Automated Quantum Program Verification in Probabilistic Dynamic Quantum Logic

Canh Minh Do[1,*], Tsubasa Takagi[2] and Kazuhiro Ogata[1]

[1]Japan Advanced Institute of Science and Technology, 1-8 Asahidai, Nomi, Japan
[2]Tokyo Institute of Technology, 2-12-1 Ookayama, Meguro-ku, Tokyo, Japan

Abstract

We proposed an automated approach to verifying quantum programs based on Basic Dynamic Quantum Logic (BDQL), a simplified version of Dynamic Quantum Logic, and developed a support tool in Maude, a specification/programming language based on rewriting logic, to automate the verification process. This paper extends BDQL to Probabilistic Dynamic Quantum Logic (PDQL) to verify probabilistic quantum programs by introducing a probabilistic operator in the formulas of PDQL. We also extend the support tool for BDQL to make a new support tool for PDQL to automatically verify both qualitative and quantitative properties for quantum programs with PDQL. As case studies, we use our support tool to verify several quantum programs: Superdense Coding, Quantum Teleportation, Quantum Secret Sharing, Entanglement Swapping, Quantum Gate Teleportation, Quantum Relay Scheme, Bidirectional Quantum Teleportation, Two-qubit Quantum Teleportation, and Quantum Network Coding, where the probability is intentionally expressed in the formalizations of quantum programs, demonstrating the expressiveness of PDQL.

Keywords

Probabilistic Dynamic Quantum Logic, Quantum Programs, Quantum Protocols, Dirac notation, Maude

1. Introduction

Quantum computing holds the potential to revolutionize various fields by harnessing the principles of quantum mechanics to perform computations that are currently infeasible for classical computers. For example, in the field of cryptography, the arrival of future quantum computers has the potential to break widely used public-key cryptography methods, such as Rivest–Shamir–Adleman (RSA) and Elliptic-curve Cryptography (ECC), by effectively factoring large numbers or solving the discrete logarithm problem using Shor's fast algorithm [1]. Quantum computing uses different principles of quantum mechanics, such as superposition, entanglement, and measurement, which are completely different from classical computing. As a result, it is challenging to accurately design and implement quantum algorithms, programs, and protocols. Therefore, it is crucial to ensure the correctness of quantum systems through verification.

The 2nd International Workshop on Formal Analysis and Verification of Post-Quantum Cryptographic Protocols, November 11, 2023, Brisbane, Australia

*Corresponding author.
canhdo@jaist.ac.jp (C. M. Do); takagi.t.ah@m.titech.ac.jp (T. Takagi); ogata@jaist.ac.jp (K. Ogata)
0000-0002-1601-4584 (C. M. Do); 0000-0001-9890-1015 (T. Takagi); 0000-0002-4441-3259 (K. Ogata)

Dynamic Logic [2] has been playing a significant role in the formal verification of classical systems for some decades. To handle quantum effects, Dynamic Quantum Logic [3], the quantum counterpart of Dynamic Logic, has been developed. However, we lack an automated approach to verifying quantum programs using Dynamic Quantum Logic. Our research group devised an automated approach to verifying quantum programs using Basic Dynamic Quantum Logic (BDQL) [4], a simplified version of Dynamic Quantum Logic, and developed a support tool in Maude [5], a specification/programming language based on rewriting logic [5], to automate the verification process by incorporating symbolic reasoning for quantum computation in [6]. This paper extends BDQL to Probabilistic Dynamic Quantum Logic (PDQL) to verify probabilistic quantum programs by introducing a probabilistic operator in the formulas of PDQL. We also extend the support tool for BDQL to make a new support tool for PDQL to automatically verify both qualitative and quantitative properties for quantum programs with PDQL. As case studies, we use our support tool to verify several quantum programs: Superdense Coding [7], Quantum Teleportation [8], Quantum Secret Sharing [9], Entanglement Swapping [10], Quantum Relay Scheme [11], Bidirectional Quantum Teleportation [12], Quantum Gate Teleportation [13], Two-qubit Quantum Teleportation [14], and Quantum Network Coding [15], where the probability is intentionally expressed in the formalizations of quantum programs, demonstrates the expressiveness of PDQL. The support tool and case studies are publicly available at https://github.com/canhminhdo/DQL.

The rest of the paper is organized as follows: Section 2 describes Basic Dynamic Quantum Logic (BDQL); Section 3 presents Probabilistic Dynamic Quantum Logic (PDQL); Section 5 demonstrates the application of PDQL for probabilistic quantum program verification; Section 5 describes the implementation of PDQL; Section 6 exhibits the experimental results with our support tool for several quantum programs; Section 7 presents some existing work; and Section 8 concludes the paper with some pieces of future work.

2. Basic Dynamic Quantum Logic

This section formulates Basic Dynamic Quantum Logic (BDQL) [4]. Let L_0 be a set of atomic formulas and Π_0 be a set of atomic programs. The set L of all formulas in BDQL and the set Π of all star-free regular programs are generated by simultaneous induction as follows:

$$L \ni A ::= p \mid \neg A \mid A \wedge A \mid [a]A,$$
$$\Pi \ni a ::= \mathbf{skip} \mid \mathbf{abort} \mid \pi \mid a\ ;\ a \mid a \cup a \mid A?,$$

where $p \in L_0$ and $\pi \in \Pi_0$. The symbols **skip** and **abort** are called constant programs. The operators ;, $\cup$, and ? are called sequential composition, non-deterministic choice, and test, respectively.

The syntax of BDQL is exactly the same as that of Propositional Dynamic Logic (PDL) [2] without the Kleene star operator *. It is not strange that two different logics have the same syntax. For example, Classical Logic and Intuitionistic Logic have the same syntax but are distinguished by their semantics (or their sets of provable formulas). Similarly, the semantics of BDQL and that of the star-free fragment of PDL are different. This paper considers only star-free regular programs, leaving the addition of * to the logic in a future paper.

We define the semantics of BDQL using frames and models as usual. This kind of semantics is called Kripke (or relational) semantics.

- A quantum dynamic frame is a pair $F = (\mathcal{H}, v)$ that consists of a Hilbert space $\mathcal{H}$ and a function v from Π_0 to the set $\mathcal{U}(\mathcal{H})$ of all unitary operators on $\mathcal{H}$. The function v is called an interpretation for atomic programs.
- A quantum dynamic model is a triple $M = (\mathcal{H}, v, V)$ that consists of a quantum dynamic frame $(\mathcal{H}, v)$ and a function V from L_0 to the set $\mathcal{C}(\mathcal{H})$ of all closed subspaces of $\mathcal{H}$. The function V is called an interpretation for atomic formulas.

