软件学报 ISSN 1000-9825, CODEN RUXUEW
Journal of Software, [doi: 10.13328/j.cnki.jos.005536]

E-mail: jos@iscas.ac.cn
http://www.jos.org.cn
Tel: +86-10-62562563

量子程序验证

冯 元[1], 应明生[1,2,3]

[1](University of Technology Sydney, Australia)

[2](中国科学院软件研究所,北京 100190)

[3](清华大学计算机系,北京 100084)

通讯作者: 冯元, E-mail: Yuan.Feng@uts.edu.au

摘要: 量子硬件设计与制造技术的飞速发展使得人们开始预言大于一百个量子比特的特定用途的量子计算机有望在 5-10 年内实现.可以想见,到那时候,量子软件的开发将变成真正发挥这些计算机能力的关键.然而,由于量子信息的不可克隆性和纠缠的非局域作用等量子特征,如何设计正确高效的量子程序和量子通信协议将是一个富有挑战性的课题.形式化验证方法,特别是模型检测技术,已经在经典软件设计和系统建模方面被证明行之有效,因此量子软件的形式化验证也开始受到越来越多的关注.本文从量子顺序程序验证和量子通信协议验证两方面,对近年来国内外学者,特别是两位作者所在的研究组在该研究领域取得的一些成果进行了系统的总结.最后,对未来可能的研究方向和面临的挑战进行了简单展望.

关键词: 量子计算;程序验证;模型检测;通信协议验证
中图法分类号: TP311

中文引用格式: 冯元,应明生.量子程序验证.软件学报,2018,29(4). http://www.jos.org.cn/1000-9825/5536.htm
英文引用格式: Feng Y, Ying MS. Verification of quantum programs. Ruan Jian Xue Bao/Journal of Software, 2018,29(4). http://www.jos.org.cn/1000-9825/5536.htm

Verification of Quantum Programs

FENG Yuan[1], YING Ming-Sheng[1,2,3]

[1](University of Technology Sydney, Australia)

[2](Institute of Software, Chinese Academy of Sciences, Beijing 100190, China)

[3](Department of Computer Science and Technology, Tsinghua University, Beijing 100084, China)

Abstract: With the rapid development of quantum hardware, people tend to believe that special-purpose quantum computers with more than 100 qubits will be available in 5 to 10 years. It is conceivable that, once this becomes a reality, the development of quantum software will be crucial inharnessing the power of quantum computers. However, due to the distinguishable features of quantum mechanics, such as the no-cloning of quantum information and the nonlocal effect of entanglement, developing correct and efficient quantum programs and communication protocols is a challenging issue. Formal verification methods, particularly model checking techniques, have proven effectivein classical software design and system modelling. Therefore, formal verification of quantum software has received more and more attention recently. This article reviews recent researchfindings in verification of both sequential quantum programs and quantum communication protocols, with the focus on the work of the two authors' research groups.Future directions and challenges in this area are also discussed.

Key words: quantum computing; program verification; model checking, verification of communication protocols

1 引言

随着量子硬件设计与制造技术的飞速发展,量子计算正快步走入我们的生活.由于在计算速度方面具有超越经典计算的潜在优势,以及可以提供绝对安全的密码方案,量子计算的理论研究受到了广泛的重视.在学术

收稿时间: 2017-12-24; 采用时间: 2018-01-03; jos在线出版时间: 2018-01-09

界：欧盟第六框架将量子信息与量子计算的基础结构作为重要研究内容,并集中了牛津大学、Bristol 大学、约克大学、巴黎七大、因斯布鲁克大学以及 McGill 大学的研究人员；由英国工程与物理科学研究理事会(EPSRC)和国家 e-Science 中心共同资助、英国计算研究委员会负责的 UK Grand Challenges Exercise 中七个主题之一的 Journeys in Non-Classical Computation 将“Quantum Software Engineering”列为最主要内容；美国国家标准技术研究所、MIT、华盛顿大学、哥伦比亚大学、英国牛津大学、法国国家科学研究中心等著名学术机构已经开始量子软件相关的研究；密歇根大学开展了量子计算机体系结构以及量子设计自动化方面的工作.在工业界：2014 年,IBM 宣布耗资 30 亿美元研发下一代芯片,主要是量子计算和神经计算；2016 年 5 月,IBM 发布了 5 个量子比特的量子计算云服务,并于 2017 年 3 月发布 IBM Q 系统,目前已可以支持 16 个量子比特；2014 年,加州大学圣巴拉拉分校的知名物理学家 John Martinis 研究组加入谷歌研发量子计算处理器,并于 2016 年 9 月提出“量子霸权(quantum supremacy)”研制计划,其目标是在 2017 年底前实现 50 个量子比特的处理器,从而超越目前传统计算机的执行能力；2017 年 10 月,Intel 发布了 17 个量子比特的量子计算芯片,交付给其荷兰合作伙伴 QuTech 使用.基于以上进展,美国总统科学技术办公室发布量子信息文件称：“预计几十个量子比特、可供早期量子计算机科学研究的系统可望在 5 年内实现”.欧盟委员会于 2016 年 7 月发布《量子宣言》,宣布将支持一项十亿欧元的量子技术旗舰计划.《量子宣言》对量子计算机的研制做出了详细部署,计划 5 年内发展出量子计算机新算法,5 到 10 年用大于 100 个量子比特的、特定用途的量子计算机解决化学和材料科学难题.澳大利亚近年来专注于硅基、磷掺杂的量子计算方案,并于 2016 年初成立了硅基半导体量子计算国家实验室.在国内,中国科技大学、清华大学、国防科大、南京大学等在量子计算和量子软件基本理论、量子密码的理论与应用、量子计算机的物理实现等方面的工作卓有成效.

