

简洁非交互零知识证明综述^{*}

李威翰¹, 张宗洋¹, 周子博¹, 邓 焱²

1. 北京航空航天大学 网络空间安全学院, 北京 100191
2. 中国科学院 信息工程研究所 信息安全部国家重点实验室, 北京 100093
通信作者: 张宗洋, E-mail: zongyangzhang@buaa.edu.cn

摘要: 区块链、隐私计算、人工智能等技术的快速发展, 极大地推动了对零知识证明尤其是简洁非交互零知识证明的研究。本文从通用构造方法、底层技术原理、协议性能表现、安全性等角度深入研究了现有的简洁非交互零知识证明。首先, 总结了简洁非交互零知识证明的通用构造方法。其次, 分别基于信息论安全证明和底层关键技术对现有的简洁非交互零知识证明进行分类并提炼了核心思路, 深入分析了典型协议的实现原理。再次, 辨析了各类协议的性能表现, 探讨其安全性并指出其适用场景。最后, 总结了简洁非交互零知识证明的研究热点和发展方向。

关键词: 零知识证明; 简洁非交互; 电路可满足性; 信息论安全证明; 承诺

中图分类号: TP309.7 **文献标识码:** A **DOI:** 10.13868/j.cnki.jcr.000525

中文引用格式: 李威翰, 张宗洋, 周子博, 邓焱. 简洁非交互零知识证明综述[J]. 密码学报, 2022, 9(3): 379–447. [DOI: 10.13868/j.cnki.jcr.000525]

英文引用格式: LI W H, ZHANG Z Y, ZHOU Z B, DENG Y. An overview on succinct non-interactive zero-knowledge proofs[J]. Journal of Cryptologic Research, 2022, 9(3): 379–447. [DOI: 10.13868/j.cnki.jcr.000525]

An Overview on Succinct Non-interactive Zero-knowledge Proofs

LI Wei-Han¹, ZHANG Zong-Yang¹, ZHOU Zi-Bo¹, DENG Yi²

1. School of Cyber Science and Technology, Beihang University, Beijing 100191, China
2. State Key Laboratory of Information Security, Institute of Information Engineering, Chinese Academy of Sciences, Beijing 100093, China

Corresponding author: ZHANG Zong-Yang, E-mail: zongyangzhang@buaa.edu.cn

Abstract: The rapid development of blockchain, privacy computation, artificial intelligence and other technologies has greatly promoted the research of zero-knowledge proofs, especially the succinct non-interactive zero-knowledge proofs. This paper studies current succinct non-interactive zero-knowledge proofs from the perspectives of general designing mechanism, underlying key techniques, performance

* 基金项目: 国家重点研发计划 (2021YFB3100400); 北京市自然科学基金 (M22038, M21033, 4202037); 国家自然科学基金 (61972017, 61972018, 72031001, 61932019, 61772521, 61772522); 中国科学院前沿科学重点研究计划 (QYZDB-SSW-SYS035); 云南省区块链应用技术重点实验室 (培育) 开放课题 (YNB202101); 中央高校基本科研业务费 (YWF-22-L-1039)

Foundation: National Key Research and Development Program of China (2021YFB3100400); Natural Science Foundation of Beijing Municipality (M22038, M21033, 4202037); National Natural Science Foundation of China (61972017, 61972018, 72031001, 61932019, 61772521, 61772522); Key Research Program of Frontier Sciences, CAS (QYZDB-SSW-SYS035); Open Project of Yunnan Key Laboratory of Blockchain Application Technology (YNB202101); the Fundamental Research Funds for the Central Universities of China (YWF-22-L-1039)

and security. First, a general mechanism of designing succinct non-interactive zero-knowledge proofs is given. Then, the existing succinct non-interactive zero-knowledge proofs are classified based on information-theoretical proofs and underlying key techniques. In addition, the core ideas and implementation principles of typical protocols are analyzed in a good technical depth. This paper then analyzes the performance and security of various typical protocols and points out their application scenarios. Finally, some future research directions of succinct non-interactive zero-knowledge proofs are summarized.

Key words: zero-knowledge proof; succinct and non-interactive; circuit satisfiability; information-theoretic proof; commitment

1 引言

零知识证明由 Goldwasser、Micali 和 Rackoff^[1] 提出, 它是运行在证明者和验证者之间的一种两方密码协议, 可用于进行成员归属命题证明或知识证明. 零知识证明具有如下三个性质:

- (1) 完备性, 用于描述协议本身的正确性. 给定某个陈述的有效证据, 如果证明者和验证者均诚实运行协议, 那么证明者能使验证者相信该陈述的正确性.
- (2) 可靠性. 可靠性用于保护诚实验证者的利益, 使其免于恶意证明者的欺骗.
- (3) 零知识性. 零知识性是指证明者能向验证者证明某个陈述的正确性而不泄露除正确性以外的其他任何信息.

零知识证明的三个性质使其具备了信任建立和隐私保护的功能, 具有良好的应用前景. 它不仅可以用子公钥加密^[2]、签名^[3]、身份认证^[4] 等经典密码学领域, 也与区块链^[5]、隐私计算^[6] 等新兴热门技术的信任与隐私需求高度契合. 例如, 在区块链匿名密码货币 (如 Zcash¹) 中, 零知识证明可在不泄露用户地址及金额的同时证明某笔未支付资金的拥有权^[7]; 在区块链扩容 (系列 zk rollup 方案, 如 zkSync²) 中, 链上的复杂计算需要转移到链下, 而零知识证明可保障该过程的数据有效性; 在匿名密码认证^[8-10] 中, 零知识证明可在不泄露用户私钥的同时证明拥有私钥, 从而实现匿名身份认证.

虽然针对通用 NP 语言均可构造零知识证明^[11], 但其落地应用仍存在若干问题. 仅以区块链为例, 由于区块链往往具有低存储的需求且建立网络实时通信的开销较高, 适配于区块链的零知识证明通常需要具有简洁性和非交互的特点, 其中简洁性指证明的通信复杂度与陈述规模成亚线性关系, 非交互指证明者只需向验证者发送 1 轮消息即可完成证明. 对于后者, 非交互可分别通过公共参考串模型 (common reference string model, CRS)^[12] 和随机预言模型 (random oracle model, ROM)^[13, 14] 实现. 然而对于简洁性, 尽管在 1992 年 Kilian^[15] 基于概率可验证证明 (probabilistic checkable proof, PCP)^[16, 17] 构造了简洁的交互式零知识证明, 而且 Micali^[18] 基于 ROM 将上述交互式证明转化为非交互证明, 但仅限于理论研究并难以实现.

直至基于二次算术程序 (quadratic arithmetic program, QAP)/线性 PCP (Linear-PCP) 的系列证明出现后, 零知识证明才得以落地实现. 该类零知识证明由 Gennaro 等人^[19] 首次提出, 其中 QAP 用于实现对待证明陈述的高效归约, Linear-PCP 用于构造高效信息论安全证明 (即对于无穷算力的恶意敌手仍具有可靠性的证明). 该类零知识证明的通信复杂度为常数个群元素, 且验证复杂度仅与陈述的公共输入输出规模成线性关系. 除了理论上的研究和优化之外, 该类零知识证明在实际隐私保护应用中也大放异彩, 例如基于 Pinocchio^[20] 的密码货币 Pinocchio coin^[21], 基于 Ben-Sasson 等人协议^[22] 的密码货币 Zcash, 基于 Plonk^[23] 用于解决以太坊扩容问题的系列 zk rollup 方案等.

然而, 即使是最高效的基于 QAP/Linear-PCP 的简洁非交互零知识证明也存在若干问题. 在性能层面, 对于每个待证明陈述都需进行较长时间的预处理, 同时协议的实际证明开销较大. 在安全性方面, 协议所基于的难题假设是不可证伪 (non-falsifiable) 假设^[24, 25], 假设本身的安全性难以完全保障; 并且为实现

¹<https://z.cash/>.

²<https://zkSync.io/>.

非交互和保障可靠性, 协议需要可信初始化 (trusted setup), 即安全生成的 CRS, 而这在去中心化的区块链中难以实现.

近年来的研究致力于从不同角度解决上述问题. 为解决证明生成效率不高的问题, 出现了实际证明速率较快的基于 DEIP 的零知识证明^[26–29]; 为解决底层假设通用性不足的问题, 出现了基于离散对数假设的零知识证明^[30–32] 和仅需单向函数存在的基于 MPC-in-the-Head 的零知识证明^[33–36]; 为解决初始化阶段可信需求高的问题, 出现了以削弱 CRS 模型下的可信初始化设置为目标的抗颠覆的零知识证明^[37–39] 和 CRS 可更新的零知识证明^[23, 40–42], 也出现了一系列不需预处理和可信初始化, 即启动阶段系统参数可独立公开生成的零知识证明 (如 STARK^[43]、Bulletproofs^[31]、Spartan^[27]、Ligero^[35] 等).

简洁非交互零知识证明虽然在多个领域具有热门和广泛的应用前景, 但一方面零知识证明种类繁多, 各类协议基于的原理驳杂, 性能侧重点也各不相同, 目前在一定程度上存在技术壁垒; 另一方面国内外针对简洁非交互零知识证明的相关综述较少, 缺乏系统的总结梳理 (见第 1.2 节), 因此有必要从通用构造方法、底层关键技术、协议性能表现、典型协议分析等角度, 对目前的简洁非交互零知识证明进行介绍, 为该领域的理论研究和应用实现提供一定参考.

1.1 本文贡献

本文详细梳理了现有的简洁非交互零知识证明, 主要贡献如下.

- (1) 总结了简洁非交互零知识证明的通用构造方法 (见图 1). 构造方法分为四步, 分别是将待证明陈述转换为电路可满足问题 (C-SAT 问题)、将电路可满足问题转换为易证明的语言 (此步可省略, 用虚线表示)、针对易证明的语言构造信息论安全证明和利用密码编译器将信息论安全证明转换为简洁非交互零知识证明.



图 1 简洁非交互零知识证明的通用构造方法

Figure 1 General method to construct succinct non-interactive zero-knowledge proof

- (2) 基于上述通用构造方法, 分类研究了现有的简洁非交互零知识证明. 根据信息论安全证明的种类, 主要分为基于 PCP、Linear-PCP、交互式 PCP (interacitve PCP, IPCP) 和交互式谕示证明 (interactive oracle proof, IOP) 的零知识证明; 根据密码编译器应用的底层关键技术, 主要分为基于 QAP、双向高效的交互式证明 (doubly efficient interactive proof, DEIP)、内积论证 (inner product argument, IPA) 和 MPC-in-the-Head 的零知识证明. 表 1 分别从信息论安全证明和密码编译器应用的底层关键技术两个角度, 总结了零知识证明, 涵盖了待证明陈述表示形式、协议性能、启动阶段参数能否公开生成等相关信息.
- (3) 总结了简洁非交互零知识证明的性能评价标准, 包括底层难题假设的通用性, 启动阶段系统参数能否公开生成, 证明、验证和通信复杂度, 是否抗量子等内容.
- (4) 分析了未来简洁非交互零知识证明的研究热点和发展方向. 基于近年来简洁非交互零知识性证明的最新研究进展, 从通用构造、性能、安全性等方面给出若干可能的未来发展方向.

1.2 相关工作

简洁非交互零知识证明是实现区块链、隐私计算等场景下隐私保护的重要技术, 近几年对零知识证明尤其是简洁非交互零知识证明的综述研究主要包括以下内容.

Goldreich^[44] 梳理了零知识证明二十余年的发展情况, 介绍了交互式证明系统与论证、计算不可区分、单向函数等零知识证明涉及的核心概念, 探讨了零知识证明的标准定义及变种, 如全局与黑盒模拟、诚实验证者零知识、计算与统计零知识、PPT 与期望多项式时间的模拟器等, 研究了零知识证明的证明能力, 并讨论了组合零知识证明、知识证明、非交互零知识证明等变种. Li 和 McMillin^[45] 介绍了零知识证明的背景、重要概念及组合零知识证明等, 并详细给出了针对若干具体 NP 问题的零知识证明, 包括三染色问题、图同构问题、哈密尔顿回路问题、背包问题、可满足性问题等. Mohr^[46] 研究了非交互零知识证

表 1 基于不同角度分类的(简洁)非交互零知识证明总结

Table 1 Summary of (succinct) non-interactive zero-knowledge proof from different perspectives

信息论 安全证明	方案	时间	待证明陈述 表示形式	证明复杂度	通信复杂度	验证复杂度	系统参数 生成方式	关键技术
PCP	Micali94 [18]	1994	算术电路	/	$O(\log C)$	/	公开	默克尔树; 抗碰撞哈希函数
	ZKBoo [33]	2016	算术/布尔 电路	$O(C) \mathbb{F}_o$	$O(C) \mathbb{F}$	$O(C) \mathbb{F}_o$	公开	
	ZKB++ [34]	2017	算术/布尔 电路	$O(C) \mathbb{F}_o$	$O(C) \mathbb{F}$	$O(C) \mathbb{F}_o$	公开	
	KKW18 [47]	2018	布尔电路	$O(C) \mathbb{F}_o$	$O(C) \mathbb{F}$	$O(C) \mathbb{F}_o$	公开	
IPCP	Ligero [35]	2017	算术/布尔 电路	$O(C \log^2 C) \mathbb{F}_o$	$O(\sqrt{ C }) \mathbb{F}$	$O(C) \mathbb{F}_o$	公开	MPC-in -the-Head
	Ligero++ [36]	2020	算术/布尔 电路	$O(C \log^2 C) \mathbb{F}_o$	$O(\log C) \mathbb{F}$	$O(C) \mathbb{F}_o$	公开	
	BooLigero [48]	2021	布尔电路	$O(C \log^2 C) \mathbb{F}_o$ $\sim \frac{\mathcal{O}(C ^{1/2})}{(\log \mathbb{F})^{1/4}} \mathbb{F}$	$O(\log \mathbb{F})^{1/2} \mathbb{F}$	$O(C) \mathbb{F}_o$	公开	
IOP	Limbo [49]	2021	算术/布尔 电路	$O(C) \mathbb{F}_o$	$O(C) \mathbb{F}$	$O(C) \mathbb{F}_o$	公开	
	Aurora [50]	2019	一阶约束 系统	$O(n \log n) \mathbb{F}_o$	$O(\log n) \mathbb{F}$	$O(n) \mathbb{F}_o$	公开	里德-所罗门码; 低度检查; 单变量求和验证
	Stark [43]	2019	对数空间 可计算电路	$O(T \log T) \mathbb{F}_o$	$O(\log T) \mathbb{F}$	$O(\log T) \mathbb{F}_o$	公开	里德-所罗门码; 低度检查
	ZKvSQL [51]	2017	分层算术 电路	$O(C \log g) \mathbb{G}_o$	$O(d \log g + d \log \mathbf{w}) \mathbb{G}$ $O(\log \mathbf{w}) P$	$O(d \log C) \mathbb{G}_o$	私密	
	Hyrax [32]	2018	数据并行 电路	$O(C + dg \log g) \mathbb{G}_o$	$O(\sqrt{w} + d \log N_g) \mathbb{G}$	$O(d \log(N_g) + dg + \sqrt{ \mathbf{w} }) \mathbb{G}_o$	公开	
	Libra [26]	2019	分层算术 电路	$O(C) + \text{in} \mathbb{F}_o$ $O(\mathbf{w} + d) \mathbb{G}_o$	$O(d \log C) \mathbb{F}$ $O(\log \mathbf{w}) \mathbb{G}$	$O(d \log C) \mathbb{F}_o$ $O(\log \mathbf{w}) P$	私密	
	Virgo [28]	2020	分层算术 电路	$O(C) \mathbb{F}_o$	$O(d \log C) \mathbb{F}$	$O(d \log C) \mathbb{F}_o$	公开	求和验证协议
	Spartan [27]	2020	一阶约束 系统	$O(n) \mathbb{G}_o$ 或 $O(n \log n) \mathbb{F}_o$	$O(\sqrt{n}) \mathbb{G}$ 或 $O(\log^2 n) \mathbb{F}$	$O(\sqrt{n}) \mathbb{G}_o$ 或 $O(\log^2 n) \mathbb{F}_o$	公开	
	Virgo++ [29]	2021	算术电路	$O(C) \mathbb{F}_o$	$\min(O(C , O(d \log C + d^2)) \mathbb{F}$	$O(d \log C + d^2) \mathbb{G}_o$	公开	
Linear -PCP	GGPR13 [19]	2013	算术电路	$O(C \log C) \mathbb{F}_o$ $O(C) \mathbb{G}_o$	9 G	$O(\text{io}) E$ $14 P$	私密	
	Pinocchio [20]	2013	算术电路	$O(C \log C) \mathbb{F}_o$ $O(C) \mathbb{G}_o$	8 G	$O(\text{io}) E$ $11 P$	私密	二次算术程序; 双线性配对
	Groth16 [52]	2016	算术电路	$O(C) \mathbb{G}_o$	3 G	$O(\text{io}) E$ $4 P$	私密	
	GKMMM18 [41]	2018	算术电路	$O(C) \mathbb{G}_o$	3 G	$O(\text{io}) E$ $5 P$	私密	
?	BCCGP16 [30]	2016	算术电路	$O(C) \mathbb{F}_o$ $O(C_M) \mathbb{G}_o$	$O(\log C_M) \mathbb{G}$ $O(\log C_M) \mathbb{F}$	$O(C) \mathbb{G}_o$ $O(\log C_M) \mathbb{F}_o$	公开	
	Bulletproofs [31]	2018	算术电路	$O(C) \mathbb{F}_o$ $O(C_M) \mathbb{G}_o$	$O(\log C_M) \mathbb{G}$ $5 F$	$O(C) \mathbb{G}_o$	公开	内积论证
	HKR19 [53]	2019	算术电路	$O(C) \mathbb{F}_o$ $O(C_M) \mathbb{G}_o$	$O(\log C_M) \mathbb{G}$ $2 F$	$O(C) \mathbb{G}_o$	公开	
DRZ20 [54]	DRZ20 [54]	2020	算术电路	$O(C) \mathbb{F}_o$ $O(C_M) \mathbb{G}_o$	$O(\log C_M) \mathbb{G}$ $O(\log C_M) \mathbb{F}$	$O(\log C_M) \mathbb{G}_o$ $O(\log C_M) P$	私密	

