SimQN: A Network-Layer Simulator for the Quantum Network Investigation

Lutong Chen, Kaiping Xue, Jian Li, Nenghai Yu, Ruidong Li, Qibin Sun, and Jun Lu

Abstract

Along with the rapid development of quantum information technologies, quantum networks that transmit quantum states have gradually shifted from laboratory to reality. However, the high cost of building a quantum network testbed seriously hinders the research progress of quantum networks, and the existing simulators have been developed with less exposure to network models. To overcome those deficiencies, this article proposes a new discrete-event driven network-layer simulation framework named SimQN, available at https://github.com/ertuil/SimQN, to provide full functionalities of the quantum networking simulation. The design goal of SimQN is to make it easy-to-configure while providing high performance and accurate simulation capability. We present multiple simulation physical backends to enable different functionalities and reduce computational overhead. The modularized SimQN also provides reusable network protocols and utilities to release the large-scale quantum networking evaluation burden. We present several demonstrations and analyses to illustrate the protocol design and the performance of SimQN.

Introduction

Quantum networks are used to transmit quantum bits (qubit) between quantum devices [1]. This breakthrough network technology enables many innovative application scenarios, such as quantum key distribution (QKD) [2], decentralized quantum computing, and quantum sensor network. Due to the unique quantum characteristics, such as the entanglement state and decoherence, quantum networks require novel architecture and protocol designs to enable the implementation of various quantum applications. Thus, one can foresee that these emerging quantum applications will boost a new wave of researches in the field of quantum networks.

Several exemplary quantum networks were constructed in the past two decades, including the DARPA quantum network, the SECOQC network, the Tokyo quantum network, and the satellite quantum network in China. However, though those pioneers validate the quantum network's feasibility, they still face many challenges in realizing large-scale qubit transmissions. First, the existing quantum networks contain quite a limited number of endpoints but still require a high cost. As the routers (or repeaters) in a real network are developed based on the principle of optical relay or trusted relay, they suffer from either a short communication distance or a harsh security assumption. Second, some building blocks, such as long lifetime quantum memory and quantum relay based on entanglement swapping [3] and teleportation techniques [4], are not ready for large-scale usage. It brings difficulties to the development of network applications and the conduction of cutting-edge quantum network research [5], especially for the studies focusing on the quantum network itself, including quantum network routing, network management, or entanglement distribution. These studies aim to improve the performance and stability of quantum networks, and a tool for low-cost performance evaluation is, however, highly demanded.

Targeted at meeting such needs, several frameworks have been provided for quantum network simulations. However, to the best of our knowledge, those existing frameworks still lack full functionality or scalability in the network layer, making them incompetent for researching interconnections. For example, NetSquid [6] provides complete quantum state operations, and it is thus suitable for quantum communication descriptions and distributed quantum computing simulations. However, it brings out high complexity in simulations. On the other hand, QuNetSim [7] is good at exploring the application scenarios of quantum networks, which, however, is with less exposure to network models that researchers may be concerned about, like time delay or errors.

The quantum networks introduce unique quantum attributions and behave differently from traditional networks, bringing new challenges to network simulation. The first challenge is the performance issue compared to the traditional network simulation. Quantum networks usually have a finer granularity to control each qubit rather than packets or flows. To represent the unique quantum attributions, the qubits must be modeled as a matrix or vector, and matrix calculation brings a non-negligible computational overhead, especially when the network scale goes large. Besides, the state of qubits changes quickly as time flies, requiring more detailed modeling. The second challenge is the complexity of use. The physical details mentioned above are exposed to users, making them hard to bootstrap their investigation. Moreover, considering quantum networks are still in the pre-standardization

Digital Object Identifier: 10.1109/MNET.130.2200481
Date of Current Version: 16 February 2024
Date of Publication: 23 January 2023

Lutong Chen, Kaiping Xue (corresponding author), Jian Li (corresponding author), Nenghai Yu, and Qibin Sun are with the School of Cyber Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, China; Ruidong Li is with the College of Science and Engineering, Kanazawa University, Kakuma, Kanazawa 920-1192, Japan; Jun Lu is with the Department of Electronic Engineering and Information Science, School of Cyber Science and Technology, University of Science and Technology of China, Hefei, Anhui 230027, China.

 0890-8044/23©2023IEEE

phase, users have to implement their own network stacks but cannot reuse the existing ones.

To fill the gap in the simulation modeling for interconnections, we intend to provide a complete description and abstraction ability on quantum networking while keeping it easy-to-configure. To this end, we construct our quantum network simulator, SimQN,[1] and introduce various innovative designs in its architecture. The first design is the multiple fine-grained physical models (named backends), currently including a density matrix-based qubit backend and a numerical entanglement backend. The qubit backend is fully functional to simulate qubits, which brings a relatively large overhead. Meanwhile, the entanglement backend hides the physical details with minimal computation overhead. As a result, users can choose a proper backend as they need. We introduce multiple approaches to leverage multiple CPU cores and further improve simulation performance. Meanwhile, we attempt to reduce the difficulty of use. We provide a modularized node model and let users focus on their protocol designs and reuse other built-in protocols provided by SimQN instead of realizing the whole network stack. Therefore, simulation results are comparable, and users' burdens can be released. We also provide many network utilities, including topology generator and built-in routing algorithms. With all these designs, users can start a large-scale network evaluation within a few lines of code.