众所周知,软件是计算机的“灵魂”.一旦量子计算机研制成功,量子软件的开发将变成真正发挥量子计算机作用的关键.另一方面,由于量子系统与经典世界相比有许多根本不同的特征(如量子信息的不可克隆性、量子纠缠的非局域作用等),正如人们所预料的,经典的软件理论、方法和技术在很大程度上不能直接适用于量子软件.这就使得量子软件的理论和方法成为一个富有挑战性的课题,未来的量子程序和量子通讯协议的设计将会非常困难和容易出错.在此背景下,对于量子程序和协议的形式化验证就显得尤为重要.本文将回顾这两个方面一些已有的研究结果,重点放在两位作者所在研究组(清华、University of Technology Sydney, Australia)的工作,并兼顾其他的相关研究.对顺序量子程序,我们将分别讨论基于程序语言的验证和基于语义模型(两种量子 Markov 链)的验证.前者的好处在于结构化和易用性；后者则与语言无关,因此发展出的理论可以适用于不同的程序语言.模型检测是计算机系统,包括硬件和软件系统形式化验证的一种重要技术,由于其完全机械化和可以提供错误诊断信息,在工业界得到了广泛的应用.本文将着重介绍这一技术在量子系统中的推广和应用.

总的来说,读者将会看到,虽然这一领域取得了一些可喜的进展,但目前的研究还非常零散,很多问题甚至还不清楚如何准确定义.这大体上有两方面的原因：一、尽管最近几年量子硬件设备的物理实现取得了长足的进展,但距离能够运行真正实用的量子程序的通用量子计算机仍然非常遥远.因此绝大部分传统计算机科学家和程序设计和验证的专家还持观望态度,并没有对这一领域给予足够多的重视；二、由于量子程序和传统计算机程序相比具有很大的不同,特别是由于量子叠加和纠缠的存在,量子程序的验证往往非常困难.但是,我们有理由相信,量子程序理论和验证的研究将会吸引越来越多计算机科学家的关注,从而带动这一领域蓬勃发展.

2 量子语言与程序逻辑

自上世纪 90 年代以来,程序语言的设计与实现就是量子程序研究的主要内容.特别是最近几年,由于受到量子计算硬件进展的刺激以及 2010 年 IARPA 设立的量子计算机科学项目的推动,量子程序设计语言有了很大的进展.主要包括：(1)微软公司的 LIQUi|>[1],及其最近与苏黎世 ETH 合作提出的量子程序编译和优化的可扩展软件设计流[2].2017 年 12 月,微软推出程序语言 Q#,以及集成在开发工具 Visual Studio 中的量子模拟器 Q# library[3]; (2) Selinger 组的 Quipper[4];(3)普林斯顿大学、加州大学圣巴拉拉分校等单位合作的 Scaffold[5]；(4) Raytheon BBNTechnologies 等的 QuaFL[6], [7].另外,应明生等定义了一种代数语言[8],以便对量子电路进行代

数操作与推理,并引入了一个量子 flowchart 语言[9].

程序分析技术在编译器设计及程序优化等方面有着重要的应用.在 Scaffold 的编译器 ScaffCC[5]中已经采用了数据流分析的一些方法,分析量子程序中的纠缠并检查是否违反不可克隆原理.文[10]首先研究了有限维希尔伯特空间中循环体(loop body)为酉算子的量子循环程序的终止问题.文[11]将[10]中的一些主要结果推广到循环体为一般超算子的情形,并提供了计算平均执行时间的一般性方法.这项工作中采用的主要技术是超算子的矩阵表示及薛定谔—海森堡对偶性.注意到微软新近推出的 Q#语言[3]已经支持循环,这一工作可以用来分析 Q#程序的正确性.Perdrix 与 Jorrand[12], [13]将抽象解释(abstract interpretation)技术引入量子程序分析,特别是程序中的纠缠及其演化的分析.Honda[14]进一步分析了可用 stablizer formalism 描述的这一类特殊量子程序中的纠缠.

2.1 量子程序的Hoare-Floyd逻辑

量子程序验证研究的一个重要方面是发展适用于量子计算的程序逻辑.Brunet 与 Jorrand[15]提出了将 Birkhoff-von Neumann 量子逻辑应用于量子程序推理的方法.Baltag 与 Smets[16]发展了一种可用于量子系统中信息流形式化的动态逻辑.冯元等[17]发现了关于量子程序的一些有用的推理规则.

1969 年 Turing 奖得主 Hoare 在另一 Turing 奖得主 Floyd 工作的基础上提出了程序的公理语义,通常称为 Floyd-Hoare 逻辑.这个逻辑的完备性定理则由 Turing 奖得主 Cook 于 1978 年证明.此后,Floyd-Hoare 逻辑成为程序验证的逻辑基础,在程序设计方法学中处于核心地位.Hoare 逻辑的核心概念是 Hoare 三元组,用来描述一段程序代码如何改变计算机的内存状态.Hoare 三元组具有形式$\{P\}C\{Q\}$,这里C是一段程序代码,P和Q是逻辑公式,分别称为C的前置条件和后置条件.Hoare 三元组可以有两种解释,一种是部分正确性(partial correctness):如果在C被执行前P成立,而且C终止,则在C执行之后Q必然成立;另一种是完全正确性(total correctness):如果在C被执行前P成立,则C必然终止,并且在C执行之后Q成立.由此可见,完全正确性比部分正确性要求更高.Hoare 逻辑通过为命令式编程语言的所有构造提供公理和推理规则,从而使得该语言下的所有程序的部分正确性和完全正确性都可以进行(部分)自动化证明.

