exponential overheads. This also helps in bookkeeping of the participating nodes in an entangled link and the state of the link itself.

Fidelity

In any real implementation of a quantum network, the generated entangled states will always differ from the perfect Bell states due to noise in the system. Fidelity measures how close a realised entangled state is to the idea state. The Bell state to be useful has to have a fidelity value.

Fidelity of a single instance of a quantum state cannot be measured. If many states are generated in succession, then fidelity can be estimated by measuring the quit error rate (QBER). For a fixed basis, X, the QBER_x is the probability of receiving equal measurement outcomes when measuring these qubits in this basis.

Decoherence

Another important fact about qubits generated today is that they cannot be stored indefinitely. They decohere very quickly. Quantum memories, which is a storage of qubits, are inherently noisy. This noise encountered by a qubit increases with the duration of storage. As a qubit encounters more noise its fidelity decreases.

Noise is also encountered in the channel. As the distance covered by a qubit increases its fidelity decreases. This practically limits the storage for long durations and transmission of qubits over long distances.

Purification

Purification raises the fidelity of an entangled state, essentially performing error correction, taking advantage of the specially-prepared initial state of the qubits. Purification takes two Bell pairs and attempts, via LOCC, into one higher fidelity pair. This is a resource expensive process.

Quantum Teleportation

Teleportation is a mechanism which makes use of entanglement between two participating entities to teleport qubits between them. Because of the no-cloning principle, when the qubit is teleported, its destroyed at the sender and recreated at the receiver. The qubit to be teleported is entangled with sender's half of the bell pair, and both these are measured. This results in one classical bit and destroys the quantum state of the qubit. When these two bits are sent to the receiver (2nd entity), they use them to perform LOCC on their half of the bell pair, eventually recreating the qubit from the sender (1st entity).

Entanglement swapping

This is an extension of quantum teleportation. In the previous case, if the receiver is entangled with a 3rd entity, then the previously recreated (or teleported) qubit from the 1st entity can be further teleported to the 3rd entity by following the same steps, effectively, teleporting the qubit from the 1st to the 3rd entity, without being direct entangled.

Design considerations

I've attempted to strike a balance between the existing quantum computing technology limitations and the larger ideal network goals.

Quantum Computing technology

The current state of quantum technology makes it difficult to develop a network framework at par with the classical network. The fidelity of the generated entanglement channel and the information qubit(s) is inversely proportional to the distance travelled, and the duration of storage. In addition, errors also creep in when qubits are generated. This effectively limits the applicability of quantum computing for communication. Given the laboratory conditions under which a lot of the experiments and research in this domain has been conducted, it further limits the real world applicability.

Communication medium

An important aspect is the physical medium between nodes over which qubits will be transported. A lot of the experiments are conducted over optical fibres and several other are conducted over air (high altitude atmosphere, between a satellite and ground stattion, as in [china]). But, that would be abstracted away from the network, as this would be handled by the underlying quantum computer.

Multipartite entanglement support and dynamic entanglement conversion

Given the advantages the multipartite entanglement hold, it makes logical sense to design a network architecture which supports this. However, given how resource expensive it is to generate them, it might not be feasible to establish a multipartite entanglement channel between nodes at all times. In such cases, the network must dynamically adopt and convert between entanglement types.

With this in mind, I have arrived at a design which enables dynamic and instant conversion between entanglement types, depending on the available quantum resources, protocols, applications etc.

Intra-domain networking

Fortunately or unfortunately, We find ourselves in a situation similar to when the classical internet was touted all those decades ago. As [3] illustrates, one of the design goals was to enable integration and communication between autonomous systems which were configured differently. Today, there are multiple entities - corporate, academic and government - which are working on different types of quantum computing technologies. There is the superconductor-based qubit generation, NV based, topological qubits etc. It is only logical to deduce that each will develop its own quantum computing ecosystem and we need to design a network framework which would enable communication and integration between these ecosystems.

Focusing on the intra domain networking provides us with an added benefit: although I'll have to stick to a particular quantum ecosystem, it will in contrast enable me to arrive at a more open ended design, as the communication is based on entanglement and teleportation.