The definition of quantum dynamic models states that atomic formulas are interpreted as a closed subspace of a Hilbert space. This interpretation is known as the algebraic semantics for Quantum Logic [16]. The set $\mathcal{C}(\mathcal{H})$ is called a Hilbert lattice [17] because it forms a lattice with meet $X \cap Y$ and join $X \sqcup Y = (X^\perp \cap Y^\perp)^\perp$ for any $X, Y \in \mathcal{C}(\mathcal{H})$, where $^\perp$ denotes the orthogonal complement. Note that $X \sqcup Y \supseteq X \cup Y$ and $X \cup Y \notin \mathcal{C}(\mathcal{H})$ in general.

Remark 2.1. Usually, Kripke frames are defined as a pair (tuple) that consists of a non-empty set S and relation(s) R on S. On the other hand, the quantum dynamic frames defined above have no relation(s). However, the relations can be recovered immediately using v. That is, the family $\{R_\pi : \pi \in \Pi_0\}$ of relations on $\mathcal{H}$ is constructed by

$$R_\pi = \{(s,t) : (v(\pi))(s) = t\}$$

for each $\pi \in \Pi_0$. For this reason, we use the word "frame" for quantum dynamic frames.

The interpretation v is defined for atomic programs, and V is defined for atomic formulas. These interpretations are extended to that for star-free regular programs and formulas, respectively. For any quantum dynamic model M, the function $[\![\ ]\!]^M : L \to \mathcal{C}(\mathcal{H})$ and family $\{R_a^M : a \in \Pi\}$ of relations on $\mathcal{H}$ are defined by simultaneous induction as follows:

1. $[\![p]\!]^M = V(p)$;
2. $[\![\neg A]\!]^M$ is the orthogonal complement of $[\![A]\!]^M$;
3. $[\![A \wedge B]\!]^M = [\![A]\!]^M \cap [\![B]\!]^M$;
4. $[\![[a]A]\!]^M = \{s \in \mathcal{H} : (s,t) \in R_a^M \text{ implies } t \in [\![A]\!]^M \text{ for any } t \in \mathcal{H}\}$;
5. $R^M_{\mathbf{skip}} = \{(s,t) : s = t\}$;
6. $R^M_{\mathbf{abort}} = \emptyset$;
7. $R^M_\pi = \{(s,t) : (v(\pi))(s) = t\}$;
8. $R^M_{a;b} = \{(s,t) : (s,u) \in R_a^M \text{ and } (u,t) \in R_b^M \text{ for some } u \in \mathcal{H}\}$;
9. $R^M_{a \cup b} = R_a^M \cup R_b^M$;
10. $R^M_{A?} = \{(s,t) : P_{[\![A]\!]^M}(s) = t\}$, where $P_{[\![A]\!]^M}$ stands for the projection onto $[\![A]\!]^M$.

Theorem 2.1. $[\![\ ]\!]^M$ is well-defined. That is, $[\![A]\!]^M \in \mathcal{C}(\mathcal{H})$ for each $A \in L$.

Proof. See our proof in [4]. □

The function $[\![\ ]\!]^M$ and family $\{R_a^M : a \in \Pi\}$ are uniquely determined if M is given. Recall that $v(\pi)$ is a function. On the other hand, R_a^M is a relation and may not be a function due to $\cup$.

Now we can understand the meaning of each program: **skip** does nothing, **abort** forces to halt without executing subsequent programs, ; is the composition operator, $\cup$ is the nondeterministic choice operator, and ? is the quantum test operator and is used to represent a result of projective measurement (see Section 4.1).

Henceforth, we write $(M, s) \models A$ for the condition $s \in [\![A]\!]^M$ as usual. That is, $(M, s) \models A$ if and only if $P_{[\![A]\!]^M}(s) = s$. A formula A is said to be satisfiable (resp. valid) if $(M, s) \models A$ for some (resp. any) M and $s \in \mathcal{H}$.

Remark 2.2. In most modal logics, a contradiction $A \wedge \neg A$ is not satisfiable. In other words, not $(M, s) \models A \wedge \neg A$ for any s. On the other hand, $A \wedge \neg A$ is satisfiable in BDQL because $(M, \mathbf{0}) \models A \wedge \neg A$, where $\mathbf{0}$ stands for the origin (zero vector) of $\mathcal{H}$. LQP [18] chooses different semantics from that in this paper to avoid this. That is, $\mathbf{0}$ (or the corresponding subspace $\{\mathbf{0}\}$) is not a state in the semantics of LQP. Unlike LQP, we allow $\mathbf{0}$ to be a state; otherwise, our definition is ill-defined (Theorem 2.1 does not hold).

The following theorem gives the theoretical background for rewriting the statement of the form $(M, s) \models A$ in our implementation explained in Section 4 of our previous work [4].

Theorem 2.2. The following holds for any M and $s \in \mathcal{H}$.

1. $(M, s) \models A \wedge B$, if and only if $(M, s) \models A$ and $(M, s) \models B$.
2. $(M, s) \models [\mathbf{skip}]A$ if and only if $(M, s) \models A$.
3. $(M, s) \models [\mathbf{abort}]A$.
4. $(M, s) \models [\pi]A$ if and only if $(M, (v(\pi))(s)) \models A$.
5. $(M, s) \models [a\ ;\ b]A$ if and only if $(M, s) \models [a][b]A$.
6. $(M, s) \models [a \cup b]A$ if and only if $(M, s) \models [a]A \wedge [b]A$.
7. $(M, s) \models [A?]B$ if and only if $(M, P_{[\![A]\!]^M}(s)) \models B$.

Proof. Straightforward. □

3. Probabilistic Dynamic Quantum Logic

In general, the outcome of a projective measurement is not determined in advance because there is a probability associated with each outcome. To capture this probabilistic ingredient, we employ a probabilistic operator $\mathbf{P}^{\geq r}$ from [3] to formulate Probabilistic Dynamic Quantum Logic (PDQL) as follows:

$$L \ni A ::= p \mid \neg A \mid A \wedge A \mid [a]A \mid \mathbf{P}^{\geq r}A,$$
$$\Pi \ni a ::= \mathbf{skip} \mid \mathbf{abort} \mid \pi \mid a\ ;\ a \mid a \cup a \mid A?,$$

where r denotes a rational number in the closed interval $[0, 1]$. The expression $\mathbf{P}^{\geq r}A$ means that a projective measurement of A on the current state of a quantum system will succeed with probability $\geq r$.

We can also use the probabilistic formula inside the dynamic modality for quantum tests as follows:

$$[A?^{\geq r}]B \triangleq \mathbf{P}^{\geq r}A \wedge [A?]B,$$

This means that if the quantum test $A?$ (or the projective measurement of A) succeeds with probability $\geq r$, then B will be the case after the successful execution of the quantum test.