在量子计算领域,Chadha, Mateus 与 Sernadas[18]给出了量子程序在只允许有界迭代的限制下的一个 Floyd-Hoare 型证明系统;Kakutani[19]试图将 den Hartog 的概率 Hoare 逻辑推广到量子情形.但这些尝试都不成功.应明生[20]建立了一个真正完整的量子 Folyd-Hoare 逻辑,证明了其(相对)完备性.值得指出的是,量子 Floyd-Hoare 逻辑完备性的证明用到了与经典 Floyd-Hoare 逻辑完备性的证明很不一样的方法,需要采用一些分析数学的技术.最近,中国科学院软件研究所詹乃军教授研究组[21]基于 Isabelle/HOL 设计实现了一个量子 Floyd-Hoare 逻辑的定理证明器.

3 基于语义模型的量子程序验证

量子程序验证的另一个思路是直接基于语义模型.在量子计算发展的初级阶段,各种量子程序设计语言和量子计算模拟环境层出不穷,但目前还没有任何一种语言占据主导地位.所以,发展基于语义模型的程序验证理论和验证工具就显得尤为重要.

基于 Girard 的 Geometry of Interaction (Abramsky-Haghverdi-Scott 范畴论形式),Hasuo 与 Hoshino[22]找到了带递归的函数式量子程序设计语言的一个语义模型.进一步地,利用线性逻辑定量语义中的一些构造,Pagani,Selinger 和 Valiron[23]定义了带递归及无限数据类型的函数式量子程序设计语言的一个指称语义.Jacobs[24]为量子程序的块(block)构造给出了一个范畴论公理化.Staton[25]为量子程序的等式推理提供了一个代数语义框架.在量子程序的最弱前置条件语义的研究中,D'Hondt 与 Panangaden[26]提出了将量子谓词定义为本征值在单位区间中的物理可观测量(Hermitian 算子).文[27]考虑了一类特殊的量子谓词——投影算子,从而使得量子逻辑中发展起来的一些方法可应用于量子程序的谓词转换器语义.需要特别指出的是,量子程序

最弱前置条件语义的研究中存在经典程序所没有的困难,即对于非交换的量子谓词,其析取和合取无法定义.

和经典程序一样,顺序量子程序也可以使用 Markov 链作为其语义模型.和以上介绍的模型相比,基于 Markov 链的语义模型更适合程序的形式化验证,特别是模型检测.在以下两个小节,我们将分别介绍两种量子 Markov 链模型以及适用于各自模型的形式化验证方法.

3.1 I型量子Markov链模型

Markov 链是描述概率系统行为和统计模型的重要数学工具,它描述了系统从一个状态到另一个状态转换的随机过程.该过程是“无记忆”的,即下一状态的概率分布只由当前状态决定,而与之前的状态无关.数学上,一个 Markov 链可以由如下三元组描述

$$M = (S, s_0, P)$$

其中S是一个有限状态的集合,$s_0 \in S$是初始状态,$P: S \times S \to [0,1]$是概率转移函数,满足归一化条件：对任意$s \in S$, $\sum_{t \in S} P(s,t) = 1$.任何顺序概率程序都可以很容易的转化成一个 Markov 链,其转移概率函数由程序中所使用的随机变量确定.这一 Markov 链模型可以很自然的推广到量子情形,我们称其为 I 型量子 Markov 链.数学上,一个 I 型量子 Markov 链用三元组$(\mathcal{H}, \rho, \mathcal{E})$来进行描述,其中$\mathcal{H}$是量子系统的状态(希尔伯特)空间, ρ是系统的初始(量子)状态,用一个密度算子(density operator)来表示,$\mathcal{E}$是描述系统状态演化的超算子(super-operator).给定一个顺序量子程序,我们首先把所有的经典变量,包括程序行标号,都用量子变量表示；程序中所涉及的所有经典运算也都表示成相应量子系统上的量子操作.这样,整个程序的行为就可以转化成一个 I 型量子 Markov 链.