Network Stack

Considering the previous design requirement I needed to design a relatively open source network stack from the ground up, which could be implemented across different quantum ecosystems, hopefully with minimal change. In this regard, each layer on the network stack would essentially be a set of fairly generic APIs and protocols.

My focus in this research will be limited to intra-domain networking which corresponds to the lower layers: physical layer, connectivity layer, link layer and network layer. Most of the operations related to quantum computing would be part of the physical layer. This would interact with the underlying quantum computer - which differs for every ecosystem - in generating qubits, entangled bits and performing numerous other quantum operations.

Design goals

When the classical internet was designed, as illustrated in [3], they had several design considerations which influenced the design on the protocols.

1. Confidentiality

This is one quality (among others) which results in a secure networking framework. The very nature of qubit and teleportation mechanism makes this possible.

When an entanglement channel is established between two nodes, then any qubit teleported between them can only be decoded by them, as the teleported qubit is entangled with the channel. Therefore, any teleported qubit, if intercepted, cannot be decoded without the other half of the entangled channel. This results in a highly confidential communication.

2. **Integrity**

The very nature of a qubit makes quantum communication highly integral. Information stored in a qubit isn't known until it's measured. Measurement causes the qubit to collapse to one of the base states. Only, we don't know which one. The only way to decode the original qubit is through the teleportation mechanism. So, if anyone tries to sniff the qubit it collapses and the information is lost, thus resulting in a highly integral communication.

3. **Accountability**

As [1] illustrates, the heralded entanglement mechanism provides a very reliable way to establish channels between nodes with accountability. And given the transparency in the teleportation mechanism, it results in a highly accountable network.

4. **Quantum Computing ecosystems**

The classical internet was designed to support multiple types of communications, as the designers didn't what would run on the Internet. However, in Quantum Networks, there might be multiple different protocols, but the underlying fundamentals which enables information exchange, happens through entanglement and teleportation. Sn entangled channel needs to be established between communicating nodes, over which qubits can be transferred (or teleported). Irrespective of the type of entanglement, the quantum technology used to generate entanglement, functionally it will remain the same. My design, therefore, is with functionality as the variable.

Technically speaking one can send a qubit to another point, but to be able to decode the information contained within, an entanglement channel is necessary. It would therefore be safe to design a well-engineered entanglement-based network despite the differences in underlying quantum technologies.

Design

I've adopted a modularised approach in my design. This concept, I drew from programming languages. For decades, many languages such as Java, Python etc have adopted this approach, which makes them extremely versatile. I therefore decided to incorporate this into my design approach.

My entire design is divided into different aspects: quantum computing components (modules), network stack including the protocols, and the packet design.

Quantum Computing components

Entanglement channel

As this is an Entanglement-based communication, this is arguably the most important component. Entanglement is first attempted between the participating nodes, over which information (qubits) is teleported. This would be responsible for establishing entanglement links between nodes. The type of entanglement would depend on the application, protocol, available resources etc. It would be essentially an API (a class or a module), which would accept arguments such as entanglement type, participating nodes, priority etc. This would also be a timeout.

Purification
Given the error in the system, the fidelity of the channel must be over a threshold for the quality of the information and the channel. This is where purification is performed on the generated entangled bits (ebits). Once an entangled state with a minimum fidelity level is generated a heralding signal is generated which is then transmitted to the participating nodes, following which the respective ebits are also sent. Now, the channel is ready to teleport information qubits.

Subcomponents
This would composed subcomponents: entanglement generation, increasing purification, generate heralding signal, swap entanglement with further nodes and teleportation of qubits.

Quantum computing packages

This is where the rubber hits the road. This is the prime module which performs all of the quantum computing operations, as this is the only module which is connected to the underlying quantum computer and has access to its resources. All of the other modules must call upon the APIs provided by this module to access quantum resources and computing operations.

This effectively abstracts away the underlying variations in quantum computing technologies. The APIs would remain the same but their implementations would differ depending on the quantum computer/ technology. Thus, once designed, developed and refined, the whole networking architecture would remain the same and theoretically could be replicated across different quantum computing ecosystems.