Similarly, we can define other probabilistic operators $\mathbf{P}^{>r}$, $\mathbf{P}^{\leq r}$, $\mathbf{P}^{<r}$, $\mathbf{P}^{=r}$, and $\mathbf{P}^{\neq r}$. Regarding the binary projective measurement of A, we have the following axioms:

$$\mathbf{P}^{\geq r}A \to \mathbf{P}^{\leq r}\neg A, \qquad \mathbf{P}^{>r}A \to \mathbf{P}^{<r}\neg A, \qquad \mathbf{P}^{\leq r}A \to \mathbf{P}^{\geq (1-r)}\neg A,$$
$$\mathbf{P}^{<r}A \to \mathbf{P}^{>r}\neg A, \qquad \mathbf{P}^{=r}A \to \mathbf{P}^{=(1-r)}A, \qquad \mathbf{P}^{\neq r}A \to \mathbf{P}^{\neq (1-r)}.$$

Notice that these axioms only hold for the binary projective measurement of A.

The function $[\![\ ]\!]^M : L \to \mathcal{C}(\mathcal{H})$ is extended to handle the probabilistic operator $\mathbf{P}^{\geq r}$ using the Born rule as follows:

$$s \in [\![\mathbf{P}^{\geq r}A]\!]^M \text{ if and only if } \left\langle s \middle| P_{[\![A]\!]^M}(s) \right\rangle \geq r,$$

where $P_{[\![A]\!]^M}(s)$ is the projection of the current state s on the closed linear subspace spanned by $[\![A]\!]^M$, and $\langle s|s'\rangle$ is the inner product of two vectors representing two states s and s', respectively. Therefore, we write $(M, s) \models \mathbf{P}^{\geq r}A$ if and only if $s \in [\![\mathbf{P}^{\geq r}A]\!]^M$ as usual.

The following theorem gives the theoretical background for rewriting the statement involving the probabilistic operator for quantum tests.

Theorem 3.1. The following holds for any M, $s \in \mathcal{H}$, and $r \in [0, 1]$.

1. $(M, s) \models [A?^{\geq r}]B$, if and only if $(M, s) \models \mathbf{P}^{\geq r}A$ and $(M, s) \models [A?]B$.
2. $(M, s) \models [A?^{>r}]B$, if and only if $(M, s) \models \mathbf{P}^{>r}A$ and $(M, s) \models [A?]B$.
3. $(M, s) \models [A?^{\leq r}]B$, if and only if $(M, s) \models \mathbf{P}^{\leq r}A$ and $(M, s) \models [A?]B$.
4. $(M, s) \models [A?^{<r}]B$, if and only if $(M, s) \models \mathbf{P}^{<r}A$ and $(M, s) \models [A?]B$.
5. $(M, s) \models [A?^{=r}]B$, if and only if $(M, s) \models \mathbf{P}^{=r}A$ and $(M, s) \models [A?]B$.
6. $(M, s) \models [A?^{\neq r}]B$, if and only if $(M, s) \models \mathbf{P}^{\neq r}A$ and $(M, s) \models [A?]B$.

Proof. Straightforward. □

4. Application to Probabilistic Quantum Program Verification

This section describes the behavior and desired properties of some specific quantum programs in the language of PDQL. These properties can be verified automatically using our support tool as shown in Sections 5 and 6.

4.1. Basic Notions

In the beginning, we briefly review quantum computation and fix our notation. We assume the readers have basic knowledge of linear algebra.

Generally speaking, quantum systems are formulated as complex Hilbert spaces. However, for quantum computation, it is enough to consider specific Hilbert spaces called qubit systems. An n-qubit system is the complex 2^n-space $\mathbb{C}^{2^n}$, where $\mathbb{C}$ stands for the complex plane. Pure states in the n-qubit system $\mathbb{C}^{2^n}$ are unit vectors in $\mathbb{C}^{2^n}$. The orthogonal basis called computational basis in the one-qubit system $\mathbb{C}^2$ is a set $\{|0\rangle, |1\rangle\}$ that consists of the column vectors $|0\rangle = (1,0)^T$ and $|1\rangle = (0,1)^T$, where ^{T} denotes the transpose operator. The linear combinations $|+\rangle = (|0\rangle + |1\rangle)/\sqrt{2}$ and $|-\rangle = (|0\rangle - |1\rangle)/\sqrt{2}$ of $|0\rangle$ and $|1\rangle$ are also pure states. In general, $|\psi\rangle = c_0|0\rangle + c_1|1\rangle$ represents a pure state in the one-qubit system $\mathbb{C}^2$ provided that $|c_0|^2 + |c_1|^2 = 1$. This notation of vectors is called bra-ket notation (also called Dirac notation). $|\psi\rangle$ is called a ket vector. The bra vector $\langle\psi|$ is defined as a row vector whose elements are complex conjugates of the elements of the corresponding ket vector $|\psi\rangle$. In the two-qubit system $\mathbb{C}^4$, there are pure states that cannot be represented in the form $|\psi_1\rangle \otimes |\psi_2\rangle$ and are called entangled states, where $\otimes$ denotes the tensor product (more precisely, the Kronecker product). For example, the EPR state (Einstein-Podolsky-Rosen state) $|\mathrm{EPR}\rangle = (|00\rangle + |11\rangle)/\sqrt{2}$ is an entangled state, where $|00\rangle = |0\rangle \otimes |0\rangle$ and $|11\rangle = |1\rangle \otimes |1\rangle$.

Quantum computation is represented by unitary operators (also called quantum gates). There are various quantum gates. For example, the Hadamard gate H and Pauli gates X, Y, and Z are typical quantum gates on the one-qubit system $\mathbb{C}^2$ and are defined as follows:

$$H = \frac{1}{\sqrt{2}}\begin{pmatrix}1 & 1\\ 1 & -1\end{pmatrix}, \quad X = \begin{pmatrix}0 & 1\\ 1 & 0\end{pmatrix}, \quad Y = \begin{pmatrix}0 & -i\\ i & 0\end{pmatrix}, \quad Z = \begin{pmatrix}1 & 0\\ 0 & -1\end{pmatrix}.$$

Two typical quantum gates on the two-qubit system $\mathbb{C}^4$ are the controlled-X gate (also called the controlled NOT gate) CX and the swap gate SWAP are defined by

$$\mathrm{CX} = |0\rangle\langle 0| \otimes I + |1\rangle\langle 1| \otimes X,$$
$$\mathrm{SWAP} = \mathrm{CX}(I \otimes |0\rangle\langle 0| + X \otimes |1\rangle\langle 1|)\mathrm{CX},$$

where I denotes the identity matrix of size 2×2. Measurement is a completely different process from applying quantum gates. Here we roughly explain specific projective measurements. For the general definition of projective measurement, see [19]. Observe that $P_0 = |0\rangle\langle 0|$ and $P_1 = |1\rangle\langle 1|$ are projections, respectively. After executing the measurement $\{P_0, P_1\}$, a current state $|\psi\rangle = c_0|0\rangle + c_1|1\rangle$ is transitioned into $P_0|\psi\rangle/|c_0| = c_0|0\rangle/|c_0|$ with probability $|c_0|^2$ and into $P_1|\psi\rangle/|c_1| = c_1|1\rangle/|c_1|$ with probability $|c_1|^2$.