3.1.1 可达性分析

可达性是经典确定性系统和概率系统模型检测的一个基本问题,很多其他的形式化验证问题都可以归结为可达性分析.对 I 型量子 Markov 链模型,应圣刚等人[28]详尽分析了可达性(reachability), 重复可达性(repeated reachability)和持续性(persistence)等性质,这些性质对于量子程序的终止(termination)、量子通信协议的公平性(fairness)等的分析具有重要意义.特别的,文[28]证明了,类似于经典 Markov 模型,量子 Markov 链的状态空间也可以分解为若干基础强连通分支(bottomstrongly connected component, BSCC)和唯一最大暂留子空间(transient subspace)的正交直和.俞能昆等[29]将这一模型推广到含有并发行为的量子程序,考虑了在公平(fairness)条件下并发量子程序的终止问题.李杨佳等[30]研究了带不确定性选择的 I 型量子自动机,其中系统的演化由一组酉算子描述,而可达状态集则为系统状态空间的闭子空间的布尔组合.在一般情况下,最终可达性(eventual reachability)、全局可达性(globalreachability)、最终永远可达性(ultimately forever reachability)和无限经常可达性(infinitely often reachability)等问题都是不可判定的.然而,如果可达集可以表现为不带否定连接词的布尔组合,那么后面三个问题则可判定.值得一提的是,经典 Markov 链的各种可达性分析往往都归结为图论中的相应问题.但由于在 I 型量子 Markov 链中,并不存在一个预先设定的有限状态结合,这些图论工具并不能直接用来进行可达性分析.为了解决这个问题,文[28]初步建立了一种希尔伯特空间中的新图论,对这种新的、能为量子 Markov 链的分析提供合适数学工具的图论的研究将是一个非常有趣和重要的问题.

3.1.2 一般性质的形式化验证

上一节我们讨论了 I 型量子 Markov 链的可达性分析.如果希望分析或验证量子程序更复杂的性质,一个首先必须解决的问题是如何定义一种可以表达这些性质的时序逻辑.由于量子测量会改变量子系统的状态,这样的时序逻辑的定义并不是一件容易的事情.目前这方面的研究刚刚起步.

3.2 II型量子Markov链模型

量子程序和量子通信协议往往也包含经典变量,按照 I 型量子 Markov 链模型的做法将这些变量全部用量子系统进行表示是非常低效的,而且会占用宝贵的存储和计算资源.因此,在实际的建模和验证中,往往需要把经典变量和量子变量区别对待.在此思想指导下,我们定义了 II 型量子 Markov 链模型,使其同时具有经典状态空间和量子状态空间.这一模型虽然并不带来表述能力的提高,但在程序和协议分析时却能带来方便.具体来

说,一个 II 型量子 Markov 链是如下的四元组$(S, \mathcal{H}, s_0, \mathrm{Q})$,其中$S, \mathcal{H}, s_0$如上定义,而Q是从$S \times S$到$\mathcal{H}$上所有超算子集合的映射,称为超算子转移函数,并满足归一化条件：对任意$s \in S$, $\sum_{t \in S} Q(s,t)$是保迹(trace-preserving)的.

II 型量子 Markov 链对于模型检测的优点主要来自如下两个方面：(1)模型检测的结果对于所有输入量子态都成立,这对于量子程序的验证尤其重要.比如说,程序的终止问题通常表现为一个超算子$\mathcal{E}$.那么,对于任何一个输入态ρ,该程序终止的概率就是$tr(\mathcal{E}(\rho))$；(2) 由于 II 型量子 Markov 链具有有限的经典状态空间,经典模型检测的技术能够有望推广到量子系统的检验中.同时,由于硬件实现的困难,在可以预见的将来量子计算机的纯量子计算部分规模不会太大,这也使得我们不必太关心模型检测技术经常会遇到的状态爆炸问题(在量子计算中也许应该是维数爆炸问题).

3.2.1 计算树逻辑模型检测

在经典模型检测中,我们往往借助于时态逻辑,特别是计算树逻辑(Computation Tree Logic, CTL)和线性时间逻辑(Linear Time Logic, LTL),来描述一大类性质.文[31]定义了一种计算树逻辑 QCTL(Quantum Computation Tree Logic),它是经典的概率计算树逻辑 PCTL 的量子推广.和 PCTL 一样,QCTL 的逻辑公式也分为状态公式和路径公式两大类,但其语义的定义由于处理的是超算子(而非 PCTL 中的概率)而变得非常复杂.对于经典 Markov 系统,PCTL 语义的定义需要借助在无限长路径的集合上引入某种概率测度[32].相应的,在 II 型量子 Markov 链中,我们需要定义超算子值的测度(super-operator valued measure).有趣的是,对于向量值测度[33]的 Kluvanek-Caratheodory-Hahn 扩展定理在这其中起到了重要的作用.在此基础上,文[31]将经典的模型检测算法推广到处理超算子的情形,从而给出了 II 型量子 Markov 链模型检测 PCTL 性质的多项式时间算法.

基于[31],我们和中国科学院软件研究所张立军教授研究组合作开发了一款模型检测工具 QPMC[34].该工具使用了一种类似于概率模型检测工具 PRISM[35]的语法,可以对简单量子程序和协议的 QCTL 性质进行自动化验证.Anticoli 等[36]将 Quipper 语言描述的电路翻译到 II 型量子 Markov 链,然后用 QPMC 进行验证.

3.2.2 线性时序逻辑模型检测

文[37]进一步考虑了线性时间性质,特别是能够用ω-正则语言表示的性质.由于超算子不具有整环性质,对ω-正则性质的模型检测具有经典 Markov 链不会遇到的困难.有趣的是,通过把 II 型量子 Markov 链转化成 I 型量子 Markov 链,并利用在[28]中发展起来的利用 I 型量子 Markov 链对希尔伯特空间进行直和分解的结论,ω-正则性质的模型检测可以转化为 I 型量子 Markov 链上的 BSCC 分解问题.由于[28]中的空间分解算法是多项式的,因此 II 型量子 Markov 链对于ω-正则性质的模型检测具有经典 Markov 链相同的时间复杂度.