Based on the above designs, we implement SimQN and conduct several demonstration examples in several scenarios to prove that our platform can easily support protocol designs and performance evaluations. With the help of our usability tools, users can concentrate on designing the algorithms or protocols without concerning bottom-layer implementations. Our examples scale over 400 nodes with multiple sessions (i.e., pairs of a source node and a destination node to distribute remote entanglements). Besides, we implement a parallel simulator to utilize multiple-thread CPUs to accelerate the simulations, and the results show that our platform is suitable for evaluating large-scale quantum networks. The contributions of this article are summarized as follows.

- We design a new network-layer simulation framework named SimQN to provide full functionalities of the large-scale quantum networking simulation. It contains multiple fine-grained physical backends to support different functionalities, usability, and performance needs.
- We present modularized network-layer utilities in SimQN to enhance protocol reusability and reduce the complexity of constructing simulation environments.
- We conduct multiple demonstration simulations for different kinds of quantum networks to illustrate the efficiency of our SimQN.

This article is organized as follows. In Section II, we brief the background and unique challenges in quantum network simulation. Then, we present our design of SimQN in Section III. We illustrate three demonstration examples of network simulation in Section IV and the performance analysis. Finally, we conclude this article in Section V.

Quantum networks usually have a finer granularity to control each qubit rather than packets or flows.

Background on Quantum Network

Motivation of Quantum Network Simulation

Quantum networks are viewed as a new network architecture that can transmit quantum information rather than classical bits. By providing this capability, quantum networks enable many breakthrough applications, including Quantum Key Distribution (QKD), Decentralized Quantum Computing (DQC), and Quantum Sensor Network (QSN). Several experimental quantum networks have been constructed in many countries, including the US, EU, Japan, and China, recently. Along with network construction, research in this area continues to march forward.

However, the current technologies need to be more mature to support the investigations of large-scale quantum networks. On the one hand, quantum channels (e.g., optical fiber used to transmit qubits) and memories (that can store qubits) suffer from unavoidable noise. Such noise limits the distance to transmit a qubit as well as the qubit's lifetime. On the other hand, the critical relay technique is still a classic relay, while the quantum relay is still in the research stage. Therefore, it restricts the application scenarios of quantum networks. Currently, most constructed quantum networks are mainly for QKD and do not support wilder use scenarios. Finally, quantum devices are costly. These factors make it difficult for researchers to investigate quantum networks, and a flexible, efficient, and low-cost evaluation approach is needed.

Thus, a quantum network simulator providing a feasible approach to carry out large-scale quantum network evaluation at a low cost is indispensable. As a result, we present our platform, SimQN, to help researchers and developers better evaluate large-scale quantum networks with flexible network functionalities, high performance, and usability.

Major Challenges in Quantum Network Simulation

After a survey of the literature, we observe several challenges in quantum network simulation, and we provide targeted features to address these issues in our platform. The challenges are summarized as follows.

Complex Physical Details. The first challenge in quantum network simulation is that the simulator needs to deal with complex physical details. Unlike traditional networks, SimQN should describe the possible noise and unique decoherence on qubits as time flies. In contrast, traditional network simulators usually do not have to care about the physical details, such as the voltage variations in the cables. On the one hand, it requires simulator users to understand these details. On the other hand, such physical details bring additional computation complexity to network simulations.

Huge Simulation Overhead. One way to present arbitrary quantum states is to use density matrices, which is another different aspect

[1] SimQN is an open-source project, and its source code and documents are available at https://github.com/ertuil/SimQN.

The simulation module provides a discrete-event driven scheduler, working as the core of SimQN.

compared to traditional simulators. On the one hand, the size of the density matrices may be large as it is relevant to the number of qubits entangled in the system. On the other hand, quantum networks require a finer granularity, focusing on transmitting qubits rather than packets. As a result, more than thousands of qubits may be handled simultaneously in a large-scale network simulation, which brings out a non-negligible computation overhead. Making things worse, the event-driven network simulation is usually hard to be parallelized or utilize GPU acceleration due to the complex time dependencies between events.

Unstandardized Network Stacks. Because quantum networks are still in the exploration stage, there is no common standard for quantum network architecture or layered designs on the protocol stack. It is observed that nowadays, users have to implement a full network stack, including link-layer protocols, routing algorithms, and other protocols. This coupled design makes it hard to perform their simulations, and the results are not comparable.

Difficulty in Large-Scale Simulation. Finally, most simulators do not provide feasible tools for users to evaluate large-scale networks conveniently, including topology generation, routing table calculation, and so on. Due to the computation complexity and the difficulty in use, it is hard for users to evaluate the performance of large-scale networks.

In this article, we design SimQN and provide corresponding solutions to address the above challenges. First, SimQN is conceived with a general solution since there is currently no recognized network architecture for quantum networks. With the universal qubit backend mentioned in Section III-A, it supports any network type, including QKD-based, entanglement-based, and other kinds of networks. Second, SimQN is both efficient and accurate for conducting simulations. SimQN needs to simulate large-scale networks in a reasonable time, and its results must be accurate and comparable. Finally, SimQN should be easy-to-configure, so users to make the slightest effort to conduct complex simulations.

