arXiv:2512.22383v1 [cs.PL] 26 Dec 2025

量子数据与操作的符号化规约与推理

应明生

量子软件与信息中心，悉尼科技大学，澳大利亚新南威尔士州尤提莫百老汇街*15*号，邮编*2007*

摘要

在量子信息与计算研究中，符号方法已被广泛用于人工规约和推理量子态与操作。同时，符号方法对于确保量子算法和程序的自动推理与验证工具具备可扩展性和高效性至关重要。然而，目前仍缺乏关于量子数据与操作的符号化规约和推理的形式化理论，这极大地限制了自动验证技术在量子计算中的实际应用。

本文提出一种通用逻辑框架，称为符号算子逻辑**SOL**，可用于对量子数据与操作进行符号化规约和推理。在该框架中，经典一阶逻辑语言被嵌入用于规约量子数据与操作（包括其递归定义）的形式算子语言中。这种嵌入使我们能够在选定的底层经典数据理论（例如布尔代数或群论）下对其性质进行推理，从而利用现有的、为经典计算开发的自动验证工具。需要强调的是，将经典一阶逻辑嵌入**SOL** 正是符号方法得以实现的关键。

我们设想，该框架可为在 Lean、Coq 等证明助手及相关系统中对量子计算与信息进行形式化验证和自动定理证明提供概念基础。

关键词：量子计算、量子信息、形式化规约、形式化验证、符号技术

1. 引言

过去十年间，人们开发了多种用于量子算法和程序的形式化验证工具。值得注意的例子包括基于 Coq 证明助手的 QWIRE [33], SQIR [23], 和 CoqQ [51], ； 基于 Isabelle/HOL 构建的 QHLProver [28], qrhl-tool [37], 和 IMD [6], ； 使用 Why3 和 SMT 求解器的 QBRICKS [9], ； 以及基于 ZX 演算的工具 [26]。由于量子态、操作和谓词（在量子霍尔逻辑中 [43]用作量子程序的前置条件/后置条件 [15], ）都以向量和矩阵的形式进行数学形式化——例如，含有 n 个量子比特的混合态的密度矩阵是一个 $2^n \times 2^n$ 矩阵，而作用于 n 个量子比特的量子门也以类似方式表示——因此，这些工具在实际应用和规模化应用中的主要瓶颈在于大型矩阵的规约与操作。为应对这一挑战，人们提出了多种方法，包括张量网络 [5], 决策图 [40], 基于自动机的表示 [10], 抽象解释 [49],和分离逻辑 [27, 50, 29]。

提高经典计算中验证技术可扩展性的一种特别有效的方法是采用符号方法（例如符号模型检测 [7] 和符号执行 [25]）。特别是在基于霍尔逻辑 [2], 对经典程序进行演绎验证时，前置条件/后置条件被表示为一阶逻辑公式，并通过形式逻辑推理证明验证条件。因此，为了实现对量子程序的可扩展、高效验证，一个重要的问题是：如何将符号方法扩展到量子计算中?

1.1. 动机示例

事实上，符号方法的理念早已广泛应用于量子信息与计算的研究中。量子数据（即纯态和混合量子态）以及量子操作——包括酉变换（或量子门）和量子测量（或观测量）——通常使用线性代数语言建模，其中状态和操作按照量子力学的形式体系表示为复数域上的向量和矩阵。然而，在量子计算与量子信息（以及量子物理）的研究中，这些向量和矩阵往往不仅包含具体的

复数，还包含符号变量、函数和表达式。此外，人们也常描述多量子比特系统，并使用符号下标表示不同的量子比特，如下所示:

例1.1（量子态的符号化规约）。 在量子计算与信息文献中，通常会以如下形式表示量子态

$$\cos\frac{\theta}{2}|2k+1\rangle_{q[m+1]}|n-5\rangle_{q[3m-4]}+\sin\frac{\theta}{2}|2k-1\rangle_{q[m+1]}|n+3\rangle_{q[3m-4]} \qquad (1)$$

其中，下标 $q[m+1]$ 和 $q[3m-4]$ 分别表示复合量子系统的第（$m+1$）个和第（$3m-4$）个子系统。从某种意义上说，量子态（*1*）由经典变量 θ、k、m、n参数化。

此外，量子计算与信息中的许多结果都是通过符号操作和推理得出的。一个简单的例子是单个量子比特的 Z–Y 分解（参见 [31]中的定理4.1）：

例1.2（关于量子门的符号推理）。 令 I、X、Y、Z表示作用于一个量子比特的泡利门，并定义绕x、y、z坐标轴的旋转算子如下:

$$R_x(\theta)=\cos\frac{\theta}{2}I-i\sin\frac{\theta}{2}X,\qquad R_y(\theta)=\cos\frac{\theta}{2}I-i\sin\frac{\theta}{2}Y,\qquad R_z(\theta)=\cos\frac{\theta}{2}I-i\sin\frac{\theta}{2}Z$$

其中，实数 θ 是旋转角。于是，单个量子比特上的门 $U=(a_{ij})_{i,j=0}^{1}$ 的 *Z–Y* 分解断言:

$$|a_{00}|^2+|a_{01}|^2=|a_{10}|^2+|a_{11}|^2=1\wedge a_{00}a_{10}^*+a_{01}a_{11}^*=0$$
$$\Rightarrow(\exists\theta,\theta_1,\theta_2,\theta_3)\left(\sum_{i,j=0}^{1}a_{ij}|i\rangle_q\langle j|=e^{i\theta}R_z(\theta_1)[q]R_y(\theta_2)[q]R_z(\theta_3)[q]\right) \qquad (2)$$

其中，$R_z(\theta_1)[q]$、$R_y(\theta_2)[q]$ 和 $R_z(\theta_3)[q]$ 分别表示量子门 $R_z(\theta_1)$ $R_y(\theta_2)$ 和$R_z(\theta_3)$ 分别作用于量子比特 q。请注意，量子门 U、R_z、R_y由经典变量 a_{ij}、θ、θ_1、θ_2、θ_3参数化。

显然，Z–Y分解定理的证明依赖于一种逻辑机制，使人能够从

满足式（2）左侧条件的复数理论，推断出式（2）右侧关于量子操作的性质。我们希望在 Lean 或 Coq 等证明助手中以符号方式验证式（2）。

让我们通过考察量子隐形传态协议，进一步探讨符号方法如何助力量子程序的验证。

例 1.3（量子程序的符号验证）。 四种贝尔态（即 *EPR* 对）

$$|\beta_{00}\rangle = \frac{|00\rangle + |11\rangle}{\sqrt{2}}, \qquad |\beta_{01}\rangle = \frac{|01\rangle + |10\rangle}{\sqrt{2}},$$
$$|\beta_{10}\rangle = \frac{|00\rangle - |11\rangle}{\sqrt{2}}, \qquad |\beta_{11}\rangle = \frac{|01\rangle - |10\rangle}{\sqrt{2}}.$$

可以用符号表示为

$$|\beta_{xy}\rangle = \frac{|0, y\rangle + (-1)^x |1, \overline{y}\rangle}{\sqrt{2}}$$

以经典布尔变量 x、y 作为参数，其中 y 表示 y的否定。¯

爱丽丝和鲍勃分别持有量子比特 q_a、q_b，并共享贝尔态 $|\beta_{xy}\rangle$。利用它们传送量子比特 q 的任意状态 $|\psi\rangle = \alpha|0\rangle + \beta|1\rangle$，可写成以下程序：

$TEL_{xy} := CNOT[q, q_a]$; $H[q]$; $m := M[q]$; $m_a := M[q_a]$; **如果** $y \oplus m_a = 1$ **则** $X[q_b]$; **如果** $x \oplus m = 1$ **则** $Z[q_b]$; **如果** $xm_a = 1$ **则** Ph

其中 M 表示在计算基 $|0\rangle$ 上进行测量，$|1\rangle$ $\oplus$表示模2加法，*Ph* 表示全局相位 -1。

程序 TEL_{xy} 在情形（x，y）=(0，0) 下的正确性已在 [31] 的第 1.3.7 节中进行分析。要验证其在其他三种情形（x，y）=(0，1)，(1，0)，(1，1)下的正确性，可以分别进行类似的分析。不过，更优雅且高效的方法是使用符号技术，通过形式参数 x和 y对TEL_{xy} 进行推理。

1.2. 本文的目标

符号化规约与推理已广泛应用于量子信息与计算研究中。但这些应用并未

以有条理或形式化的方式开展。尤其是，这类推理仍缺乏严谨的逻辑基础。这与经典计算的情况形成鲜明对比：在经典计算中，符号方法以成熟的逻辑框架为基础。在量子情境下缺少此类基础，严重阻碍了符号方法在量子算法和程序的自动化形式验证中的有效应用。

本文旨在概述一种通用逻辑框架，以支持对量子数据和操作进行符号化规约与推理，如示例 1.1 至 1.3 所示。该框架的基本设计决策如下：

1. 我们采用一类句法实体，称为形式算子，来规约量子数据和操作。此设计背后的理念很简单：表示一个 n 维量子态的向量可以视为一个 $n\times 1$ 矩阵。事实上，这种设计的优点与量子物理中的狄拉克符号类似：只要量子态、酉变换和可观测量都用狄拉克符号表示，它们的操作往往就可以结合起来进行。尤其是，这使我们能够以统一的方式给出针对量子数据和操作进行符号推理的推理规则。随后，我们将通过为形式算子采用不同的签名 （或类型）来区分量子数据和操作。

我们的符号化规约与推理背后的核心思想是：形式算子可以包含经典表达式，因此由经典变量参数化。这正是它们被称为形式化算子的原因。这确实是对[45, 46]中引入的形式化量子态和形式量子谓词的扩展。

2. 我们假设存在一个经典一阶逻辑语言 $\mathcal{L}$。在此基础上，我们定义形式算子的语法，其中复形式算子（记作 A, B, ...）由原子算子通过一组构造器构造而成。例如，我们可以取 $|s\rangle_q$、 $|s\rangle_q\langle t|$ 等作为原子算子，其中 s, t 是 $\mathcal{L}$ 中的经典表达式，而 q 是一个量子数组变量。构造器可以包括线性组合和张量积 $\otimes$。请注意，表达式 s 和 t 不仅可以包含常量（例如复数），还可以包含

经典变量。这体现了（1）中讨论的由经典变量参数化的形式算子的思想。此外，在这种语法中使用量子数组，为标记多体量子系统中的不同子系统提供了一种便捷机制。

引入这种语法，通常能够比矩阵表示更紧凑地表示量子数据和操作，尤其是在涉及的矩阵稀疏时。这类似于一阶逻辑公式能够比布尔矩阵更紧凑地表示经典谓词这一事实。此外，这种语法也是对量子数据和操作进行符号操作、执行与推理的关键。

本质上，(1) 和 (2) 对狄拉克符号背后的思想进行了形式化表述和符号扩展；狄拉克符号广泛应用于量子物理、计算和信息领域。这里的关键思想是将经典一阶表达式嵌入狄拉克符号中，使形式算子能够由相关的经典变量参数化（见图1）。

符号算子逻辑 **SOL**

⇑ 嵌入

一阶逻辑 $\mathcal{L}$

图1：双层逻辑框架。

3. 下一步，我们将在式 (2) 中定义的形式算子公式之上构建一阶逻辑，并将其称为符号算子逻辑（**SOL**）。在这种逻辑中，形式算子被视为表达式（即项）。随后，我们将引入形式算子的各种基本性质或关系——例如，$A \otimes B = C$——并将它们用作原子命题，以构成逻辑公式，描述量子数据和操作更复杂的性质。

此外，我们定义如下形式的蕴涵

$$\Sigma \models \mathcal{A}, \tag{3}$$

其中 Σ 是逻辑 $\mathcal{L}$中的一个理论， $\mathcal{A}$ 是符号算子逻辑**SOL**中的一个逻辑公式。这使我们能够针对以**SOL**中的逻辑公式（例如 $\mathcal{A}$）表示的量子数据和操作的性质进行（条件）推理——例如利用重写技术进行等式推理——并以选定的理论（例如 Σ）为模，该理论描述了逻辑 $\mathcal{L}$中涉及的经典数据和操作。例如，我们可以进行如下推理：

$$\left(k = \frac{3}{2}m - 2\right) \wedge \left(n = \frac{5}{7}l - 1\right) \models \mathsf{CNOT}[q[2k+3], q[5l-2]];\ \mathsf{CNOT}[q[3\overline{m-1}], q[7\overline{n+5}]] \equiv \textbf{跳过。} \tag{4}$$

另一个例子是，在示例 1.3 中验证程序 TEL^{xy} 时，需要用到关于变量x 和 y 的布尔算术理论。蕴涵式 (3) 可以进一步推广为

$$\Sigma, \Gamma \models \mathcal{A} \tag{5}$$

其中 Γ 是 **SOL** 中一组逻辑公式，表示（经典）理论Σ 和（量子）理论 Γ 共同蕴涵量子实体的一项性质 $\mathcal{A}$ 。

引入蕴涵式 (3) 和 (5) 及其支持的推理，为利用经典计算中开发的验证工具来推理量子数据和操作提供了便捷机制。

1.3. 论文结构

为方便读者并统一符号记法，第2节简要回顾了我们定义符号算子逻辑 **SOL** 所依据的经典一阶逻辑 $\mathcal{L}$ 。第3节介绍**SOL**的基本要素——量子类型、数组和变量。第4节定义**SOL** 中作为项的形式算子的语法和语义。第5节介绍形式算子上的各种基本函数与谓词。第6节定义**SOL**中逻辑公式的语法和语义；特别是，在以逻辑 $\mathcal{L}$表达的经典一阶理论为条件时，**SOL** 中逻辑公式之间的蕴涵这一概念。第7节给出了关于量子数据