¹ 上述零知识证明均是知识论证(见定义 14),涉及的关键技术定义详见第 2.2 节和各章的定义及概念部分。² \mathbb{F} 表示域元素, \mathbb{F}_o 表示域上操作; \mathbb{G} 表示群元素, \mathbb{G}_o 表示群上操作, 主要包括群幂运算 E 和群上乘法。一般而言, 群上操作尤其是群幂运算比域上操作开销大得多。 P 表示双线性群上的配对运算。 $|C|$ 表示电路规模, $|C_M|$ 表示电路中乘法门的数目, d 表示电路深度, g 表示电路宽度, $|\text{io}|$ 表示 C-SAT 问题的公共输入输出规模, n 表示 R1CS 可满足问题的规模, T 表示对数空间可计算电路的时间上限。数据并行电路(data-parallel circuit)是指可以分为 N 份宽度为 g 、深度为 d 的子电路的电路,且有 $|C| = Ngd$ 。启动阶段系统参数可公开生成意味着可通过基于 ROM 的 Fiat-Shamir 启发式实现非交互;启动阶段系统参数需私密生成意味着证明者和验证者需拥有 CRS。“/”指当前协议没有具体可查的系统参数。“?”指本文所知尚不知道明确关系。³ 表中列出的证明、通信和验证复杂度,且均为协议 1 轮的主要开销,协议实际运行轮数及实际证明、验证计算开销和通信量与可靠性误差 ϵ 及安全级别有关。此外,一些基于 DEIP 的零知识证明的复杂度省去了部分因子,详见第 6 章。

明在密码学中的应用，并重点探讨了 Fiat-Shamir 认证协议是如何应用于零知识认证协议中的。上述工作侧重于（非交互）零知识证明理论层面的研究，而本文同时着力于区块链等应用背景下简洁非交互零知识证明及典型协议的总结和研究。

Nitulescu^[55] 详细定义了 zk-SNARK (zero-knowledge succinct non-interactive argument of knowledge)，并探讨了通用构造方法。Nitulescu 将 zk-SNARK 分类为基于 PCP、QAP、LIP 和 PIOP (polynomial interactive oracle proof) 的零知识证明，系统整理了各类证明的构造思路，并总结了典型方案。Nitulescu 详细描述了基于 QAP 的零知识证明的协议流程、底层难题假设及安全性等细节，包括如何将 C-SAT 问题归约为 QAP 可满足问题、如何针对 QAP 可满足问题构建 Linear-PCP 等。然而，该工作着重于对 zk-SNARK 的研究（对应于第 5 章），而本文除探讨 zk-SNARK 外，还详细对比研究了其他类别的简洁非交互零知识证明，尤其是系统参数可公开生成的系列零知识证明。Morais 等人^[56] 对比了构造零知识范围证明的不同方法，详细说明了 Bulletproofs 中范围证明的实现细节，但仅涉及对（零知识）范围证明的研究。Sun 等人^[57] 研究并总结了在区块链背景下零知识证明的框架、模型及应用，指出了目前区块链中零知识证明的应用现状、面临挑战及未来发展方向。然而，该工作不涉及对具体零知识证明方案的研究。

相比于前人的工作，本文的亮点主要有二。第一，本文总结了简洁非交互零知识证明的通用构造方法，并基于该通用构造方法对现有简洁非交互零知识证明进行了分类。分类共有两个维度，一是信息论安全证明，二是将信息论安全证明转换为简洁非交互零知识证明所基于的关键技术。基于这两个维度，本文较为深入地分类研究了现有的简洁非交互零知识证明，总结了每一类证明的待证明陈述表示形式、协议性能等。第二，本文基于上述分类维度详细梳理了每一类零知识证明的构建思路、优化方向及后续改进，分析了安全性、复杂度及性能优缺点。

1.3 本文结构

本文结构如图 2 所示。第 2 章介绍相关表示及全局定义。第 3 章介绍简洁非交互零知识证明的通用构造方法（图 1）及简洁非交互零知识证明的性能评价标准。根据通用构造方法中信息论安全证明的不同，第 4 章简要介绍基于概率可验证证明类的零知识证明，主要包括概率可验证证明 (PCP)、交互式概率可验证证明 (IPCP)、交互式预言证明 (IOP) 和线性概率可验证证明 (linear-PCP)；根据密码编译器应用的底层关键技术，第 5–8 章分别介绍基于二次算术程序 (QAP)、双向高效交互式证明 (DEIP)、内积论证 (IPA) 和 MPC-in-the-Head 的零知识证明，详见图 2（其中基于线性概率可验证证明和基于二次算术程序的零知识证明为同一类协议的两个维度）。第 9 章介绍未来研究方向。

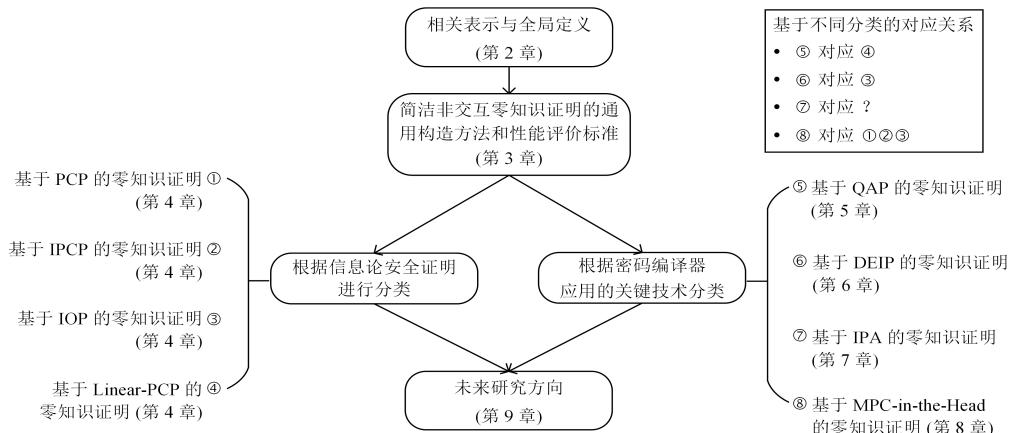


图 2 本文结构
Figure 2 Structure of this paper

2 相关表示与全局定义

2.1 相关表示

在本文中, 小写粗体字母表示向量, 例如 $\mathbf{a} \in \mathbb{F}^{1 \times n}$ 表示域 \mathbb{F} 上维度为 n 的行向量 (a_1, a_2, \dots, a_n) , 为表述方便, 常将 $\mathbb{F}^{1 \times n}$ 简记为 \mathbb{F}^n . $f(\cdot)$ 、 $g(\cdot)$ 等表示多项式, $\mathbf{f}(\cdot)$ 等表示向量多项式. 大写粗体字母 (如 \mathbf{A}) 表示矩阵, 例如, $\mathbf{A} \in \mathbb{F}^{m \times n}$ 表示 m 行 n 列的矩阵且 $a_{i,j}$ 表示第 i 行、第 j 列的矩阵元素, $\mathbf{A}\mathbf{a}$ 表示矩阵 \mathbf{A} 与向量 \mathbf{a} 的矩阵乘法. 本文用 \cdot 表示乘法, 特别的, $\mathbf{a} \cdot \mathbf{b} = \sum_{i=1}^n a_i \cdot b_i$ 表示向量 \mathbf{a} 与向量 \mathbf{b} 的内积. \odot 表示哈达玛积, 例如 $\mathbf{a} \odot \mathbf{b} = (a_1 \cdot b_1, a_2 \cdot b_2, \dots, a_n \cdot b_n)$. $y \leftarrow A(x, r)$ 表示算法 A 以 x 为输入、 r 为随机输入生成 y 的过程, 用 $y \xleftarrow{\$} S$ 表示从集合 S 中均匀随机地挑选 y , \rightarrow 表示函数映射关系, 用 $a \stackrel{?}{=} b$ 表示验证 a 是否等于 b . 对于正整数 n , $[n]$ 表示集合 $\{1, 2, \dots, n\}$.

本文中的算法输入均包含安全参数 λ . 如果对于任意的多项式 $p(\cdot)$ 都存在常数 c 使得当 $\lambda > c$ 时有 $\text{negl}(\lambda) < 1/p(\lambda)$, 则称 $\text{negl}(\lambda)$ 为对于 λ 的可忽略函数. 记 $f(\lambda) \approx g(\lambda)$ 当 $|f(\lambda) - g(\lambda)| \leq \text{negl}(\lambda)$. 用 PPT 表示概率多项式时间 (probabilistic polynomial time). 记 $O_\lambda(\cdot)$ 为省略安全参数 λ 多项式因子的大 O 记法, 本文常省略 λ . 本文常用的缩写词及含义对照表如表 2 所示.

表 2 缩略词及其含义对照表

Table 2 Table for abbreviations and their meanings

缩略词	含义	中文含义
IP ^[1]	interactive proof	交互式证明
LIP ^[58]	linear interactive proof	线性交互式证明
PCP ^[16, 17, 59]	probabilistic checkable proof	概率可验证证明
DEIP ^[60, 61]	doubly efficient interactive proof	双向高效的交互式证明
linear-PCP ^[58, 62]	-	线性 PCP
sum-check 协议 ^[63]	-	求和验证协议
IPCP ^[64]	interactive PCP	交互式概率可验证证明
R1CS ^[65]	rank-1 constraint system	一阶约束系统
IOP ^[66, 67]	interactive oracle proof	交互式谕示证明
IPA ^[30]	inner product argument	内积论证
CRS ^[12]	common reference string	公共参考串
DLOG 假设	discrete logarithm assumption	离散对数假设
ROM ^[13, 14]	random oracle model	随机预言模型
MPC ^[68]	secure multiparty computation	安全多方计算
C-SAT 问题	circuit satisfiability problem	电路可满足问题
RS 码 ^[69]	reed solomon code	里德-所罗门码
CRHF ^[15]	collision-resistant hash function	抗碰撞哈希函数
NIZK AoK	non-interactive zero-knowledge argument of knowledge	非交互零知识知识论证
QAP ^[19]	quadratic arithmetic program	二次算术程序
zk-SNARK ^[70]	zero-knowledge succinct non-interactive argument of knowledge	简洁非交互零知识知识论证

2.2 全局定义

本节介绍本文涉及的主要基础知识, 第 2.2.1 小节介绍电路及相关定义, 第 2.2.2 小节介绍承诺及相关定义, 第 2.2.3 小节介绍零知识证明及相关定义.

2.2.1 电路及相关定义

记算术电路 (arithmetic circuit) 为 $C : \mathbb{F}^{|x|+|w|} \rightarrow \mathbb{F}^{|y|}$, 它由若干域上的加法门和乘法门组成. 布尔电路 (Boolean circuit) 是算术电路的子类, 其仅有与门、异或门等布尔逻辑门, 变量取值仅为 0 或 1. 可以证明, 通过增加常数级别的电路门和深度, 任何布尔电路都可以转换为算术电路^[61]. 不失一般性, 本文中出现的电路均为二输入电路 (circuit with fan-in 2 gates). 记电路规模为电路中门的数量, 用 $|C|$ 表示, d 表示电路深度, g 表示电路宽度.

定义 1 (分层算术电路) 分层算术电路 (layered arithmetic circuit) 是指可以分为 d 层、且任意层的电路门的输入导线全部位于上一层的算术电路.

可以证明, 通过增加电路深度级别的电路门, 任意的算术电路都可转换为分层算术电路^[32].

定义 2 (电路可满足问题) 电路可满足问题 (circuit satisfiability problem, C-SAT) 是指给定电路 C 、电路的部分输入 x (x 可为空) 和电路输出 y , 判断是否存在证据 w (电路的另一部分输入, 视为秘密输入) 使得 $C(x, w) = y$. 如无特殊说明, 本文中的零知识证明均是针对 C-SAT 问题的.

针对布尔电路可满足问题的零知识证明可通过调用针对算术电路可满足问题的零知识证明高效构造 (只需扩大域并增加对变量为 0 或 1 的约束即可), 反之, 尚不清楚是否有高效的转化方式.

定义 3 (一阶约束系统^[27, 50]) 一个一阶约束系统 (rank-1 constraint system, R1CS) 是七元组 $(\mathbb{F}, \mathbf{A}, \mathbf{B}, \mathbf{C}, \mathbf{io}, m, n)$, 其中 \mathbf{io} 表示公共输入输出向量, $\mathbf{A}, \mathbf{B}, \mathbf{C} \in \mathbb{F}^{m \times m}$, $m \geq |\mathbf{io}| + 1$, n 是所有矩阵中非零值的最大数目. 称 R1CS 问题是可满足的当且仅当对于一个 R1CS 组, 存在证据 $w \in \mathbb{F}^{m - |\mathbf{io}| - 1}$ 使得 $(\mathbf{Az}) \odot (\mathbf{Bz}) = (\mathbf{Cz})$, 其中 $z = (\mathbf{io}, 1, w)^T$.