Architecture of SimQN

SimQN is a modular discrete-event driven network simulator designed for quantum networks and implemented in Python. The operations on qubits in quantum networks are entirely different from bit operations in the current network, and they bring a non-eligible computational overhead which poses significant challenges in designing SimQN. Fig. 1 illustrates those basic modules of SimQN. The SimQN consists of a discrete-event driven simulation module, a physical backend module, a quantum entity module, a network utility module, a network application module, and other auxiliary tools.

The simulation module provides a discrete-event driven scheduler, working as the core of SimQN. To provide the optimal performance, the event pool is implemented as a priority queue (i.e., minimum heap), where both procedures of inserting an event and fetching the most recent event can be handled in $O(\log(n))$.

The physical backend module is for the physical description of qubits, which is introduced in Section III-A. To achieve a balance between performance and functionality, we propose multiple physical models, i.e., a density matrix-based qubit model and a high-level entanglement model by default. In Section III-B, we present the quantum entity module to represent the devices in the network, such as the nodes, channels, and others. Last, we bring in a network-layer utility module for more straightforward large-scale network evaluation, such as generating universal topology, building routing tables, and managing multiple sessions in Section III-C. SimQN also contains several network applications to present built-in protocols and algorithms in Section III-C.

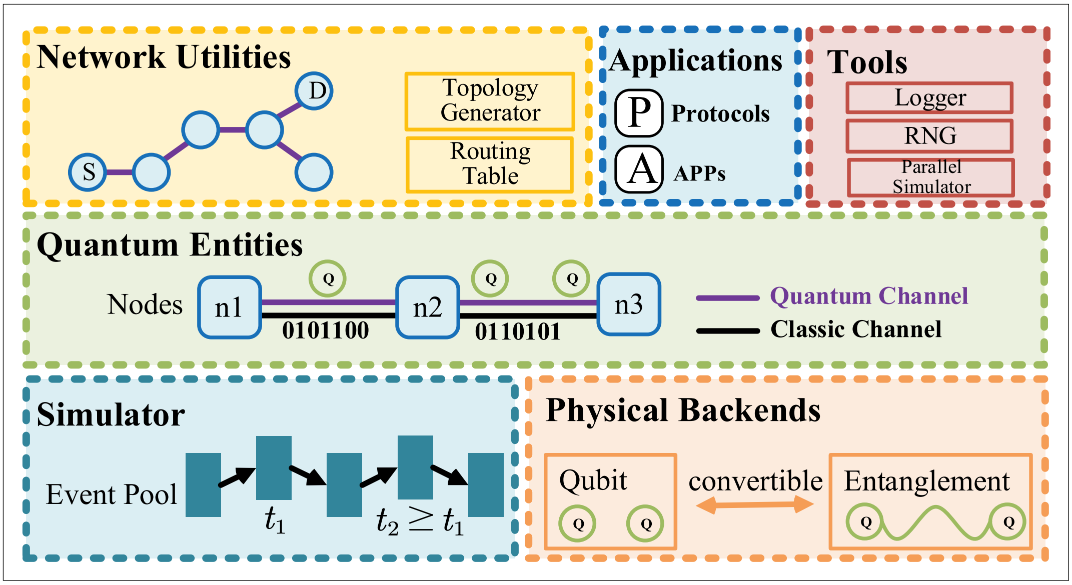

FIGURE 1. The architecture and modules of the SimQN.

Physical Backend Module

The physical backend module is designed to model quantum attributions. We propose two major backend models: a density matrix-based qubit backend and a numerical evaluation-based entanglement backend. Both backends are highly configurable so that users can set physical parameters (including noise models) to represent different physical quantum models (e.g., photons or iron traps) as they need.

In the qubit backend, we give priority to the functionality, which enables it to be suitable for most use scenarios. It should also be capable of modeling arbitrary noise and decoherence. Thus, we adopt a density matrix to represent quantum states. Each quantum system — an ensemble of several entangled qubits — is represented by a complex density matrix. In this way, qubits can be operated by quantum gates or measured. Qubit backend can also be used to illustrate quantum errors or noises during operating, measuring, transmitting, and storing qubits, and we further provide some commonly used decoherence models, including bit flipping, depolarizing, dephasing, and dissipation noises. Consequently, the qubit backend is capable of simulating most evaluation scenarios because of the general capability of describing qubits.

Though the qubit backend is fully functional, it generates a non-eligible computation overhead. It is because the size of the density matrices is correlated with the number of qubits, and quantum networks need to handle a large number of matrix operations. To reduce such computational overhead and speed up simulations, we inspect the mainstream types of quantum networks. We observe that two kinds of quantum networks are studied in depth. One is the QKD network [2] that is usually based on single non-entangled qubits. The second is the entanglement distribution networks [8], where Bell State entangled pairs are the basic elements. Therefore, we propose entanglement backends for the second scenario to provide high simulation performance.