和**SOL** 中操作的性质进行符号推理的示例，第8节则提供了若干在**SOL** 中递归定义量子数据与操作的示例。第9节讨论**SOL**的潜在后续应用。第10节作简要结论。

2. 经典一阶语言

我们的符号算子逻辑 **SOL**建立在经典一阶逻辑$\mathcal{L}$之上。为方便读者，本节先简要介绍 $\mathcal{L}$ 的语法和语义。我们采用类型化一阶语言 $\mathcal{L}$，遵循教科书 [2] 中经典程序验证的介绍。

2.1. 类型

直观而言，类型是一个名称或符号记法，用来表示预期的值集合。在逻辑 $\mathcal{L}$中，我们假设存在一组基本类型。例如，

- **Bool** 表示集合 {**假**，**真**}；
- **Int** 表示所有整数构成的集合……，-2，-1，0，1，2，……；
- 特别地，我们需要复数的类型 **C** ，因为本节定义的语言将在后文用于描述量子态和操作。

对于任意基本类型 $T_1, ..., T_n (n \geq 1)$ 和 T，我们引入一种高阶类型 $T_1 \times ... \times T_n \to T$。直观地说，它表示从由 $T_1, ..., T_n$ 表示的集合的笛卡尔积到由 T 表示的集合的所有函数的集合。这里，$T_1, ..., T_n$ 称为参数类型，而 T 称为值类型。

2.2. 语法

我们的一阶语言 $\mathcal{L}$ 的**字母表**由以下部分组成：

- 一个变量集合 Var；
- 一个常量集合；
- 逻辑联结词 $\neg$ (否定)，$\wedge$(合取)，$\vee$(析取)，$\to$(蕴含)；
- 全称量词 $\forall$、存在量词 $\exists$。

每个变量 x 都具有一个类型 T_x。如果 T_x 是基本类型，那么 x称为简单变量；如果 T_x 是高阶类型，那么 x 称为数组变量。同样，每个常量也具有一个类型。例如，整数常量 $...,-2,-1,0,1,2,...$ 具有基本类型 **Int**，布尔常量 **true**,**false** 具有基本类型 **Bool**，每个复数都具有基本类型 **C**。此外，函数符号可以视为高阶类型 $T_1 \times ... \times T_n \to T$的常量。同样，谓词符号可以视为高阶类型 $T_1 \times ... \times T_n \to \mathbf{Bool}$的常量。

表达式（即项）由变量和常量通过应用函数符号构成。

定义 2.1. $\mathcal{L}$ 中的表达式按以下方式归纳定义：

1. 基本类型为 T 的简单变量和常量都是类型为 T 的表达式；2. 若 s_1, ..., s_n 分别是类型为 T_1, ..., T_n 的表达式，且 op 是类型为 $T_1 \times ... \times T_n \to T$ 的常量，则

$$op(s_1, ..., s_n) \tag{6}$$

是类型为 T 的表达式；

3. 若 s_1, ..., s_n 分别是类型为 T_1, ..., T_n 的表达式，且 a 是类型为 $T_1 \times ... \times T_n \to T$ 的数组变量，则

$$a[s_1, ..., s_n] \tag{7}$$

是类型为 T 的表达式，通常称为下标变量；

4. 如果 b 是一个布尔常量，且 s_0和 s_1 都是类型为 T的表达式，那么

$$\textbf{if } b \textbf{ then } s_1 \textbf{ else } s_0 \textbf{ fi} \tag{8}$$

就是一个类型为 T的表达式。

根据定义，很明显所有表达式都有一个基本类型。我们用 变量(*s*) 表示表达式 s中的变量集合。

逻辑公式由原子命题（即应用于表达式（即项）的谓词符号）通过逻辑联结词和量词构造而成。

定义 2.2。 $\mathcal{L}$中的一阶逻辑公式按以下方式归纳定义：

1. 如果 p 是一个谓词符号，即类型为 $T_1 \times \cdots \times T_n \to$**Bool**的常量，并且 $s_1, ..., s_n$ 分别是类型为 $T_1, ..., T_n$的表达式，那么$p(s_1, ..., s_n)$ 是一个逻辑公式，称为原子命题；2. 如果 φ, ψ 是逻辑公式，那么 $\neg\varphi, \varphi \wedge \psi$, $\varphi \vee \psi$ 和 $\varphi \to \psi$也都是逻辑公式；3. 如果 φ是一个逻辑公式，且 x是一个变量，那么$(\forall x)\varphi$和$(\exists x)\varphi$都是逻辑公式。

我们用$free(\varphi)$表示 φ的自由变量集合，即未被量词约束的变量。 $\mathcal{L}$ 中的逻辑公式集合记为$Wff_{\mathcal{L}}$。

代换 是一种基本操作，用特定的项或表达式替换表达式或逻辑公式中的变量。它使我们能够将一般逻辑规则应用于具体情形。

定义 2.3. 设 x 是类型为 T 的简单变量或下标变量，且t 是同一类型的表达式。那么：

1. 对于任意表达式 s，将 t代入 s 中的x 所得的表达式为 $s[t/x]$ 。更精确地说，它按以下方式归纳定义：

   (a) 如果 $s = y$ 是带有 $y \neq x$ 的简单变量，或者 $s = c$ 是常量，那么 $s[t/x] = s$；如果 $s = x$，那么 *then* $s[t/x] = t$。(b) 如果 $s =$ ***op***$(s_1, ..., s_n)$，那么 $s[t/x] =$ ***op***$(s_1[t/x], ..., s_n[t/x])$。(c) 设 $s = a[s_1, ..., s_n]$ 是一个数组 a 的项。如果 x 是简单变量或带有 $a' \neq a$ 的$x = a'[s'_1, ..., s'_m]$，那么 $s[t/x] = a[s_1[t/x], ..., s_n[t/x]]$；如果 $x = a[s'_1, ..., s'_m]$，那么

$$s[t/x] = \textbf{if } \bigwedge_{i=1}^{n} s_i[t/x] = s'_i \textbf{ then } t \textbf{ else } a[s_1[t/x], ..., s_n[t/x]].$$

2. 对于任意逻辑公式 φ，代换 $\varphi[t/x]$ 归纳定义如下：

   (a) 如果 $\varphi =$ ***p***$(s_1, ..., s_n)$ 是一个原子命题，则 $\varphi[t/x] =$ ***p***$(s_1[t/x], ..., s_n[t/x])$。***(b)*** 如果 $\varphi = \neg\psi$，则 $\varphi[t/x] = \neg\psi[t/x]$；如果 $* \in \{\wedge, \vee, \to\}$ 且 $\varphi = \varphi_1 * \varphi_2$，则 $\varphi[t/x] = \varphi_1[t/x] * \varphi_2[t/x]$。***(c)*** 设 $Q \in \{\forall, \exists\}$ 且 $\varphi =$ ***(Qy)***ψ。如果 $y = x$，则 $\varphi[t/x] = \varphi$；如果 $y \neq x$，则 $\varphi[t/x] =$ ***(Qy')***$\psi[y'/y][t/x]$，其中 $y' \notin$ ***free(Q)*** $\cup$ ***var (t)*** $\cup \{x\}$。

上述代换的概念可以推广为同时代换：将一串表达式 t 代入一串互不相同的变量 $\bar{x}$ 中。 $s[\bar{t}/\bar{x}]$

2.3. 语义

作为对我们的逻辑语言 $\mathcal{L}$ 的解释，我们引入一阶结构的概念。它定义为一对 $\mathcal{S} = (\mathcal{D},\ \mathbb{I})$，其中：

- $\mathcal{D}$ 是一个族 $\{\mathcal{D}_T\}$，对于每个基本类型 T，$\mathcal{D}_T$ 都是一个非空集合，称为类型 T 的定义域；特别地，$\mathcal{D}_{\mathbf{Int}} = \{..., -2, -1, 0, 1, 2, ...\}$、$\mathcal{D}_{\mathbf{Bool}} = \{$**真**，**假**$\}$，而 $\mathcal{D}_{\mathbf{C}}$ 是复数的集合。接下来，对于高阶类型，我们定义：

$$\mathcal{D}_{T_1 \times ... \times T_n \to T} = \mathcal{D}_{T_1} \times ... \times \mathcal{D}_{T_n} \to \mathcal{D}_T \tag{9}$$

（从$\mathcal{D}_{T_1}$、**.....**、$\mathcal{D}_{T_n}$的笛卡尔积到$\mathcal{D}_T$的所有映射的集合）；

- $\mathbb{I}$ 是一个解释；也就是说，每个基本类型或高阶类型T 的常量 c都被解释为 $\mathcal{D}_T$中的一个元素 $c^{\mathbb{I}}$。特别地：

  – 类型为 $T_1 \times ... \times T_n \to T$的函数符号 F被解释为映射 $F^{\mathbb{I}}$：$\mathcal{D}_{T_1} \times ... \times \mathcal{D}_{T_n} \to \mathcal{D}_T$；而– 类型为 $T_1 \times ... \times T_n \to \mathbf{Bool}$ 的谓词符号 P被解释为映射 $P^{\mathbb{I}}$：$\mathcal{D}_{T_1} \times ... \times \mathcal{D}_{T_n} \to \{$**真**，**假**$\}$。

给定一个具有定义域 $\mathcal{D}$ 的结构 $\mathcal{S}$。那么，$\mathcal{S}$ 中的状态是一个映射 σ，它将每个基本类型或高阶类型 T 的变量 x 映射到 $\mathcal{D}_T$ 中的一个元素 $\sigma(x)$。$\mathcal{S}$ 中所有状态的集合记作 状态 $_{\mathcal{S}}$。如果结构 $\mathcal{S}$ 已固定或可从上下文中识别，我们通常将 状态$_{\mathcal{S}}$ 简写为 状态。

设 σ 为一个状态，x 为一个变量，$\mathcal{D}_T$ 中的 d 为一个值，其中 T 是 x 的类型。那么，更新状态 $\sigma[x{:=}\, d]$ 定义如下：对于任意变量 x'，

$$\sigma[x := d](x') = \begin{cases} d & \text{if } x' = x; \\ \sigma(x) & \text{if } x' \neq x. \end{cases} \tag{10}$$

定义 2.4. 状态 $\sigma \in \Sigma$ 中表达式 s 的值 $\sigma(s)$ 归纳定义如下：

1. 如果 s 是简单变量，那么 $\sigma(s)$ 由 σ给出；如果 s是基本类型的常量，那么 $\sigma(s) = s^{\mathbb{I}}$；*2.* 如果 $s = op(s_1, ..., s_n)$，那么 $\sigma(s) = op^{\mathbb{I}}(\sigma(s_1) ..., \sigma(s_n))$；*3.* 如果 $s = a[s_1, ..., s_n]$，那么 $\sigma(s) = \sigma(a)(\sigma(s_1) ..., \sigma(s_n))$；

4. 如果 $s =$**if** b **then** s_1 **else** s_0 **fi**，那么

$$\sigma(s) = \begin{cases} \sigma(s_1) & if\ \sigma(b) = \mathbf{true}, \\ \sigma(s_0) & otherwise. \end{cases}$$

定义 2.5. 满足关系 $\sigma \models \varphi$，即状态 $\sigma \in \Sigma$满足逻辑公式 φ，归纳定义如下：

1. $\sigma \models p(s_1, ..., s_n)$ 当且仅当 $p^{\mathbb{I}}(\sigma(s_1) ..., \sigma(s_n)) =$**为真**；2. $\sigma \models \neg\varphi$ 当且仅当非 $\sigma \models \varphi$，$\sigma \models \varphi \wedge \psi$ 当且仅当 $\sigma \models \varphi$ 且 $\sigma \models \psi$，$\sigma \models \varphi \vee \psi$ 当且仅当$\sigma \models \varphi$ 或 $\sigma \models \psi$，$\sigma \models \varphi \to \psi$ 当且仅当 $\sigma \models \varphi$ 蕴含 $\sigma \models \psi$；3. 若 x 是类型为 T的变量，则：

- $\sigma \models (\forall x)\varphi$ *iff* $\sigma[x := d] \models \varphi$ *for any* $d \in \mathcal{D}_T$,
- $\sigma \models (\exists x)\varphi$ *iff* $\sigma[x := d] \models \varphi$ *for some* $d \in \mathcal{D}_T$.