3.2.3 带递归的 II 型量子 Markov 链模型

为了描述带过程调用(procedure call)的量子程序,文[38]定义了 II 型递归量子 Markov 链.该模型结合了经典递归 Markov 链[39]和 II 型量子 Markov 链的特征,其表述能力等价于将转移函数的取值由概率替换为超算子的概率下推自动机.根据不同的应用场景,文[38]考虑了三种 II 型递归量子 Markov 链的可达性问题：经典状态可达性(classical state reachability)、量子状态可达性(quantum state reachability)和量子状态空间可达性(quantum subspace reachability).有趣的是,这三种可达性问题都可以转化成计算某个多项式方程组的最小解的问题,其中每个方程的系数都是量子状态空间上的超算子.对于 II 型递归量子 Markov 链的一个重要的子类,即线性 Markov 链,三种可达性问题都可以在多项式时间内解决.对于一般的情形,量子状态空间可达性问题也存在多项式时间算法；而程序分析的牛顿迭代法[40]则可用来有效的近似描述前面两种可达性所对应的超算子.

4 量子通信并发系统验证

量子通信并发系统研究的主要动因有二.首先,普遍认为量子通信的实用化会比量子计算机早很多,而量子通信并发系统的研究可以为量子通信协议的设计、分析与验证提供理论基础和工具.其次,大型的量子计算机还非常遥远,但最近美国总统科学技术办公室发布的量子信息文件预计几十个量子比特、可供早期量子计算机科学研究的系统可望在 5 年内出现[41].因此,一个自然的想法是如何将多个小型量子计算机组成分布式系统以

解决经典计算机所不能解决的问题.实际上,十多年前物理学家就已经开始研究分布式量子计算,而并发问题在分布式量子程序设计中是不可避免的.

为了形式化地描述量子通信并发系统,法国国家科学研究中心莱布尼茨实验室的 Jorrand 和 Lalire 首先提出了一种量子进程代数 QAlg[42],并为其建立了概率分支互模拟语义[43].但该方法只是经典进程代数的简单扩充,甚至无法避免出现违背量子不可克隆原理等基本原则的进程.英国诺丁汉大学的 Gay 和 Nagarajan 提出了一种量子语言 CQP[44],融合了 Pi 演算的通信原语和量子力学的酉变换和测量.CQP 的一个显著特点是提供了一个类型系统以保证量子进程的物理可实现性.然而,这些方法都无法有效刻画含纠缠的量子输入和输出.基于经典 CCS 理论,文[45], [46]提出了量子进程代数 qCCS.通过引入量子自由变量,在语法的层面上保证了所有量子进程都是物理可实现的.这样做既可以刻画量子态不可任意克隆等量子系统的特征,又可以避免 CQP 中定义的繁杂的类型系统,从而使得并发系统语义的研究更为简洁和方便.为了对进程中出现的量子自由变量进行赋值,文中定义了量子上下文的概念,用来精确表述含纠缠的量子输入和输出.

4.1 量子进程的互模拟

互模拟关系是进程代数的核心概念,刻画了系统的行为等价性.直观的讲,如果两个系统中任何一个的任何行为都可以被另一个系统模拟,并且这种模拟可以递归的一直进行下去,它们就称为是可互模拟的.对于经典系统,互模拟具有某种同余性质,即任意进程构造符都保持互模拟关系.这一性质保证了在一个复杂系统中,将任何一个子进程替换为与其互模拟的另一个进程并不会改变整个系统的任何可观测行为.这对于复杂系统的模块化设计,以及验证通信系统的通用可组合安全性(Universally composable security)具有非常重要的意义.

对量子通信并发系统,由于纠缠的非定域性特征,复合系统中某一部分的行为可能会改变其他部分的状态,这使得寻找能够在并行复合下保持的互模拟关系变得非常困难.QPAlg 和 CQP 中已有的互模拟都不是同余关系.通过引入超算子作用下的闭条件(closed under super-operator application),冯元等人[47], [48]为 qCCS 中的进程定义了一种真正意义上同余的互模拟关系,并解决了在这种关系下 qCCS 的递归方程解的唯一性问题.不久后,邓玉欣等[49]将这一思想简化,提出了类似于 Pi 演算中的开互模拟关系(open bisimulation).文[50]进一步把符号互模拟[51]的概念推广到 qCCS,从而给出了判定量子进程互模拟的有效算法.

4.2 近似互模拟

进程的互模拟关系是一种“刚性”的概念,系统各部分概率的细微改变都可能破坏这种脆弱的关系.考虑到在量子计算与量子通信中,噪音都是一个不可忽略的因素,文[46], [48]将经典进程代数的近似互模拟概念推广到 qCCS,并由此出发定义了量子进程空间上的伪度量(psudo-metric),这一度量在不同进程构造符下保持单调(nonexpansiveness).单调性是互模拟关系的同余性的推广,它保证了在复杂系统中,将子进程P替换为另一个进程P'时,整个系统的行为改变不超过P和P'之间的距离.由于全体量子操作构成一个连续谱,因此量子进程之间的近似互模拟显得更加直观和自然.作为应用,文[46], [48]证明了由任何完备的量子逻辑门集合生成的量子进程在所有量子进程的集合中关于该伪度量是稠密的,从而说明了 qCCS 框架下逻辑门的完备性和量子电路框架下量子逻辑门的完备性是一致的.