4.2. Standard Interpretation

To describe the quantum programs discussed in this paper, we fix

$$\Pi_0 = \{\mathtt{H}(i), \mathtt{X}(i), \mathtt{Y}(i), \mathtt{Z}(i), \mathtt{CX}(i,j), \mathtt{SWAP}(i,j) : i,j \in \mathbb{N}, i \neq j\},$$
$$L_0 = \{p(i, |\psi\rangle), p(i, i+1, |\Psi\rangle) : i \in \mathbb{N}, |\psi\rangle \in \mathbb{C}^2, |\Psi\rangle \in \mathbb{C}^4\},$$

where $\mathbb{N}$ stands for the set of all natural numbers (including 0). Because now atomic programs and atomic formulas are restricted, we only need to consider specific interpretations called the standard interpretations $\bar{v}$ and $\overline{V}$ instead of v and V, respectively. The standard interpretations are defined below.

A function $\bar{v} : \Pi_0 \to \mathcal{U}(\mathbb{C}^{2^n})$ is called the standard interpretation on $\mathbb{C}^{2^n}$ for atomic programs if

$$\begin{aligned}
\bar{v}(\mathtt{H}(i)) &= I^{\otimes i} \otimes H \otimes I^{\otimes n-i-1}, \quad \bar{v}(\mathtt{X}(i)) = I^{\otimes i} \otimes X \otimes I^{\otimes n-i-1},\\
\bar{v}(\mathtt{Y}(i)) &= I^{\otimes i} \otimes Y \otimes I^{\otimes n-i-1}, \quad \bar{v}(\mathtt{Z}(i)) = I^{\otimes i} \otimes Z \otimes I^{\otimes n-i-1},\\
\bar{v}(\mathtt{CX}(i,j)) &= I^{\otimes i} \otimes |0\rangle\langle 0| \otimes I^{\otimes n-i-1} + (I^{\otimes i} \otimes |1\rangle\langle 1| \otimes I^{\otimes n-i-1})(I^{\otimes j} \otimes X \otimes I^{\otimes n-j-1}),\\
\bar{v}(\mathtt{SWAP}(i,j)) &= \bar{v}(\mathtt{CX}(i,j)\ ;\ \mathtt{CX}(j,i)\ ;\ \mathtt{CX}(i,j)),
\end{aligned}$$

where

$$I^{\otimes i} = \overbrace{I \otimes \cdots \otimes I}^{i}.$$

That is, under the standard interpretation, $\mathtt{H}(i)$, $\mathtt{X}(i)$, $\mathtt{Y}(i)$, $\mathtt{Z}(i)$ execute the corresponding quantum gate on the i-th qubit, $\mathtt{CX}(i,j)$ executes the Pauli gate X on the target qubit (j-th qubit) depending on the state of the control qubit (i-th qubit), and $\mathtt{SWAP}(i,j)$ swaps the i-th and j-th qubits.

A functionato is called the standard interpretation on $\mathbb{C}^{2^n}$ for atomic formulas if

$$\begin{aligned}
\overline{V}(p(i,|\psi\rangle)) &= \mathbb{C}^{2^i} \otimes \mathrm{span}\{|\psi\rangle\} \otimes \mathbb{C}^{2^{n-i-1}},\\
\overline{V}(p(i,i+1,|\Psi\rangle)) &= \mathbb{C}^{2^i} \otimes \mathrm{span}\{|\Psi\rangle\} \otimes \mathbb{C}^{2^{n-i-2}},
\end{aligned}$$

where $\mathrm{span}\{|\psi\rangle\}$ (resp. $\mathrm{span}\{|\Psi\rangle\}$) stands for the subspace spanned by $\{|\psi\rangle\}$ (resp. $\{|\Psi\rangle\}$).

In what follows, we write $\overline{M}_n$ for $(\mathbb{C}^{2^n}, \bar{v}, \overline{V})$, where the index n represents the number of qubits. In addition, we use the following abbreviation to conventionally describe quantum programs for quantum tests with probability in PDQL:

$$\textbf{if } \mathbf{P}^{\geq r} A \textbf{ then } a \textbf{ else } b \textbf{ fi} = (A?^{\geq r}\ ;\ a) \cup (\neg A?^{\leq (1-r)}\ ;\ b).$$

This program means the selection depends on the outcomes of projective measurement with respect to the probabilities. That is, execute a or b depending on the result of the measurement $\{P_{[\![A]\!]^M}, P_{[\![\neg A]\!]^M}\}$ with respect to $\mathbf{P}^{\geq r} A$ and $\mathbf{P}^{\leq (1-r)} \neg A$.

4.3. Case Studies

In the sequel, we use PDQL to verify some case studies involving probability, which intentionally is added to the formalizations of quantum programs to demonstrate the expressiveness of PDQL. For the sake of brevity, this section only describes some quantum programs, including Quantum Relay Scheme [11], Bidirectional Quantum Teleportation [12], Two-qubit Quantum Teleportation [14], and Quantum Network Coding [15]. Meanwhile, other quantum programs, including Superdense Coding [7], Quantum Teleportation [8], Quantum Secret Sharing [9], Entanglement Swapping [10], and Quantum Gate Teleportation [13], can be derived in a similar way. The reader interested in them is referred to our previous work [4] for their description without probability.

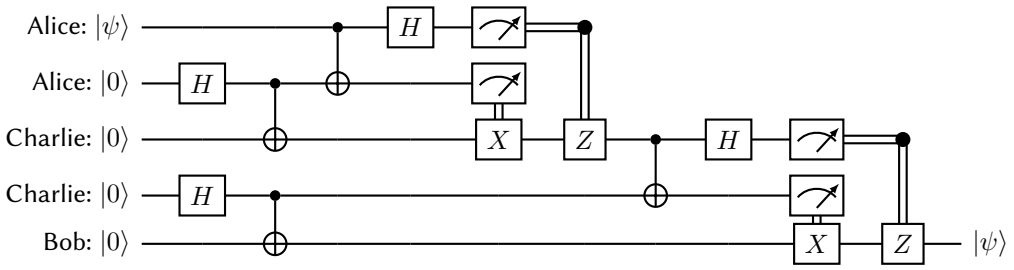

Figure 1: Quantum Relay Scheme

Quantum Relay Scheme

Quantum Relay Scheme [11] allows us to teleport a quantum state from one quantum device to another wirelessly even though these two devices do not share EPR pairs mutually, which differs from Quantum Teleportation [8]. The quantum circuit of Quantum Relay Scheme is depicted in Figure 1. The program of Quantum Relay Scheme with probability intentionally added is described as follows:

$$\begin{aligned}\mathbf{relay} = {} & \mathtt{H}(1)\ ;\ \mathtt{CX}(1,2)\ ;\ \mathtt{H}(3)\ ;\ \mathtt{CX}(3,4)\ ;\ \mathtt{CX}(0,1)\ ;\ \mathtt{H}(0) \\ & ;\ \mathbf{if}\ p(1,|0\rangle)^{\geq 1/2}\ \mathbf{then\ skip\ else}\ \mathtt{X}(2)\ \mathbf{fi} \\ & ;\ \mathbf{if}\ p(0,|0\rangle)^{\geq 1/2}\ \mathbf{then\ skip\ else}\ \mathtt{Z}(2)\ \mathbf{fi} \\ & ;\ \mathtt{CX}(2,3)\ ;\ \mathtt{H}(2) \\ & ;\ \mathbf{if}\ p(3,|0\rangle)^{\geq 1/2}\ \mathbf{then\ skip\ else}\ \mathtt{X}(4)\ \mathbf{fi} \\ & ;\ \mathbf{if}\ p(2,|0\rangle)^{\geq 1/2}\ \mathbf{then\ skip\ else}\ \mathtt{Z}(4)\ \mathbf{fi}.\end{aligned}$$

The desired property of Quantum Relay Scheme is that "a pure state $|\psi\rangle$ is correctly teleported." In PDQL, this property is expressed as follows:

$$(\overline{M}_5, |\psi\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle) \models [\mathbf{relay}]p(4,|\psi\rangle).$$

Bidirectional Quantum Teleportation

Bidirectional Quantum Teleportation [12] is based on EPR pairs and entanglement swapping to allow two users to simultaneously transmit an unknown single qubit to each other. The quantum circuit of Bidirectional Quantum Teleportation is depicted in Figure 2. The program of Bidirectional Quantum Teleportation with probability intentionally added is described as follows:

$$\begin{aligned}\mathbf{biTeleport} = {} & \mathtt{H}(2)\ ;\ \mathtt{CX}(2,3)\ ;\ \mathtt{H}(4)\ ;\ \mathtt{CX}(4,5)\ ;\ \mathtt{CX}(0,2)\ ;\ \mathtt{CX}(1,5)\ ;\ \mathtt{H}(0)\ ;\ \mathtt{H}(1) \\ & ;\ \mathbf{if}\ p(2,|0\rangle)^{\geq 1/2}\ \mathbf{then\ skip\ else}\ \mathtt{X}(3)\ \mathbf{fi} \\ & ;\ \mathbf{if}\ p(0,|0\rangle)^{\geq 1/2}\ \mathbf{then\ skip\ else}\ \mathtt{Z}(3)\ \mathbf{fi} \\ & ;\ \mathbf{if}\ p(5,|0\rangle)^{\geq 1/2}\ \mathbf{then\ skip\ else}\ \mathtt{X}(4)\ \mathbf{fi}\end{aligned}$$

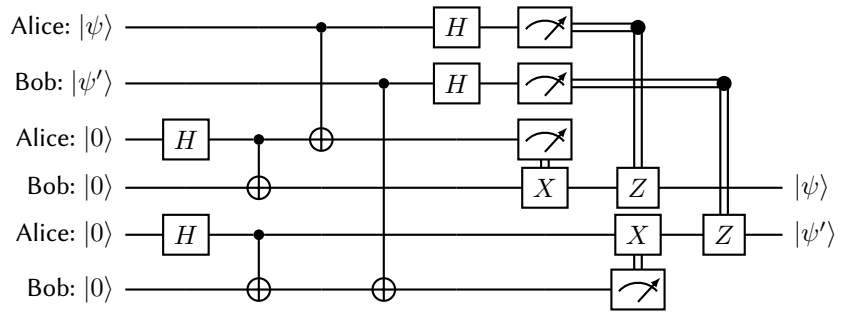

Figure 2: Bidirectional Quantum Teleportation

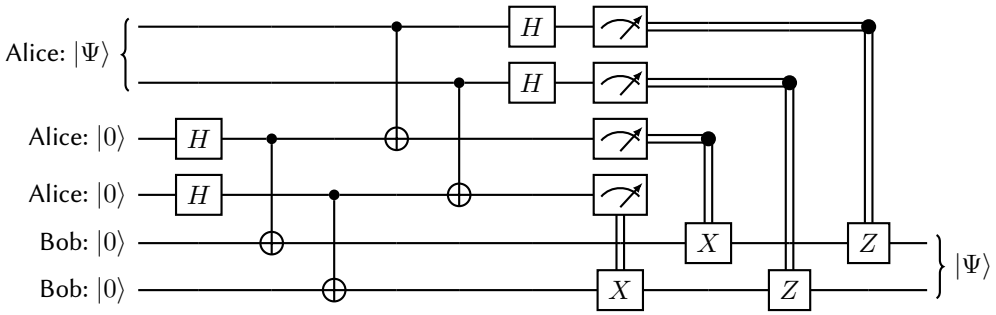

Figure 3: Two-qubit Quantum Teleportation

$$; \textbf{if } p(1, |0\rangle)^{\geq 1/2} \textbf{ then skip else } \mathtt{Z}(4) \textbf{ fi}.$$

The desired property of Bidirectional Quantum Teleportation is that "two pure states $|\psi\rangle$ and $|\psi'\rangle$ owned by two users are correctly teleported to each other." In PDQL, this property is expressed as follows:

$$(\overline{M}_6, |\psi\rangle \otimes |\psi'\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle) \models [\textbf{biTeleport}]p(3, |\psi\rangle) \wedge p(4, |\psi'\rangle).$$

Two-qubit Quantum Teleportation

Two-qubit Quantum Teleportation [14] allows us to teleport an arbitrary two-qubit pure states $|\Psi\rangle = c_{00}|00\rangle + c_{01}|01\rangle + c_{10}|10\rangle + c_{11}|11\rangle$ from one source to another, where $|c_{00}|^2 + |c_{01}|^2 + |c_{10}|^2 + |c_{11}|^2 = 1$. The quantum circuit of Two-qubit Quantum Teleportation is depicted in Figure 3. The program of Two-qubit Quantum Teleportation with probability intentionally added is described as follows:

$$\begin{aligned}\textbf{twoTeleport} = {} & \mathtt{H}(2) \,;\, \mathtt{H}(3) \,;\, \mathtt{CX}(2,4) \,;\, \mathtt{CX}(3,5) \,;\, \mathtt{CX}(0,2) \,;\, \mathtt{CX}(1,3) \,;\, \mathtt{H}(0) \,;\, \mathtt{H}(1) \\ & ; \textbf{if } p(3, |0\rangle)^{\geq 1/2} \textbf{ then skip else } \mathtt{X}(5) \textbf{ fi} \\ & ; \textbf{if } p(2, |0\rangle)^{\geq 1/2} \textbf{ then skip else } \mathtt{X}(4) \textbf{ fi} \\ & ; \textbf{if } p(1, |0\rangle)^{\geq 1/2} \textbf{ then skip else } \mathtt{Z}(5) \textbf{ fi}\end{aligned}$$

$$; \textbf{if } p(0,|0\rangle)^{\geq 1/2} \textbf{ then skip else } \mathtt{Z}(4) \textbf{ fi}.$$