以下引理阐明了代换与状态更新之间的联系。

引理 2.1 (代换**).** 对于任意变量 x、表达式 s、t 和逻辑公式 φ，以及任意状态 σ，有：

1. $\sigma(s[\vec{t}/\vec{x}]) = \sigma[\vec{x} := \sigma(\vec{t})](\vec{s})$；2. $\sigma \models \varphi[t/x]$ 当且仅当 $\sigma[x := \sigma(t)] \models \varphi$。

我们不再进一步回顾经典一阶逻辑的证明系统 $\mathcal{L}$。作为本节的总结，我们想指出，本节回顾的内容将发挥双重作用：

- 我们的符号算子逻辑 **SOL** 将建立在逻辑 $\mathcal{L}$之上；换言之，$\mathcal{L}$ 嵌入于 **SOL**中。
- 从本质上说，**SOL** 也是一种一阶逻辑（尽管其中的个体是算子）。因此，**SOL**将按照 $\mathcal{L}$的模式来发展。

3. 量子类型、数组和变量

从现在起，我们固定一阶逻辑语言 $\mathcal{L}$ 以及作为 $\mathcal{L}$ 的解释的一个一阶结构。我们用于规约和推理量子数据与操作的符号算子逻辑 **SOL**建立在它们之上。本节开始介绍 **SOL**的一些基本要素，即量子类型、数组和变量。这些概念最初在 [46, 47]中提出。

3.1. 量子类型

首先，我们介绍一种将某些（但未必是全部）经典类型量子化的方法:

- 对于基本类型 T，我们假定存在一个相应的量子类型，也用 T 表示（这是对符号记法的稍微滥用）。类型T 的量子域是由 T的经典域张成的希尔伯特空间:

$$\mathcal{H}_T = \mathrm{span}\left\{|d\rangle : d \in \mathcal{D}_T\right\}. \tag{11}$$

也就是说，经典域 $\mathcal{D}_T$ 是 $\mathcal{H}_T$的一组标准正交基，称为标准基。例如，类型**Bool** 的量子域 $\mathcal{H}_{\mathbf{Bool}}$ 是二维希尔伯特空间，即一个量子比特的状态空间；类型 **Int** 的量子域是

$$\mathcal{H}_{\mathbf{Int}} = \left\{ \sum_{i=-\infty}^{\infty} \alpha_i |i\rangle : \sum_{i=-\infty}^{\infty} |\alpha_n|^2 < \infty \right\}.$$

- 对于高阶类型 $T_1 \times ... \times T_n \to T$，我们只对其值类型$T$进行量子化，而将其参数类型 $T_1, ..., T_n$保留为经典类型。因此， $T_1 \times ... \times T_n \to T$的量子域为

$$\mathcal{H}_{T_1 \times ... \times T_n \to T} = \mathcal{H}_T^{\otimes(\mathcal{D}_{T_1} \times ... \times \mathcal{D}_{T_n})} = \bigotimes_{d_1 \in \mathcal{D}_{T_1}, ..., d_n \in \mathcal{D}_{T_n}} \mathcal{H}_T \tag{12}$$

也就是说，这是多个 $\mathcal{H}_T$ 的张量积，分别由类型 $T_1, ..., T_n$的值$d_1, ..., d_n$索引。从本质上说，式（12）是将式（9）中的 $\mathcal{D}_T$替换为 $\mathcal{H}_T$得到的（但保持$\mathcal{D}_{T_1}, ...,$ $\mathcal{D}_{T_n}$不变）。

3.2. 量子变量

对应于经典语言 $\mathcal{L}$ 中使用的变量，我们引入两类量子变量:

- 基本类型的简单量子变量，例如 T;
- 更高量子类型的数组量子变量，例如 $T_1 \times ... \times T_n \to T$。

值得注意的是，经典变量和量子变量的类型相同，但它们的解释不同；更准确地说，具有类型 T 的经典变量 x 的值是 $\mathcal{D}_T$ 中的一个元素，而具有相同类型 T 的量子变量 q 的值则是希尔伯特空间 $\mathcal{H}_T$ 中的一个向量。因此， q 的状态空间定义为 $\mathcal{H}_q = \mathcal{H}_T$。

定义 2.1(3) 中下标变量的概念可以直接推广到量子情境：

定义 3.1。 设 q 是类型为 $T_1 \times ... \times T_n \to T$ 的数组量子变量，且对每个 $1 \leq i \leq n$，令 s_i 为类型为 T_i 的经典表达式。那么

$$q[s_1, ..., s_n] \tag{13}$$

称为类型为 T 的带下标量子变量。

带下标量子变量 (13) 的定义方式与带下标经典变量 (7) 类似。直观地说，数组变量 q 表示一个由按 $(d_1、...、d_n) \in \mathcal{D}_{T_1} \times ... \times \mathcal{D}_{T_n}$ 编号的子系统组成的量子系统。因此，每当表达式 s_i 对每个 i 都求值得到值 $d_i \in \mathcal{D}_{T_i}$ 时，$q[s_1, ..., s_n]$ 就表示索引元组为 $(d_1、...、d_n)$ 的子系统。为简化表述，简单变量 q 通常被视为当 $n = 0$ 时的下标变量 $q[s_1, ..., s_n]$。

例 3.1。 设 q 是一个类型为 $\mathbf{Int} \times \mathbf{Int} \to \mathbf{Bool}$的量子比特数组。那么 $q[2x + y, 7 - 3y]$ 是一个带下标量子变量。如果在经典状态 σ中， $x = 5$ 和 $y = -1$，那么它表示量子比特 $q[9, 10]$；也就是说， $\sigma(q[2x + y, 7 - 3y]) = q[9, 10]$ （见下文定义 *3.3*）。

设 $\overline{q} = q_1, ..., q_n$，其中 q_i 是一个类型为 T_i 的简单量子变量或带下标量子变量，适用于 $1 \leq i \leq n$。那么， q 的维数定义为 $\dim q = \prod_{i=1}^{n} |T_i|$，其中 $|T_i|$ 表示（经典）类型 T_i 的大小。因此，q 的状态空间是维数为 $\dim q$的希尔伯特空间 $\mathcal{H}_q = \bigotimes_{i=1}^{n} \mathcal{H}_{q_i}$。

量子信息的不可克隆定理禁止对两个相同的量子比特执行量子操作。为了在语法中体现这一要求，我们引入：

定义 3.2. 设 $q = q_1[s_{11}, ..., s_{1n_1}]$, ..., $q_k[s_{k1}, ..., s_{kn_k}]$ 是一列简单量子变量或带下标量子变量。那么， q 的互异性定义为以下一阶逻辑公式：

$$Dist(\overline{q}) = \bigwedge_{i,j \ with \ q_i = q_j} (\exists l \leq n_i) (s_{il} \neq s_{jl}) . \tag{14}$$

本文将频繁使用这种互异性，以确保涉及多个量子比特的概念定义良好。对于给定的经典结构 $\mathcal{S}$ （作为一阶逻辑 $\mathcal{L}$ 的解释）以及状态$\sigma \in Stat_{\mathcal{S}}$，满足关系：

$$\sigma \models Dist(q_1[t_{11}, ..., s_{1n_1}], ..., q_k[s_{k1}, ..., s_{kn_k}])$$

表示在经典状态 σ 中， $q_1[s_{11}, ..., s_{1n_1}]$，**.....**， $q_k[s_{k1}, ..., s_{kn_k}]$ 表示k 个不同的量子系统。

我们将简单量子变量或$q[d_1, ...d_n]$称为<styIe id='1'>基态量子变量，其中 q 是类型为 $T_1 \times ... \times T_n$ 的数组量子变量，且对于 $1 \leq i \leq n$，$d_i \in \mathcal{D}_{T_i}$。然后我们可以引入：

定义 3.3。 设 σ 是给定一阶结构$\mathcal{S}$中的经典状态。那么：

- 对于简单量子变量 q，我们定义 $\sigma(q) = q$；
- 对于带下标量子变量，我们定义：

$$\sigma(q[t_1, ..., t_n]) = q[\sigma(t_1), ..., \sigma(t_n)] \tag{15}$$

其中 $\sigma(t_i)$ 表示表达式 t_i 在状态 σ 中的值（参见定义 *2.4*）。

此外，对于由简单量子变量或带下标量子变量组成的串 $q = q_1, ..., q_n$，我们定义：

$$\sigma(\overline{q}) = \sigma(q_1), ..., \sigma(q_n). \tag{16}$$

需要注意的是，(15) 是一个基态变量，而 (16) 是一个基态变量串。

4. 形式算子

现在我们开始构建符号算子逻辑**SOL**。首先，本节将定义形式算子的语法和语义。如前所述，在逻辑中，算子被视为个体 $\mathcal{L}$。因此，本节定义的形式算子实际上是**SOL**中的表达式（即个体项）。

4.1. 算子变量和常量

在**SOL**的字母表中，我们假设：

1. 一组算子变量，其范围由元符号 X、X_1、X_2.....表示；以及 2. 一组基本算子符号，称为算子常量，其范围由元符号 O、O_1、O_2.....表示。

每个算子变量 X 都具有如下形式的量子类型：

$$T_1 \times ... \times T_m \to T'_1 \times ... \times T'_n \tag{17}$$

其中 T_1、.....、T_m、T'_1.....、T'_n 均为基本类型。直观地说，X 的一个值是从希尔伯特空间 $\mathcal{H}_{T_1\times...\times T_m}$ 到 $\mathcal{H}_{T'_1\times...\times T'_n}$ 的线性算子，其中乘积类型的量子域定义为

$$\mathcal{H}_{T_1\times...\times T_m} = \bigotimes_{i=1}^{m} \mathcal{H}_{T_i}. \tag{18}$$

应注意，类型 (18) 中允许出现 $m = 0$ 或 $n = 0$。我们用 [] 表示空类型，即 $T_1 \times ... \times T_m$ 且 $m = 0$。于是：

- 一个类型为 $[] \to T'_1 \times ...T'_n$ 的算子变量 X 表示希尔伯特空间 $\mathcal{H}_{T'_1\times...\times T'_n}$ 中的刁矢量 $\langle\varphi|$;
- 类型为 $T_1 \times ...T_m \to []$ 的算子变量 X 表示希尔伯特空间 $\mathcal{H}_{T_1\times...\times T_m}$ 中的右矢 $|\varphi\rangle$;
- 类型为 $[] \to []$ 的算子变量 X 表示一个复数 c。

每当 $T_1 \times ... \times T_m = T'_1 \times ... \times T'_n$ 时，我们就简单地说，类型为（17）的算子变量 X 具有类型 $T_1 \times ... \times T_m$。

类似地，每个算子常量 O 都配有一个类似于（17）的量子类型。此外，它还配有一个用于其经典参数的类型，记作 $T''_1 \times ... \times T''_k$。这意味着 O 可以携带 k 经典（形式）参数 x_1，.....，x_k，使得对于每个 $1 \le i \le k$，x_i 都是一个类型为 T''_i 的经典变量

例4.1。1. 作为算子常量，哈达玛门 H 和泡利门 I, X, Y, Z 的量子类型为 **Bool**，受控非门 *CNOT* 的量子类型为 **Bool** $\times$**Bool**。由于它们不携带任何参数，因此其参数类型为空 []。

2. 例*1.2*中的旋转算子 R_x, R_y 和 R_z 的量子类型为 **Bool**，参数类型为**R**（实数），对应参数 θ。3. 例*1.1*中的量子态（*1*）可视为一个算子常量，其量子类型为 **Int** ×**Int**，参数类型为**R** × **Int** × **Int**，因为它有三个参数 θ, k 和 m。

算子变量、算子常量，或更一般地说，形式算子，将用于描述从一列量子变量（表示一个复合量子系统）到另一列量子变量的变换。为明确指定算子的定义域和陪域，我们引入：

定义4.1(签名**)**。1. 签名定义为如下形式的公式：

$$\overline{q} \to \overline{q'} \tag{19}$$

其中，^{-}q 和 $\overline{q'}$ 是由简单量子变量或带下标量子变量构成的字符串（也可以是空字符串 ϵ）。
2. 如果 q 和 $\overline{q'}$ 是由基态量子变量构成的字符串，那么（*19*）称为基签名。

令 $\overline{q} = q_1, ..., q_m$ 和 $\overline{q'} = q'_1, ..., q'_n$。对于每个 i，令 q_i 具有T_i的类型，且 q'_i 具有类型 T'_i。则签名（19）的类型定义为$T_1 \times ... \times T_m \to T'_1 ... \times T'_n$。特别地，令 ϵ 表示量子变量的空字符串，则签名$q \to \epsilon$ 的类型为 $T_1 \times ... \times T_m \to []$, 签名$\epsilon \longrightarrow q'$的类型为 $[] \to T'_1 \times ... \times T'_n$，签名 $\epsilon \to \epsilon$的类型为 $[] \to []$。我们很快会看到，当算子符号 O 应用于签名 $q \to q'$时，它们的类型必须匹配。– —

4.2. 形式算子的语法

现在我们可以介绍形式算子的语法了。我们的形式算子语言的字母表包括第2节中经典一阶逻辑语言的字母表 $\mathcal{L}$，以及：

- 第3节中给出的量子变量；
- 基本算子符号 O，O_1，O_2，.....，称为算子常量符号，每个符号都被赋予一种量子类型和一种参数类型（如第4.1小节所述）；