In the entanglement backend, we adopt scale variables to represent the quantum states rather than density matrices. It is observed that several numerical models are usually adopted in evaluation, such as Werner states, Bell states, and Bell-diagonal states. In these models, scalar variables are used instead of vectors or matrics. One of the variables is fidelity, which quantifies the quality of entangled pairs. We present several quantum operations that are usually used in entanglement-based networks, including quantum teleportation [4], entanglement swapping [3], and entanglement distillation [9], [10]. Specifically, we implement DEJMPS [9] distillation protocol for Bell-diagonal state and BBPSSW [10] for Werner state. Corresponding noises and errors are also presented in the entanglement backend. In conclusion, the entanglement backend leverages scale calculation to vastly speed up the simulation, with the cost of the loss of generality.

The qubit backend and entanglement backend are convertible for users to take advantage of both of them at the same time. In this way, users can reduce the simulation overhead by using the entanglement backend and then convert them to a qubit backend to perform arbitrary quantum operations whenever they want.

Network Entity Module

The network entity module implements quantum devices in constructing a quantum network, including quantum nodes, memories, classic channels, quantum channels, etc.

The quantum nodes can be end-point users, quantum repeaters [11], or routers in classical networks. By equipping operators, memories, and other devices, nodes are capable of transmitting, operating, and storing qubits. As a result, users can instruct nodes to perform these behaviors as they need. To decouple the complex node behaviors and improve protocol reuse capability, quantum nodes in SimQN can embed multiple applications in the simulator. The applications can refer to a network protocol (such as distributing remote entangled pairs) or application-layer programs. For example, users can embed the built-in *ClassicNetworkStackApp* into nodes to communicate with each other via classical channels and a *RoutingApp* to calculate routing tables. In Fig. 2, we design *BB84SendApp* and *BB84RecvApp* to work as the two parties in the BB84 QKD protocol. This modularized application design can further make the simulation results more comparable.

Other entities are equipped on or between the nodes, including quantum memories, operators, quantum channels, and classic channels. Quantum memories can store qubits for a while. We use an error model to describe the noise in memories and a delay model to present the time delay in read or write qubits to simulate different techniques of quantum memories. Similarly, users can use the operator entity to represent a quantum circuit and operate the qubits, and the noise and delay model is also supported. Besides, quantum and classic channels connect multiple nodes and transmit qubits or classic messages. Like the links in traditional networks, the channels are modeled in several attributions, like "drop_rate", "delay", and "bandwidth" to simulate multiple transmission media, including optical fiber and free space. The unique characteristics of quantum networks are that the qubits suffer from quantum noises or decoherence during being stored, transmited, or operated, and we present several commonly used error models, including bit flip, dephasing, depolarizing, and dissipation errors.

Note that the discrete-event driven simulation is driven by entities' generating events and handling the incoming events. In Fig. 2, four nodes on a path run QKD protocols to distribute secret keys between Alice and Bob. As a quantum node, Alice can produce an event to send a qubit to Repeater 1 at t_1. Consequently, Repeater 1 needs to handle the incoming qubit after a time delay t_2.

Network-Layer Utility Module

Along with the built-in applications mentioned in Section III-B, we also provide several network utilities to bootstrap large-scale network evaluations, including a double-stack topology generator, double-stack routing algorithms, data collector, and parallel simulator. With all these utilities, users can

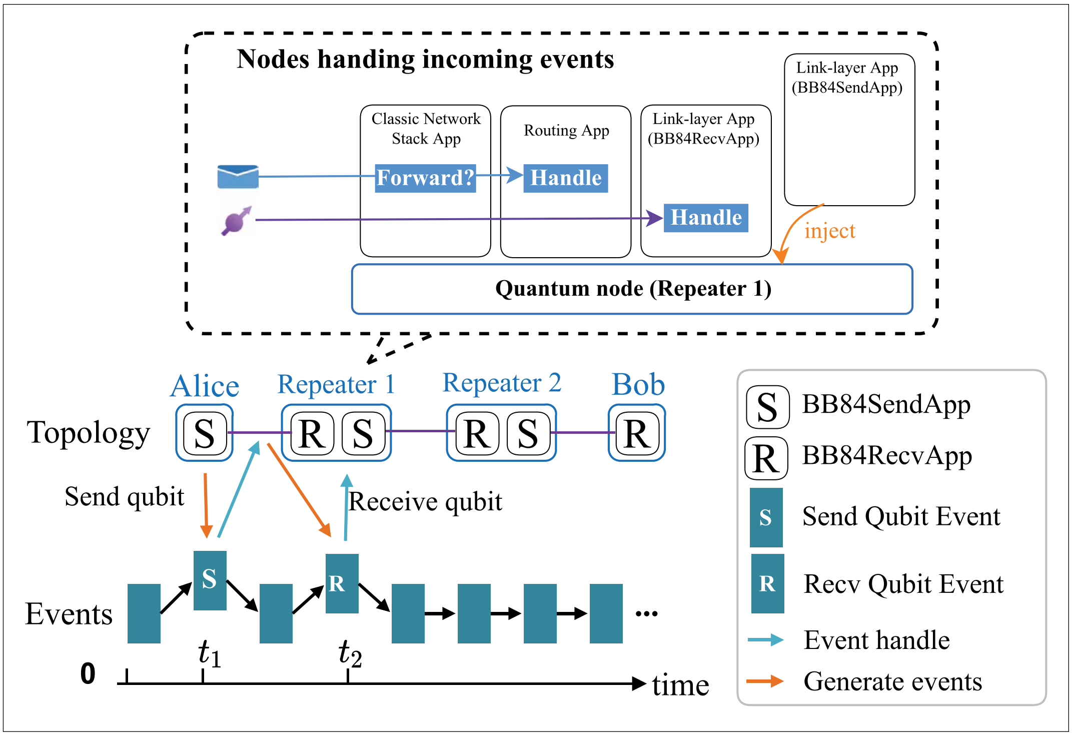

FIGURE 2. An example of quantum entities in a QKD network.