As an example, I could implement this module with SImulaQron simulator and design the whole architecture. Then, if I decide to change the underlying implementation using a different simulator, it wont affect the network design in any way. Only, the performance and metrics would change.

This component performs some of the basic quantum operations such as:

1. Generating qubit - This submodule provides the API which is to be used to request a qubit from the underlying quantum computer. This API call triggers the computer and once a qubit is generated, it returns with a pointer to it.

2. Generating Entanglement - Once you have the qubits, generating ebits is a sequence of unitary operations which entangles the participating qubits. So, this module would invoke operations in the Unitary Operations submodule.

3. Entanglement purification: [This paper] provides a purification scheduling algorithm to increase the fidelity of the generated ebits. While this is a fairly straight forward implementation, it is resource expensive. *I have to conduct more literature survey to explore different purification mechanisms.

4. Qubit error correction:

5. Unitary operations: Unitary transformations (Hadamard, X, Z, etc)

Quantum nodes

This is an important abstraction. This makes use of the various modules and provides an abstraction whose functions are a heterogenous combinations of:

1. **Quantum computing** - This includes performing necessary quantum computing operations, quantum communications

2. **Networking** - Some degree of decision making like dynamic routing, packet (datagram) creation, protocol implementations and enforcement, network quality, establishing entanglement links based on protocols etc.

3. **Miscellaneous** - Maintaining the *node* and *network* state which consists of the available quantum resources in terms of entanglement types, entanglement generation capacity, any issues or errors during operation.

These are of different types: end nodes, intermediate nodes: automated (dumb) nodes and intelligent-master nodes, which are described below.

1. End hosts have very limited quantum networking capabilities: sending and receiving qubits, generating entanglement channel and . These do not have have any routing decision making abilities. Their quantum computing capacity is also limited and related their networking abilities.

2. Automated intermediate nodes, as illustrated in [1], have minimal or no decision making ability, although have a higher quantum computing capability. Their primary responsibility is to teleport the received qubit as per the received routing decision. One of the changes to the quantum packet here is that it strips the network state header of its sender and attaches its own.

3. Intelligent-master nodes have more decision making abilities and higher quantum computing capabilities than the automated nodes. The latter often report to these nodes in case of anything: sending frequent heartbeats, reporting in case of any issue etc. They receive regular heartbeats from all the other nodes about their network state, which are all collected and further transmitted to the master cluster.

This also has certain dynamic routing decision making ability, which is determined by the available quantum resources, entanglement conversion, etc. based on the updated network state. If all is good it teleports the received packet as is. If not, it makes an ad-hoc route change.

Quantum Network state

This state is divided into **private** and **public**. The private node state is the primary state of every node, which consists of details about the quantum computer, internal operating capacity, and any the issues and bugs of the (sub)modules of the quantum computing package. The private isn't visible to any entity outside of this node.

The public state information is is abstracted atop the private state. This state is transmitted to its neighbours through every quantum packet transmission included in the datagram, and to a nearby master node frequently in the form of *heartbeats*. Only the node's neighbours and some intelligent nodes are privy to this public version of the node state.

The *Node State* consists of, but not limited to:

1. Qubit creation capacity
    1. Number of qubits
    2. Minimum fidelity
2. Entanglement generation capacity
    1. Number of qubits in the entangled state
    2. Entanglement type
    3. Minimum fidelity
3. Entanglement Purification
    1. Scheduling
    2. Estimated time
    3. Minimum fidelity
    4. Rate of purification
4. Error correction
    1. Capacity
    2. Rate
5. Transmission rate

The network state also consists of the Entanglement Channel States between the nodes. Furthermore, channel metrics are illustrated in detail later on. The Entanglement channel state consists of, but not limited to:

1. Entanglement Type - Bell pair, GHZ, W, etc.
2. Participating nodes
3. Fidelity at creation
4. Decoherence rate
5. Propagation rate/ channel capacity
6. Creation/ Expiry timestamp

Quantum Operating System (QOS)

Every node, as mentioned before, is a collection of heterogenous operations, from quantum computing, to quantum networking, to classical networking, to node state maintenance, and other classical operations. Furthermore, these different aspects are inter-dependent: the node state depends on the available quantum computing resources, routing and teleportation also depends on the same. To handle all of this cohesively and seamlessly there needs to be a comprehensive software for every node.