- 算子变量 X， X_1， X_2，.....，每个变量都被赋予一种量子类型；以及

- 操作符号 $\cdot$(数乘)、 † (伴随)、 $+$ (加法)、 $\cdot$(乘法)和 $\otimes$ (张量积)。

指定量子数据和操作的一种常见方式是将它们表示为矩阵或算子。然而，当所考虑的希尔伯特空间维数过大时，显式指定它们变得不切实际。因此，我们需要引入：

定义 4.2（语法）。形式算子（由 A、A_1、A_2.....表示）由以下语法定义：

$$\begin{aligned} A ::= \ & c \mid |s\rangle_q \mid \langle s|_q \mid X[\overline{q} \to \overline{q'}] \mid O(t_1, ..., t_k)[\overline{q} \to \overline{q'}] \\ & \mid cA \mid A^\dagger \mid A_1 + A_2 \mid A_1 A_2 \mid A_1 \otimes A_2 \end{aligned} \tag{20}$$

更准确地说，形式算子 A 及其签名 $Sign(A)$ 定义如下：

1. 类型为 C（复数）的经典表达式 c 是形式算子，且 $Sign(c) = [] \to []$；*2.* 如果 s 是与简单量子变量或带下标量子变量 q 类型相同的经典表达式，那么 $|s\rangle_q$ 和 $\langle s|_q$ 都是形式算子，且 $Sign(|s\rangle_q) = q \to []$、$Sign(\langle s|_q) = [] \to q$；*3.* 如果 X 是与签名 $q \to q'$ 类型相同的算子变量，那么 $X[q \to q']$ 是形式算子，且 $Sign(X[q \to q']) = q \to q'$。特别地，当 $q = q'$ 时，$X[q \to q']$ 可缩写为 $X[q]$；*4.* 如果 t 是一个经典表达式字符串，其类型分别为 T_1、...、T_k，且 O 是与签名 $q \to q'$ 具有相同量子类型并被赋予参数类型 $T_1 \times ... \times T_k$ 的算子常量，那么 $O(t)[q \to q']$ 是形式算子，且 $Sign(O(t)[q \to q']) = q \to q'$。特别地，当 $q = q'$ 时，$O(t)[q \to q']$ 可缩写为 $O(t)[q]$；*5.* 如果 c 是类型为 C 的经典表达式，A 是形式算子，那么 cA （作为 $c \cdot A$ 的缩写）是形式算子，且 $Sign(cA) = Sign(A)$；*6.* 如果 A 是满足 $Sign(A) = q \to q'$ 的形式算子，那么 $A^\dagger$ 也是形式算子，且 $Sign(A^\dagger) = q' \to q$；*7.* 如果 A_1、A_2 是具有相同签名的形式算子，那么 $A_1 + A_2$ 是形式算子，且 $Sign(A_1 + A_2) = Sign(A_1)$；

8. 如果 A_1、A_2 是满足 $\mathit{Sign}(A_1) = q \to q'$ 且 $\mathit{Sign}(A_2) = q' \to q''$ 的形式算子，那么 $A_1 A_2$（作为 $A_1 \cdot A_2$ 的缩写）是形式算子，且 $\mathit{Sign}(A_1 A_2) = q \to q''$；9. 如果 A_1、A_2 是形式算子，且对于 $i = 1, 2$，满足 $\mathit{Sign}(A_i) = q_i \to q_i'$，那么 $A_1 \otimes A_2$ 是形式算子，且 $\mathit{Sign}(A_1 \otimes A_2) = q_1$、$q_2 \to q_1'$、$q_2'$。

值得注意的是，基本形式算子的签名 $q \to q'$ (t) $[q \to q']$ 和 $X[q \to q']$ 用于表示算子常量 O定义域与陪域中的量子变量。在上述定义中，仅允许 $\cdot$（数乘）、†（伴随）、$+$（加法）、$\cdot$（乘法）和$\otimes$（张量积）作为（算子）操作符号。在未来的应用中，可以引入更多操作符号，例如 e^A、$\log A$、$\sqrt{A}$ 以及对某些量子变量进行偏迹。为使表述简洁，我们常常引入形式算子的某些派生操作（作为语法糖）。例如，如果 I 是一个表示恒等算子的基本算子符号，那么对于任意算子 A，我们写作 $\neg A := I - A = I + (-1) A$.

显然，形式算子 A 并不总是有意义的。令签名$(A) = q \to q'$。那么，仅当 q 中的量子变量互不相同时，A才有意义；q'中的量子变量也必须互不相同。然而，如第3.2小节所述，这无法通过语法确定。此问题将通过在第4.4小节引入签名规则来解决。

上文定义的形式算子语法使我们能够通过诸如 $|s\rangle_q, \langle s|_q, O[q \to q']$ 和 $X[q \to q']$ 这样的原子形式算子，以及诸如 $^\dagger, +, \cdot, \otimes$ 等操作，构造出越来越复杂的形式算子。下面举例说明，这如何显著简化量子数据和操作的规约。

例4.2. 对于类型为 **Int** →**Bool** 的任意经典比特数组 J，我们用片段 $J[k : l]$ 表示 J 在区间 $[k : l] = \{$整数 $i | k \leq i \leq l\}$ 上的限制。设 q 是一个具有相同类型 **Int** → **Bool** 的量子比特数组。我们考虑片段 $q[k : l]$ ——一个由 $l - k + 1$ 个量子比特构成的复合系统。于是：

1. $q[k : l]$ 的每个基态都可以写成一个形式算子

$$|J[k : l]\rangle_{q[k:l]} = |J[k]\rangle_{q[k]} \otimes ... \otimes |J[l]\rangle_{q[l]}$$

对于某个经典比特数组 J。

2. 特别是，投影到由所有基态 $|J[k:\ \ l]\rangle_{q[k:l]}$ 张成的子空间上的投影算子，其中 $J[i] \neq 1$，对于某个 $k \leq i \leq l$，可以简洁地写作

$$\neg|J_0[k:l]\rangle_{q[k:l]}\langle J_0[k,l]|$$

其中 J_0 是满足以下条件的比特数组：对所有 $k \leq i \leq l.,\ \ J_0[i] = 0$

事实上，在量子信息与计算文献中，由较简单的矩阵和算子构造复杂矩阵和算子是一种常见做法。定义 4.2 只是用形式逻辑的语言对这一思想进行了形式化。

4.3. 代换

代换是符号推理中的一项关键技术。在定义 2.3 中，已针对经典逻辑 $\mathcal{L}$ 定义了代换。为了在关于量子数据和操作的推理中运用这项技术，我们引入：

定义 4.3 (经典变量代换**)。** 设 x 和 t 与定义 *2.3* 中相同。则对于任意形式算子 A，将 t 代入 A 中的 x，可得到形式算子 $A[t/x]$。更准确地说，它按如下方式归纳定义：

1. 若 $A = c$，则 $A[t/x] = c$；*2.* 若 $A = |s\rangle_q$，则 $A[t/x] = |s[t/x]\rangle_{q[t/x]}$，其中 $s[t/x]$ 由定义 *2.3* 给出，且当 $q = q'[s_1, ..., s_n]$ 时 $q[t/x] = q'[s_1[t/x], ..., s_n[t/x]]$（特别地，若 q 是简单量子变量，则 $q[t/x] = q$）；*3.* 若 $A = \langle s|_q$，则 $A[t/x] = \langle s[t/x]|_{q[t/x]}$；*4.* 若 $A = X[q \to q']$，则 $A[t/x] = X[q[t/x] \to q'[t/x]]$，其中若 $q = q_1$，则 $q[t/x] = q_1[t/x]$、...、$q_m[t/x]$，...，q_m（特别地，$\epsilon[t/x] = \epsilon$）；*5.* 若 $A = O(s)[q \to q']$，则 $A[t/x] = O(s[t/x])[q[t/x] \to q'[t/x]]$，其中若 $s = s_1$，则 $s[t/x] = s_1[t/x]$、...、$s_k[t/x]$，...、s_k（特别地，$\epsilon[t/x] = \epsilon$）；*6.* 若 $A = cA'$，则 $A[t/x] = cA'[t/x]$；*7.* 若 $A = A'^{\dagger}$，则 $A[t/x] = (A'[t/x])^{\dagger}$；*8.* 若 $A = A_1 + A_2$，则 $A[t/x] = A_1[t/x] + A_2[t/x]$；*9.* 若 $A = A_1A_2$，则 $A[t/x] = A_1[t/x] \cdot A_2[t/x]$；*10.* 若 $A = A_1 \otimes A_2$，则 $A[t/x] = A_1[t/x] \otimes A_2[t/x]$。

本质上，上述定义将底层逻辑中的代换概念提升到我们的符号算子逻辑 **SOL** 的表达式中。这种提升还将进一步扩展到 **SOL** 中的逻辑公式。我们还可以引入：

定义 4.4（算子变量替换）。设 X 是一个算子变量，A、B 是满足 $Sign(B) = q \to q'$ 的形式算子。那么，通过在 A 中用 B 替换 X，可得到形式算子 $A[B/X]$。形式化地，其归纳定义如下：

1. 若 $A = c, |s\rangle_\varphi, \langle s|_\varphi, O(t)[\overline{r} \to \overline{r'}]$，或 $Y[r \to \overline{r'}]$ 与 $Y \neq X$，则 $A[B/X[q \to \overline{q'}]] = A$；2. 若 $A = X[r \to r']$，则 $A[B/X] = B[r, r'/q, q']$（将 r, r' 代换为 q, q'，在 B 中）；3. $(cA)[B/X] = cA[B/X]$；4. $(A^\dagger)[B/X] = (A[B/X])^\dagger$；5. $(A_1 + A_2)[B/X]] = A_1[B/X] + A_2[B/X]$；6. $(A_1A_2)[B/X] = A_1[B/X] \cdot A_2[B/X]$；7. $(A_1 \otimes A_2)[B/X] = A_1[B/X] \otimes A_2[B/X]$。

上述两个定义可以直接推广到同时替换 $A[t/x]$ 和 $A[B/X]$.

4.4. 赋予签名

为了定义形式算子的语义，我们固定一个第2节中定义的一阶结构$\mathcal{S}$。令 $\sigma \in Stat_{\mathcal{S}}$ 为给定结构中的一个经典状态。正如第4.2小节所指出的，并非 σ 中的每个形式算子 A 的语义都有定义。为确保其语义定义良好，我们先定义 A在σ中的签名。对于每个类型为 T的简单量子变量或带下标量子变量 q，$\sigma(q)$是一个基态量子变量，它表示一个状态空间为 $\mathcal{H}_T$的量子系统。对于签名 $q \to q'$，我们将 $\sigma(q \to q') = \sigma(q) \to \sigma(q')$ 定义为基签名。此外，我们引入：

定义 4.5. 1. 我们称形式算子 A 在经典状态 σ 中是良签名的，如果可以推导出如下形式的基项签名判断：

$$\sigma \models A : \overline{q} \to \overline{q'} \tag{21}$$

可根据表*1*中的签名规则推导得出。

2. 设 Σ 是一组经典一阶逻辑公式，A 是形式算子，而 q、q' 是由简单量子变量或带下标量子变量组成的字符串。那么根据

$$\Sigma \models A : \overline{q} \to \overline{q'}$$

我们的意思是，对于任意经典状态 σ，只要 $\sigma \models \Sigma$，那么 $\sigma \models A$：$\sigma(\overline{q} \to \overline{q'})$。

(Sign-Cons) $\sigma \models c : \epsilon \to \epsilon$ (Sign-Stat) $\sigma \models |s\rangle_q : \sigma(q) \to \epsilon \quad \sigma \models \langle s|_q : \epsilon \to \sigma(q)$

(Sign-OpV) $\dfrac{\sigma \models Dist(\overline{q}) \wedge Dist(\overline{q'})}{\sigma \models X[\overline{q} \to \overline{q'}] : \sigma\left(\overline{q} \to \overline{q'}\right)}$ (Sign-OpC) $\dfrac{\sigma \models Dist(\overline{q}) \wedge Dist(\overline{q'})}{\sigma \models O(\overline{t})[\overline{q} \to \overline{q'}] : \sigma\left(\overline{q} \to \overline{q'}\right)}$

(Sign-Scal) $\dfrac{\sigma \models A : \overline{q} \to \overline{q'}}{\sigma \models cA : \overline{q} \to \overline{q'}}$ (Sign-Diag) $\dfrac{\sigma \models A : \overline{q} \to \overline{q'}}{\sigma \models A^\dagger : \overline{q'} \to \overline{q}}$

(Sign-Add) $\dfrac{\sigma \models A_i : \overline{q} \to \overline{q'} \ (i = 1, 2)}{\sigma \models A_1 + A_2 : \overline{q} \to \overline{q'}}$ (Sign-Mult) $\dfrac{\sigma \models A_1 : \overline{q'} \to \overline{q''} \quad \sigma \models A_2 : \overline{q} \to \overline{q'}}{\sigma \models A_1 A_2 : \overline{q} \to \overline{q''}}$

(Sign-Tensor) $\dfrac{\sigma \models A_i : \overline{q_i} \to \overline{q_i'} \ (i = 1, 2) \quad \overline{q_1} \cap \overline{q_2} = \overline{q_1'} \cap \overline{q_2'} = \emptyset}{\sigma \models A_1 \otimes A_2 : \overline{q_1}; \overline{q_2} \to \overline{q_1'}; \overline{q_2}}$