记 n 为 R1CS 可满足问题的规模, 可以证明, 任意 C-SAT 问题都可用 R1CS 可满足问题表示^[27], 且 $n = O(|C|)$ ^[50].

R1CS 是高级语言编译器的常见目标程序^[62, 65] 且形式较简单, 同时任意 C-SAT 问题都可用 R1CS 可满足问题表示^[27], 故有一些零知识证明^[19, 20] 在应用实现时是先将 C-SAT 问题转化为 R1CS 可满足问题, 再针对 R1CS 可满足问题构造的; 也有部分零知识证明^[27, 50] 直接针对 R1CS 可满足问题.

2.2.2 承诺及相关定义

定义 4 (承诺) 一个承诺方案 (commitment scheme) 包含发送者和接收者两个参与方及三个 PPT 算法 (Setup , Com , Open). 具体的, 算法 Setup 用于生成承诺用公共参数 pp . Com_{pp} 定义了函数映射 $\mathbf{M} \times \mathbf{R} \rightarrow \mathbf{C}$, 其中 \mathbf{M} 、 \mathbf{R} 和 \mathbf{C} 分别表示明文空间、随机数空间和承诺空间. Open_{pp} 算法定义了函数映射 $\mathbf{C} \times \mathbf{M} \times \mathbf{R} \rightarrow \{0/1\}$. 具体的, 对于消息 $m \in \mathbf{M}$ 和随机数 $r \in \mathbf{R}$, 承诺 c 的生成方式为 $c \leftarrow \text{Com}_{\text{pp}}(m; r)$, Open 算法为 $0/1 \leftarrow \text{Open}_{\text{pp}}(c, m, r)$. 为表示方便, 本文常省略 pp 和随机数 r .

承诺有两个基本性质, 隐藏性 (hiding) 和绑定性 (binding). 其中, 隐藏性是指敌手获得承诺 c 后无法获知 m 的值, 绑定性是指一个承诺 c 在 Open 阶段只能打开为一个值.

定义 5 (隐藏性) 计算隐藏性 (computational hiding) 是指对于任意的 PPT 敌手 \mathcal{A} , 有

$$\Pr \left[\begin{array}{l} \text{pp} \leftarrow \text{Setup}(1^\lambda); (m_0, m_1) \leftarrow \mathcal{A}(\text{pp}); b \stackrel{\$}{\leftarrow} \{0, 1\} : b = b' \\ r \stackrel{\$}{\leftarrow} \mathbf{R}; c \leftarrow \text{Com}_{\text{pp}}(m_b; r); b' \leftarrow \mathcal{A}(\text{pp}, c) \end{array} \right] - \frac{1}{2} \leq \text{negl}(\lambda). \quad (1)$$

对应的, 完美隐藏性 (perfect hiding) 是将不等式 (1) 中的敌手 \mathcal{A} 修改为无穷算力且 “ $\leq \text{negl}(\lambda)$ ” 替换为 “ $= 0$ ”.

定义 6 (绑定性) 计算绑定性 (computational binding) 是指对于任意的 PPT 敌手 \mathcal{A} , 有

$$\Pr \left[\text{pp} \leftarrow \text{Setup}(1^\lambda); (m_0, m_1, r_0, r_1) \leftarrow \mathcal{A}(\text{pp}) : \text{Com}_{\text{pp}}(m_0; r_0) = \text{Com}_{\text{pp}}(m_1; r_1) \wedge m_0 \neq m_1 \right] \leq \text{negl}(\lambda). \quad (2)$$

对应的, 完美绑定性 (perfect binding) 是将不等式 (2) 中的敌手 \mathcal{A} 修改为无穷算力且 “ $\leq \text{negl}(\lambda)$ ” 替换为 “ $= 0$ ”.

定义 7 (加性同态承诺) 加性同态承诺 (additive homomorphic commitment) 是指具有加性同态性质的承诺, 即给定承诺 $\text{Com}(x; r_x)$ 和 $\text{Com}(y; r_y)$, 存在运算 \oplus , 满足

$$\text{Com}(x; r_x) \oplus \text{Com}(y; r_y) = \text{Com}(x + y; r_x + r_y). \quad (3)$$

定义 8 (Pedersen 承诺) Pedersen 承诺的明文空间、随机数空间和承诺空间分别为 $M = R = \mathbb{Z}_q, C = \mathbb{G}_q$. Setup 算法生成公共参数, 即 \mathbb{G} 上生成元 g, h , 其承诺和承诺打开算法为 $g^x h^r \leftarrow \text{Com}(x; r)$ 和 $0/1 \leftarrow \text{Open}(\text{Com}(x; r), x, r)$.

容易证明, Pedersen 承诺具有完美隐藏性、计算绑定性和加性同态性质.

定义 9 (Pedersen 向量承诺) Pedersen 向量承诺是对定义 8 的自然扩展, 其明文空间、随机数空间和承诺空间分别为 $M = \mathbb{Z}_q^n, R = \mathbb{Z}_q, C = \mathbb{G}_q$. Setup 算法生成公共参数, 即 \mathbb{G}_q^n 上生成元 $\mathbf{g} = (g_1, g_2, \dots, g_n)$ 和 \mathbb{G}_q 上生成元 h , 向量承诺的承诺和打开承诺算法分别为

- $c = \mathbf{g}^m h^r \leftarrow \text{Com}(\mathbf{m}; r)$. 承诺算法以消息 \mathbf{m} 和随机数 r 为输入, 输出承诺 $c = \mathbf{g}^m h^r$.
- $0/1 \leftarrow \text{Open}(c, \mathbf{m}, r)$. 承诺打开算法以承诺 c 、消息 \mathbf{m} 和随机数 r 为输入, 验证承诺正确性.

2.2.3 零知识证明及相关定义

本小节介绍零知识证明, 并给出若干相关概念的简要定义.

给定二元关系 $\mathcal{R} : \{0, 1\}^* \times \{0, 1\}^* \rightarrow \{0, 1\}$, 记语言 $\mathcal{L}(\mathcal{R})$ 为集合 $\{x : \exists \mathbf{w} \text{ s.t. } \mathcal{R}(x, \mathbf{w}) = 1\}$. 称一个语言 $\mathcal{L}(\mathcal{R})$ 是 NP 语言当如下两个条件成立:

- (1) $|\mathbf{w}| = \text{poly}(|x|)$.
- (2) 给定任意的 x, \mathbf{w} , 存在多项式时间算法能够高效判定 $\mathcal{R}(x, \mathbf{w}) \stackrel{?}{=} 1$.

记 $\langle A, B \rangle$ 为一对交互式图灵机. 记 $\langle A(y), B(z) \rangle(x)$ 为在 A, B 的随机输入带均匀独立选取, 公共输入为 x , A 的辅助输入为 y , B 的辅助输入为 z 时, 图灵机 B 与图灵机 A 交互后输出的随机变量.

定义 10 (交互式证明系统^[1, 72]) 给定二元关系 \mathcal{R} 及其对应语言 $\mathcal{L}(\mathcal{R})$, 则针对该语言的交互式证明系统 (interactive proof system) 是 $\langle \mathcal{P}(y), \mathcal{V}(z) \rangle$, 其中图灵机 \mathcal{P} 、 \mathcal{P}^* (也被称为证明者) 可以为无穷算力的, 图灵机 \mathcal{V} (也被称为验证者) 是 PPT 的. $\langle \mathcal{P}(y), \mathcal{V}(z) \rangle$ 满足如下两个性质.

- **完备性** (completeness): 对于任意的 $x \in \mathcal{L}(\mathcal{R})$, 存在 y , 使得对于任意的 $z \in \{0, 1\}^*$, $\Pr[\langle \mathcal{P}(y), \mathcal{V}(z) \rangle(x) = 1] \geq 1 - \text{negl}(|x|)$. 完美完备性 (perfect completeness) 是指上述概率等于 1.
- **可靠性** (soundness): 对于任意的 $x \notin \mathcal{L}(\mathcal{R})$, 任意的恶意证明者 \mathcal{P}^* , 任意的 $y, z \in \{0, 1\}^*$, 有 $\Pr[\langle \mathcal{P}^*(y), \mathcal{V}(z) \rangle(x) = 1] \leq \text{negl}(|x|)$.

定义 11 (交互式论证^[72, 73]) 交互式论证与交互式证明系统的区别在于, 论证可靠性定义中恶意证明者 \mathcal{P}^* 被限制为 PPT 的图灵机. 此外, 通常也限制完备性中的 \mathcal{P} 为 PPT 的 [72, §4.8.1]. 具体的, 给定二元关系 \mathcal{R} 及其对应语言 $\mathcal{L}(\mathcal{R})$, 则针对该语言的交互式论证 (interactive argument) 是 $\langle \mathcal{P}(y), \mathcal{V}(z) \rangle$, 其中图灵机 \mathcal{P} 、 \mathcal{P}^* 和 \mathcal{V} 均是 PPT 的. 与证明系统类似, $\langle \mathcal{P}(y), \mathcal{V}(z) \rangle$ 也具有完备性和可靠性.

零知识证明是具有零知识性的交互式证明系统或论证, 零知识性的直观含义是当以满足 $x \in \mathcal{L}(\mathcal{R})$ 的 x 为公共输入时, 任何在与 \mathcal{P} 交互后高效计算出的信息也都可仅根据 x 高效计算得出 (此时没有交互).

定义 12 (计算零知识证明^[1, 72]) 考虑 \mathcal{V} 在交互过程中的视图, 记为 $\text{view}_{\mathcal{V}}^{\mathcal{P}}(x, z)$, 即 \mathcal{V} 在交互中拥有的所有信息, 其包括 \mathcal{V} 的公共输入、随机输入、辅助输入, 从 \mathcal{P} 处收到的消息及 \mathcal{V} 可自行计算的信息. 显然, 最后一项可根据前几项计算得出. 给定二元关系 \mathcal{R} 及其对应语言 $\mathcal{L}(\mathcal{R})$, 令 $\langle \mathcal{P}, \mathcal{V} \rangle$ 是针对该语言的交互式证明系统 (论证), 如果对于任意 PPT 的 \mathcal{V}^* , 都存在一个期望 PPT 的模拟器 \mathcal{S} , 使得 $\{\text{view}_{\mathcal{V}^*}^{\mathcal{P}}(x, z)\}_{x \in \mathcal{L}(\mathcal{R})}$ 和 $\{\mathcal{S}(x, z)\}_{x \in \mathcal{L}(\mathcal{R})}$ 这两个随机变量族是计算不可区分的, 则称该证明系统 (论证) 是计算零知识的.

其中计算不可区分是指对于所有的概率性算法 D (运行时间受 $\text{poly}|x|$ 限制)、所有的多项式 $p(\cdot)$ 、所

有的 $z \in \{0, 1\}^*$, 有

$$|\Pr[D(\{\text{view}_{\mathcal{V}^*}^{\mathcal{P}}(x, z)\}_{x \in \mathcal{L}(\mathcal{R})}) = 1] - \Pr[D(\{\mathcal{S}(x, z)\}_{x \in \mathcal{L}(\mathcal{R})}) = 1]| < 1/p(|x|). \quad (4)$$

定义 13 (诚实验证者零知识)^[72] 诚实验证者零知识 (honest verifier zero-knowledge) 是指模拟过程中的验证者是按照事先确定好的协议步骤运行的. 给定二元关系 \mathcal{R} 及其对应语言 $\mathcal{L}(\mathcal{R})$, 令 $\langle \mathcal{P}, \mathcal{V} \rangle$ 是针对该语言的交互式证明系统 (论证), 如果存在一个期望 PPT 的模拟器 \mathcal{S} 使得 $\{\text{view}_{\mathcal{V}}^{\mathcal{P}}(x, z)\}_{x \in \mathcal{L}(\mathcal{R})}$ 和 $\{\mathcal{S}(x, z)\}_{x \in \mathcal{L}(\mathcal{R})}$ 这两个随机变量族是计算不可区分, 则称该证明系统 (论证) 是诚实验证者计算零知识的.

定义 14 (知识论证)^[74] 知识论证 (argument of knowledge) 指具有知识可靠性的论证. 给定一个多项式时间内可判定的二元关系 \mathcal{R} 及其对应的 NP 语言 $\mathcal{L}(\mathcal{R})$, 知识可靠性是指对于任意的 PPT 敌手 \mathcal{P}^* , 都存在期望 PPT 的提取器 \mathcal{E} , 使得对于 $\mathcal{L}(\mathcal{R})$ 及任意的陈述 x, w' , 如果有 $\Pr[w' \leftarrow \mathcal{P}^*(x) : \langle \mathcal{P}^*(w'), \mathcal{V}(z) \rangle(x) = 1] \geq 1/p(|x|)$, 其中 $p(\cdot)$ 为某个多项式, 则有 $\Pr[w' \leftarrow \mathcal{E}^{\mathcal{P}^*}(x) : \mathcal{R}(x, w') = 1] \geq 1/p(|x|) - \text{negl}(|x|)$.

本文中部分论证^[30–32, 53, 54] 援引的知识可靠性为统计意义的证据扩展可仿真性 (statistical witness-extended emulation)^[75].

定义 15 (公开抛币) 如果验证者在一个证明 (论证) 的交互过程中发送的信息是公开抛币的直接结果, 则称该证明 (论证) $\langle \mathcal{P}, \mathcal{V} \rangle$ 是公开抛币 (public coin) 的.

定义 16 (简洁性)^[25] 对于一个交互式论证, 如果 \mathcal{P} 和 \mathcal{V} 之间的通信复杂度不超过 $\text{poly}(\lambda)(|x| + |w|)^{o(1)}$, 则称该论证 $\langle \mathcal{P}, \mathcal{V} \rangle$ 为简洁 (succinct) 的; 如果通信复杂度不超过 $\text{poly}(\lambda)(|x| + |w|)^c + o(|x| + |w|)$, 则称论证为略显简洁的 (slightly succinct), 其中 c 为某个小于 1 的常数. 本文将简洁和略显简洁的零知识证明统称为简洁零知识证明.

基于标准假设^[24](敌手仅受可用时间和算力限制) 无法实现简洁且具有统计级别可靠性的证明系统^[76]. 如无特殊说明, 本文涉及到的所有具体零知识证明协议均是零知识论证或零知识知识论证, 为叙述方便, 本文将非交互零知识知识论证 (non-interactive zero-knowledge argument of knowledge) 简记为 NIZK AoK.

Σ 协议^[77] 给定二元关系 \mathcal{R} 及其对应语言 $\mathcal{L}(\mathcal{R})$, 则针对该语言的 Σ 协议是公开抛币的诚实验证者零知识论证, 其有以下三步.

- 承诺阶段: 证明者 \mathcal{P} 向验证者 \mathcal{V} 发送承诺 a .
- 挑战阶段: \mathcal{V} 向 \mathcal{P} 发送随机挑战 e .
- 响应阶段: \mathcal{P} 向 \mathcal{V} 发送响应函数 $f(w, r, e)$, 其中 f 是某公开函数, w 是证据, r 是随机数.

Σ 协议具有完备性、特殊知识可靠性和诚实验证者零知识性. 其中特殊知识可靠性 (s -special soundness) 是指给定对于任意的陈述 x 及 s 个接受副本 $\{(a, e_i, z_i)\}_{i \in [s]}$, x 所对应的证据 w 可被高效提取.

非交互零知识证明 交互式证明需要证明者和验证者时刻保持在线状态, 而这会因网络延迟、拒绝服务等原因难以保障. 而在非交互零知识证明中, 证明者仅需发送一轮消息即可完成证明. 然而, 在标准假设下已证明无法构造针对非平凡语言的非交互零知识证明系统^[78], 因此必须引入新的假设. 目前主流的非交互零知识证明的构造方法有两种, 一是基于 CRS 模型实现的, 二是基于 ROM 并利用 Fiat-Shamir 启发式实现的.