quickly construct a simulation environment with only a few lines of code and achieve a high evaluation performance simultaneously.

Some utilities are designed to reduce the repetitive work in simulation and bootstrap network simulation during the whole period. For example, we design a double-stack topology generator to easily configure the network topology and set network parameters in one step. SimQN provides several topology demonstrations, such as grid topology, linear topology, and the Waxman topology, and users can also build any topologies. The topology generator will help users to build an experimental testbed with a few lines of code. After building the topology, we also implement an injectable double-stack routing model to calculate routing tables for both quantum and classic networks. We currently implement the shortest-path routing algorithm based on Dijkstra's algorithm and Yen's multiple-path routing algorithm used in many existing papers with arbitrary metric functions. The metric functions calculate the priority of the channels. For example, the channel length is an optional metric function. SimQN also provides several practical modules for better experiments. For example, the built-in request management can select random requests (source-destination pairs) and assign parameters. As mentioned in Section III-B, quantum nodes can embed multiple applications, and many are directly built-in in SimQN. Finally, after the simulation is completed, the data collector in SimQN provides easy access to the results.

Other utilities improve the simulation performance. On the one hand, we provide an optional Cython accelerated version of SimQN. It compiles the computationally intensive codes into machine instructions to speed up the simulation. It speeds up the discrete-driven scheduler and other heavy algorithms like calculating routing tables. On the other hand, we provide a multiple-thread parallel simulator so that users can conduct multiple simulation tasks simultaneously by dispatching these tasks to multiple CPUs. The experiments in Section IV show the efficiency of the parallel simulator.

Conclusion. Leveraging the modules mentioned above, SimQN meets all challenges in Section II-B. For example, we present the entanglement backend in Section III-A to reduce the computational overhead and hide the physical details from users. We also provide injectable network applications to meet the requirements of different scenarios and overcome the disadvantage of unstandardized architecture in Section III-B. In addition, SimQN brings several network utilities in Section III-C and built-in applications to bootstrap the simulations.

Comparisons With the Existing Platforms

The typical existing quantum network simulation software include SeQUeNCe [12], QuISP [13], QuNetSim [7], and NetSquid [6]. Here, we brief their features and compare our work with them.

QuISP [13] is a simulation based on describing errors instead of the state vectors for the performance issue. For a similar reason, SimQN provides an entanglement physical backend. Still, we keep the full functional qubit backend for universal quantum operations and noises. Another innovation of QuISP is the ruleset-based network protocols. At this point, SimQN provides Python interfaces making the simulation more accessible and flexible.

NetSquid [6] and SeQUeNCe [12] are fully functional quantum network platforms based on a discrete-event driven simulator. They provide detailed physical hardware components, while SimQN, as a network-layer simulator, focuses on describing the network attributes instead of the physical ones. Besides, SimQN is designed to be easy-to-configure. We provide an optional entanglement backend to reduce the computational overhead. Extra network utilities in SimQN further reduce the burden on users.

QuNetSim [7] is a real-time simulator for small-scale quantum networks. It uses a strict network architecture inspired by the OSI model and hides the network model. Thus, QuNetSim is suitable for developing upper-layer applications rather than network-layer investigations. SimQN is capable of these tasks with no assumption of the network architecture.

Evaluations

This section presents three demonstrations: a QKD network, an entanglement distribution network, and a performance analysis. The first two examples show the capability of SimQN for evaluating both QKD and entanglement-based networks by using the two physical backends, respectively. These experiments also demonstrate how easy to use the tools provided by SimQN to conduct large-scale simulations. In the third example, we conduct a performance evaluation to show the efficiency of different physical backends and compare them with other platforms.

Validation Example: Remote Quantum Key Distribution

In the first validation example, we simulate a QKD network on a linear topology, as shown