表1：形式算子的签名规则。

下面的引理给出了形式算子的签名与其代换之间的联系。

引理4.1 (替换的签名**)**。 对于任意基签名 $q \to q'$，我们有：

$$\sigma \models A[\overline{t}/\overline{x}] : \overline{q} \to \overline{q'} \ \textit{iff}\ \sigma[\overline{x} := \sigma(\overline{t})] \models A : \overline{q} \to \overline{q'}. \tag{22}$$

证明。 我们可以通过对 A 的结构进行归纳来证明此引理。 □

4.5. 语义

现在我们可以给出形式算子的语义定义了。本质上，形式算子的语义是通过将经典表达式的语义嵌入其中而得到的。给定逻辑 $\mathcal{L}$ 的一阶结构 $\mathcal{S}$。我们首先引入：

定义4.6。 给定经典结构 $\mathcal{S}$ 上的量子结构是一个有序对 $\mathcal{Q} = (\mathcal{H}, \mathbb{I})$，其中：

1. $\mathcal{H}$ 是希尔伯特空间族 $\{\mathcal{H}_T\}$，对于每个基本类型 T， T 的量子域为

$$\mathcal{H}_T = span\{|d\rangle \mid d \in \mathcal{D}_T\}$$

其中， $\mathcal{D}_T$ 是 $\mathcal{S}$中 T 的经典域，而高阶类型 $T_1 \times ... \times T_n' \to T$的量子域由公式 *(12)* 定义；

2. $\mathbb{I}$ 将每个量子类型 $T_1 \times ... \times T_m \to T'_1 \times ... \times T'_n$、参数类型为 $T''_1 \times ... \times T''_k$ 的算子常量 O 解释为一族 $O^{\mathbb{I}} = \{O^{\mathbb{I}}(d)\}$ 线性算子，这些算子从希尔伯特空间 $\bigotimes_{i=1}^{m} \mathcal{H}_{T_i}$ 映射到 $\bigotimes_{i=1}^{n} \mathcal{H}_{T'_i}$，其中指标为 $d \in \prod_{i=1}^{k} \mathcal{D}_{T''_i}$。（为简便起见，我们滥用符号记法 $\mathbb{I}$，用它同时表示经典结构和量子结构中的解释。）

量子结构 $\mathcal{Q}$ 中算子变量的赋值 η 被定义为：将每个类型为 $T_1 \times ... \times T_m \to T'_1 \times ... \times T'_n$ 的算子变量 X 映射到从希尔伯特空间 $\bigotimes_{i=1}^{m} \mathcal{H}_{T_i}$ 到 $\bigotimes_{i=1}^{n} \mathcal{H}_{T'_i}$ 的线性算子 $\eta(X)$。

定义 4.7。设 $\zeta = (\sigma, \eta)$，其中 σ 是 $\mathcal{S}$ 中的一个经典状态，η 是 $\mathcal{Q}$ 中算子变量的一个赋值。那么，σ 中形式算子 A 的语义 $\zeta(A)$ 归纳定义如下：

1. 如果 $A = c$，则 $\zeta(A) = c^{\mathbb{I}}$；*2.* 如果 $A = |s\rangle_q$，则 $\zeta(A) = |\sigma(s)\rangle_{\sigma(q)}$（量子系统的标准基态，以 $\sigma(q)$ 表示），其中 $\sigma(s)$ 和 $\sigma(q)$ 由定义 *2.4* 和 *3.3* 给出；*3.* 如果 $A = \langle s|_q$，则 $\zeta(A) = \langle \sigma(s)|_{\sigma(q)}$，其中 $\sigma(s)$ 和 $\sigma(q)$ 如第（2）条所述；*4.* 如果 $A = X[q \to q']$，则 $\zeta(A) = \eta(X) : \mathcal{H}_{\sigma(q)} \to \mathcal{H}_{\sigma(q')}$；*5.* 如果 $A = O(t)[q \to q']$，则 $\zeta(A) = O^{\mathbb{I}}(\sigma(t)) : \mathcal{H}_{\sigma(q)} \to \mathcal{H}_{\sigma(q')}$；*6.* 如果 $A = cA'$，则 $\zeta(A) = c^{\mathbb{I}} \cdot \sigma(A')$；*7.* 如果 $A = A'^{\dagger}$，则 $\zeta(A) = \zeta(A')^{\dagger}$；*8.* 如果 $A = A_1 + A_2$，则 $\zeta(A) = \zeta(A_1) + \zeta(A_2)$；*9.* 如果 $A = A_1 A_2$，则 $\zeta(A) = \zeta(A_1) \cdot \zeta(A_2)$；*10.* 如果 $A = A_1 \otimes A_2$，则 $\zeta(A) = \zeta(A_1) \otimes \zeta(A_2)$。

应注意，条款（6）至（10）中 $\zeta(A)$ 定义方程右侧的操作，是相应希尔伯特空间上线性算子的操作。

下面的引理表明，经典状态 σ 中形式算子 A 的签名确实描述了语义 $\zeta(A)$ 的定义域和陪域。

引理 4.2。设 $\zeta = (\sigma, \eta)$。若 $\sigma \models A : \overline{q} \to \overline{q'}$，则 $\zeta(A)$ 作为从 $\mathcal{H}_q$ 到 $\mathcal{H}_{q'}$ 的线性算子是定义良好的。

证明。 对 A的结构进行归纳即可。 □

因此，形式算子 A 的语义可以理解为一个偏函数，其输入为经典状态与算子变量赋值构成的对 $\zeta = (\sigma, \eta)$，输出为 $\mathcal{H}_q \to \mathcal{H}_{q'}$ 中满足 $\sigma \models A : q \to q'$的算子

$$[\![A]\!] : \zeta = (\sigma, \eta) \mapsto \zeta(A) \text{ whenever } \zeta(A) \text{ is well-defined.}$$

因此，从语义上看， A 是一个由其中的经典变量参数化的算子。这种理解与 [19]中定义的cq态和cq谓词，以及 [48]中的酉表达式的概念相一致。我们的方法与 [19, 48] 中的方法有一个根本区别：我们引入了参数化算子的语法。值得注意的是，语义映射 $[\![A]\!]$ 的陪域$\mathcal{H}_q \to \mathcal{H}_{q'}$ 取决于输入 $\zeta =$ 中的 σ （σ， η）；换言之，应将 A视为依赖类型化的（即依赖类型理论意义上的依赖类型化）。

显然，如果 A 不包含任何算子变量，并且 $\zeta =$ （σ， η），那么ζ（A）仅取决于 σ，而不取决于 η。此时， ζ（A）通常简写为σ（A）。

示例4.3。 考虑形式算子：

$$s = \cos\frac{\theta}{2}|x - \frac{1}{2}\rangle_{q[3n-2]} + \sin\frac{\theta}{2}|x + \frac{1}{2}\rangle_{q[3n-2]}.$$

它表示一个由经典变量 θ、 x 和 n参数化的量子态。设经典状态 σ 由 $\sigma(\theta) = \frac{\pi}{2}$、$\sigma(x) = \frac{1}{2}$、 $\sigma(n) = 3$给定。则 $\sigma(s)$ 是量子比特 $q[7]$的状态 $|+\rangle = \frac{1}{\sqrt{2}}(|0\rangle + |1\rangle)$ 。

经典表达式的代换引理（引理2.1(1)）可以推广到形式算子：

引理4.3 (代换**)**。对于任意经典变量 x 和表达式 t，对于任意 $\zeta = (\sigma, \eta)$，以及任意形式算子 A，我们有：

$$\zeta(A[t/x]) = (\sigma[x := \sigma(t)], \eta)(A);$$
$$\zeta(A[B/X]) = (\sigma, \eta[X := \zeta(B)])(A)$$

其中 $\eta[X := \zeta(B)]$ 表示由以下定义的算子变量的更新后赋值

$$\eta[X := \zeta(B)](Y) = \begin{cases} \eta(Y) & \text{if } Y \neq X, \\ \zeta(B) & \text{if } Y = X. \end{cases}$$

*Pr*哎。 对 A 的结构进行归纳，使用引理 2.1(1) 和 4.1。 □

5. 形式算子上的函数与谓词

我们回顾一下第2节中的内容：在经典一阶逻辑中 $\mathcal{L}$，函数被定义为从个体（即定义域中的元素）到其自身的映射，而谓词则是从个体到真值的映射。在符号算子逻辑 **SOL**这一层，算子被视为个体，而形式算子被视为表达式。因此，在本节中，我们将介绍一些关于形式算子的基本函数与谓词，**SOL**的逻辑公式将基于这些函数与谓词构造。

5.1. 形式算子上的函数

上一节定义的形式算子本质上是矩阵和算子的语法表示（由经典变量参数化）。因此，量子计算与信息中使用的各种矩阵和算子函数 [31]可以自然地推广到形式算子上。本节仅列举其中几个示例，以说明如何进行此类推广。未来的应用中还可以引入更多形式算子上的函数。

例 5.1. 我们可以为形式算子定义弗罗贝尼乌斯范数函数 $\|A\|$ A。给定一个一阶结构 $\mathcal{S}$，作为经典逻辑 $\mathcal{L}$ 的解释，以及一个状态 σ，其中 $\mathcal{S}$，再给定一个定义在$\mathcal{S}$ 之上的量子结构 $\mathcal{Q}$，以及对 $\mathcal{Q}$中算子变量的一个赋值 η，令 $\zeta = (\sigma, \eta)$。那么， ζ 中 $\|A\|$ 的语义 $\zeta(\|A\|)$ 定义如下：

$$\zeta(\|A\|) = \|\zeta(A)\| = \sqrt{\sum_{i,j} |a_{ij}|^2} \ \textit{if}\ \zeta(A)\ \textit{is matrix}\ (a_{ij}).$$

示例 5.2。 形式算子 A 的迹 $tr(A)$ 可按照示例 *5.1* 中相同的方式定义。对于给定的一对 $\zeta = (\sigma, \eta)$ 经典状态 σ 和算子变量的赋值 η，

$$\zeta(tr(A)) = \begin{cases} tr(\zeta(A)) & \textit{if}\ \sigma \models A : \overline{q} \to \overline{q}\ \textit{for some}\ \overline{q}, \\ \textit{undefined} & \textit{otherwise}. \end{cases}$$

算子的各种性质都可以推广到形式算子，例如迹的线性性：$tr(\lambda A + \mu B) = \lambda tr(A) + \mu tr(B)$。

5.2. 形式算子上的谓词

符号算子逻辑**SOL** 是一种一阶逻辑，其中算子是个体（即对象），而第4节定义的形式算子则是**SOL**中的表达式（即项）。在上一小节中，我们展示了如何定义形式算子上的函数。在本小节中，我们将通过一些简单示例进一步说明，如何将形式算子上的谓词定义为**SOL**中的原子命题。

示例5.3 (量子态、酉变换和可观测量**)**。我们为纯量子态、混合量子态、酉变换和可观测量分别引入形式算子上的一元谓词 $\mathcal{S}_p$、 $\mathcal{S}_m$、 $\mathcal{U}$ 和 $\mathcal{O}$ 。其语义定义如下。给定一个经典结构 $\mathcal{S}$ 以及其上的量子结构 $\mathcal{Q}$ 。设 σ 是 $\mathcal{S}$ 中的一个经典状态， η 是 $\mathcal{Q}$ 中算子变量的一个赋值，并且 $\zeta = (\sigma, \eta)$，设 A 为一个形式算子， q 为由简单量子变量或带下标量子变量组成的序列。

1. 判断 $\zeta \models \mathcal{S}_p(A) : q$ 表示，在上下文 ζ 中， A 表示量子变量 q 的一个纯态；也就是说， $\sigma \models \mathit{Dist}\,(q), \sigma \models A : \sigma(q) \to \epsilon$ ，且 $\zeta(A)$ 是希尔伯特空间 $\mathcal{H}_{\sigma(q)}$ 中的单位向量。*2.* 判断 $\zeta \models \mathcal{S}_m(A) : q$ 表示，在上下文 ζ 中， A 表示量子变量 q 的一个混合态；也就是说， $\sigma \models \mathit{Dist}\,(q), \sigma \models A : \sigma(q) \to \sigma(q)$，且 $\zeta(A)$ 是希尔伯特空间 $\mathcal{H}_{\sigma(q)}$ 上的密度算子（即迹为 *1* 的正算子）。*3.* 判断 $\zeta \models \mathcal{U}(A) : q$ 表示，在上下文 ζ 中， A 表示量子变量 q 上的酉变换；也就是说， $\sigma \models \mathit{Dist}(q), \sigma \models A : \sigma(q) \to \sigma(q)$，且 $\zeta(A)$ 是希尔伯特空间 $\mathcal{H}_{\sigma(q)}$ 上的酉算子。*4.* 判断 $\zeta \models \mathcal{O}(A) : q$ 表示，在上下文 ζ 中， A 表示量子变量 q 上的可观测量；也就是说， $\sigma \models \mathit{Dist}\,(q), \sigma \models A : \sigma(q) \to \sigma(q)$，且 $\zeta(A)$ 是希尔伯特空间 $\mathcal{H}_{\sigma(q)}$ 上的厄米算子。