The desired property of Two-qubit Quantum Teleportation is that "arbitrary two-qubit pure states $|\Psi\rangle$ is correctly teleported." In PDQL, this property is expressed as follows:

$$(\overline{M}_6, |\Psi\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle \otimes |0\rangle) \models [\textbf{twoTeleport}]p(4,5,|\Psi\rangle).$$

Quantum Network Coding

Quantum Network Coding [15] enables the simultaneous generation of EPR pairs by using only local operations and classical communication (LOCC) and the sharing of EPR pairs between adjacent quantum repeaters. Simultaneous quantum communication can be achieved by using Teleportation with these EPR states. This facilitates the transmission of quantum information on complex networks at higher rates than straightforward routing strategies, such as the simple entanglement swapping strategy. For the sake of simplicity, we do not depict the quantum circuit of Quantum Network Coding here, while the reader is referred to their original paper [15]. The protocol introduces two new techniques called "Connection" and "Removal" to manipulate entangled states and systematize the methods of encoding, which are described as follows:

$$\begin{aligned}
\textbf{con}(N1,(N2 \to N3)) &= \mathtt{CX}(N1,N2) \\
&\quad ; \textbf{ if } p(N2,|0\rangle)^{\geq 1/2} \textbf{ then skip else } \mathtt{X}(N3) \textbf{ fi}. \\
\textbf{fanout}(N1,(N2 \to N3),(N4 \to N5)) &= \mathtt{con}(N1,(N2 \to N3)) \ ; \mathtt{con}(N1,(N4 \to N5)). \\
\textbf{add}(N1,N2,(N3 \to N4)) &= \mathtt{CX}(N1,N3) \ ; \mathtt{CX}(N2,N3) \\
&\quad ; \textbf{ if } p(N3,|0\rangle)^{\geq 1/2} \textbf{ then skip else } \mathtt{X}(N4) \textbf{ fi}. \\
\textbf{rem}(N1 \to N2) &= \mathtt{H}(N1) \\
&\quad ; \textbf{ if } p(N1,|0\rangle)^{\geq 1/2} \textbf{ then skip else } \mathtt{Z}(N2) \textbf{ fi}. \\
\textbf{remadd}(N3 \to (N1,N2)) &= \mathtt{H}(N3) \\
&\quad ; \textbf{ if } p(N3,|0\rangle)^{\geq 1/2} \textbf{ then skip} \\
&\quad \textbf{else } \mathtt{Z}(N1) \ ; \mathtt{Z}(N2) \textbf{ fi} \ .
\end{aligned}$$

where $N1$, $N2$, $N3$, and $N4$ are natural numbers in the range $[0,9]$. Then, the program of Quantum Network Coding with probability intentionally added is described via **con**, **fanout**, **add**, **rem**, and **remadd** as follows:

$$\begin{aligned}
\textbf{networkcoding} = \ & \mathtt{H}(0) \ ; \mathtt{CX}(0,1) \ ; \mathtt{H}(2) \ ; \mathtt{CX}(2,3) \\
& ; \mathtt{H}(4) \ ; \mathtt{CX}(4,5) \ ; \mathtt{H}(6) \ ; \mathtt{CX}(6,7) \\
& ; \mathtt{H}(8) \ ; \mathtt{CX}(8,9) \ ; \mathtt{H}(10) \ ; \mathtt{CX}(10,11) \\
& ; \mathtt{H}(12) \ ; \mathtt{CX}(12,13) \\
& ; \mathtt{con}(0,(2 \to 3) \ ; \mathtt{con}(4,(6 \to 7)) \ ; \mathtt{add}(3,7,(8 \to 9)) \\
& ; \mathtt{fanout}(9,(10 \to 11),(12 \to 13)) \ ; \mathtt{CX}(13,5) \ ; \mathtt{CX}(11,1) \\
& ; \mathtt{rem}(11 \to 9) \ ; \mathtt{rem}(13 \to 9) \ ; \mathtt{remadd}(9 \to (3,7))
\end{aligned}$$

$$; \texttt{rem}(3 \to 0) \; ; \texttt{rem}(7 \to 4)$$
$$; \texttt{SWAP}(2, 4) \; ; \texttt{SWAP}(0, 4).$$

The desired property of Quantum Network Coding is that "the two EPR states (in this case $|\mathrm{EPR}\rangle$) are simultaneously generated." In PDQL, this property is expressed as follows:

$$(\overline{M}_{14}, |0\rangle^{\otimes 14}) \models [\mathbf{networkcoding}]p(1, 2, |\mathrm{EPR}\rangle) \wedge p(4, 5, |\mathrm{EPR}\rangle).$$

5. Implementation of Probabilistic Dynamic Quantum Logic

This section describes the implementation of PDQL in Maude [5], a specification/programming language based on rewriting logic [20]. Hence, the notations used in this section follow the Maude syntax.

5.1. Syntax of Probabilistic Dynamic Quantum Logic

Recalling *the syntax of Basic Dynamic Quantum Logic* (BDQL) in our previous work [4], we defined two sorts `AtomicProg` and `Prog` for atomic programs Π_0 and star-free regular programs Π, respectively, where `AtomicProg` is a subsort of `Prog`. Additionally, we defined several operators for atomic programs: `I(_)`, `H(_)`, `X(_)`, `Y(_)`, and `Z(_)`. These operators take a natural number as input, denoting the index of a qubit of a pure state on which the quantum gates I, H, X, Y, and Z, will be applied, respectively. Furthermore, `CX(_,_)` and `SWAP(_,_)` operators take two natural numbers as inputs, denoting the indices of two qubits of a pure state on which CX and $SWAP$ gates will be applied. We also defined `abort`, `skip`, `_;_` , `_U_`, and `_?` operators to construct star-free regular programs in BDQL, as specified in Section 2.