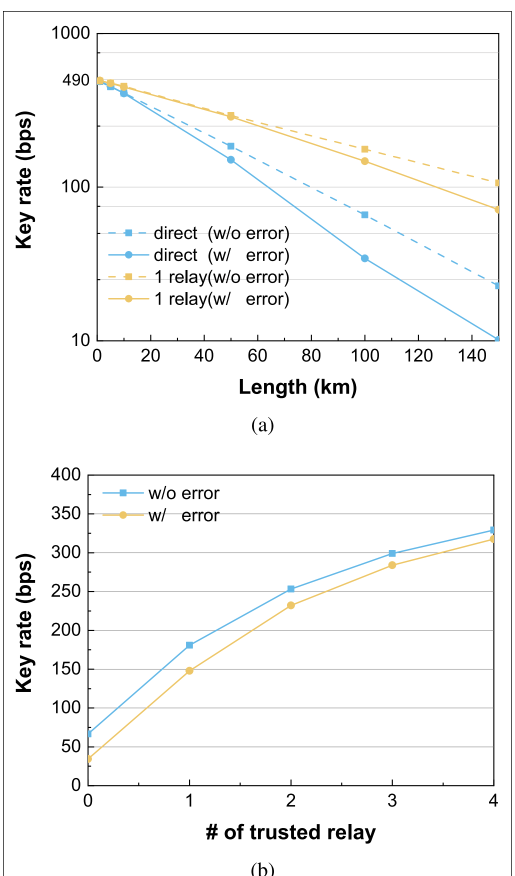

FIGURE 3. Evaluation results of the remote QKD experiments. a) Key rate vs channel length. b) Key rate vs number of repeaters.

We use the topology generator to build a linear topology and set network parameters.

in Fig. 2. We use the topology generator to build a linear topology and set network parameters. We set the attenuation of the quantum channels to 0.2 dB/km and the propagation delay according to light speed. In this experiment, adjacent nodes first generate link-layer secret keys, implemented in two applications, i.e., *BB84SendApp* and *BB84RecvApp*, to represent the qubit sender and receiver in the BB84 protocol, respectively. After that, we adopt the protocol in [14] to generate end-to-end secret keys. We also introduce a bit-flip error in transmission with a probability depending on the channel length.

First, we explore how the channel length influences the key distribution rate. We set the qubit sending rate to 1 kHz and the channel length from 1 km to 150 km without a trusted repeater. We conduct each game ten times. Fig. 3(a) shows that the key rate drops exponentially with the increasing channel length, which is in line with the results of other practical experiments. The key rate decreases when we introduce the bit flip error, significantly when the channel length increases. Further, we can observe that trusted relays can be used to slow the decline of the key rate.

In the second experiment, we fix the total length to 100 km and insert 0-4 trusted relays on the path. The key rate increases as shown in Fig. 3(b). However, as the number of trusted relays increases, the revenue of adding a new trusted relay will gradually decrease. It is mainly due to the increasing calculation and communication delay.

Validation Example: Remote Entanglement Distribution

In the second example, we aim to distribute remote entangled pairs with multiple quantum repeaters in a large-scale random quantum network. We use the network utilities mentioned in Section III-C to quickly set up a network simulation environment within only 30 lines of code, including generating a random topology with up to 400 nodes, setting the network parameters, and building classical and quantum network stacks.

Specifically, quantum nodes are connected by the quantum channels and equipped with a limited quantum memory size of 50. We use the built-in routing algorithm to calculate the routing table of all nodes and use request management to select up to 40 random requests. In order to speed up the simulation, we use the Werner state entanglement backend to model entangled pairs. The initial fidelity is set to 0.99, and we set a depolarizing noise with a coherency time T_{coh} = 5 s. In the simulation, we design a decentralized entanglement distribution protocol named *EntanglementDistributionApp*, installed on every node. In this protocol, source nodes start entanglement distribution at a constant sending rate (number of concurrent entanglement distribution), and then all nodes will perform entanglement swapping hop-by-hop on the path selected by the routing algorithm [15].

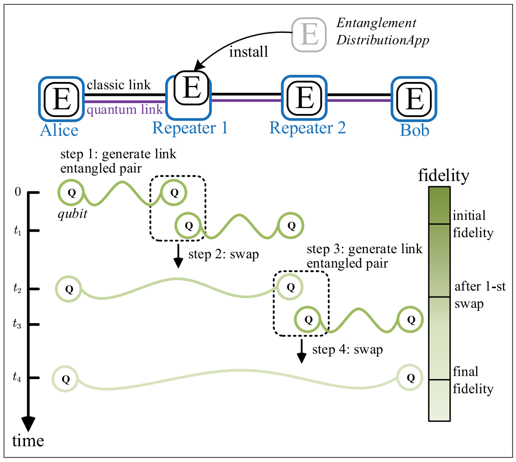

FIGURE 4. The decentralized entanglement distribution protocol on a 4-hop path.

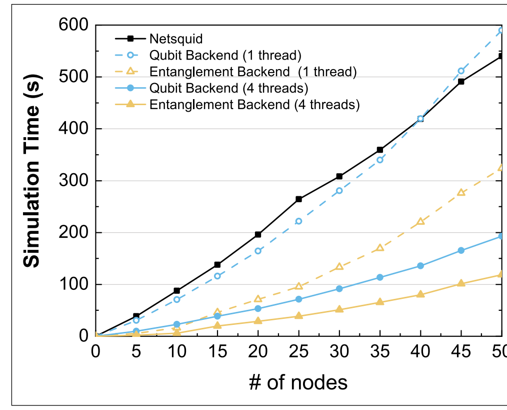

FIGURE 6. Performance between different platforms, backends and threads.