CRS 模型由 Blum、Feldman 和 Micali^[12] 提出, 其假设存在一个证明者和验证者可获得的由可信第三方生成的公共字符串, 其可由 MPC 生成^[79, 80]. 本文中基于 QAP 的零知识证明就是基于 CRS 模型实现非交互的, 该类证明也被称为 zk-SNARK^[25, 58, 81], 其通信量为常数个群元素, 验证复杂度可通过预处理达到常数次配对运算.

然而, zk-SNARK 存在两个较为严重的问题, 一是需要可信初始化用以预处理, 二是基于非可证伪假设才是安全的. 对于问题一, 虽然 Feige、Lapidot 和 Shamir^[82] 基于单向函数存在的假设构造了同一个 CRS 可用于证明多个陈述的非交互零知识证明, 但为实现更低的通信量和验证计算开销, 在绝大多数 zk-SNARK 中 CRS 都是与陈述相关的. 也就是说, 在给定待证明陈述后, 需先由可信第三方先进行相应预处理. 预处理虽然使得亚线性甚至常数级别的验证复杂度成为了可能 (在无预处理情况下验证者读取陈

述就会导致线性级别的验证复杂度), 但对于每个陈述都进行预处理也会带来较大的计算开销. 对于问题二, Gentry 和 Wichs^[25] 从理论上证明了基于可证伪假设无法构造 zk-SNARK. 可证伪假设是指可用敌手和高效挑战者之间的游戏模型描述的假设, 在游戏结束时, 挑战者能够高效判断敌手是否攻击成功. 常见的标准假设, 如单向函数存在、离散对数假设等都属于可证伪假设.

在 ROM^[13,14] 下, 验证者的随机挑战可由哈希函数的输出替代, 由此任何公开抛币的交互式零知识论证(如 Σ 协议)可转换为非交互零知识论证^[54,83-85]. 在 Σ 协议中, 证明者计算 $e \leftarrow \text{RO}(x, a)$ 并用该 e 替代验证者的随机挑战, 其中 RO 代表随机预言函数. ROM 是一种理想的密码学模型, 其假设协议的所有参与方都有访问请求一个理想随机谕示的权限, 该谕示在实际应用中通常启发地用哈希函数替代. 在 ROM 下, 该类非交互零知识证明可基于标准假设实现. 本文中基于 PCP、IPCP、IOP、DEIP、IPA 和 MPC-in-the-Head 的零知识证明均是通过 ROM 下的 Fiat-Shamir 启发式实现非交互的.

定义 17 (Schwartz-Zippel 引理^[86,87]) 若域 \mathbb{F} 上多项式 $p(x_1, x_2, \dots, x_n)$ 度为 d 且不为零多项式, 令集合 $S \subseteq \mathbb{F}$, 则对于 $(y_1, y_2, \dots, y_n) \xleftarrow{\$} S$, 有与 n 无关的以下概率关系成立

$$\Pr[p(y_1, y_2, \dots, y_n) = 0] \leq \frac{d}{|S|}. \quad (5)$$

Schwartz-Zippel 引理可将针对多项式约束的证明转化为针对多项式上某一点的证明, 其能够在保障可靠性(但会带来一定误差)的同时降低通信复杂度, 是构造简洁非交互零知识证明的重要技术支撑之一.

3 简洁非交互零知识证明的通用构造方法及性能评价标准

3.1 简洁非交互零知识证明的通用构造方法

简洁非交互零知识证明的通用构造方法如图 1 所示, 简要介绍如下.

- (1) 将待证明陈述归约为 C-SAT 问题. 一方面 C-SAT 问题是 NPC 问题, 也就是说任意 NP 问题都可在多项式时间内归约为 C-SAT 问题, 另一方面大多数实际问题都可用电路形式表达, 故现有的简洁非交互零知识证明的待证明陈述表示形式大都为 C-SAT 问题. 事实上, 也存在一系列的算术电路生成器, 可将格式化计算程序转化为算术电路, 例如 Meiklejohn 等人^[88] 提出的 ZKPDL 和 Ben-Sasson 等人^[81] 提出的 TinyRAM. 然而, 这些库的实际归约效果可能并不好, 例如, 针对 SHA-256 采用 Pinocchio³ 中的电路生成器所生成的算术电路门数为 58 160, 而根据 SHA-256 的算法可手动生成门数仅为 27 904 的相应算术电路^[89].
- (2) 将 C-SAT 问题转换为易证明的语言. 针对 C-SAT 问题直接构造零知识证明往往无法实现简洁性, 一个构造零知识证明的平凡思路是在掩藏电路中每个导线值的同时完成验证计算, 而这会导致 $\Theta(|C|)$ 级别的通信复杂度. 因此, 通常需将 C-SAT 问题转换为易证明的语言(此步可能没有), 而易证明的语言在不同具体场景下也是不尽相同的. 例如, 在基于 Linear-PCP 的零知识证明中, 需将 C-SAT 问题转化为 QAP 可满足问题, 即判断是否存在一个多项式能够被某个公开多项式整除; 在基于 IPA 的零知识证明中, 需将 C-SAT 问题中所有的线性约束和乘法门约束归约为判断一个多项式是否为零多项式的问题; 在基于 DEIP 的零知识证明中, 需将 C-SAT 问题转换为多元多项式的求和验证问题.
- (3) 针对易证明的语言构造信息论安全证明. 许多简洁零知识证明都是基于信息论安全证明构造的, 例如, 第一个简洁零知识论证^[15] 就是基于 PCP 构造的; 基于 MPC-in-the-Head 的零知识证明则是首先利用 MPC-in-the-Head 构造零知识的 PCP(或 IPCP、IOP), 然后调用合适的 MPC 协议构造的; Bitansky 等人^[58] 和 Setty 等人^[62] 也分别指出, 基于 QAP 的零知识证明本质上是基于 Linear-PCP 构造的. 具体的, 本文中出现的信息论安全证明包括 PCP^[16,17,59]、IPCP^[64]、IOP^[67,90] 和 Linear-PCP^[58,62], 详见第 4 章.
- (4) 利用密码编译器将信息论安全证明转换为简洁非交互零知识证明. 密码编译器的作用包括:

³Pinocchio v0.5.3. <https://github.com/amiller/pinocchio>.

- 实现谕示. 信息论安全证明需要理想谕示 (Ideal Oracle), 而在实际的交互式证明中理想谕示是不存在的, 因此需借助承诺、哈希等密码工具实现. 由于这些密码工具大都是基于计算意义上的难题假设 (如 Pedersen 承诺的绑定性是基于 DLOG 假设的), 因此证明也往往被削弱为论证.
- 实现非交互. 根据实现非交互所基于的模型, 分为基于 CRS 模型和基于 ROM 的零知识证明. 例如, 基于 QAP 的零知识证明均基于 CRS 模型实现非交互, 而基于 DEIP、IPA 和 MPC-in-the-Head 的零知识证明均基于 ROM 实现非交互. 需要指出的是, 即使是基于 ROM 实现非交互的零知识证明, 也可能需要公共参考串, 如 ZKvSQL^[51]、Libra^[26] 等.
- 实现零知识. 由于信息论安全证明本身可能不具备零知识性, 因此可能需利用密码编译器实现零知识性. 例如, 基于 DEIP 和 IPA 的零知识证明是通过承诺的隐藏性及盲化多项式实现的零知识性, 基于 MPC-in-the-Head 的零知识证明是通过 MPC 协议的隐私性实现的零知识性.
- 降低通信复杂度. 部分密码编译器有助于降低通信复杂度, 例如, 基于 DEIP 的零知识证明可通过多项式承诺降低通信复杂度, 基于 MPC-in-the-Head 的零知识证明可通过选取合适的底层 MPC 协议降低通信复杂度.

3.2 简洁非交互零知识证明的性能评价标准

性能评价标准是衡量一个简洁非交互零知识证明协议优劣的准绳, 本部分从效率和安全性角度简要介绍简洁非交互零知识证明的性能评价标准, 见表 3.

表 3 简洁非交互零知识证明的性能评价标准
Table 3 Performance evaluation of succinct NIZKAoK

性能评价标准		协议
证明复杂度	线性级别	Groth16 ^[52] 、Virgo ^[28] 等
	准线性级别	Ligero ^[35] 、Aurora ^[50] 等
通信复杂度	根号级别	Ligero ^[38] 、BooLigero ^[48]
	对数级别	Ligero++ ^[36] 、Aurora ^[50] 、Bulletproofs ^[31] 等
效率	常数级别	Pinocchio ^[20] 、Groth16 ^[52] 等
	线性级别	Ligero ^[35] 、Bulletproofs ^[31] 等
验证复杂度	亚线性级别 (需要预处理或 CRS)	DRZ20 ^[54] 、Pinocchio ^[20] 等
	需要 (尤其是群幂运算)	Pinocchio ^[20] 、Virgo ^[28] 、Bulletproofs ^[31] 等
是否需要公钥 密码操作	仅需对称密码操作	Aurora ^[50] 、Ligero ^[35] 等
	标准假设	Aurora ^[50] 、Ligero ^[35] 、Bulletproofs ^[31] 等
底层难题假设	非标准假设 (主要指不可证伪假设)	Pinocchio ^[20] 、Groth16 ^[52] 等
	系统参数私密生成	Pinocchio ^[20] 、Libra ^[26] 、DRZ20 ^[54] 等
安全性	系统参数公开生成	Ligero ^[35] 、Aurora ^[50] 、Virgo ^[28] 等
	仅包含对称密钥操作基于单向函数 存在被认为是抗量子的	Aurora ^[50] 、Ligero ^[35] 等
是否抗量子	非抗量子	Bulletproofs ^[31] 、Pinocchio ^[20] 、Virgo ^[28] 等

在效率层面, 主要分为证明复杂度、验证复杂度、通信复杂度和是否需要公钥密码操作. 特别注意的是, 本文中的证明、通信和验证复杂度均为协议一轮的开销, 协议实际运行轮数及实际证明、验证计算开销和通信量与可靠性误差及安全级别有关.

- 证明复杂度: 证明复杂度是指在一轮协议中证明者生成证明所需计算步数的渐近复杂度. 在其他条件不变的情况下, 一个协议的证明复杂度越低, 协议的性能越好. 然而, 几乎所有的简洁非交互零知识证明协议都具有线性或准线性级别的证明复杂度, 仅从证明复杂度来看, 这些协议的差距

并不大.

- (2) 通信复杂度: 通信复杂度是指一轮协议证明规模的渐近复杂度. 在其他条件不变的情况下, 一个协议的通信复杂度越低, 协议的性能越好. 常见的亚线性通信复杂度为常数级别 (zk-SNARK, 见第5章)、根号级别 (包括 Ligero^[35]、BooLigero^[48])、对数级别 (包括 Aurora^[50] 等基于 IOP 的零知识证明、Bulletproofs^[31] 等基于 IPA 的零知识证明和 Ligero++^[36] 等基于 MPC-in-the-Head 的零知识证明). 特别的, 基于 DEIP 的零知识证明只有满足电路深度与电路规模成对数关系时, 通信复杂度才是对数级别的.
- (3) 验证复杂度: 验证复杂度是指一轮协议验证者验证证明有效性所需计算步数的渐近复杂度. 在其他条件不变的情况下, 一个协议的验证复杂度越低, 协议的性能越好. 需要指出的是, 一般而言由于验证者起码要读取整个陈述, 因此验证复杂度起码为线性. 但是, 可通过 CRS 和预处理降低验证者参与协议后的计算开销. 例如 DRZ20^[54] 就是利用 CRS 中的结构化承诺密钥实现了对数级别的验证复杂度, Pinocchio 也可利用预处理实现常数级别配对操作的验证复杂度.
- (4) 是否需要公钥密码操作: 对称密码操作是指移位、异或、域上多项式运算等对称密码中常出现的运算操作, 而公钥密码操作包括椭圆曲线上运算等公钥密码中常出现的运算操作. 一般而言, 公钥密码操作尤其是群幕运算的实际开销较高, 因此是否需要公钥密码操作会影响零知识证明协议的实际效率. 基于 QAP 的零知识证明、基于 DEIP 的零知识证明和基于 IPA 的零知识证明均需要一定的公钥密码操作.

特别指出的是, 上述复杂度均与协议的安全参数成多项式关系. 因此, 即使是通信复杂度仅为常数个群元素的协议, 如 Pinocchio^[20], 其实际通信量也会随协议安全参数的改变而改变.

在安全性层面, 主要分为底层难题假设、系统参数生成方式和是否抗量子, 分别介绍如下.

- (1) 底层难题假设: 不同简洁非交互零知识证明协议的底层难题假设通用性有一定的差距. 例如, 基于 MPC-in-the-Head 的零知识证明所基于的假设是单向函数存在, 这是一种密码学中较为常见的假设. 第7章中的零知识证明基于的难题假设是离散对数假设, 其相比于单向函数存在的假设通用性较低, 但也属于标准假设^[24]. 第5章中的零知识证明基于的假设是不可证伪假设, 其不属于标准假设. 底层难题假设的通用性是零知识证明落地应用的重要考量因素之一.
- (2) 系统参数生成方式: 基于 CRS 的简洁非交互零知识证明 (如 Pinocchio^[20]、Libra^[26] 和 DRZ20^[54]) 的系统参数必须由可信第三方生成, 这在去中心化的区块链应用中会带来安全性问题. 而一些基于 ROM 实现非交互的简洁非交互零知识证明可利用某些哈希函数由证明者自行生成随机数, 在某种程度上具有更高的安全性.
- (3) 是否抗量子: 基于 MPC-in-the-Head 的零知识证明和部分基于 IOP 的零知识证明 (如 Aurora) 只需假设单向函数存在, 且仅有对称密钥操作, 被认为是抗量子的.

4 基于 PCP、Linear-PCP、IPCP 和 IOP 的零知识证明

1992 年, Kilian^[15] 基于概率可验证证明 (PCP) 利用默克尔树和抗碰撞哈希函数构造了第一个简洁的交互式零知识论证, 后续的零知识证明大都是基于 PCP 及其变种实现的. 值得一提的是, 虽然基于二次算术程序的简洁非交互零知识证明^[19, 91] 似乎“摆脱了” PCP, 但这些协议本质是基于一种特殊的 PCP, 即 Linear-PCP 实现的^[58, 62]. 除 Linear-PCP 外, 基于其他对 PCP 的扩展, 即 IPCP 和 IOP, 也可构造零知识证明. 本章第4.1节介绍相关定义及概念, 第4.2节分别介绍基于 PCP、Linear-PCP、IPCP 和 IOP 的零知识证明.

4.1 定义及概念

定义 18 (概率可验证证明^[16, 17, 59]) 给定二元关系 \mathcal{R} , 概率可验证证明 (probabilistically checkable proof, PCP) 指在证明者针对语言 $\mathcal{L}(\mathcal{R})$ 生成证明谕示 π 后, 给定验证者访问请求谕示 π 任意位置比特的权限, 则验证者可通过生成最多 $r(\lambda)$ 长度的随机数进而访问请求谕示 π 的 $q(\lambda)$ 个比特值选择是否接受 $x \in \mathcal{L}(\mathcal{R})$. 该证明也具有完美完备性和可靠性 (可靠性误差不多于 $1/2$).

Arora 和 Safra^[17] 指出, $\text{NP} = \text{PCP}(\log n, 1)$, 即陈述长为 n 的 NP 问题平凡证明与允许使用随机

数长度为 $O(\log n)$ 、允许访问谕示数为 $O(1)$ 的 PCP 等价.