4.3 基于概率分布的互模拟

以上所讨论的互模拟关系都是基于进程格局(configuration,表现为一个进程－量子状态对)的,在很多情况下对于进程的区分度太强.Kubota 等[52]为 qCCS 的量子测量提供了一种额外的语义,即当测量产生的概率分布中所有格局都具有相同的可见动作时,其语义描述为合并所有测量结果所对应的超算子(因此不会产生任何概率分支).这样扩充的 qCCS 可以有效解决概率行为都是由测量所引起的量子进程的互模拟问题.然而,正如在量子密钥分发 BB84 协议中那样,如果概率行为是系统本身的随机性引起的,这种方法就不适用了.另外,判断一个量子测量是否引起不同可见行为的量子格局本身就是一个困难的问题,有时候甚至依赖于后续的环境输入.为解决这些困难,文[53]采用了一种不同的处理方式,即保持 qCCS 原有的量子测量语义,但把基于格局的互模拟

推广到格局的概率分布上.判断两个概率分布(不管其是由测量引起的,还是系统本身的随机性引起的)是否互模拟时,相互模拟的量子进程对应的量子状态总是组合在一起进行考虑.类似的思想也用来讨论量子 Lambda 演算中项(Lambda terms)的等价性[54].文[53]还把近似互模拟推广到格局的概率分布上,用以检验量子通讯协议的复杂安全性特征,这些特性无法在以往的互模拟概念下得到证明.作为应用,文[53]证明了 BB84 协议在接收－重发攻击下,与理想系统的互模拟距离随着安全参数(Security Parameter,对于 BB84 协议来说就是所消耗的 qubit 个数)的增加按指数级降低,从而说明了该协议的正确性和渐进安全性.

4.4 互模拟检测工具

可以预见,量子通信并发系统的研究将对复杂量子协议的设计、分析和验证起到积极的推动作用.Ardeshir-Larijani 等[55], [56]研究并实现了一种简单的量子进程等价性检测,可用来验证 Teleportation 等函数式量子程序和协议的正确性,即对给定的量子输入状态,任何不确定性或并发分支都产生同样的量子输出状态.Kubota 等开发出基于 qCCS 的软件工具[52],用于检测量子进程的互模拟.英国 Glasgow 大学的研究小组正在着手 qCCS 符号互模拟的软件实现.最近,我们和新加坡南洋理工大学刘扬教授研究组合作开发了工具 Qubet (Bisimulation Checking and Emulation Tool for Quantum Communication Systems),利用 qCCS 对量子通信系统进行仿真和互模拟检测,并提供了易扩展的架构和用户友好的图形化界面.

5 展望

需要指出的是,本文所讨论的量子程序和量子协议都遵循“程序的数据流是量子的,但控制流仍然是经典的”这样一个基本的思路.Selinger 将这个思路总结为一句口号:“量子数据,经典控制(quantum data,classical control)”.这个思路的提出是十分自然的,也是相对来说比较容易实现的,只需要在经典的程序设计语言中增加对于量子数据的操作,如酉变换和量子测量等.

但是,人们也认识到“量子数据,经典控制”的思想并不能最为有效地发挥量子计算特有的优势.Altenkirch 与 Grattage[57]提出了一个带量子控制流的函数式量子程序设计语言 QML,并在其后一系列论文中系统地发展了相应的理论,甚至实现了编译器.但是,现在看来这个理论并没有完全成功,特别是一般程序的量子叠加的严格语义这一最为核心的问题并没有得到明确的定义.在没有递归(也没有循环)的情况下,如果整个程序中测量都可以移到最后一步,带量子控制流的程序的语义比较容易定义,本质上就是量子的 multiplexor.如果中间过程有量子测量,这个问题就变得非常困难了.Badescu 与 Panangaden 在[58]中对这个问题有一些有趣的讨论.直到最近,受到量子随机游走(quantum random walks)的启发,书[59]找到了一种真正能实现量子控制流的量子程序设计模型.书[59]第 6 章通过引入算子值函数的卫式复合,进一步定义超算子的卫式复合,从而由量子程序的半经典语义导出纯量子语义,并解决了这一问题.正如[58]中指出的,带量子控制流的量子递归是一个非常困难的问题,需要全新的思想.书[59]第 6 章提出采用二次量子化方法(secondquantization)解决了这个问题, 定义了波色子(对称的)与费米子(反对称)量子递归模式.必须承认的是,我们对于量子递归还远没有完全彻底地理解,这也是量子程序理论和方法需要进一步研究的重要问题.

6 参考文献

[1] D. Wecker and K. M. Svore, “LIQUi|>: A Software Design Architecture and Domain-Specific Language for Quantum Computing,” *arXiv.org*, vol. quant-ph. p. arXiv:1402.4467, 19-Feb-2014.

[2] T. Häner, D. S. Steiger, K. Svore, and M. Troyer, “A Software Methodology for Compiling Quantum Programs,” *arXiv.org*, vol. cs.PL. p. arXiv:1604.01401, 06-Apr-2016.

[3] “The Q# Programming Language,” *https://docs.microsoft.com/en-us/quantum/quantum-qr-intro?view=qsharp-preview*.

[4] A. S. Green, P. L. Lumsdaine, N. J. Ross, P. Selinger, and B. Valiron, “Quipper - a scalable quantum programming language,” presented at the PLDI, 2013.

