UNIVERSITÀ
DI PARMA

Anonymous communication of quantum messages

Matteo Gianvenuti

Academic year 2023/2024

Summary

Project description

The intention of this project is to use Quantum Network Explorer (QNE)[1] to Simulate Advanced Quantum Security Protocols with the purpose of evaluating the usability of QNE. I decided to implement the paper "Anonymity for practical quantum networks"[2] for the importance of the privacy, especially now days.

Protocol inputs

The protocol inputs define which is our context.

- In the application we have four roles (agents): one sender and three other agents. One of the other agents must be selected as receiver to establish the anonymous entanglement with the sender. By default, the receiver is the agent two. All the four agents in the network are honest nodes.
- The security parameter S is an important value used in various subprotocols. It increases the randomness and the security. More S is great, and more rounds are executed in the subprotocols. By default, I set it to the value '2', because the agents are honest and a higher value would slow down the protocol, especially for the LogicalOR subprotocol.

Protocol variations

The unique variation is in the step 2 of Parity (protocol 6). Instead of sending the j-th bit to the j-th agent it is sent with the broadcast. The result still be the same, every agent gets her bit and ignore all others. This variation was necessary because the number of sockets needed to be handled synchronously was too high (e.g. if an agent do a 'send', the other should be in 'recv'). Even with an ordering, the agents were unable to communicate in a synchronized way.

Focusing on QNE

What is QNE-ADK?

QNE-ADK is the Application Development Kit for Quantum Network Explorer. It includes almost all necessary tools to build an application with QNE in Python. "almost" because you need to install also SquidASM to run your application. SquidASM is a quantum network simulator (library based on NetSquid). It allows you to run your application without the need of a physical quantum network. To use SquidASM is necessary register on the NetSquid forum.

The operating system for development must be a modern version of Linux (Ubuntu is suggested) or MacOS. If you use Windows, you can use either Windows Subsystem for Linux or a Virtual Machine.[3] I used Ubuntu 22.04.4 LTS (GNU/Linux 5.15.146.1-microsoft-standard-WSL2 x86_64).

Known issues

- Considering the methods 'send' and 'recv' of the classical socket, which send and receive only strings, there may be data leaks if the string is longer than one character. I mean, if you send "1101", "1010", etc, sometimes you may receive "0" (from experimental tests). This happen when multiple messages are sent sequentially, the 'recv' could compact them (may depend by my hardware). I solved this problem sending and receiving one bit per time e.g. "1" or "0". This is not a problem since the amount of information to be exchanged is very small. Then it is not a real issue for this project.

- Rarely, the shared GHZ verification can fail. Because the angles used for the rotations, required to measure in the specified basis (protocol 4 a.k.a. Verification, step 2), are approximated by the library. Theoretically the verification should always be correct, given that all agents are honest and the shared GHZ is prepared properly.

- The first step of the main protocol sometimes behaves unexpectedly: Instead of notifying the receiver, could notify more agents or none. All the subprotocols involved in this step are correctly implemented. The problem could be due to my hardware. Because most of the time, when I run the application after a long period (which may include a reboot) the notification is properly done.

- At the end of the execution is possible to see a warning for each role. This is a known issue from NetSquid.[4] This warning has no impact.

Advantages of QNE

- Allow you to build and test a quantum network without the need of quantum computers. Which is a very good advantage since quantum computers are expensive.

- It allows writing application code in Python, a common programming language. It is also possible to write the code directly in "quantum assembly" (NetQASM language, which resembles assembly code).[5]

Limitations of QNE

- Not all the Linux distribution are supported: At the beginning of this project, I attempted to use a different Linux distribution instead of the suggested Ubuntu, but even if I was running the same example of entanglement in the documentation it did not work. NetQASM, a fundamental component of QNE-ADK, has been developed and tested on Ubuntu.[5]

- Documentation and functionalities: In the documentation there are objects not completely implemented then abstract objects! For example, the broadcast, I had to complete it by myself. Since the library is open source on GitHub you can see the broadcast file still be the same from three years, not so good.[6]

- Just few qubits in the simulator: To run the project in the classical way with QNE you should create an experiment and run it. My application in the sender role needs to use seven qubits (to create the 4-qubit GHZ state and share it with other three agents, through the quantum teleportation), but seven qubits are too many to create and run an experiment so the application should be run with "netqasm simulate". Then QNE can be used only with little projects if you do not have real quantum computers. From experimental test, each role can create no more than three qubits (may depend by the hardware).

- Lack of tools for synchronization: With QNE every role is a node which correspond to a Python script that runs in competition with all other roles, so if you need to exchange classical information with many multiple roles you may have some issue as here. This lack is understandable because QNE is focused on quantum computing and not on parallel computing.

- Lack of support: The developers appear uncontactable, if you contact them for issues in their library, they never answer you (e.g. broadcast not completely implemented). Fun fact: Instead, if you contact who handle public communications for QNE they answer you, but they cannot say nothing more than "implement it by yourself".

Conclusion

I may have been too critical of this library, because I see it as a very little library to be used to develop complex projects. Anyway, it could be very useful to test little projects if your budget is limited and you do not have quantum computers.

Bibliography

[1] QNE site: https://quantum-network.com/.

[2] The paper (a.k.a. the protocol): "Unnikrishnan, A., MacFarlane, I. J., Yi, R., Diamanti, E., Markham, D., & Kerenidis, I. (2019). Anonymity for practical quantum networks. Physical review letters, 122(24), 240501." Available at https://github.com/Mqtth3w/QNE-anonymity-quantum-networks-nipr/blob/main/Anonymity_for_practical_quantum_networks(paper).pdf.

[3] QNE-ADK: https://www.quantum-network.com/adk/.

[4] NetSquid issue: https://netqasm.readthedocs.io/en/latest/known_issues.html.

[5] NetQASM information: https://github.com/QuTech-Delft/netqasm.

[6] broadcast "abandoned": https://github.com/QuTech-Delft/netqasm/tree/develop/netqasm/sdk/classical_communication.