定义 19 (线性 PCP 与线性交互式证明^[58]) 相比于 PCP, 在线性 PCP (linear-PCP) 中验证者 \mathcal{V} 的访问请求 $\mathbf{q} \in \mathbb{F}^m$ 为行向量, 而证明者 \mathcal{P} 的回答 $a \leftarrow \boldsymbol{\pi}_o \cdot \mathbf{q}$ 为谕示和访问请求的内积.

线性交互式证明 (linear interactive proof, LIP) 是指证明中证明者 \mathcal{P} 仅能利用验证者 \mathcal{V} 的消息进行线性/仿射运算的一类证明, 又由于 \mathcal{V} 的消息蕴含在 CRS 中, 故 LIP 中证明 $\boldsymbol{\pi}_o$ 与 CRS 成线性/仿射关系.

利用 Linear-PCP 可自然构造两轮 LIP. 在 LIP 中, 记 Linear-PCP 中的证明谕示为 $\boldsymbol{\pi}_o$, \mathcal{V} 的访问请求为 $(\mathbf{q}_1, \mathbf{q}_2, \dots, \mathbf{q}_k, \mathbf{q}_{k+1})$, 且 $\mathbf{q}_{k+1} = \alpha_1 \mathbf{q}_1 + \dots + \alpha_k \mathbf{q}_k$. 证明者返回 $\{a_i = \boldsymbol{\pi}_o \cdot \mathbf{q}_i\}_{i \in [k+1]}$. 由于限制了证明者只能计算 \mathbf{q}_i 的线性组合, 因此若 \mathcal{V} 验证 $a_{k+1} \stackrel{?}{=} \alpha_1 a_1 + \dots + \alpha_k a_k$ 通过, 其会以较大概率相信 \mathcal{P} 使用了一致的 $\boldsymbol{\pi}_o$ (可靠性误差不多于 $1/|\mathbb{F}|$).

定义 20 (交互式谕示证明^[67, 90]) 对于一个 $k(\lambda)$ 轮公开抛币的交互式谕示证明 (interactive oracle proof, IOP), 在第 i 轮, 验证者 \mathcal{V} 发送随机消息 m_i 给证明者 \mathcal{P} , 随后 \mathcal{P} 返回消息 π_i , $k(\lambda)$ 轮交互结束后, \mathcal{P} 可构造证明谕示 $\boldsymbol{\pi} = (\pi_1, \pi_2, \dots, \pi_k)$. 随后 \mathcal{V} 向 $\boldsymbol{\pi}$ 发起 $q(\lambda)$ 数目的访问请求并决定接受或拒绝.

在某种程度上, IOP 是 IP 与 PCP 的结合. 与 IP 类似, $k(\lambda)$ 轮 IOP 也具有完备性和可靠性; 与 PCP 类似, IOP 中也有描述谕示规模的参数和描述验证者访问请求次数的参数. 具体的, 记 r_i 和 r_q 分别为交互和访问请求的随机比特长度, 则对于 $x \notin \mathcal{L}(\mathcal{R})$ 和恶意敌手 \mathcal{P}^* , 有

$$\Pr \left[\begin{array}{l} (m_1, m_2, \dots, m_k) \xleftarrow{\$} \{0, 1\}^{r_i}; \\ (\pi_1, \pi_2, \dots, \pi_k) \leftarrow \langle \mathcal{P}^*, \mathcal{V}(r_i) \rangle(x) \end{array} : \Pr_{r \xleftarrow{\$} \{0, 1\}^{r_q}} [\mathcal{V}^{\pi_1, \pi_2, \dots, \pi_k}(x, (m_1, m_2, \dots, m_k); r) = 1] \geq \epsilon_q(\lambda) \right] \leq \epsilon_i(\lambda) \quad (6)$$

不等式 (6) 的含义为随机性使得 \mathcal{P}^* 以至少 $\epsilon_q(\lambda)$ 大小概率欺骗成功的概率不超过 $\epsilon_i(\lambda)$. 记 IOP 的可靠性误差为 $\epsilon_q(\lambda) + \epsilon_i(\lambda)$, 则 PCP 是 $\epsilon_q(\lambda) = 0$ 的特殊 IOP, IP 是 $\epsilon_i(\lambda) = 0$ 的特殊 IOP, 对 NP 问题的平凡证明中 $\epsilon_q(\lambda) = \epsilon_i(\lambda) = 0$. 此外, IPCP 是 $k(\lambda) = 1$ 的特殊 IOP.

定义 21 (默尔克树^[92]) 默尔克树 (Merkle tree) 是一个二叉树, 其任意父节点的值等于左右子节点值连接后的哈希值. 记默尔克树根 Root 为无父节点的节点, 叶子节点为无子节点的节点. 给定一组叶子节点值, 称该组叶子节点的节点路径为足以计算 Root 所需的最少节点哈希值. 容易证明, 任意一组叶子节点的节点路径规模为 $O(\log n)$, 其中 n 为叶子节点的个数.

4.2 典型协议分析

本节第 4.2.1 小节介绍基于 PCP 的零知识证明, 第 4.2.2 小节介绍基于 Linear-PCP 的零知识证明, 第 4.2.3 小节介绍基于 IPCP 的零知识证明, 第 4.2.4 小节介绍基于 IOP 的零知识证明.

4.2.1 基于 PCP 的零知识证明

Kilian92/Micali94. 1992 年, Kilian^[15] 基于 PCP 利用默克尔树和 CRHF 构造了简洁的交互式论证, 其思路如下. 首先验证者 \mathcal{V} 和证明者 \mathcal{P} 约定抗碰撞哈希函数 H ; 其次 \mathcal{P} 生成 PCP 的谕示证明 $\boldsymbol{\pi}$, 并利用默克尔树对 $\boldsymbol{\pi}$ 承诺 (哈希函数选用 H), 将默克尔树根 Root 发送给 \mathcal{V} ; 然后 \mathcal{V} 生成 $r(n)$ 个随机数发给 \mathcal{P} , \mathcal{P} 和 \mathcal{V} 根据随机数、公共输入和 Root 共同确定访问请求 $\boldsymbol{\pi}$ 的位置; 最后 \mathcal{P} 揭示访问请求的位置值并将对应的节点路径哈希值发送给 \mathcal{V} , \mathcal{V} 计算 Root 的值验证一致性并根据访问请求的位置值验证 PCP 的正确性.

由于默克尔树的结构特点, 上述协议的通信复杂度可达到对数级别, 此外, 该协议的可靠性来自于哈希函数 H 的抗碰撞性. 值得一提的是, 上述协议本身不具备零知识性, 其零知识性是通过一种在不揭示承诺值的同时证明承诺值具有某种性质的方法实现的 (notarized envelopes).

1994 年, Micali^[18] 利用 ROM 下的 Fiat-Shamir 启发式将上述协议修改为非交互. Valiant^[93] 进一步指出 Micali 提出的方案是知识论证. 后续的工作通过引入 PIR (private information retrieval) 将 Kilian92 中的四轮协议改进为两轮^[94], 并利用抗碰撞可提取哈希函数 (extractable CRHF) 替代了 Micali94 中的 ROM^[70]. 近年, Chiesa 和 Yogev 改进了 Micali94 的通信复杂度^[95], 并对其安全性进行了更为详细的探讨^[96].

然而, 这些论证的实际性能均较差, 难以落地应用。例如, 针对 25×25 的矩阵乘法问题, 如果利用一个基于 PCP 的经典零知识证明方案^[97] 构造协议, 那么该协议的实际证明和验证时间就已超过亿年^[20]。

ZKBoo/ZKB++/KKW18. ZKBoo^[33]、ZKB++^[34] 和 KKW18^[47] 的主要思路均是基于 MPC-in-the-Head^[98] 的思想直接构造了高效零知识 PCP, 随后将该零知识 PCP 转换为 NIZKAO_K。其核心思路是证明者在脑海中模拟一个安全多方计算协议的运行并生成 MPC 参与方数目的视图, 然后验证者随机挑选若干视图验证正确性和一致性, 协议的零知识性由安全多方计算协议的隐私性保障。从 PCP 的角度, 证明者生成的视图就是谕示, 验证者挑选的视图就是访问请求。本文第 8.3.1 小节介绍 ZKBoo 和 ZKB++, 第 8.3.2 小节介绍 KKW18。

4.2.2 基于 Linear-PCP 的零知识证明

Bitansky 等人^[58] 指出, zk-SNARK (包括 GGPR13^[19]、Pinocchio^[20]、Groth16^[52]、GKMM18^[41] 等) 均是基于 Linear-PCP 实现的。其步骤为首先将 Linear-PCP 转换为 LIP, 再将 LIP 转换为 SNARK, 最后将 SNARK 转换为 zk-SNARK。首先, 将 Linear-PCP 转换为 LIP 是自然的, 见定义 19。其次, 利用一种特殊的密码编码方法 (可基于 KEA 假设实现, 其需具有单向性、允许公开验证二次等式、保障证明者只能进行线性运算, 详见第 5.2.2 小节), 任意 LIP 都可转换为特定验证者的 SNARK, 具有低度验证者 (low-degree verifier) 的 LIP 可转换为公开可验证的 SNARK。然后, 通过随机化处理, SNARK 可转换为 zk-SNARK。此外, Bitansky 等人给出了将若干具体 PCP^[59, 99, 100] 转换为 Linear-PCP 的方法。基于 Bitansky 等人构造 zk-SNARK 的思路, Ben-Sasson 等人^[81] 及 Groth^[52] 通过构造高效的 Linear-PCP 和 LIP 优化了 zk-SNARK 的理论和实际性能。本文第 5.3.1 小节介绍 Pinocchio, 第 5.3.2 小节介绍 Groth16, 第 5.3.3 小节介绍 GKMM18。

4.2.3 基于 IPCP 的零知识证明

Ligero. Ligero 系列协议^[35, 36, 48] 均是基于 IPCP 的零知识证明, 其也是利用 MPC-in-the-Head 的思想直接构造了零知识的 IPCP。与基于 PCP 的零知识证明不同, 基于 IPCP 的零知识证明允许证明者生成谕示后根据验证者的随机挑战构造新谕示并进一步完成证明。具体的, 在 Ligero 系列协议中, 证明者首先利用 RS 码将证据编码为一个 $m \times n$ 的矩阵, 然后根据验证者的随机挑战 $r \in \mathbb{F}^n$ 计算矩阵行与 r 的线性组合返回给验证者, 验证者通过该线性组合验证协议的正确性, 协议的可靠性由 RS 码保障, 零知识性由 RS 码和随机掩藏多项式保障。本文第 8.3.3 小节介绍 Ligero 和 Ligero++, 第 8.3.4 小节介绍 BooLigero。

4.2.4 基于 IOP 的零知识证明

Ben-Sasson、Chiesa 和 Spooner^[90] 指出, 在 ROM 下 IOP 可以被转换为非交互论证, 并可利用已有的零知识编译方法, 构造系统参数可公开生成的简洁 NIZKAO_K。同基于 PCP 的零知识证明类似, 基于 IOP 的零知识证明也具有仅需对称密钥操作、可抗量子的优点; 不同的是, 基于 IOP 的零知识证明具有更好的性能^[66, 101]。基于不同底层关键技术构造 IOP, 如 sum-check 协议、RS 码、MPC-in-the-Head 等可构造性能不同的简洁 NIZKAO_K, 分列如下。

基于 DEIP 的零知识证明. Sum-check 协议^[63] 可用于证明某函数的遍历求和值等于公开值 (具体见定义 30)。Sum-check 协议本质上属于 IOP, 其中 \mathcal{P} 的证明谕示为 $g(x_1, x_2, \dots, x_\ell)$, \mathcal{V} 的访问请求次数为 1, 即 $g(r_1, r_2, \dots, r_\ell)$ 。基于 sum-check 协议, Goldwasser、Kalai 和 Rothblum^[61] 提出了一个针对分层算术电路求值问题的交互式证明, 该协议由于证明和验证复杂度均较低, 故被称为双向高效的交互式证明 (doubly efficient interactive proof, DEIP)。后续的研究如文献 [26–29, 32, 51] 利用 Cramer 和 Damgård 的转换方法^[102] 将 DEIP 转换为简洁零知识知识论证, 并利用 Fiat-Shamir 启发式转换为简洁 NIZKAO_K。该类零知识证明的主要思路、构造方法、性能表现和典型协议详见第 6 章。

Aurora. Aurora 由 Ben-Sasson 等人^[50] 提出, 他们构造了针对 R1CS 可满足问题的 IOP, 并利用已有的零知识编译方法^[101], 实现了证明复杂度为 $O(n \log n)$ 、通信复杂度为 $O(\log n)$ 、验证复杂度为 $O(n)$ 的简洁 NIZKAO_K, 其中 n 为 R1CS 可满足问题的规模。Aurora 的主要思路如下。

首先将 R1CS 可满足问题转换为两种检查, 即列检查和行检查。其中, 列检查为给定向量 $a, b, c \in \mathbb{F}^{m \times 1}$, 检查 $a \odot b = c$; 行检查为给定向量 $a, b \in \mathbb{F}^{m \times 1}$ 及矩阵 $M \in \mathbb{F}^{m \times m}$, 检查 $Ma = b$ 。显然, R1CS

可满足问题是可满足的当且仅当对 $\mathbf{y}_A = \mathbf{Az}$, $\mathbf{y}_B = \mathbf{Bz}$, $\mathbf{y}_C = \mathbf{Cz}$ 的行检查成立和对 $\mathbf{y}_A \odot \mathbf{y}_B = \mathbf{y}_C$ 的列检查成立.

然而, 平凡的列检查和行检查均需要 $O(n)$ 级别的通信量. 为实现亚线性级别的通信复杂度, Aurora 随后将向量 $\mathbf{x}, \mathbf{y}, \mathbf{z}$ 利用 RS 码进行编码. 这是因为 RS 码具有纠错性质, 也就是即使是距离很小的消息向量在经过 RS 码编码后其码距也会显著增大, 所以只需访问请求编码的一部分即可保障可靠性. 令 $\text{RS}[L, \rho]$ 表示码长集合为 L 、码率为 ρ 的 RS 码. 记 RS 码 $f, g, h \in \text{RS}[L, \rho]$, 其对应的唯一码多项式(度小于 $\rho|L|$) 分别为 $f(\cdot), g(\cdot), h(\cdot)$, 则列检查等价于给定集合 $H \subset \mathbb{F}$, 检查对于任意的 $a \in H$, $f(a)g(a) - h(a) \stackrel{?}{=} 0$; 行检查等价于给定 RS 码 $f, g \in \text{RS}[L, \rho]$, 其对应的唯一码多项式分别为 $f(\cdot), g(\cdot)$, 给定集合 $H \subset \mathbb{F}$ 和矩阵 $\mathbf{M} \in \mathbb{F}^{|H| \times |H|}$, 检查对于任意的 $a \in H$, $g(b) = \sum_{a \in H} \mathbf{M}_{b,a} \cdot f(a)$.

接着, 对于列检查, Aurora 援引 Ben-Sasson 和 Sudan 的标准概率检查方法^[103] 构造了对应的 IOP, 其主要思路如下. 对于任意的 $a \in H$, $f(a)g(a) - h(a) = 0$ 等价于存在 $p(x)$, 使得 $f(x)g(x) - h(x) = p(x)Z_H(x)$, 其中 $Z_H(x)$ 是度为 $|H|$ 且在集合 H 上均为 0 的唯一多项式, 则列检查可等价转换为对多项式 $p(x)$ 的低度检查问题, 即检查码 $p \in \text{RS}[L, 2\rho - |H|/|L|]$. 对于行检查, 注意到其问题形式为求和, 故可利用 sum-check 协议; 然而与一般 sum-check 协议不同的是, 行检查的求和函数不是多变量而是单变量, sum-check 协议的核心思路是每轮将多变量求和函数转换为单变量函数, 针对单变量求和函数直接调用 sum-check 协议是困难的. 基于此, Aurora 指出当集合 H 是 \mathbb{F} 的陪集时, 可构造单变量求和验证协议, 并利用该协议构造通信复杂度为 $O(\log d)$ 的 IOP, 其中 d 为求和函数的度. 最后, Aurora 利用 Ben-Sasson 等人的零知识编译方法^[101] 实现了上述 IOP 的零知识并最终构造了简洁 NIZK AoK.