These nodes must also be reachable remotely to pull updates and push upgrades. There needs to be some generic, but secure APIs which accomplishes this. The design of the modern operating systems are good enough to be carried over, in whatever capacity, here. But, they must be modified to support the nature of the quantum computing technologies, which are starkly different from the classical computing technologies.

Multithreading and parallel computing will play a huge role in the efficiency of this architecture. A lot of operations involve qubits, they are still classical in nature.

Distributed Quantum Resource manager

As mentioned before, every node maintains a state, a public version of which it sends encoded in the quantum packet. The intelligent-master nodes collect the states from surrounding nodes via frequent heartbeats and updates its local network state. This updated local state is then transmitted to the master cluster which updates the network state of the entire quantum computing ecosystem. This is similar to distributed computing framework as implemented in HDFS.

Furthermore, the master cluster and/or intelligent-master nodes will ping individual nodes if they haven't received heartbeats from them in a while. The network state is dynamic and updated.

Quantum Network stack (inverse) and protocols

[1] and [2] have provided a good design base and approach to build upon. I've tried to incorporate both into my design, as appropriate. [2] has provided a good abstractive stack and descriptions of the layers.
protocols

1. Physical

Goal: Establish physical connect between quantum networking devices, perform necessary quantum computations

Functions: Quantum Computing operations, Generating qubits, generate entangled state, qubit error correction, entanglement purification, entanglement swapping, heralded signal generation

2. Connectivity

Goals: Ensure long-distance P2P connectivity, Node State, Repeaters

Functions: Entanglement generation scheme, Node state maintenance
Quantum repeaters, Long-distance P2P entanglement, Entanglement distillation

3. Link

Goals: Generate network state for a network, Intra-network routing, Switching

Functions: Generate arbitrary network states, Network state maintenance - Node states, adjacency matrix, Switches, Intra-network routing, communicate to connectivity layer, P2P entanglement

4. Network

Goal: Inter-network communication

Functions: Routing between Network states, Boundaries - quantum-classical network, bipartite-multipartite, different Quantum Computing ecosystems, Router, Intra-domain decision making, Share network states - internally and externally

Quantum packet (datagram)

Much of the datagram will remain the same. However, with another added layer (connectivity), some changes will be incorporated.

As I mentioned earlier, the node state is also included in the datagram when the quantum packet is teleported to the neighbouring node.
A more detailed packet design is yet to come.

Implementation

Design

In the design framework elaborated previously, with regards to the network stack, I've implemented the *Physical* and *Connectivity* layer to an extent and yet to implement the Link and Network layers and the Protocols.

With regards to the components, I've implemented to an extent, Entanglement channel and Quantum computing packages. I've yet to implement the quantum node abstraction, node and network state, Quantum OS and quantum distributed resource manager. The latter two are out of scope of this research, as they need a much deeper research into Operating Systems.

All of the design and simulation implementation has been programmed in Python.

Metrics

I’ve considered to capture the metric of the channel and the teleported data (qubits).

Channel

1. **Degradation rate** - The fidelity of the generated entangled state isn’t 100% and it also decoders with time and distance
2. **Capacity** - (Max) throughput of the channel, excluding the transmission delay
3. **Propagation rate/ delay** - Currently, with the given quantum technology,

Data/ Node

1. **Error rate** - The percentage of qubits generated with QBER > threshold
2. **Error correction rate** - the percentage of qubits with corrected
3. **Throughput** - This is similar to the classical definition of throughput.
4. **Transmission delay** - transmitting qubits between nodes depends on the underlying quantum technology and the established entanglement channel. The number of qubits in the entangled state that could be used for teleportation determines how many qubits can be teleported at once. The quantum technology can be atomic or concurrent with access to its quantum computing resources which also determines the transmission delay.

Simulation

SimulaQron is the simulator developed by QuTech.

1:1 topology

Line topology

Random graph topology (adjacency matrix)

Results

Data

Graphs

Future work

References