上例中定义的形式算子上的谓词，可以从另一个角度来理解。正如引言中所述，该框架经过刻意设计，将所有形式算子视为逻辑系统中的同一类句法实体，用于描述量子数据和操作。然而，前面的例子对这一框架作了细化，明确区分了纯态与混合量子态、酉变换和可观测量——这种做法在实际应用中往往是必需的。

例 5.4。我们可以在形式算子上引入二元谓词 $=$ *(*等式*)* 和 $\sqsubseteq$*(*勒夫纳序*)*。对于任意两个形式算子 A 和B，以及任意一对 $\zeta = (\sigma, \eta)$ 经典状态 σ 和算子变量的赋值 η：

1. 判断 $\zeta \models A = B : q \to q'$ 当且仅当 $\sigma^{\frown} \models$ *Dist* $(q) \wedge$ *Dist* $(q')^{\frown}$，$\sigma \models A : \overline{\sigma}(q) \to \sigma(q')^{\frown}$，$\sigma \models \overline{B} : \sigma(q) \to \overline{\sigma}(q')$ 且 $\overline{\zeta}(A) = \zeta(B)$ 时为真。

2. 判断 $\zeta \models A \sqsubseteq B : q$ 当且仅当 $\overline{\sigma} \models$ *Dist* (q)，$\sigma \models \bar{A} : \sigma(q) \to \sigma^{\frown}(q)$，$\sigma \models \overline{B} : \sigma(q) \to \sigma^{\frown}(q)$ 且 $\overline{\zeta}(B) - \zeta(A)$ 是一个正算子时为真。

上述两个示例中引入的算子上的谓词符号属于最基本的一类。针对不同的应用，还可以引入更多谓词符号。

6. 符号算子逻辑

在第4节中，符号算子逻辑 **SOL** 将形式算子定义为表达式（即个体项）。在此基础上，我们现在可以定义逻辑系统 **SOL** 的语法和语义，以及其中蕴涵的概念。

6.1. 语法

借助第5节介绍的基本函数与谓词，我们可以在**SOL**中定义原子命题。然后，利用逻辑联结词和量词，可以由此构造出更复杂的**SOL** 逻辑公式。

定义6.1（语法）。 算子上的逻辑公式（即**SOL**中的合式公式）由以下语法定义：

$$\begin{aligned}
\mathcal{A} := {} & \|A\| \propto \lambda \mid tr(A) \propto \lambda \mid \ldots && (23)\\
& \mid \mathcal{S}_p(A) : \overline{q} \mid \mathcal{S}_m(A) : \overline{q} \mid \mathcal{U}(A) : \overline{q} \mid \mathcal{O}(A) : \overline{q} \mid \cdots && (24)\\
& \mid A_1 = A_2 \mid A_1 \sqsubseteq A_2 \mid \cdots && (25)\\
& \mid \neg\mathcal{A} \mid \mathcal{A}_1 \wedge \mathcal{A}_2 && (26)\\
& \mid (\forall x)\mathcal{A} \mid (\forall X)\mathcal{A} && (27)
\end{aligned}$$

其中，λ 是常量，$\propto \in \{=, <, >\}$、x 是经典变量，X 是算子变量。

一些基本原子命题见（23）至（25）。未来的应用中可能会引入其他原子命题。命题联结词 $\neg$ （否定）和 $\wedge$ （合取）见（26）。经典变量上的全称量词（$\forall x$）和算子变量上的全称量词（$\forall X$）见（27）。其他逻辑联结词——$\vee$ （析取）、$\rightarrow$ （蕴含）、 $\leftrightarrow$ （双蕴含）——以及存在量词 $\exists$，均可用 $\neg$、$\vee$和 $\forall$ 以熟知的方式定义。我们用 $Wff_{\mathbf{SOL}}$表示算子上所有逻辑公式的集合。

下面给出几个简单示例，展示如何方便地使用上述符号算子逻辑 **SOL** 语法来描述量子数据和操作的各种性质。

示例 6.1 (效应 $=$ 量子谓词**)**。 量子谓词在 *[15]* 中被引入，用作量子霍尔三元组（即量子程序正确性公式）中的前置条件和后置条件。它们被定义为介于零算子0和恒等算子 I之间的厄米算子（即可观测量）。事实上，在量子基础文献中，它们被称为效应。形式上，可以用**SOL**语言将其刻画如下：

$$\mathcal{P}(A) : \overline{q} ::= \mathcal{O}(A) : \overline{q} \wedge 0 \sqsubseteq A \sqsubseteq I.$$

这里， $\mathcal{P}(A)$ 表示 A 是关于量子变量 q的量子谓词。

示例 6.2 (不可克隆定理**)**。 不可克隆定理指出，不存在能够为任意未知量子态创建独立且完全相同副本的复制机器（例如参见 *[31], 第 1.3.5 节*）。设 q_1和q_2 是两个类型相同的量子变量。那么，不可克隆定理可以用**SOL** 形式化表述如下：

$$\neg(\exists U)\left[\mathcal{U}(U) : q_1, q_2 \wedge (\forall|\psi\rangle)\left(\mathcal{S}_p(|\psi\rangle) : q_1 \rightarrow U(|\psi\rangle \otimes |0\rangle) = |\psi\rangle \otimes |\psi\rangle\right)\right]. \tag{28}$$

通过采用熟悉的α-转换技术（见定义 2.3 的第（2）条款），第4.3小节中为形式算子 A 定义的替换 $A[t/x]$ 和 $A[B/X]$可以轻易扩展为针对逻辑公式 $\mathcal{A} \in Wff_{\mathbf{SOL}}$ 的替换$\mathcal{A}[t/x]$和 $\mathcal{A}[B/X]$。

6.2. 语义

为了定义**SOL**的语义，我们假设给定一个经典一阶结构 $\mathcal{S}$ 和一个基于它的量子结构 $\mathcal{Q}$。

定义 6.2 （语义）。设 $\zeta = (\sigma, \eta)$，其中 σ 是 $\mathcal{S}$中的一个经典状态，η 是 $\mathcal{Q}$中算子变量的一个赋值，并设 $A \in \mathit{Wff}^{\mathrm{SOL}}$ 是关于算子的逻辑公式。那么，满足关系 $\zeta \models \mathcal{A}$ 根据 $\mathcal{A}$的结构，通过归纳定义如下：

1. 若 $\mathcal{A}$ 是原子公式，则 $\zeta \models \mathcal{A}$ 已在第5节中定义；2. $\zeta \models \neg\mathcal{A}$ 当且仅当并非 $\zeta \models \mathcal{A}$为真；3. $\zeta \models \mathcal{A}_1 \wedge \mathcal{A}_2$ 当且仅当 $\zeta \models \mathcal{A}_1$ 且 $\zeta \models \mathcal{A}_2$；4. $\zeta \models (\forall x)\mathcal{A}$ 当且仅当 $(\sigma[x := d], \eta) \models \mathcal{A}$ 对任何与x类型相同的值 d 都成立；以及5. $\zeta \models (\forall X)\mathcal{A}$ 当且仅当 $(\sigma, \eta[X := A]) \models \mathcal{A}$ 对任何与 X 类型相同的算子 A 都成立。

设 $\Gamma \subseteq \mathit{Wff}^{\mathrm{SOL}}$ 是关于算子的逻辑公式。则 $\zeta \models \Gamma$表示，对于每个 $\mathcal{A} \in \Gamma$，都有 $\zeta \models \mathcal{A}$。

经典一阶逻辑的代换引理 $\mathcal{L}$(引理 2.1) 可以推广到符号算子逻辑**SOL**。

引理 6.1 (代换)。对于任意逻辑公式 $\mathcal{A} \in \mathit{Wff}^{\mathrm{SOL}}$ 以及任意经典状态 σ 和算子变量的赋值 η 所构成的 $\zeta = (\sigma, \eta)$，我们有：

$$\zeta \models \mathcal{A}[\overline{t}/\overline{x}][\overline{A}/\overline{X}] \text{ iff } (\sigma[\overline{x} := \sigma(\overline{t}], \eta[\overline{X} := \zeta(\overline{A})]) \models \mathcal{A}. \tag{29}$$

证明。对 $\mathcal{A}$ 的结构进行归纳，并使用引理4.3。 □

6.3. 蕴涵

基于**SOL**的语义，我们可以引入如下概念：在一组经典一阶逻辑公式 Σ 作为假设的情况下，**SOL** 中的一组逻辑公式 Γ 蕴涵**SOL** 中的逻辑公式 $\mathcal{A}$。

定义 6.3 (蕴涵)。设 $\Sigma \subseteq \mathit{Wff}_{\mathcal{L}}$ 是一组经典一阶逻辑公式，$\Gamma \subseteq \mathit{Wff}^{\mathrm{SOL}}$ 是一组关于算子的逻辑公式，且 $\mathcal{A} \in \mathit{Wff}^{\mathrm{SOL}}$。那么我们称 Σ 和 Γ 蕴涵 $\mathcal{A}$，记作

$$\Sigma, \Gamma \models \mathcal{A} \tag{30}$$

如果对于任意经典结构 $\mathcal{S}$ 以及其上的任意量子结构 $\mathcal{Q}$，对于 $\mathcal{S}$ 中的所有经典状态 σ，以及对于 $\mathcal{Q}$ 中算子变量的所有赋值 η，只要 $\sigma \models \Sigma$ 和（σ、η）$\models \Gamma$成立，我们就有 $\sigma \models \mathcal{A}$。

特别地，如果 $\Sigma=\emptyset$，则（30）可简写为 $\Gamma\models\mathcal{A}$；如果 $\Gamma=\emptyset$，则（30）可简写为 $\Sigma\models\mathcal{A}$；如果 $\Sigma=\Gamma=\emptyset$，那么我们称 $\mathcal{A}$在逻辑上有效，并简写为 $\models\mathcal{A}$。

例 6.3 （量子比特的布洛赫球可视化 [31],第 15 页）。

$$|\alpha|^2+|\beta|^2=1\models(\exists\theta,\varphi,\gamma)\left(\alpha|0\rangle+\beta|1\rangle=e^{i\gamma}\left(\cos\frac{\theta}{2}|0\rangle+e^{i\varphi}\sin\frac{\theta}{2}|1\rangle\right)\right)$$

经典一阶逻辑中的各种有效逻辑公式都可以直接推广到符号算子逻辑 **SOL**；例如，

1. $\models(\forall x)\mathcal{A}\to\mathcal{A}[t/x]$ 对于任意经典变量 x 和经典表达式t；2. $\models(\forall X)\mathcal{A}\to\mathcal{A}[B/X]$ 对于任意算子变量 X 和形式算子公式 B；3. $\models(\forall x)(\mathcal{A}\to\mathcal{B})\to(\mathcal{A}\to(\forall x)\mathcal{B})$，若经典变量 x 在 $\mathcal{A}$中自由出现；4. $\models(\forall X)(\mathcal{A}\to\mathcal{B})\to(\mathcal{A}\to(\forall X)\mathcal{B})$，若算子变量 X 在 $\mathcal{A}$中自由出现。

经典一阶逻辑中的推理规则，例如肯定前件式和概括规则，可以扩展到 **SOL**。这些逻辑公式的有效性以及这些推理规则的可靠性都可以根据定义轻松验证。此外，我们有：

定理 6.1 (演绎定理)。 对于任意 $\Sigma\subseteq Wff_{\mathcal{C}}$ 和 $\Gamma\subseteq Wff_{\mathbf{SOL}}$，以及任意 $\mathcal{A}$, $\mathcal{B}\in Wff_{\mathbf{SOL}}$，有

$$\Sigma,\Gamma\cup\{\mathcal{A}\}\models\mathcal{B}\ \textit{iff}\ \Sigma,\Gamma\models\mathcal{A}\to\mathcal{B}.$$

证明。 根据蕴涵的定义，很容易得出。 □

定理 6.2 (代换)。 若 $\Sigma,\Gamma\models\mathcal{A}$，则对于任意经典变量$x$、算子变量 X、经典表达式 t 和形式算子公式B，有：

$$\Sigma[\overline{t}/\overline{x}],\Gamma[\overline{t}/\overline{x}][\overline{B}/\overline{X}]\models\mathcal{A}[\overline{t}/\overline{x}][\overline{B}/\overline{X}].$$

证明。 由引理 6.1 可立即得出。 □

(统计1) $\models \mathcal{U}(A_1) \wedge \mathcal{S}_p(A_2) \to \mathcal{S}_p(A_1A_2)$ (统计2) $\models \mathcal{U}(A_1) \wedge \mathcal{S}_m(A_2) \to \mathcal{S}_m(A_1A_2A_1^\dagger)$ (统计3) $\models 0 \sqsubseteq A \wedge \mathsf{tr}(A) = 1 \to \mathcal{S}_m(A)$ (幺正1) $\models \mathcal{U}(A) \to \mathcal{U}(A^\dagger)$ (幺正2) $\models \mathcal{U}(A_1) \wedge \mathcal{U}(A_2) \to \mathcal{U}(A_1A_2)$ (观测量1) $\models \mathcal{O}(A) \to A = A^\dagger$ (观测量2) $\models \mathcal{U}(A_1) \wedge \mathcal{O}(A_2) \to \mathcal{O}(A_1A_2A_1^\dagger)$ (观测量3) $\models \mathcal{O}(A_1) \wedge \mathcal{O}(A_2) \wedge A_1A_2 = A_2A_1 \to \mathcal{O}(A_1A_2)$