We defined two sorts `AtomicFormula` and `Formula` for atomic formulas L_0 and general formulas L in BDQL, respectively, where `AtomicFormula` is a subsort of `Formula`. Additionally, we defined several operators for constructing formulas in BDQL: `P(_,_)` and `P(_,_,_)` operators are atomic formulas representing projections of the forms $p(i, |\psi\rangle)$ and $p(i, j, |\Psi\rangle)$, respectively; and the `neg_`, `_/\_`, and `[_]_` operators to construct formulas in BDQL, as specified in Section 2. Moreover, the `if_then_else_fi` command in Section 4.2 was also implemented for convenient use.

Regarding *the syntax of Probabilistic Dynamic Quantum Logic* (PDQL), we extend the syntax of BDQL by introducing the probabilistic operator as follows:

```
op _with prob >=_  : Formula Scalar -> Formula [ctor] .
op _with prob >_   : Formula Scalar -> Formula [ctor] .
op _with prob <=_  : Formula Scalar -> Formula [ctor] .
op _with prob <_   : Formula Scalar -> Formula [ctor] .
op _with prob ==_  : Formula Scalar -> Formula [ctor] .
op _with prob =/=_ : Formula Scalar -> Formula [ctor] .
```

where the five operators above explicitly denote the probabilistic operators $\mathbf{P}^{\geq r}A$, $\mathbf{P}^{>r}A$, $\mathbf{P}^{\leq r}A$, $\mathbf{P}^{<r}A$, $\mathbf{P}^{=r}A$, $\mathbf{P}^{\neq r}A$, respectively, and r is a rational number $\in [0, 1]$. However, we define r as belonging to a sort of Scalar in our formalization because we also want to reason on symbolic complex constants included in the value of probability.

5.2. Semantics of Probabilistic Dynamic Quantum Logic

We extend the semantics of BDQL to describe the semantics of PDQL by introducing the following equations, which strictly follow Theorem 3.1. For the sake of simplicity, we only describe equations used for `_with prob >=_` operator representing the probabilistic operator $\mathbf{P}^{\geq r} A$, while other operators are similar.

We define some Maude variables before use as follows: `Q` is a Maude variable of quantum states, `Phi` is a Maude variable of formulas, `Prob` is a Maude variable of scalars, `M` is a Maude variable of matrices, and `N`, `N1`, and `N2` is Maude variables of natural numbers.

We first handle the probabilistic operator in a formula as follows:

```
ceq Q |= P(N, M) with prob >= Prob = emptyJS if (Q).Prob(N, M) .>= Prob .
ceq Q |= P(N1, N2, M) with prob >= Prob = emptyJS if (Q).Prob(N1, N2, M) .>= Prob.
```

where `emptyJS` denote success in checking and `.>=` operator denotes "greater than or equal to" when comparing two complex numbers.

We then handle the probabilistic operator in the dynamic modality as follows:

```
ceq Q |= [(P(N, M) with prob >= Prob)?] Phi = Q |= [P(N, M)?] Phi
                                          if (Q).Prob(N, M) .>= Prob .
```

With respect to the equations specified in our support tool, $(M, s) \models A$ if `s |= A` is simplified to `emptyJS`. Otherwise, $(M, s) \not\models A$ if `s |= A` is not simplified to `emptyJS`.

6. Experimental Results

In this section, we illustrate the application of our support tool for verifying Quantum Relay Scheme in Maude, using it as an example. Similar procedures can be done for verifying other quantum programs, which are publicly available at https://github.com/canhminhdo/DQL. Following that, we summarize the experimental results for several quantum programs employed in our experiments, including Superdense Coding [7], Quantum Teleportation [8], Quantum Secret Sharing [9], Entanglement Swapping [10], Quantum Relay Scheme [11], Bidirectional Quantum Teleportation [12], Two-qubit Quantum Teleportation [14], Quantum Gate Teleportation [13], and Quantum Network Coding [15].

Assuming that `RELAY-SCHEME` represents the specification of Quantum Relay Scheme, `initQState` represents for the initial state for `RELAY-SCHEME`, and `qubitAt` denotes the function to retrieve a single qubit at a specific index. We can verify the correctness of Quantum Relay Scheme using our support tool by executing the following `reduce` command in Maude:

```
red in RELAY-SCHEME : initQState |= [
    H(1) ; CX(1, 2) ; H(3) ; CX(3, 4) ; CX(0, 1) ; H(0) ;
    if P(1, |0>) with prob >= 1/2 then skip else X(2) fi ;
    if P(0, |0>) with prob >= 1/2  then skip else Z(2) fi ;
    CX(2, 3) ; H(2) ;
    if P(3, |0>) with prob >= 1/2  then skip else X(4) fi ;
    if P(2, |0>) with prob >= 1/2  then skip else Z(4) fi
] P(4, qubitAt(initQState, 0)) .
```

Table 1
Experimental results with our support tool for several quantum programs in PDQL

Protocol	Qubits	Rewrite Steps	Verification Time
Superdense Coding	2	2,451	1ms
Quantum Teleportation	3	9,034	4ms
Quantum Secret Sharing	4	39,041	18ms
Entanglement Swapping	4	14,272	6ms
Quantum Relay Scheme	5	44,939	26ms
Bidirectional Quantum Teleportation	6	47,717	27ms
Two-qubit Quantum Teleportation	6	660,313	238ms
Quantum Gate Teleportation	6	667,806	250ms
Quantum Network Coding	14	11,568,281	4,811ms

The `reduce` command automatically performs an equational simplification process based on the equations defined in our tool. Within just a few moments, the command returns an `emptyJS` result. Consequently, this confirms the correctness of Quantum Relay Scheme with probability intentionally added by using our support tool, the implementation of PDQL in Maude. The property being verified informally says that the first qubit of the initial quantum state at index 0 is correctly teleported to the fifth qubit at index 4 in the final quantum state.

We conducted experiments on an iMac equipped with a 4 GHz microprocessor with eight cores and 32 GB memory of RAM. Table 1 presents the experimental results. We have successfully confirmed the correctness of Superdense Coding [7], Quantum Teleportation [8], Quantum Secret Sharing [9], Entanglement Swapping [10], Quantum Relay Scheme [11], Bidirectional Quantum Teleportation [12], Quantum Gate Teleportation [13], Two-qubit Quantum Teleportation [14], and Quantum Network Coding [15] in accordance with the specified properties outlined in Section 4. Comparing to our previous work [4], we extend our case studies by adding additional quantum programs to be verified with and without the probability. For all case studies from two to 14 qubits, we can quickly verify their correctness within just a few moments using our support tool despite the considerable number of rewrite steps involved. These results would have been extremely challenging to obtain without the assistance of computer programs like our support tool, especially in the case of Quantum Network Coding. This shows the usefulness of our automated approach for verifying quantum programs in PDQL, employing the symbolic approach for quantum computation from [6].