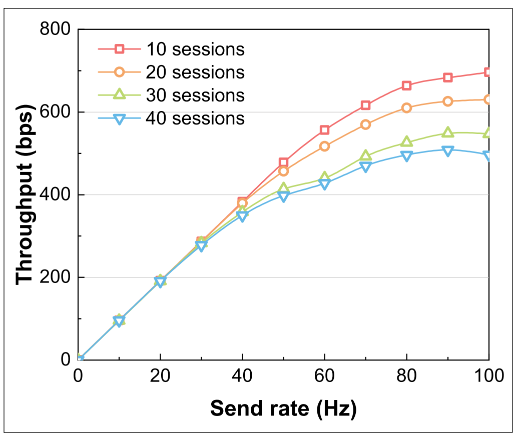

FIGURE 5. Throughput of entanglement distribution in multiple sessions.

Fig. 4 shows an example of remote entanglement distribution on a 4-hop path.

We conduct our experiments under 10, 20, 30, and 40 concurrent sessions and examine the average throughput (i.e., entanglement distribution rate, EDR). Fig. 5 shows how the number of concurrent sessions and the send rates influence the average throughput. The average throughput is almost proportional to the send rate if the network is not congested (when the send rate is lower than 30 Hz). At this time, the average throughput is independent of the number of concurrent sessions. Otherwise, more sessions bring out worse congestion and limit the average throughput.

Further investigation finds that a larger send rate and concurrency lead to less fairness for multiple sessions. We also measure the time used in running a large-scale simulation game on a typical consumer computer with an Intel i5-9400 CPU and 32 GB memory. The results show that the number of sessions affects the simulation time, but it is acceptable.

Performance Analysis

We improve the simulation performance from several aspects. First, we propose an entanglement backend to reduce the calculation overhead. Then, SimQN leverages Cython to improve the performance further. Lastly, we provide a parallel simulator to utilize multiple-thread CPUs and run multiple experiments concurrently.

We further evaluate the performance under different physical backends and different numbers of threads. We simulate entanglement distribution on a 10-nodes linear topology. The distance between two adjacent nodes is 10 km, and the medium speed is set to 200,000 km/s. We implement the entanglement distribution protocols in both the qubit backend and Werner state entanglement backend. In this protocol, each adjacent node pair first generates entangled pairs at 1000 Hz. Then, the repeaters perform joint measurements concurrently and send the measurement results to the destination node to form end-to-end entangled pairs. In the qubit backend, qubits suffer from depolarizing decoherence, and the decoherence rate is 200 Hz. We also implement the same protocol on other platforms. Since QuNetSim does not expose network interfaces and QuISP uses different error models, we construct the protocol in NetSquid. The results are shown in Fig. 6.

The simulation results illustrate that SimQN provides better performance than NetSquid. The qubit backend is overall 11.1% faster than NetSquid. Furthermore, the entanglement backend provides an average of 2.68 times performance improvement compared to the qubit backend in single CPU simulations and is 3.07 times faster than NetSquid. We also evaluate the performance using the parallel simulator. We use 4 cores to run simulations concurrently. We record the average time for each simulation. The results prove that parallel simulation provides a significant performance improvement. The performance of the qubit backend and entanglement backend models increases 3.07 times and 2.60 times, respectively. Compared to NetSquid, SimQN can provide an average of 8.02 times performance improvement. In conclusion, the three demonstrations above illustrate the functionality and performance advantage of SimQN.

Conclusion

Due to the rapid development of quantum information technology, quantum networks are becoming more of a reality than ever. In this article, we presented a modularized discrete-event driven simulation platform, SimQN, available at https://github.com/ertuil/SimQN, to help answer the questions in quantum network investigations and application development. The primary goal is to provide convenience for large-scale quantum network research while balancing performance and functionality. We implemented multiple quantum backends to achieve this goal. Furthermore, we provided full-stack network utilities and modularized injectable applications to bootstrap the simulation. Demonstration evaluations and performance analyses illustrate the capability and performance of the SimQN, indicating that the platform can competently simulate large-scale quantum networks. Next, we will continue to push SimQN forward in the next stage. We plan to construct a visual GUI interface for our SimQN. Besides, we will also update and provide more built-in protocols and physical modules to make our platform more fully functional and usable. We look forward to the peers using our platform to carry out relevant research and put forward valuable suggestions.

Acknowledgment

This work was supported in part by the Anhui Initiative in Quantum Information Technologies under Grant AHY150300 and in part by the Youth Innovation Promotion Association Chinese Academy of Sciences (CAS) under Grant Y202093.

References

[1] Z. Li et al., "Building a large-scale and wide-area quantum Internet based on an OSI-alike model," *China Commun.*, vol. 18, no. 10, pp. 1–14, Oct. 2021.

[2] M. Mehic et al., "Quantum key distribution: A networking perspective," *ACM Comput. Surv.*, vol. 53, no. 5, pp. 1–41, 2020.

[3] H. de Riedmatten et al., "Long-distance entanglement swapping with photons from separated sources," *Phys. Rev. A, Gen. Phys.*, vol. 71, May 2005, Art. no. 050302.

[4] C. H. Bennett et al., "Teleporting an unknown quantum state via dual classical and Einstein–Podolsky–Rosen channels," *Phys. Rev. Lett.*, vol. 70, no. 13, pp. 1895–1899, Mar. 1993.