表 2：量子态、酉变换和可观测量的公理。

7. 关于量子数据和操作的符号推理

本节将通过一系列示例，说明如何运用上一节提出的逻辑机制，对量子数据和操作进行符号推理。在其中一些示例中，在一些示例中，蕴涵 $\Sigma, \Gamma \models \mathcal{A}$ 将用以下图示表示：

$$\frac{\Sigma \quad \Gamma}{\mathcal{A}}$$

7.1. 量子数据和操作的基本性质

首先，我们来说明如何在符号算子逻辑 **SOL** 中表述并推导量子数据和操作的一些基本性质。

示例 7.1. 表 *2* 展示了逻辑**SOL**中量子态、酉变换和可观测量的一些基本公理的表述。

例 7.2. 以下是**SOL** 中酉变换和可观测量定义的形式化表述：

$$\bigwedge_{i=1}^{n}\left(\sum_{l=1}^{n}|a_{il}|^2=1\right)\wedge\bigwedge_{1\le i<j\le n}\left(\sum_{l=1}^{n}a_{il}a_{jl}^*=0\right)\models\mathcal{U}\left(\sum_{i,j=1}^{n}a_{ij}|i\rangle\langle j|\right) \tag{31}$$

$$\bigwedge_{i,j=1}^{n}\left(a_{ij}=a_{ji}^*\right)\models\mathcal{O}\left(\sum_{i,j=1}^{n}a_{ij}|i\rangle\langle j|\right). \tag{32}$$

值得注意的是，*(31)* 和 *(32)* 的左侧是经典一阶逻辑公式，而右侧则是

SOL 中的逻辑公式。由于

$$\textit{The theory of trigonometric functions} \models |\cos\frac{\theta}{2}|^2 + |\sin\frac{\theta}{2}|^2 = 1$$

我们可以推导出绕坐标轴 x、y 和 z 的旋转的幺正性：

$$\models \mathcal{U}(R_x(\theta)) \wedge \mathcal{U}(R_y(\theta)) \wedge \mathcal{U}(R_z(\theta))$$

由 *(31)* 得出。同样，我们也可以在 **SOL** 中推导出量子傅里叶变换的幺正性：$\models \mathcal{U}(\mathit{QFT})$。

7.2. （条件）等式推理

由于 **SOL** 是一种带等式的逻辑，因此可以在其中对算子进行等式推理。具体而言，只要所涉及的算子以某些经典变量作为参数，就可以根据关于这些变量的经典一阶理论进行条件等式推理。以下是一些简单示例：

示例 7.3。1. 线性性：$\models \alpha(\sum_i A_i) = \sum_i \alpha A_i$.

2. 量子态相等：$\models \bigwedge_{i<j}\ s_i \neq s_j \wedge \sum_i \alpha|s_i\rangle = \sum_i \beta_i|s_i\rangle \ \to ()()$

()3. 系数加法：$\bigwedge_i\ \alpha_i = \beta_i\ .$

$$\frac{s_1 = s_2 \wedge q_1 = q_2}{\alpha_1|s_1\rangle_{q_1} + \alpha_2|s_2\rangle_{q_2} = (\alpha_1+\alpha_2)|s_1\rangle_{q_1}}$$

4. 自外积：

$$\frac{s_2 = s_3 \wedge q_2 = q_3}{\left(|s_1\rangle_{q_1}\left(|s_2\rangle_{q_2}^{\dagger}\right)\right)|s_3\rangle_{q_3} = |s_1\rangle_{q_1}}$$

5. 恒等式：

$$\frac{\dim \overline{q} = m \quad (\forall 1 \leq i < j \leq m)(\overline{s}_i \neq \overline{s}_j)}{\sum_{i=1}^{m} |\overline{s}_i\rangle_{\overline{q}}\langle\overline{s}_i|_{\overline{q}} = I[\overline{q}]}$$

6. 矩阵表示：

$$\frac{A : \overline{q} \to \overline{q'} \quad \dim \overline{q} = m \quad \dim \overline{q'} = n \quad \overline{s}_i \neq \overline{s}_j\ (1 \leq i < j \leq m) \quad \overline{s'}_k \neq \overline{s'}_l\ (1 \leq k < l \leq n)}{A = \sum_{i=1}^{m}\sum_{k=1}^{n}\langle\overline{s}_i|A|\overline{s'}_k\rangle|\overline{s}_i\rangle_{\overline{q}}\langle\overline{s'}_k|_{\overline{q'}}}$$

量子物理和量子信息理论中常用的带标签的狄拉克记号在 [51] 中被形式化，以便在量子程序验证中方便地处理多个量子变量。此外， [41, 42] 中收集并实现了一系列针对带标签的狄拉克记号的重写规则。显然，借助第3节中介绍的量子数组数据结构，可以轻松地重新表示带标签的狄拉克记号。因此， [41, 42] 中的重写规则可以推广到**SOL**框架中。与带标签的狄拉克记号相比，**SOL** [51, 41, 42] 还提供了其他功能：可以对量子寄存器或存储器进行地址算术，并在经典一阶逻辑语言中表达相关经典变量的某些假设，从而对其进行条件推理；一个简单示例见式（4）。

7.3.（条件）不等式推理

由于逻辑 **SOL** 也配备了二元谓词 $\sqsubseteq$，因此也可以方便地在其中对符号算子进行不等式推理，并且可以根据所涉及算子中的经典参数的某些假设进行条件推理。下面是一个简单的例子：

例7.4。 概率组合：

$$\frac{(\forall i)\,(0 \le p_i \le p_i' \wedge 0 \le A_i \sqsubseteq A_i')}{\sum_i p_i A_i \sqsubseteq \sum_i p_i' A_i'}$$

以下两个命题给出了一些关于符号算子进行等式推理和不等式推理的基本技巧：

命题7.1。 1. 反对称性和传递性：

(a) $\Sigma, \Gamma \models A \sqsubseteq B$ 当且仅当 $\Sigma, \Gamma \models A \equiv B$; (b) 若 $\Sigma, \Gamma \models A \sqsubseteq B$ 且 $\Sigma, \Gamma \models B \sqsubseteq C$, 则 $\Sigma, \Gamma \models A \sqsubseteq C$.

2. 由操作保持的不等式：若 $\Sigma, \Gamma \models A \sqsubseteq B$, 则：

(a) $\Sigma, \Gamma \models \neg cA \sqsubseteq \neg cB$ 若 $c \ge 0$ 且 Σ, 则 $\Gamma \models cB \sqsubseteq cA$ 若 $c \le 0$;(b) $\Sigma, \Gamma \models A^\dagger \sqsubseteq B^\dagger$;(c) $\Sigma, \Gamma \models A + C \sqsubseteq B + C$ 且 $\Sigma, \Gamma \models C + A \sqsubseteq C + B$;(d) $\Sigma, \Gamma \models AC \sqsubseteq BC$ 且 $\Sigma, \Gamma \models CA \sqsubseteq BC$;

(e) $\Sigma, \Gamma \models A \otimes C \sqsubseteq B \otimes C$ 且 $\Sigma, \Gamma \models C \otimes A \sqsubseteq C \otimes B$。

证明。 根据定义显然成立。 □

作为定理 6.2 的推论，我们有：

命题 7.2 (代换)。 **1.** 如果 $\Sigma,\ \Gamma \models A \sqsubseteq B$，那么 $\Sigma[t/x],\ \Gamma[t/x] \models A[t/x] \sqsubseteq B[t/x]$. **2.** 如果 $\Sigma,\ \Gamma \models A = B$，那么 $\Sigma[t/x],\ \Gamma[t/x] \models A[t/x] = B[t/x]$.

算子变量的代换也同样如此。

8. 量子数据与量子谓词的递归定义

符号算子逻辑**SOL**最重要的应用之一，是将经典一阶逻辑 $\mathcal{L}$ 嵌入其中，从而能够在**SOL**中方便而优雅地书写量子数据和算子的递归定义，其中将 $\mathcal{L}$ 中的变量用作参数。本节给出几个简单示例：先定义一个递归的经典数据结构，再通过从 $\mathcal{L}$到**SOL**的提升，递归定义一个量子数据结构。

例 8.1 （均匀叠加与 GHZ 态）。 考虑量子比特数组区间 $q[m\colon\ n]$。其上的均匀叠加态和 *GHZ*（*Greenberger-Horne-Zeilinger*）态可以看作由经典变量 m 和 n参数化的量子态，并按如下方式递归定义：

$$|S(m,n)\rangle = \begin{cases} \frac{1}{\sqrt{2}}\left(|0\rangle_{q[m]} + |1\rangle_{q[m]}\right) & if\ n = m, \\ |S(m,n-1)\rangle \otimes \frac{1}{\sqrt{2}}\left(|0\rangle_{q[n]} + |1\rangle_{q[n]}\right) & if\ n > m; \end{cases}$$

$$|GHZ(m,n)\rangle = \begin{cases} \frac{1}{\sqrt{2}}\left(|0\rangle_{q[m]} + |1\rangle_{q[m]}\right) & if\ n = m, \\ CNOT[q[n-1], q[n]]\left(|GHZ(m,n-1)\rangle \otimes |0\rangle_{q[n]}\right) & if\ n > m. \end{cases}$$

容易验证蕴涵 $m = n \models |S(m,n)\rangle = |GHZ(m,n)\rangle$。

示例 8.2 （量子傅里叶变换中的数据结构）。 设 j 是类型为 **Int** $\to$ **Bool** 的经典比特数组。对于任意整数 k、l，且满足 $1 \le k \le l$，我们用 $0.j[k\colon\ l]$ 表示其二进制表示

$$0.j_k ... j_l = \sum_{r=k}^{l} j[r] \cdot 2^{k-r-1}.$$

然后，量子数组 $|(j, k\colon\ l)\rangle$ 和 $|QFT(j, k\colon\ l)\rangle$ 被定义为以下状态

量子比特 $q[k:\ l]$ 递归地:

$$\begin{cases} |(j,k:l)\rangle = |j[l]\rangle \ \textit{if } k=l, \\ |(j,k:r+1)\rangle = |(j,k:r)\rangle \otimes |j[r+1]\rangle \ \textit{for } k \le r \le l-1; \end{cases}$$

$$\begin{cases} |QFT(j,k:l)\rangle = \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i 0.j[l]}|1\rangle\right) \ \textit{if } k=l, \\ |QFT(j,k:r+1)\rangle = |QFT(j,k:r)\rangle \otimes \frac{1}{\sqrt{2}}\left(|0\rangle + e^{2\pi i 0.j[l-r:l]}|1\rangle\right) \ \textit{for } k \le r \le l-1. \end{cases}$$

由 *[31]* 中的方程（*5.4*）很容易看出，l 量子比特上的量子傅里叶变换可以改写为:

$$|(j,1:l)\rangle \mapsto |QFT(j,1:l)\rangle.$$

还可以用类似的方式递归定义更多量子数据结构；例如，正则语言量子态 [20] 和图态[22]。

此外，读者可以在 [46, 47] 中找到一些递归定义量子门和量子算法的有趣示例。特别是，符号算子逻辑 **SOL** 可作为量子程序形式化验证中的断言逻辑。

9. 潜在应用

本节将讨论本文提出的逻辑框架在规约和推理量子数据与操作方面的其他几种潜在应用。

9.1. 面向量子信息理论家的证明助手

Lean-QuantumInfo 库最近开发完成，并成功应用于验证量子信息理论中的一系列定理；尤其是广义量子斯坦引理 [30]。本文定义的符号算子逻辑 **SOL**似乎可以用作此类库的形式化规范语言。为此，我们来看一个简单的例子，即 [30] 中介绍的不可克隆定理形式化。仅从该形式化中并不能立即看出它确实表示不可克隆定理，而且其形式对于实际应用而言不太方便。相比之下，使用**SOL**语言，不可克隆定理可以像示例 6.2 那样进行形式化。显然，这种新的

形式化更有效地体现了不可克隆定理背后的直觉，并且可以直接在应用中调用。此外，将 Lean-QuantumInfo 库与**SOL**的推理能力相结合，可以开发出一款专为量子信息理论家打造的强大证明助手。

9.2. 量子程序的规约与验证

人们已经开发了多种量子程序逻辑，用于推理量子程序的正确性 [43, 36, 19, 14, 50, 27, 29] 以及量子密码协议的安全性 [37, 3]。然而，正如引言中所述，目前仍缺少能够规约量子数据性质，并能与这些程序逻辑顺畅整合，从而实现高效、可扩展验证的断言逻辑。为定义此类断言语言，人们已经进行了若干尝试，其中包括作者先前的工作 [44, 46]。事实上，本文可视为对 [46] 中提出的断言语言的延续与拓展。我们预计，**SOL**可与量子程序逻辑结合，并在 Lean 和 Coq 等证明助手中实现，以便对量子程序进行实用且可扩展的验证。