Stark. Stark 由 Ben-Sasson 等人^[43] 提出, 是一种针对可用对数空间可计算电路表示的陈述的非交互零知识论证, 比如随机存取机 (random access machine) 上的有界停机问题 (bounded halting problem). 给定一个程序 P 和时间上界 $T(n)$, 若其可用空间大小为 $O(\log T(n))$ 的电路表示, 则 Stark 可以证明 “ P 可在 T 步内接受”, 且生成证明的时间为 $O(T \log T)$, 证明的长度为 $O(\log T)$, 验证时间为 $O(|P| + \text{poly log } T)$, 这相比于平凡验证时间 $\Omega(|P| + T)$ 有显著优化.

同 Aurora 相比, Stark 仅支持均匀计算 (uniform computation) 问题, 而 Aurora 可支持非均匀计算问题 (如非均匀电路). 事实上, $O(|C|)$ 规模的 C-SAT 问题也可转换为 $\{|P|, T = \Omega(|C|)\}$ 的有界停机问题. 此时, Stark 的证明复杂度为 $O(|C| \log^2 |C|)$, 通信复杂度为 $O(\log |C|)$, 验证复杂度为 $O(|C|)$. 由于 Stark 不能直接针对 C-SAT 问题, 本文不再详述.

Limbo. Limbo 由 Guilhem、Orsini 和 Tanguy^[49] 提出, 其拓展了 Ligero 中的 MPC 模型并基于 IOP 实现, 是一种虽然通信复杂度与电路规模成线性关系, 但是实际性能良好的 NIZK AoK. 对于算术电路, 相比于文献 [33–36], Limbo 实现了针对中等规模 C-SAT 问题 (乘法门少于 500 000 个) 的最优实际性能. 具体的, Limbo 给出了一个较为适合 MPC-in-the-Head 的 MPC 模型, 即客户-服务器模型 (client-server model), 然后基于该模型和 IOP 实现了乘法门约束的高效验证进而构造了简洁 NIZK AoK. 本文第 8.3.5 小节介绍 Limbo.

4.3 本章小结

本章简要介绍了基于 PCP、IPCP、Linear-PCP 和 IOP 的零知识证明, 简单给出了若干典型协议的主要思路和底层关键技术, 并指出了这些简洁非交互零知识证明的构造方法——首先构建信息论安全证明, 然后利用密码编译器将信息论安全证明转换为简洁非交互零知识证明. 本书第 5–8 章将以底层关键技术为线索, 较为详细地介绍目前较为主流的几种简洁非交互零知识证明.

5 基于 QAP 的零知识证明

本章介绍基于二次算术程序 (QAP) 的零知识证明. 该类零知识证明又被称为 zk-SNARK, 其均基于 CRS 模型实现非交互. 从信息论安全证明的角度, zk-SNARK 是基于 Linear-PCP^[58, 62] 及 LIP^[58] 实现的; 从密码编译器应用的底层关键技术角度, 其大多利用 QAP^[19] 将 C-SAT 问题归约为一组多项式的约束并利用双线性配对验证上述约束进而实现. 虽然早期的 zk-SNARK^[91, 104] 并不是基于 QAP 实现的, 并且 Bitansky 等人^[70] 指出利用知识可提取假设就足以构造 zk-SNARK, 但是不论是在理论还是实际应

用层面, 大部分 zk-SNARK 均是基于 QAP 及其变种^[19, 20, 22, 105–110] 实现的, 故本文将该类零知识证明称为基于 QAP 的零知识证明.

zk-SNARK 协议较多, 且 Nitulescu^[55] 对 zk-SNARK 已有较为详细的介绍, 本文仅介绍若干典型协议, 其性能表现、底层难题假设、关键技术等列于表 4, zk-SNARK 的其他协议及具体细节可参考文献 [55].

表 4 部分 zk-SNARK 总结
Table 4 Summary of several zk-SNARK

对比项	协议				
	Groth10 ^[91]	GGPR13 ^[19]	Pinocchio ^[20]	Groth16 ^[52]	GKMM18 ^[41]
CRS 规模	$\Theta(C ^2) \mathbb{G}$	$O(C) \mathbb{G}$	$O(C) \mathbb{G}$	$O(C) \mathbb{G}$	全局: $O(C_M ^2) \mathbb{G}$ 具体关系: $O(C) \mathbb{G}$
待证明陈述表示形式	算术电路	算术电路	算术电路	算术电路	算术电路
证明复杂度	$\Theta(C ^2) E$	$O(C \log C) \mathbb{F}_o$ $O(C) E$	$O(C \log C) \mathbb{F}_o$ $O(C) E$	$O(C) E$	$O(C) E$
验证复杂度	$\Theta(C) M$ $\Theta(1) P$	$O(\text{io}) E$ $14 P$	$O(\text{io}) E$ $11 P$	$O(\text{io}) E$ $4 P$	$O(\text{io}) E$ $5 P$
通信复杂度	42 \mathbb{G}	9 \mathbb{G}	8 \mathbb{G}	2 $\mathbb{G}_1, 1 \mathbb{G}_2$	2 $\mathbb{G}_1, 1 \mathbb{G}_2$
底层难题假设	q -PKE, q -PDH	q -PKE, q -PDH	q -PKE, q -PDH, q -SDH	GGM, LO	q -MK, q -MC ^[41]
关键技术	向量承诺 双线性配对	QAP 双线性配对	QAP 双线性配对	QAP, LIP 双线性配对	QAP 双线性配对

¹ \mathbb{G} 表示群元素, \mathbb{F} 表示域元素, \mathbb{F}_o 表示域上运算, E 表示群幂运算, P 表示双线性群上的配对运算. $|C|$ 表示电路规模, $|C_M|$ 表示电路中乘法门的数目, io 指公共输入输出, 且有 $|\text{io}| = |\mathbf{x}| + |\mathbf{y}|$. \mathbb{G}_1 和 \mathbb{G}_2 分别指双线性映射群上的元素. GGM 指通用群模型 (generic group model), LO 指假设敌手只能利用验证者的消息进行线性/仿射运算. q -MK 假设指 q 阶单项式知识假设 (q -monomial knowledge assumption), 是对 q -PKE 假设的拓展, q -MC 假设指 q 阶单项式计算假设 (q -monomial computational assumption).

² GKMM18 在生成 CRS 时分为两步, 首先生成一个与待证明陈述独立的全局 CRS, 然后再针对具体关系生成对应的 CRS.

³ 基于 QAP 的 zk-SNARK 可将验证者的群幂运算移至预处理中, 此时验证者计算开销可为常数个配对运算. 此外, 与公共输入输出成线性关系的验证复杂度为理论最低, 因为验证者起码要读取公共输入输出, 而这一步的开销起码为 $O(|\text{io}|)$.

⁴ 证明、通信和验证复杂度均为协议 1 轮的主要开销, 协议运行轮数及实际证明、验证计算开销和通信量与可靠性误差和目标可靠性误差有关.

5.1 定义及概念

定义 22 (SNARG 与 SNARK^[25, 58, 81]) SNARG 是算法三元组 $\Pi = (\text{Setup}, \text{Prove}, \text{Verify})$. 给定一个多项式时间内可判定的二元关系 \mathcal{R} 及其对应的 NP 语言 $\mathcal{L}(\mathcal{R})$, 启动算法 $(\sigma, \tau) \leftarrow \text{Setup}(1^\lambda, \mathcal{R})$ 以安全参数 λ 的一元表示和关系 \mathcal{R} 为输入, 生成参考串 σ 和模拟陷门 τ . 证明生成算法 $\pi \leftarrow \text{Prove}(\mathcal{R}, \sigma, x, w)$ 用于生成证明 π . 验证算法 $0/1 \leftarrow \text{Verify}(\mathcal{R}, \sigma, x, \pi)$ 用于验证证明.

称三元组算法 $(\text{Setup}, \text{Prove}, \text{Verify})$ 为公开可信预处理 SNARG (publicly verifiable preprocessing succinct non-interactive argument, SNARG) 当如下条件满足.

- 完备性. 对于任意的 $x \in \mathcal{L}(\mathcal{R})$, 若算法 Setup 和 Prove 正确运行, 则验证者 \mathcal{V} 一定会接受.
- 可靠性. 对于任意 PPT 的 \mathcal{P}^* , 任意的 $x \notin \mathcal{L}(\mathcal{R})$, \mathcal{P}^* 能使 \mathcal{V} 接受的概率不超过 $\text{negl}(\lambda)$.
- 高效性. Setup 的运行时间为 $\text{poly}(\lambda + |x|)$, Prove 的运行时间为 $\text{poly}(\lambda + |x|)$, Verify 的运行时间为 $\text{poly}(\lambda + |x|)$.
- 简洁性. 证明者发送的消息规模不超过 $\text{poly}(\lambda)(|x| + |w|)^{o(1)}$.

如果 Setup 的运行时间为 $\text{poly}(\lambda + \log |x|)$, 则称一个 SNARG 为完全简洁的 (fully succinct)^[111]. SNARK (succinct non-interactive argument of knowledge) 是指具有 (计算意义) 的知识可靠性的 SNARG, zk-SNARK 是指具有零知识性的 SNARK. 其中, (计算意义) 的知识可靠性是指如果敌手能够生成一个针对某语言的有效证据, 那么就存在一个多项式的提取器可将这个有效证据提取出来, 且该提取器能够访问敌手的任意状态. 具体而言, 知识可靠性是指对于任意的多项式敌手 \mathcal{A} , 都存在一个多项式

时间的提取器 $\mathcal{X}_{\mathcal{A}}$, 使得下式成立

$$\Pr[(\sigma, \tau) \leftarrow \text{Setup}(1^\lambda, \mathcal{R}); (x, \pi) \leftarrow \mathcal{A}(\mathcal{R}, \sigma); w \leftarrow \mathcal{X}_{\mathcal{A}}(R, x, \sigma, \pi) : (x, w) \notin \mathcal{R} \wedge \text{Verify}(\mathcal{R}, \sigma, x, \pi) = 1] \leq \text{negl}(\lambda). \quad (7)$$

定义 23 (二次算术程序^[19]) 二次算术程序 (quadratic arithmetic program, QAP) 是二次张成程序 (quadratic span program, QSP)^[19] 在算术电路上的自然扩展, 而后者是对张成程序 (span program)^[112] 的扩展。域 \mathbb{F} 上的 QAP $\mathcal{Q} = (t(z), \mathcal{U}, \mathcal{W}, \mathcal{Y})$ 包含三组多项式 $\mathcal{U} = \{u_k(z)\}, \mathcal{W} = \{w_k(z)\}, \mathcal{Y} = \{y_k(z)\} (k \in 0 \cup [m])$ 和目标多项式 $t(z)$ 。记公共输入为 (c_1, c_2, \dots, c_N) , 则称 \mathcal{Q} 是可满足的当且仅当存在系数 $(c_{N+1}, c_{N+2}, \dots, c_m)$ 使得 $t(z)$ 整除 $p(z)$, 其中

$$p(z) = \left(u_0(z) + \sum_{k=1}^m c_k \cdot u_k(z) \right) \cdot \left(w_0(z) + \sum_{k=1}^m c_k \cdot w_k(z) \right) - \left(y_0(z) + \sum_{k=1}^m c_k \cdot y_k(z) \right), \quad (8)$$

即存在多项式 $h(z)$ 使得 $p(z) - h(z)t(z) = 0$. 称 \mathcal{Q} 的规模为 m , \mathcal{Q} 的度为 $t(z)$ 的度, 即 d .

强 QAP (strong QAP)^[19] 是指对于任意一组 $(a_1, a_2, \dots, a_m, b_1, b_2, \dots, b_m, c_1, c_2, \dots, c_m)$, 如果其构成的 $p(z)$ 可被 $t(z)$ 整除, 则有 $(a_1, a_2, \dots, a_m) = (b_1, b_2, \dots, b_m) = (c_1, c_2, \dots, c_m)$, 其中

$$p(z) = \left(u_0(z) + \sum_{k=1}^m a_k \cdot u_k(z) \right) \cdot \left(w_0(z) + \sum_{k=1}^m b_k \cdot w_k(z) \right) - \left(y_0(z) + \sum_{k=1}^m c_k \cdot y_k(z) \right). \quad (9)$$

自然的, QAP 可视为 3 组 Linear-PCP, 每组访问请求次数为 1.

定义 24 (q -PKE 假设^[91]) 指数知识假设 (knowledge of exponent assumption, KEA)^[113] 是指给定 p 阶群 \mathbb{G} 和群 \mathbb{G}_T 、双线性映射关系 $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T, \mathbb{G}$ 上生成元 g 及 g^α , 在不知道 a 的情况下难以构造 \hat{c}, c 且 $\hat{c} = c^\alpha$, 其中 $c = g^a, \hat{c} = (g^\alpha)^a$. q 阶指数知识假设 (q -Power PKE, 即 q -PKE 假设) 是指给定 $(g, g^x, \dots, g^{x^q}, g^\alpha, g^{\alpha x}, \dots, g^{\alpha x^q})$, 在不知道 a_0, a_1, \dots, a_q 的情况下难以构造 \hat{c}, c 且 $\hat{c} = c^\alpha$, 其中 $c = \prod_{i=0}^q (g^{x^i})^{a_i}, \hat{c} = \prod_{i=0}^q (g^{\alpha x^i})^{a_i}$. 值得注意的是, q -PKE 假设是一种不可证伪 (non-falsifiable) 的假设^[24].

定义 25 (q -PDH 假设^[91]) q -PDH 假设 (q -power Diffie-Hellman assumption) 是指给定 p 阶群 \mathbb{G} 和群 \mathbb{G}_T 、双线性映射关系 $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T, \mathbb{G}$ 上生成元 $g, x \xleftarrow{\$} \mathbb{Z}_p^*$ 及 $(g, g^x, \dots, g^{x^q}, g^{x^{q+2}}, \dots, g^{x^{2q}})$, 难以计算 $g^{x^{q+1}}$.

定义 26 (q -SDH 假设^[114]) q -SDH 假设 (q -strong Diffie-Hellman assumption) 是指给定 p 阶群 \mathbb{G} 和群 \mathbb{G}_T 、双线性映射关系 $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T, \mathbb{G}$ 上生成元 $g, x \xleftarrow{\$} \mathbb{Z}_p^*$ 及 (g, g^x, \dots, g^{x^q}) , 难以计算得到 (y, c) , 其中 $y = e(g, g)^{1/(x+c)}, c \in \mathbb{Z}_p^*$.

5.2 背景及主要思路

5.2.1 背景

2006 年, Groth、Ostrovsky 和 Sahai^[115] 利用双线性配对构造了第一个基于标准假设且通信复杂度为线性的非交互零知识证明. 2010 年, Groth^[91] 基于 CRS 模型、 q -PKE 假设和 q -PDH 假设, 利用双线性配对实现了第一个不需依赖 ROM 且通信量为 42 个群元素的 zk-SNARK. Groth 的核心思路是将 C-SAT 问题归约为一组等式并利用双线性配对验证等式成立, 然而该协议的 CRS 规模和证明复杂度均为 $O(|C|^2)$. Lipmaa^[104] 将上述协议的 CRS 规模降低到了 $O(|C| \log |C|)$, 但证明复杂度仍为 $O(|C|^2)$.