[5] A. JavadiAbhari, S. Patil, D. Kudrow, J. Heckey, A. Lvov, F. T. Chong, and M. Martonosi, “ScaffCC: Scalable Compilation and Analysis of Quantum Programs,” *arXiv.org*, vol. quant-ph. pp. arXiv:1507.01902–17, 08-Jul-2015.

[6] A. Lapets and M. Roetteler, “Abstract resource cost derivation for logical quantum circuit descriptions,” presented at the Proceedings of the 1st annual workshop on Functional programming concepts in domain-specific languages - FPCDSL '13, New York, USA, 2013, pp. 35–42.

[7] A. Lapets, M. P. da Silva, M. Thome, A. Adler, J. Beal, and M. Roetteler, “QuaFL: a typed DSL for quantum programming,” presented at the Proceedings of the 1st annual workshop on Functional programming concepts in domain-specific languages - FPCDSL '13, New York, USA, 2013, pp. 19–26.

[8] M. Ying and Y. Feng, “An Algebraic Language for Distributed Quantum Computing,” *IEEE Transactions on Computers*, vol. 58, no. 6, pp. 728–743, 2009.

[9] M. Ying and Y. Feng, “A Flowchart Language for Quantum Programming,” *IEEE Transactions on Software Engineering*, vol. 37, no. 4, pp. 466–485, 2011.

[10] M. Ying and Y. Feng, “Quantum loop programs,” *Acta Informatica*, vol. 47, no. 4, pp. 221–250, Jun. 2010.

[11] M. Ying, N. Yu, Y. Feng, and R. Duan, “Verification of quantum programs,” *Science of Computer Programming*, vol. 78, no. 9, pp. 1679–1700, Sep. 2013.

[12] S. Perdrix, “Quantum Entanglement Analysis Based on Abstract Interpretation,” presented at the SAS, 2008, pp. 270–282.

[13] P. Jorrand and S. Perdrix, “Abstract Interpretation Techniques for Quantum Computation,” in *Semantic Techniques in Quantum Computation*, no. 6, S. Gay and I. Mackie, Eds. Cambridge: Cambridge University Press, 2009, pp. 206–234.

[14] K. Honda, “Analysis of Quantum Entanglement in Quantum Programs using Stabilizer Formalism,” *Electronic Proceedings in Theoretical Computer Science*, vol. 195, no. 1, pp. 262–272, Nov. 2015.

[15] O. Brunet and P. Jorrand, “Dynamic quantum logic for quantum programs,” *International Journal of Quantum Information*, vol. 2, no. 1, pp. 45–54, Nov. 2011.

[16] A. Baltag and S. Smets, “LQP: the dynamic logic of quantum information,” *Math. Struct. in Comp. Science*, vol. 16, no. 3, pp. 491–525, Jul. 2006.

[17] Y. Feng, R. Duan, Z. Ji, and M. Ying, “Proof rules for the correctness of quantum programs,” *Theoretical Computer Science*, vol. 386, no. 1, pp. 151–166, Oct. 2007.

[18] R. Chadha, P. Mateus, and A. Sernadas, “Reasoning About Imperative Quantum Programs,” *Electr. Notes Theor. Comput. Sci.*, 2006.

[19] Y. Kakutani, “A Logic for Formal Verification of Quantum Programs,” presented at the ASIAN, 2009.

[20] M. Ying, “Floyd-Hoare logic for quantum programs,” *TOPLAS*, vol. 33, no. 6, pp. 1–49, Dec. 2011.

[21] T. Liu, Y. Li, S. Wang, M. Ying, and N. Zhan, “A Theorem Prover for Quantum Hoare Logic and Its Applications,” *arXiv.org*, vol. cs.LO. p. arXiv:1601.03835, 15-Jan-2016.

[22] I. Hasuo and N. Hoshino, “Semantics of higher-order quantum computation via geometry of interaction.,” *Ann. Pure Appl. Logic*, 2017.

[23] M. Pagani, P. Selinger, and B. Valiron, “Applying quantitative semantics to higher-order quantum computing,” presented at the POPL, 2014.

[24] B. Jacobs, “On Block Structures in Quantum Computation,” *Electronic Notes in Theoretical Computer Science*, vol. 298, pp. 233–255, Nov. 2013.

[25] S. Staton, “Algebraic Effects, Linearity, and Quantum Programming Languages,” presented at the the 42nd Annual ACM SIGPLAN-SIGACT Symposium, New York, New York, USA, 2015, pp. 395–406.

[26] E. D'Hondt and P. Panangaden, “Quantum weakest preconditions.,” *Math. Struct. in Comp. Science*, 2006.

[27] M. Ying, R. Duan, Y. Feng, and Z. Ji, “Predicate Transformer Semantics of Quantum Programs,” in *Semantic Techniques in*

Quantum Computation, no. 8, S. Gay and I. Mackie, Eds. Cambridge: Cambridge University Press, 2009, pp. 311–360.

[28] S. Ying, Y. Feng, N. Yu, and M. Ying, "Reachability Probabilities of Quantum Markov Chains," presented at the CONCUR, 2013, pp. 334–348.

[29] N. Yu and M. Ying, "Reachability and Termination Analysis of Concurrent Quantum Programs," presented at the CONCUR, 2012, pp. 69–83.

[30] Y. Li and M. Ying, "(Un)deciddable Problems about Reachability of Quantum Systems," presented at the CONCUR, 2014, pp. 482–496.