[5] L. Chen et al., "A heuristic remote entanglement distribution algorithm on memory-limited quantum paths," *IEEE Trans. Commun.*, vol. 70, no. 11, pp. 7491–7504, Nov. 2022.

[6] T. Coopmans et al., "NetSquid: A NETwork Simulator for QUantum Information using Discrete events," *Commun. Phys.*, vol. 4, no. 1, p. 164, Jul. 2021.

[7] S. DiAdamo et al., "QuNetSim: A software framework for quantum networks," *IEEE Trans. Quantum Eng.*, vol. 2, pp. 1–12, 2021.

[8] Z. Li et al., "Connection-oriented and connectionless remote entanglement distribution strategies in quantum networks," *IEEE Netw.*, vol. 36, no. 6, pp. 150–156, Nov./Dec. 2022.

[9] D. Deutsch et al., "Quantum privacy amplification and the security of quantum cryptography over noisy channels," *Phys. Rev. Lett.*, vol. 77, no. 13, pp. 2818–2821, 1996.

[10] C. H. Bennett et al., "Purification of noisy entanglement and faithful teleportation via noisy channels," *Phys. Rev. Lett.*, vol. 76, pp. 722–725, Jan. 1996.

[11] N. Sangouard et al., "Quantum repeaters based on atomic ensembles and linear optics," *Rev. Mod. Phys.*, vol. 83, pp. 33–80, Mar. 2011.

[12] X. Wu et al., "SeQUeNCe: A customizable discrete-event simulator of quantum networks," *Quantum Sci. Technol.*, vol. 6, no. 4, 2021, Art. no. 045027.

[13] T. Matsuo, "Simulation of a dynamic, ruleset-based quantum network," 2019, *arXiv:1908.10758*.

[14] H. Bechmann-Pasquinucci and A. Pasquinucci, "Quantum key distribution with trusted quantum relay," 2005, *arXiv:quant-ph/0505089*.

[15] J. Li et al., "A connection-oriented entanglement distribution design in quantum networks," *IEEE Trans. Quantum Eng.*, vol. 3, pp. 1–13, 2022.

Biographies

Lutong Chen (lutong98@mail.ustc.edu.cn) received the bachelor's degree from the School of Cyber Science and Technology, University of Science and Technology of China (USTC), in 2020. He is currently pursuing the Ph.D. degree in information security with the School of Cyber Science and Technology, USTC. His research interests include quantum networking and network security.

Kaiping Xue (Senior Member, IEEE) (kpxue@ustc.edu.cn) received the bachelor's degree from the Department of Information Security, University of Science and Technology of China (USTC), in 2003, and the Ph.D. degree from the Department of Electronic Engineering and Information Science (EEIS), USTC, in 2007. He is currently a Professor with the School of Cyber Science and Technology, USTC. His research interests include future Internet architecture design, transmission optimization, network security, and quantum networking. He is a Fellow of the IET.

Jian Li (Senior Member, IEEE) (lijian9@ustc.edu.cn) received the bachelor's degree from the Department of Electronics and Information Engineering, Anhui University, in 2015, and the Ph.D. degree from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC), in 2020. He is currently a Post-Doctoral Fellow with the School of Cyber Science and Technology, USTC. His research interests include future Internet architecture design, satellite-terrestrial integrated networks, and quantum networking.

Nenghai Yu (ynh@ustc.edu.cn) received the bachelor's degree from the Nanjing University of Posts and Telecommunications, Nanjing, China, in 1987, the M.E. degree from Tsinghua University, Beijing, China, in 1992, and the Ph.D. degree from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC), Hefei, China, in 2004. He is currently a Professor with the School of Cyber Science and Technology, USTC. He is the Executive Dean of the School of Cyber Science and Technology, USTC, and the Director of the Information Processing Center, USTC. His research interests include multimedia security and quantum networking.

Ruidong Li (Senior Member, IEEE) (lrd@se.kanazawa-u.ac.jp) received the bachelor's degree in engineering from Zhejiang University, China, in 2001, and the Ph.D. degree from the University of Tsukuba in 2008. He is currently an Associate Professor with the College of Science and Engineering, Kanazawa University, Japan. Before joining Kanazawa University, he was a Senior Researcher with the Network System Research Institute, National Institute of Information and Communications Technology (NICT). His research interests include big data networking, blockchain, information-centric network, the Internet of Things, network security, wireless networks, and quantum Internet.

Qibin Sun (Fellow, IEEE) received the Ph.D. degree from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC), in 1997. He is currently a Professor with the School of Cyber Science and Technology, USTC. He has published more than 120 papers in international journals and conferences. His research interests include multimedia security, network intelligence and security, and so on.

Jun Lu (lujun2019@ustc.edu.cn) received the bachelor's degree from Southeast University in 1985, and the master's degree from the Department of Electronic Engineering and Information Science (EEIS), University of Science and Technology of China (USTC), in 1988. He is currently a Professor with the School of Cyber Security and Technology and the Department of EEIS, USTC. His research interests include theoretical research and system development in the field of integrated electronic information systems. He is an Academician of the Chinese Academy of Engineering (CAE).