2013 年, Gennaro 等人^[19] 提出了 QAP, 其是一种新的 NP 语言且利用 QAP 可将算术电路可满足问题快速归约为 QAP 可满足问题, 即判断是否存在一个多项式能够被某个公开确定多项式整除的问题. Gennaro 等人利用强 QAP 构造了 CRS 规模为 $O(|C|)$ 、证明复杂度为 $O(|C| \log |C|)$ 、通信量为 9 个群元素的 zk-SNARK (记作 GGPR13). 同年, Parno 等人^[20] 提出了 Pinocchio, 在将通信量进一步降低到了 8 个群元素的同时弱化了 GGPR13 对 QAP 的限制, 即利用一般 QAP 构造了 zk-SNARK, 这

将 GGPR13 中的 CRS 规模降低了约 2/3, 同时也降低了预处理时间和证明者计算开销. Pinocchio 具有良好的实际性能, 在一定程度上促使了零知识证明的落地应用. Zcash [7] 就是基于该类零知识证明 [22] 所构造的一种隐私密码货币, 能够在防止双花的同时实现交易的匿名性. 在 Pinocchio 之后, 一系列研究着力于优化该类零知识证明的实际性能 [22, 107–109] 并应用于不同场景中, 例如对认证数据的隐私保护证明 [108]、大数据计算 [116, 117] 和可验证计算中. 特别的, 对于可验证计算, 其思路主要是先将 C 语言程序转换为某种编程语言 (例如具有固定内存访问和控制流的 C 语言程序 [20]、RISC [22, 81]、RAM [65, 116] 等), 再将中间语言转换为电路并调用针对电路可满足问题的零知识证明完成计算; 也有直接构造针对算术电路变种 (如集合电路 [107]) 的零知识证明从而完成可验证计算的.

除了改进 zk-SNARK 的性能之外, 还有系列研究探讨了 zk-SNARK 的特征和性质. Gentry 和 Wichs [25] 指出基于黑盒归约和可证伪假设无法构造 SNARG. Bitansky 等人 [70] 指出构造 zk-SNARK 必须依赖于可提取抗碰撞哈希函数, 同时文献 [111] 指出基于 PCD 系统 (proof-carrying data system), 任何 zk-SNARK 都可高效转换为完全简洁的 zk-SNARK. Groth [52] 基于非对称双线性映射构造了通信量仅为 3 个群元素、验证者计算开销为 3 个配对运算的 zk-SNARK. 此外, Groth 还指出基于通用非对称双线性群模型 (general asymmetric bilinear group model) [118] 无法构造通信复杂度为 1 个群元素的 zk-SNARK.

Linear-PCP. Bitansky 等人 [58] 指出, zk-SNARK 均是基于 Linear-PCP 实现的, 其步骤如下. 首先将线性概率可验证证明转换为线性交互式证明, 该转换是自然的, 见定义 19; 再将线性交互式证明转换为 SNARK, 利用一种特殊的密码编码方法 (可基于指数知识假设假设实现, 其需具有单向性、允许公开验证二次等式、保障证明者只能进行线性运算, 详见第 5.2.2 小节), 任意 LIP 都可转换为特定验证者的 SNARK, 具有低度验证者 (low-degree verifier) 的 LIP 可转换为公开可验证的 SNARK; 最后通过引入随机化处理, 可将 SNARK 转换为 zk-SNARK. 此外, Bitansky 等人也给出了将若干具体 PCP [59, 99, 100] 转换为 Linear-PCP 的方法. Bitansky 等人提供了一种构造 zk-SNARK 的新思路. 基于该种思路, Ben-Sasson 等人 [81] 及 Groth [52] 通过构造高效 Linear-PCP 和 LIP 优化了 zk-SNARK 的理论和实际性能.

可更新的零知识证明. 上述 zk-SNARK 存在两个问题, 第一个问题是协议需要安全生成的公共参考串, 若公共参考串中含有的秘密信息被攻击者获知, 则整个协议就不再具备可靠性. 启动阶段的私密性和区块链的去中心化产生了较为严重的矛盾, 在一定程度上影响了 zk-SNARK 的进一步应用. 为解决该问题, Ben-Sasson 等人 [79] 和 Bowe、Gabizon 及 Green [80] 指出可以利用安全多方计算生成公共参考串, 但如何选择参与方及如何保障安全性是一个新的难题. 第二个问题是公共参考串与陈述是相关的, 即待证明陈述改变后需重新进行预处理, 然而预处理阶段的复杂度为 $O(|C|^2 \log^2 |C|)$, 多次预处理会严重影响协议性能.

针对上述问题, Groth 等人 [41] 基于 QAP 提出了一种 CRS 全局且可更新 (updatable universal CRS) 的 zk-SNARK (记作 GKMMM18). 可更新是指对 CRS 安全性持怀疑态度的用户可以发起对 CRS 的更新请求, 只要旧 CRS 和更新发起者中有一个是诚实的, 新 CRS 就是安全的. 全局是指根据一个全局公共参考串可以生成多个针对具体电路的公共参考串, 其中全局公共参考串与待证明陈述独立, 可预先生成; 针对具体电路的公共参考串与陈述相关, 给定陈述后才能生成. GKMMM18 可根据 $O(|C_M|^2)$ 级别的全局 CRS 生成 $O(|C_M|)$ 级别的陈述相关 CRS. 此外, Groth 等人还指出 CRS 中仅含单项式的 zk-SNARK 易实现可更新, 而 CRS 中含有非单项式的 zk-SNARK 难以实现, 他们还证明了 Pinocchio 无法实现可更新.

GKMMM18 虽然实现了全局可更新的 CRS, 但一方面根据全局 CRS 构造陈述相关 CRS 需进行额外预处理, 另一方面更新 CRS 需要平方级别的群幂运算. 在此基础上, Maller 等人 [40] 提出的 Sonic 通过置换论证 (permutation argument)、大积论证 (grand-product argument) 等技术在代数群模型 (algebraic group model) [119] 下实现了 CRS 全局可更新的、规模为 $O(|C|)$ 、不需额外预处理的简洁 NIZKAoK. 后续的工作, 如 Plonk [23]、Marlin [42] 和 AuroraLight [120], 改进了 Sonic 的实际性能, 但也是基于代数群模型或知识假设实现的. 2020 年, Daza、Rafols 和 Zacharakis [54] 基于离散对数假设 (见第 7.1 节) 通过结构化承诺密钥改进了内积论证从而实现了 CRS 可更新的简洁 NIZKAoK (记作 DRZ20). 值得注意的是,

在这些可更新的零知识证明中, 只有 GKMM18 是基于 QAP 的, 本文第 5.3.3 小节介绍 GKMM18, 第 7.3.3 小节介绍 DRZ20.

5.2.2 主要思路

本小节以 Pinocchio^[20] 为例, 介绍 zk-SNARK 的构造思路. 首先介绍如何将 C-SAT 问题归约为 QAP 可满足问题, 然后介绍如何利用“掩藏”编码和双线性配对实现针对 QAP 可满足问题的 zk-SNARK.

将 C-SAT 问题归约为 QAP 可满足问题. 考虑一个算术电路 C . 记电路中的乘法门数为 $d = |C_M|$, 且为每个乘法门取随机根 $(r_1, r_2, \dots, r_d) \in \mathbb{F}^d$, 则目标多项式为 $t(z) = \prod_{g=1}^d (z - r_g)$. 记电路中导线的数目为 m (不计加法门的输出导线), 记每条导线值为 (c_1, c_2, \dots, c_m) , 记 (c_1, c_2, \dots, c_N) 为公共输入输出, $(c_{N+1}, c_{N+2}, \dots, c_m)$ 为秘密输入, 且记集合 $I_{\text{mid}} = \{N+1, N+2, \dots, m\}$. 定义贡献函数 $\{u_i(\cdot)\}, \{w_i(\cdot)\}, \{y_i(\cdot)\}$, 对于乘法门 $g \in [d]$, 若导线 $i (i \in [m])$ 对该门的左输入“有贡献”, 则有 $u_i(r_g) = 1$, 否则 $u_i(r_g) = 0$. 同理, $w_i(\cdot)$ 和 $y_i(\cdot)$ 则分别描述了导线对门右输入和输出是否“有贡献”. 其中对于左输入, “有贡献”是指导线 i 本身或仅经过若干加法门后为门 g 的左输入. 经此法构造 $p(z)$, 则 $p(z)$ 在 r_g 的取值即为门 g 所满足的约束. 因此, 电路是可满足的当且仅当对于所有的 $r_{g,g \in [m]}$ 都有 $p(r_g) = 0$, 即多项式 $p(z)$ 可被 $t(z) = \prod_{g=1}^d (z - r_g)$ 整除, 而这也就是 QAP 可满足问题的形式. 同时, QAP 串的规模为电路中的非加法门输出导线数 m , 度为乘法门的数目 d .

为了更好地理解“有贡献”的概念, 本小节在图 3 中给出一个例子. 对门 r_6 的左输入“有贡献”的是导线 c_1 和 c_2 , 故 $u_1(r_6) = u_2(r_6) = 1$; 对门 r_5 的左输入“有贡献”的是导线 c_3 , 故 $u_3(r_5) = 1$. $w_i(r_g)$ 和 $y_i(r_g)$ 同理, 其分别考虑的是对门右输入和输出的贡献, 且度至少为 $d-1$.

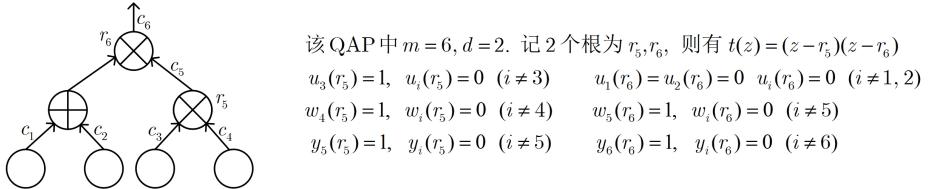


图 3 算术电路可满足问题与 QAP 可满足问题的归约^[19, 20]

Figure 3 Reduction from arithmetic C-SAT problem to QAP satisfiability problem

除 QAP 外, 还可将 C-SAT 问题归约为其他形式的可满足问题, 针对不同形式的具体问题, 其归约过程更为高效. 如针对布尔电路的 QSP^[19]、利用纠错码构造的高效 QSP^[105] 及 SSP (square span programs)^[106] 和针对算术电路的 SAP (square arithmetic programs)^[110]. 这些可满足问题的形式与 QAP 可满足问题的形式类似, 且基于其构造 zk-SNARK 的思路也是类似的, 本文只详细介绍 QAP.

针对 QAP 可满足问题构造 zk-SNARK 的主要思路. 针对 QAP 可满足问题构造 zk-SNARK 的主要思路列于图 4. 直接证明多项式 $p(z)$ 可被 $t(z)$ 整除可能是困难的, 而这可等价转换为证明者证明其拥有 $p(z)$ 和 $h(z)$, 且 $p(z) = h(z)t(z)$. 由于 $u(z), w(z), y(z)$ 和 $t(z)$ 可由验证者根据电路结构自行计算, 因此证明者可发送 $(c_{N+1}, c_{N+2}, \dots, c_m)$ 及 $h(z)$. 然而, 考虑到 $h(z)$ 的度约为 d , 直接发送 $h(z)$ 将导致 $O(|C_M|)$ 级别的通信复杂度. 利用 Schwartz-Zippel 引理可将传输多项式简化为传输多项式在某点的取值进而降低通信复杂度, 即验证者 \mathcal{V} 挑选随机挑战 $s \xleftarrow{\$} \mathbb{F}$, 随后 \mathcal{P} 返回 $p(s)$ 和 $h(s)$, \mathcal{V} 验证 $p(s) - h(s)t(s) \stackrel{?}{=} 0$.

然而, 上述方法既需要交互, 又不能保障恶意证明者在获知 s 后无法伪造 $p(s)$ 和 $h(s)$. 前者可通过 CRS 解决, 但在 CRS 中直接存储 s 仍无法解决后者, 因此需引入某种编码方式“掩藏” s . 需要注意的是, 若记该“掩藏”方式为 Enc , 则 Enc 至少需具备四个特点, 一是 $\text{Enc}(s)$ 具有一定的单向性, 获取 $\text{Enc}(s)$ 难以推知 s ; 二是为了实现公开可验证的 zk-SNARK, CRS 中不能存储秘密而只能存储公共信息, 如 $\text{Enc}(1), \text{Enc}(s), \dots, \text{Enc}(s^{d-1})$; 三是利用 $\text{Enc}(1), \text{Enc}(s), \dots, \text{Enc}(s^{d-1})$ 可构造多项式 $\text{Enc}(h(s)), \text{Enc}(p(s)), \text{Enc}(t(s))$, 即 Enc 支持线性运算; 四是 Enc 需支持二次等式的验证, 该运算用于验证

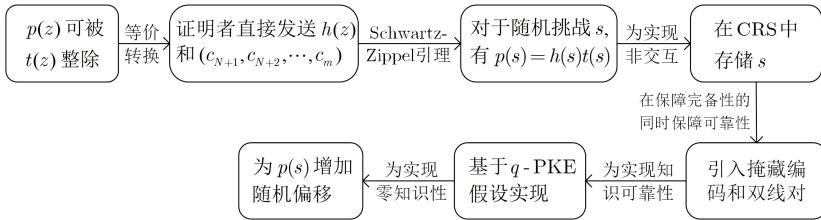


图 4 针对 QAP 可满足问题构造 zk-SNARK 的主要思路 [19, 20]

Figure 4 Main idea of constructing zk-SNARK for QAP satisfiability problem

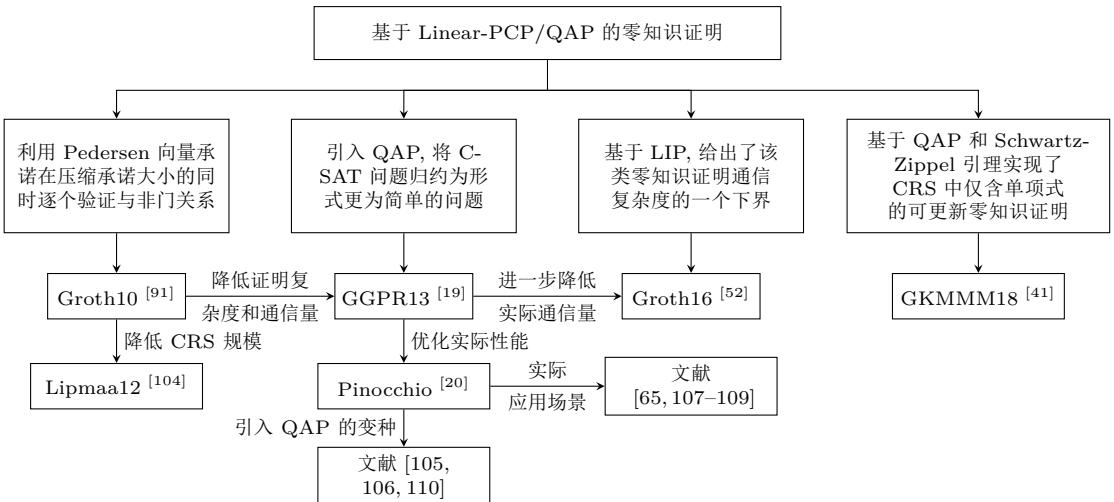
$\text{Enc}(p(s) - h(s)t(s)) \stackrel{?}{=} \text{Enc}(0)$. 事实上, 给定素数阶 p 的循环群 \mathbb{G}, \mathbb{G}_T , 若有双线性映射 $e : \mathbb{G} \times \mathbb{G} \rightarrow \mathbb{G}_T$ 且对于任意的 $a, b \in \mathbb{Z}_p$ 有 $e(g^a, g^b) = e(g, g)^{ab}$, 则 $\text{Enc}(a) = g^a$ 即可满足上述要求.