9.3. 量子程序的符号执行

近年来，符号执行已扩展至量子程序分析 [8, 4, 17]。然而，它在量子计算中的全部潜力尚未发挥出来，因为这些工作只能对非常受限类别的量子态和操作进行恰当的符号化，例如 [17] 中的稳定子态和 Clifford算子。这一局限很大程度上源于缺乏一个通用的逻辑框架，用于规约和推理任意量子算子与量子态。本文提出的符号算子逻辑 **SOL**似乎非常适合纳入符号执行，从而有望为量子程序分析与验证提供一种表达能力更强、功能更强大的方法。

9.4. 量子系统的符号模型检测

模型检测技术也已应用于量子系统的分析与验证，包括量子电路、量子程序和量子通信协议。事实上，已有若干用于量子系统的模型检查器被实现 [21, 18, 12]。不过，这些工具只能处理涉及少量量子比特的简单示例。

符号模型检测使用逻辑公式和基于决策图（DD）的数据结构来表示状态和转移集合，使可达状态空间能够得到更高效的探索：每一步都可通过同时操作大量状态，而非单个状态来完成。已有少数尝试 [16] 将这一方法扩展到量子情境，但这些尝试仍依赖逻辑语言来表示量子数据和操作。本文提出的符号算子逻辑 **SOL** 可用于此目的。

10. 结论

本文定义了一个双层逻辑框架，其中符号化算子逻辑 **SOL** 建立在一阶逻辑之上 $\mathcal{L}$。本研究实质上扩展了 [46, 47]中引入的形式化参数化量子态和谓词。两种逻辑 $\mathcal{L}$ 和 **SOL** 分别用于规约和推理经典数据与操作以及量子数据与操作。

单层与双层：有人可能会认为，可以将**SOL**和 $\mathcal{L}$压缩到同一层，形成一种单一逻辑，前提是将量子数据和经典数据置于同一层，但赋予不同类型。原则上，这是可行的。但在实际实现验证工具时，我们认为，双层框架能够以结构良好的方式验证量子计算与信息。

符号化规约与推理：引入**SOL** 的主要好处在于，我们可以方便地在 $\mathcal{L}$ 中将经典变量作为参数来定义量子数据和操作，然后借助 $\mathcal{L}$的强大能力，以符号化方式推理其性质。正如我们的示例所示，与非符号化规约和推理相比，这种方式通常经济得多。

我们必须承认，本文仅概述了用于量子数据与操作的符号化规约与推理的逻辑框架。许多细节仍有待补充。不过，我们相信，在 Lean 和 Coq 等证明助手中实现这一框架，将为量子计算与信息的形式化验证奠定有效基础。

最后，我们指出这一研究方向上值得进一步探索的几个课题：

• 本文仅展示了一些非常简单的示例，不足以有效体现逻辑**SOL**的优势。若能找出更有说服力的应用示例，将大有裨益。

• 超算子被广泛应用于量子信息与计算研究 [31] 中，既作为量子信道和噪声的数学模型，也作为量子程序的语义模型。然而，在现阶段，它们尚未在逻辑**SOL**中形式化。因此，有必要通过引入第三层超算子，将超算子纳入**SOL** 。

• 文献中已经提出了若干关于量子计算符号推理的方案。未来研究的一个重要课题是，如何将**SOL**中的形式算子和逻辑公式与量子电路的其他表示连接起来，包括各种决策图（DD），例如 QMDD [32], TDD [24], CFLOBDD [35], LimDD [38], 和 FeynmanDD [39]；路径和表示 [1]; ZX 演算 [11]；树自动机 [10]；Coq [34]和 Maude [16]中的符号推理框架；以及酉表达式 [48]。

参考文献

[1] M. Amy，迈向通用量子电路的大规模函数式验证，第*15*届量子物理与逻辑国际会议（*QPL*）论文集2018年。[2] K. R. Apt、F. S. de Boer 和 E. -R. Olderog，《顺序与并发程序验证》，施普林格，伦敦，2009年。[3] G. Barthe、J. Hsu、M. S. Ying、N. K. Yu 和 L. Zhou，量子程序的关系证明，*ACM* 编程语言会议论文集4(2020) POPL：论文编号21:1-29。[4] F. Bauer-Marquart、S. Leue 和 C. Schilling，symQV：量子程序的自动符号验证，第*25*届形式化方法国际研讨会（*FM*）论文集，2023年，第181-198页。[5] A. Berezutskii、M. Z. Liu、A. Acharya 等，量子计算的张量网络，*arXiv*: 2503.08626。

[6] A. Bordg、H. Lachnitt 和 Y. J. He，在 Isabelle/HOL 中经验证的量子计算，《自动推理杂志》65(2021) 691-709。[7] J. R. Burch、E. M. Clarke、K. L. McMillan、D. L. Dill 和 L. J. Hwang，符号模型检测：10^{20}状态及更多，《信息与计算》98(1992) 142-170。[8] J. Carette、G. Ortiz 和 A. Sabry，Hadamard-Toffoli 量子电路的符号执行，*2023*年*ACM*部分求值与程序操纵国际研讨会（*PEPM*）论文集，第14-26页。[9] C. Chareton、S. Bardin、F. Bobot、V. Perrelle 和 B. Valiron，针对电路构建量子程序的自动演绎验证框架，第*30*届欧洲编程语言研讨会（*ESOP*）论文集，施普林格，2021，第148-177页。[10] Y. F. Chen、K. M. Chung、O. Lengál、J. A. Lin、W. L. Tsai 和 D. D. Yen，一种用于量子电路验证与漏洞搜寻的基于自动机的框架，《*ACM*编程语言会议论文集》7(PLDI) 1218-1243。[11] B. Coecke 和 R. Duncan，交互式量子可观测量，第*35*届自动机、语言与程序设计国际研讨会（*ICALP*）论文集，施普林格，2008年，第298-310页。[12] A. C. Dai 和 M. S. Ying，QReach：量子马尔可夫链可达性分析工具，国际计算机辅助验证会议（*CAV*）论文集，2024年，第520-532页。[13] H. W. Deng、R. Z. Tao、Y. X. Peng 和 X. D. Wu，递归量子酉程序综合的一个案例，*ACM* 编程语言会议论文集 8(POPL)(2024): 1759-1788。[14] Y. X. Deng、H. L. Wu 和 M. Xu，经典-量子程序概率行为的局部推理，第*25*届验证、模型检测与抽象解释国际会议（*VMCAI*）论文集，2024年，第163-184页。[15] E. D'Hondt 和 P. Panangaden，量子最弱前置条件，《计算机科学中的数学结构》，16(2006)429-451。

[16] C. M. Do 和 K. Ogata, 《Maude 中量子电路的符号模型检测》, *PeerJ* 计算机科学（2024）。[17] W. Fang 和 M. S. Ying, 《量子纠错程序的符号执行》, *ACM* 编程语言会议论文集8（2024）PLDI: 1040-1065。[18] Y. Feng、E. M. Hahn、A. Turrini 和 L. J. Zhang, 《QPMC：量子程序和协议的模型检查器》, *2015*年国际形式化方法研讨会论文集（*FM*）, 第265-272页。[19] Y. Feng 和 M. S. Ying, 《带经典变量的量子霍尔逻辑》, *ACM* 量子计算汇刊2（2021）文章16：1-43。[20] M. Florido-Llin as、A. M. Alhambra、D. Pérez-García 和 J. I. Cirac, 《正则语言量子态》, $arXiv$: 2407.17641。[21] S. J. Gay、R. Nagarajan 和 N. Papanikolaou, 《QMC：量子系统的模型检查器》, *CAV 2008*会议论文集，第543-547页。[22] M. Hein、W. Dür、J. Eisert、R. Raussendorf、M. Van den Nest 和 H.-J. Briegel, 《图态中的纠缠及其应用》, 恩里科·费米国际物理学学校量子计算机、算法与混沌会议论文集，2005。[23] K. Hietala、R. Rand、S. -H. Hung、X. D. Wu 和 M. Hicks, 《量子电路的经过验证的优化器》, *ACM* 编程语言会议论文集5（POPL）（2021）37：1-29。[24] X. Hong、X. Z. Zhou、S. J. Li、Y. Feng 和 M. S. Ying, 《一种基于张量网络的量子电路表示决策图》, *ACM* 电子系统设计自动化汇刊27:6（2022）1-30。[25] J. C. King, 《符号执行与程序测试》, *ACM* 通讯19（1976）385-394。[26] A. Kissinger 和 J. van de Wetering, 《PyZX：大规模自动化图示推理》, 第*17*届量子物理与逻辑国际会议（*QPL*）论文集2020，第229-241页。

[27] X. -B. Le、S. -W. Lin、J. Sun 和 D. Sanán，《基于分离逻辑的量子程序局部推理中分离合取的量子解释》，*ACM* 编程语言会议论文集6（2022）POPL：1-27。[28] J. Y. Liu、B. H. Zhan、S. L. Wang、S. G. Ying、T. Liu、Y. J. Li、M. S. Ying 和 N. J. Zhan，《使用量子霍尔逻辑对量子算法进行形式化验证》，第*31*届计算机辅助验证国际会议（*CAV*）论文集，施普林格，2019，第187-207页。[29] Y. Matsushita、K. Hirata、R. Wakizaka 和 E. D'Osualdo，《RapunSL：通过分离、线性组合与混合理清量子计算》，$POPL$ 2026。[30] A. Meiburg、L. A. Lessa 和 R. R. Soldati，《Lean 中广义量子 Stein 引理的形式化》，$arXiv$: 2510.08672。[31] M. A. Nielsen 和 I. L. Chuang，《量子计算与量子信息》，剑桥大学出版社，2010。[32] P. Niemann、R. Wille、D. M. Miller、M. A.Thornton 和 R. Drechsler，《量子多终端决策图：高效的量子函数表示与操作》，*IEEE*集成电路系统计算机辅助设计汇刊35:1（2015）86-99。[33] R. Rand、J. Paykin 和 S. Zdancewic，《QWIRE实践：在 Coq 中对量子电路进行形式化验证》，第*14*届量子物理与逻辑国际会议（*QPL*）论文集，2017，第119-132页。[34] W. J. Shi、Q. X. Cao、Y. X. Deng、H. R. Jiang 和 Y. Feng，《在 Coq 中对量子电路进行符号推理》，《计算机科学与技术学报》36（2021）1291-1306。[35] M. A. Sistla、S. Chaudhuri 和 T. Reps，《上下文无关语言有序二元决策图》，*ACM* 编程语言与系统汇刊46:2（2023），文章7。[36] D. Unruh，《带幽灵变量的量子霍尔逻辑》，第*34*届*ACM/IEEE*计算机科学逻辑年会（*LICS*）论文集，2019。

[37] D. Unruh, 《量子关系霍尔逻辑》, *ACM* 编程语言会议论文集 3(POPL)(1019) 33:1-31。[38] L. Vinkhuijzen、T. Coopmans、D. Elkouss、V. Dunjko 和 A. Laarman, 《LIMDD：一种用于模拟量子计算（包括稳定子态）的决策图》, *Quantum* 7(2023) 1108。[39] Z. Y. Wang、B. Cheng、L. X. Yuan 和 Z. F. Ji, 《FeynmanDD：使用经典决策图进行量子电路分析》, 第*37*届国际计算机辅助验证会议（*CAV*）论文集，2025，第28-52页。[40] R. Wille、S. Hillmich 和 L. Burgholzer, 《基于决策图的量子计算工具》, *ACM* 量子计算汇刊3(2022) 13:1-17。[41] Y. T. Xu、G. Barthe 和 L. Zhou, 《自动化狄拉克符号中的等式证明》, 《*ACM*编程语言会议论文集》9:POPL(2025) 42:1-33。[42] Y. T. Xu、L. Zhou 和 G. Barthe, 《D-Hammer：带标签的狄拉克记号的高效等式推理》, 第*37*届国际计算机辅助验证会议（*CAV*）论文集。[43] M. S. Ying, 《量子程序的弗洛伊德–霍尔逻辑》, *ACM*《编程语言》汇刊, *39* (2011)，文章编号19。[44] M. S. Ying, 《伯克霍夫–冯·诺伊曼量子逻辑作为量子程序的断言语言》, *arXiv* 2205.01959 (2022)。[45] M. S. Ying, 《量子编程基础》（第二版），Morgan Kauffman，2024。[46] M. S. Ying, 《带经典变量的实用量子霍尔逻辑，I》, *arXiv*: 2412.09869。[47] M. S. Ying 和 Z. C. Zhang, 《递归定义的量子电路的验证》, *arXiv*: 2404.05934。[48] E. Younis, 《酉表达式：可扩展量子编程语言和系统所必需的抽象》, *arXiv* 2501.09667。

[49] N. K. Yu 和 J. Palsberg, 《量子抽象解释》, 第*42*届*ACM*程序设计语言设计与实现国际会议（*PLDI*）论文集，2021，542-558。[50] L. Zhou、G. Barthe、J. Hsu、M. S. Ying 和 N. K. Yu, 《分簇逻辑与量子分离逻辑的量子解释》, *LICS* 2021：1-14。[51] L. Zhou、G. Barthe、P. -Y. Strub、J. Y. Liu 和 M. S. Ying, 《CoqQ：量子程序的基础验证》, *ACM* 编程语言会议论文集7(POPL) 833-865。