7. Related Work

Quantum Hoare Logic (QHL), as introduced by M. Ying in his work [21], was conceived with the aim of serving as a quantum counterpart to Hoare Logic. When examining the logical aspects, BDQL, in contrast to QHL, possesses the capability to express more foundational elements within quantum programs. Specifically, QHL faces limitations when dealing with the $\mathbf{if} \cdots \mathbf{fi}$ statement, which represents non-deterministic measurements, as it cannot be further divided. In contrast, BDQL can explicitly express its non-deterministic characteristic by employing the choice operator $\cup$. Additionally, it's worth noting that QHL lacks the inclusion of a test operator in its syntax and does not support probability in its formulas.

In our paper, we used Maude as our implementation language. In contrast, the work by [22] utilized PRISM, a probabilistic symbolic model checker, to verify quantum protocols. Unlike our approach, [22] requires the task of enumerating states and precomputing state transitions, which are then encoded into a specification in PRISM. Conversely, our method doesn't necessitate this state enumeration and precomputing state transitions because we formalize both the quantum computation and the semantics of PDQL using equations. The verification problem is carried out automatically through an equational simplification process within Maude. Moreover, they only support a small number of qubits, saying less than five qubits. As demonstrated, we can verify Quantum Network Coding with 14 qubits within a few moments, demonstrating the scalability of our approach.

8. Conclusion

We have extended Basic Dynamic Quantum Logic (BDQL) to Probabilistic Dynamic Quantum Logic (PDQL) to verify probabilistic quantum programs by introducing the probabilistic operator $\mathbf{P}^{\geq r}$ and some others in the formulas of PDQL. We have also developed a support tool in Maude to automate the entire verification process. Several quantum programs have been verified automatically using the support tool, demonstrating the usefulness of PDQL and our support tool. As one piece of our future work, we would like to conduct more case studies where the probabilistic properties are realistically expressed, such as Quantum Search Algorithm and Quantum Leader Election Protocol.

Acknowledgments

The research was supported by JAIST Research Grant for Fundamental Research. The research of the first author was supported by JSPS KAKENHI Grant Number JP23K19959. The research of the first and the third authors was supported by JST SICORP Grant Number JPMJSC20C2. The research of the second author was supported by Grant-in-Aid for JSPS Research Fellow Grant Number JP22KJ1483.

References

[1] P. Shor, Algorithms for quantum computation: discrete logarithms and factoring, in: Proceedings 35th Annual Symposium on Foundations of Computer Science, 1994, pp. 124–134. doi:10.1109/SFCS.1994.365700.

[2] D. Harel, D. Kozen, J. Tiuryn, Dynamic Logic, MIT Press, 2000.

[3] A. Baltag, S. Smets, Reasoning about quantum information: An overview of quantum dynamic logic, Applied Sciences 12 (2022). URL: https://www.mdpi.com/2076-3417/12/9/4458. doi:10.3390/app12094458.

[4] T. Takagi, C. M. Do, K. Ogata, Automated quantum program verification in a dynamic quantum logic (to appear), in: DaLí: Dynamic Logic – New trends and applications, 2023.

[5] M. Clavel, et al., All About Maude, volume 4350 of *Lecture Notes in Computer Science*, Springer, 2007. doi:10.1007/978-3-540-71999-1.

[6] C. M. Do, K. Ogata, Symbolic model checking quantum circuits in maude, in: The 35th International Conference on Software Engineering and Knowledge Engineering, SEKE 2023, 2023, pp. 103–108. doi:10.18293/SEKE2023-014.

[7] C. H. Bennett, S. J. Wiesner, Communication via one- and two-particle operators on Einstein-Podolsky-Rosen states, Phys. Rev. Lett. 69 (1992) 2881–2884. doi:10.1103/PhysRevLett.69.2881.

[8] C. H. Bennett, G. Brassard, C. Crépeau, R. Jozsa, A. Peres, W. K. Wootters, Teleporting an unknown quantum state via dual classical and Einstein-Podolsky-Rosen channels, Phys. Rev. Lett. 70 (1993) 1895–1899. doi:10.1103/PhysRevLett.70.1895.

[9] M. Hillery, V. Bužek, A. Berthiaume, Quantum secret sharing, Phys. Rev. A 59 (1999) 1829–1834. doi:10.1103/PhysRevA.59.1829.

[10] M. Żukowski, A. Zeilinger, M. A. Horne, A. K. Ekert, "Event-ready-detectors" Bell experiment via entanglement swapping, Phys. Rev. Lett. 71 (1993) 4287–4290. doi:10.1103/PhysRevLett.71.4287.

[11] S.-T. Cheng, C.-Y. Wang, M.-H. Tao, Quantum communication for wireless wide-area networks, IEEE Journal on Selected Areas in Communications 23 (2005) 1424–1432. doi:10.1109/JSAC.2005.851157.

[12] S. Hassanpour, M. Houshmand, Bidirectional quantum teleportation and secure direct communication via entanglement swapping, 2014. doi:10.48550/arXiv.1411.0206. arXiv:1411.0206.

[13] D. Gottesman, I. L. Chuang, Demonstrating the viability of universal quantum computation using teleportation and single-qubit operations, Nature 402 (1999) 390–393. doi:10.1038/46503.

[14] G. Rigolin, Quantum teleportation of an arbitrary two-qubit state and its relation to multipartite entanglement, Phys. Rev. A 71 (2005) 032303. URL: https://link.aps.org/doi/10.1103/PhysRevA.71.032303. doi:10.1103/PhysRevA.71.032303.

[15] T. Satoh, F. L. Gall, H. Imai, Quantum network coding for quantum repeaters, Physical Review A 86 (2012). doi:10.1103/physreva.86.032331.

[16] G. Birkhoff, J. von Neumann, The logic of quantum mechanics, Annals of mathematics 57 (1936) 823–843. doi:10.2307/1968621.

[17] M. Rédei, Quantum logic in algebraic approach, volume 91 of *Fundamental Theories of*

Physics, Springer, 1998. doi:10.1007/978-94-015-9026-6.

[18] A. Baltag, S. Smets, LQP: the dynamic logic of quantum information, Mathematical structures in computer science 16 (2006) 491–525. doi:10.1017/S0960129506005299.

[19] M. A. Nielsen, I. L. Chuang, Quantum Computation and Quantum Information: 10th Anniversary Edition, Cambridge University Press, 2011.

[20] J. Meseguer, Twenty years of rewriting logic, The Journal of Logic and Algebraic Programming 81 (2012) 721–781. doi:10.1016/j.jlap.2012.06.003.

[21] M. Ying, Floyd–hoare logic for quantum programs, ACM Transactions on Programming Languages and Systems (TOPLAS) 33 (2012) 1–49. doi:10.1145/2049706.2049708.

[22] S. Gay, R. Nagarajan, N. Papanikolaou, Probabilistic model–checking of quantum protocols, arXiv preprint quant-ph/0504007 (2005). doi:10.48550/arXiv.quant-ph/0504007.