[31] Y. Feng, N. Yu, and M. Ying, "Model checking quantum Markov chains," *Journal of Computer and System Sciences*, vol. 79, no. 7, pp. 1181–1198, 2013.

[32] M. Y. Vardi, "Automatic Verification of Probabilistic Concurrent Finite-State Programs," presented at the FOCS, 1985.

[33] J. Diestel and J. Uhl, *Vector Measures*, vol. 15. Providence, Rhode Island: American Mathematical Society, 1977.

[34] Y. Feng, E. M. Hahn, A. Turrini, and L. Zhang, "QPMC: A Model Checker for Quantum Programs and Protocols," presented at the FM, 2015, pp. 265–272.

[35] M. Z. Kwiatkowska, G. Norman, and D. Parker, "PRISM 4.0 - Verification of Probabilistic Real-Time Systems.," presented at the CAV, 2011.

[36] L. Anticoli, C. Piazza, L. Taglialegne, and P. Zuliani, "Towards Quantum Programs Verification - From Quipper Circuits to QPMC," presented at the RC, 2016.

[37] Y. Feng, E. M. Hahn, A. Turrini, and S. Ying, "Model Checking Omega-regular Properties for Quantum Markov Chains .," presented at the CONCUR, 2017, pp. 35:1–35:16.

[38] Y. Feng, N. Yu, and M. Ying, "Reachability Analysis of Recursive Quantum Markov Chains," presented at the MFCS, 2013, pp. 385–396.

[39] K. Etessami and M. Yannakakis, "Recursive Markov chains, stochastic grammars, and monotone systems of nonlinear equations.," *J. ACM*, p. 1, 2009.

[40] J. Esparza, S. Kiefer, and M. Luttenberger, "Newtonian program analysis," *J. ACM*, vol. 57, no. 6, pp. 1–47, Oct. 2010.

[41] "Advancing Quantum Information Science: National Challenges and Opportunities," https://www.whitehouse.gov/sites/whitehouse.gov/files/images/Quantum_Info_Sci_Report_2016_07_22%20final.pdf.

[42] P. Jorrand and M. Lalire, "Toward a quantum process algebra," presented at the Proceedings of the first conference on computing frontiers on Computing frontiers, New York, USA, 2004, pp. 111–119.

[43] M. Lalire, "Relations among quantum processes: bisimilarity and congruence," *Math. Struct. in Comp. Science*, vol. 16, no. 3, p. 407, Jul. 2006.

[44] S. J. Gay and R. Nagarajan, "Communicating quantum processes.," presented at the POPL, 2005.

[45] Y. Feng, R. Duan, Z. Ji, and M. Ying, "Probabilistic bisimulations for quantum processes," *Information and Computation*, vol. 205, no. 11, pp. 1608–1639, Nov. 2007.

[46] M. Ying, Y. Feng, R. Duan, and Z. Ji, "An algebra of quantum processes," *ACM Transactions on Computational Logic (TOCL)*, vol. 10, no. 3, pp. 19–36, Apr. 2009.

[47] R. Duan, Y. Feng, and M. Ying, "Bisimulation for quantum processes," presented at the POPL, New York, USA, 2011, pp. 523–534.

[48] Y. Feng, R. Duan, and M. Ying, "Bisimulation for Quantum Processes," *TOPLAS*, vol. 34, no. 4, pp. 1–43, Dec. 2012.

[49] Y. Deng and Y. Feng, "Open Bisimulation for Quantum Processes," presented at the IFIP TCS, 2012.

[50] Y. Feng, Y. Deng, and M. Ying, "Symbolic Bisimulation for Quantum Processes," *ACM Trans. Comput. Logic*, vol. 15, no. 2, pp. 1–32, Apr. 2014.

[51] M. Hennessy and H. Lin, "Symbolic bisimulations," *Theoretical Computer Science*, vol. 138, no. 2, pp. 353–389, 1995.

[52] T. Kubota, Y. Kakutani, G. Kato, Y. Kawano, and H. Sakurada, “Semi-automated verification of security proofs of quantum cryptographic protocols,” *Journal of Symbolic Computation*, vol. 73, pp. 192–220, Mar. 2016.

[53] Y. Feng and M. Ying, “Toward Automatic Verification of Quantum Cryptographic Protocols,” presented at the CONCUR, 2015, pp. 441–455.

[54] Y. Deng, Y. Feng, and U. D. Lago, “On Coinduction and Quantum Lambda Calculi,” presented at the CONCUR, 2015, pp. 427–440.

[55] E. Ardeshir-Larijani, S. J. Gay, and R. Nagarajan, “Equivalence Checking of Quantum Protocols,” presented at the TACAS, Berlin, Heidelberg, 2013, vol. 7795, no. 33, pp. 478–492.

[56] E. Ardeshir-Larijani, S. J. Gay, and R. Nagarajan, “Verification of Concurrent Quantum Protocols by Equivalence Checking,” presented at the TACAS, Berlin, Heidelberg, 2014, vol. 8413, no. 42, pp. 500–514.

[57] T. Altenkirch and J. Grattage, “A Functional Quantum Programming Language,” presented at the LICS, 2005.

[58] C. Bădescu and P. Panangaden, “Quantum Alternation: Prospects and Problems,” *Electronic Proceedings in Theoretical Computer Science*, vol. 195, no. 1, pp. 33–42, Nov. 2015.

[59] M. Ying, *Foundations of Quantum Programming*. Elsevier Science, 2016.