上述方法仍存在几个问题. 一是无法保障知识可靠性, 即无法保障证明者确实利用了 $(c_{N+1}, c_{N+2}, \dots, c_m)$ 构造 $\text{Enc}(p(s))$, 而知识可靠性可基于 q -PKE 假设得以保障; 二是可能无法保障零知识性, 验证者虽难以通过 $\text{Enc}(p(s))$ 和 $\text{Enc}(h(s))$ 直接推知私密信息, 但仍可能获知某些隐私信息, 这可通过随机化处理 $p(s)$ 从而实现统计意义的零知识性.

基于以上思路可构造通信复杂度为常数个群元素的零知识证明, 具体见第 5.3 节.

5.3 典型协议分析

本节介绍基于 QAP 的零知识证明典型协议 (见图 5), 分析各协议的构造思路、协议流程、复杂度及安全性. 本节第 5.3.1 小节介绍 Pinocchio, 第 5.3.2 小节介绍 Groth16, 第 5.3.3 小节介绍 GKMM18.

图 5 基于 QAP 的部分零知识证明协议优化思路
Figure 5 Optimization of several zero-knowledge proof based on QAP

5.3.1 Pinocchio

Pinocchio 由 Parno 等人 [20] 提出, 其通信复杂度为 8 个群元素, 且验证复杂度仅与输入输出成线性关系. 本小节简要介绍 Pinocchio 的主要思路、协议流程 (见协议 1) 和复杂度及安全性.

主要思路. Pinocchio 的主要思路与第 5.2.2 小节是类似的. 为实现非交互, 在 CRS 中存储随机挑战 s ; 为保障可靠性, 需要“掩藏” s , 即在 CRS 中存储 g^s ; 为保障完备性, 应使证明者根据 CRS 可计算 $h(s)$, 由于 $h(s)$ 的度为 $d-2$, 因此在 CRS 中需存储 $\{g^{s^i}\}$, 其中 $i \in 0 \cup [d-2]$. 为保障证明者构造了正确形式的 $p(s)$, 即 $p(s)$ 中证明者提供的 $(c_{N+1}, c_{N+2}, \dots, c_m)$ 和 $\{u_k(s)\}, \{w_k(s)\}$ 和 $\{y_k(s)\}$ 是形式正确的, 其中 $k \in I_{\text{mid}}$, $I_{\text{mid}} = \{N+1, N+2, \dots, m\}$, 第一需在 CRS 中存储 $\{g_u^{u_k(s)}\}, \{g_w^{w_k(s)}\}$ 和 $\{g_y^{y_k(s)}\}$; 第二验证者需通过公共输入输出构造完整的 $p(s)$ 和 $h(s)$ 进行检查从而保障可靠性; 第三需基于 q -PKE

协议 1 Pinocchio [20]

公共输入: 域 \mathbb{F} , 算术电路 $C : \mathbb{F}^{|\mathbf{x}|+|\mathbf{w}|} \rightarrow \mathbb{F}^{|\mathbf{y}|}$. 其中 $(\mathbf{x}, \mathbf{y}) = (c_1, c_2, \dots, c_N)$, $|\mathbf{x}| + |\mathbf{y}| = N$.
证明者秘密输入: $\mathbf{w} = (c_{N+1}, c_{N+2}, \dots, c_m)$, 记集合 $I_{\text{mid}} = \{N+1, N+2, \dots, m\}$.

1. 可信初始化阶段.

- (1) 由可信第三方将算术电路 C 的可满足性问题归约为 QAP 可满足问题, 构造对应 QAP 串 $(t(z), \mathcal{U}, \mathcal{W}, \mathcal{Y})$, 其规模为 m , 度为 d .
- (2) 由可信第三方生成相应参数. 生成生成元为 g 的群 \mathbb{G} 及双线性映射群 \mathbb{G}_T . 选取随机数 $r_u, r_w, r_y, s, \alpha_u, \alpha_w, \alpha_y, \beta, \gamma \xleftarrow{\$} \mathbb{F}$ 并令 $r_y \leftarrow r_u r_w, g_u \leftarrow g^{r_u}, g_w \leftarrow g^{r_w}, g_y \leftarrow g^{r_y}$.
- (3) 由可信第三方生成公共参考字符串. 证明者 \mathcal{P} 的参考串为

$$\begin{aligned} & \left\{ g_u^{u_k(s)} \right\}_{k \in I_{\text{mid}}}, \left\{ g_w^{w_k(s)} \right\}_{k \in I_{\text{mid}}}, \left\{ g_y^{y_k(s)} \right\}_{k \in I_{\text{mid}}}, \\ & \left\{ g_u^{\alpha_u u_k(s)} \right\}_{k \in I_{\text{mid}}}, \left\{ g_w^{\alpha_w w_k(s)} \right\}_{k \in I_{\text{mid}}}, \left\{ g_y^{\alpha_y y_k(s)} \right\}_{k \in I_{\text{mid}}}, \\ & \left\{ g_u^{s^i} \right\}_{i \in 0 \cup [d-2]}, \left\{ g_u^{\beta u_k(s)} g_w^{\beta w_k(s)} g_y^{\beta y_k(s)} \right\}_{k \in I_{\text{mid}}}, \\ & g_u^{\alpha_u t(s)}, g_w^{\alpha_w t(s)}, g_y^{\alpha_y t(s)}, g_u^{\beta t(s)}, g_w^{\beta t(s)}, g_y^{\beta t(s)}. \end{aligned}$$

验证者 \mathcal{V} 的参考串为

$$\left(g, g^{\alpha_u}, g^{\alpha_w}, g^{\alpha_y}, g^\gamma, g^{\beta\gamma}, g_y^{t(s)}, \left\{ g_u^{u_k(s)}, g_w^{w_k(s)}, g_y^{y_k(s)} \right\}_{k \in 0 \cup [N]} \right).$$

2. 证明者 \mathcal{P} 生成证明. \mathcal{P} 首先选取 $\delta_u, \delta_w, \delta_y \xleftarrow{\$} \mathbb{F}$, 然后他利用证据 \mathbf{w} 构造 $p(z)$ 并计算

$$\begin{aligned} p'(z) &\leftarrow (u_0(z) + u_{io}(z) + u_{\text{mid}}(z) + \delta_u t(z)) \cdot (w_0(z) + w_{io}(z) + w_{\text{mid}}(z) + \delta_w t(z)) \\ &\quad - (y_0(z) + y_{io}(z) + y_{\text{mid}}(z) + \delta_y t(z)), \\ h'(z) &\leftarrow p'(z)/t(z), \end{aligned}$$

其中, $u_{\text{mid}}(z) = \sum_{k \in I_{\text{mid}}} c_k \cdot u_k(z)$, $w_{\text{mid}}(z)$ 和 $y_{\text{mid}}(z)$ 同理. \mathcal{P} 利用参考串生成证明 π

$$\left(g_u^{u'_{\text{mid}}(s)}, g_w^{w'_{\text{mid}}(s)}, g_y^{y'_{\text{mid}}(s)}, g_u^{\alpha_u u'_{\text{mid}}(s)}, g_w^{\alpha_w w'_{\text{mid}}(s)}, g_y^{\alpha_y y'_{\text{mid}}(s)}, g^{h'(s)}, g_u^{\beta u'_{\text{mid}}(s)} g_w^{\beta w'_{\text{mid}}(s)} g_y^{\beta y'_{\text{mid}}(s)} \right),$$

其中, $u'_{\text{mid}}(s) = u_{\text{mid}}(s) + \delta_u t(s)$, 可根据 $\left\{ g_u^{u_k(s)} \right\}_{k \in I_{\text{mid}}}$ 计算得出, $w'_{\text{mid}}(s), y'_{\text{mid}}(s)$ 同理. 记 $h'(z) = \sum_{i=0}^{d-2} h'_i z^i$, 则 $g^{h'(s)}$ 可由 $\prod_{i=1}^d (g^{s^i})^{h'_i}$ 计算得出.

3. 验证者 \mathcal{V} 验证证明. 记 \mathcal{V} 收到的证明为 $\pi = (g^{U_{\text{mid}}}, g^{W_{\text{mid}}}, g^{Y_{\text{mid}}}, g^{\tilde{U}_{\text{mid}}}, g^{\tilde{W}_{\text{mid}}}, g^{\tilde{Y}_{\text{mid}}}, g^H, g^Z)$, 则 \mathcal{V} 进行如下三项检查.

- (1) \mathcal{V} 检查 \mathcal{P} 可以构造 U_{mid} , 即 \mathcal{P} 拥有 U_{mid} 的系数. $W_{\text{mid}}, Y_{\text{mid}}$ 同理. 具体的, \mathcal{V} 检查

$$e(g_u^{\tilde{U}_{\text{mid}}}, g) \stackrel{?}{=} e(g_u^{U_{\text{mid}}}, g^{\alpha_u}), e(g_w^{\tilde{W}_{\text{mid}}}, g) \stackrel{?}{=} e(g_w^{W_{\text{mid}}}, g^{\alpha_w}), e(g_y^{\tilde{Y}_{\text{mid}}}, g) \stackrel{?}{=} e(g_y^{Y_{\text{mid}}}, g^{\alpha_y}).$$

- (2) \mathcal{V} 检查 $t(s)$ 可以整除 $p(s)$. 具体的, \mathcal{V} 先计算 $g_u^{u_{io}(s)} \leftarrow \prod_{k \in [N]} (g_u^{u_k(s)})^{c_k}$, $g_w^{w_{io}(s)}, g_y^{y_{io}(s)}$ 同理.

\mathcal{V} 随后验证

$$e(g_u^{u_0(s)} g_u^{u_{io}(s)} g_u^{U_{\text{mid}}}, g_w^{w_0(s)} g_w^{w_{io}(s)} g_w^{W_{\text{mid}}}) \stackrel{?}{=} e(g_y^{t(s)}, g^H) e(g_y^{y_0(s)} g_y^{y_{io}(s)} g_y^{Y_{\text{mid}}}, g).$$

- (3) \mathcal{V} 检查 $U_{\text{mid}}, W_{\text{mid}}, Y_{\text{mid}}$ 是由同一组系数生成的. 具体的, \mathcal{V} 检查

$$e(g^Z, g^\gamma) \stackrel{?}{=} e(g_u^{U_{\text{mid}}} g_w^{W_{\text{mid}}} g_y^{Y_{\text{mid}}}, g^{\beta\gamma}).$$

输出: 比特 b , 当且仅当上述三项检查均通过, 输出 $b = 1$; 否则输出 $b = 0$.

假设证明拥有 $(c_{N+1}, c_{N+2}, \dots, c_m)$. 为保证证明者在 $p(s)$ 中使用的系数是一致的, 还需检查一致性, 即协议 1 中的 β . 为实现零知识性, 需引入随机化处理, 即协议 1 中的 δ .

协议流程. Pinocchio 的协议流程如协议 1 所示. 在可信初始化阶段, 由可信第三方将 C-SAT 问题归约为 QAP 可满足问题, 并生成证明者和验证者的公共参考串. 在生成证明阶段, 证明者利用证据 w 生成证明 π , 其规模为 8 个群元素. 在验证证明阶段, 验证者共需验证五个配对等式, 其中第一项检查用于验证证明者确实拥有 $U_{\text{mid}}(s)$ 、 $W_{\text{mid}}(s)$ 和 $Y_{\text{mid}}(s)$ 的系数; 第二项检查用于验证 $t(s)$ 可以整除 $p(s)$; 第三项检查用于验证 U_{mid} 、 W_{mid} 和 Y_{mid} 是由同一组系数生成的.

讨论总结. 现简要分析 Pinocchio 可被视为基于 Linear-PCP 构造的原因. 首先, 可以认为谕示证明是

$$\left(g_u^{u'_{\text{mid}}(z)}, g_w^{w'_{\text{mid}}(z)}, g_y^{y'_{\text{mid}}(z)}, g_u^{\alpha_u u'_{\text{mid}}(z)}, g_w^{\alpha_w w'_{\text{mid}}(z)}, g_y^{\alpha_y y'_{\text{mid}}(z)}, g^{h'(z)}, g_u^{\beta u'_{\text{mid}}(z)} g_w^{\beta w'_{\text{mid}}(z)} g_y^{\beta y'_{\text{mid}}(z)} \right),$$

其中 $u'_{\text{mid}}(z)$ 、 $w'_{\text{mid}}(z)$ 、 $y'_{\text{mid}}(z)$ 、 $h'(z)$ 均是以 z 为自变量的多项式. 而验证者仅访问了多项式上的一个点, 即

$$\left(g_u^{u'_{\text{mid}}(s)}, g_w^{w'_{\text{mid}}(s)}, g_y^{y'_{\text{mid}}(s)}, g_u^{\alpha_u u'_{\text{mid}}(s)}, g_w^{\alpha_w w'_{\text{mid}}(s)}, g_y^{\alpha_y y'_{\text{mid}}(s)}, g^{h'(s)}, g_u^{\beta u'_{\text{mid}}(s)} g_w^{\beta w'_{\text{mid}}(s)} g_y^{\beta y'_{\text{mid}}(s)} \right),$$

因此可被视为是一种 PCP. 另外, 显然证明 π 与证明者的参考串存在线性关系, 因此 Pinocchio 可被视为基于 Linear-PCP 构造.

接着分析 Pinocchio 的复杂度和安全性. 首先分析复杂度. 在预处理阶段, 将算术电路可满足问题归约为 QAP 可满足问题, 其主要开销为根据点值构造度为 d 的共 $3m$ 个多项式, 若利用 FFT 计算拉格朗日插值^[121], 该阶段的计算开销为 $O(md \log^2 d) < O(|C|^2 \log^2 |C_M|)$ 次域上运算. 由于 CRS 中至少包含 d 个值且与 m 有关, 故其长度为 $O(|C|)$ 个群元素. 证明者的主要计算开销为 $O(|C| \log |C|)$ 次域上运算和 $O(m + d)$ 次群幂运算, 前者用于计算 $h'(x)$, 后者用于生成证明 π . 通信复杂度为 8 个群元素. 验证者的计算开销为 $O(|x| + |y|)$ 次群幂运算和 11 次配对运算. 若将计算 $u_{io}(s)$ 、 $w_{io}(s)$ 和 $y_{io}(s)$ 的任务交与预处理阶段, 则验证复杂度可为 $O(1)$ 次配对运算.

接着分析安全性. 基于 d -PKE、 q -PDH 和 $2q$ -SDH 假设, Pinocchio 可被证明是知识可靠的^[20]. 其中 d -PKE 假设可保障证明者 \mathcal{P} 拥有证据 $w = (c_{N+1}, c_{N+2}, \dots, c_m)$ (协议 1 第 3 步的 (1)), q -PDH 假设和 $2q$ -SDH 假设可保障知识可靠性. 对于零知识性, Pinocchio 通过随机化处理 $u_{\text{mid}}(z)$, 即引入 $\delta_{ut}(z)$ ($w_{\text{mid}}(z), y_{\text{mid}}(z)$ 同理) 从而实现了统计意义的零知识性.

5.3.2 Groth16

在限定敌手只能进行线性/仿射运算的情况下, Groth^[52] 基于 QAP 构造了通信量仅为 3 个元素的 LIP, 并基于该 LIP 构造了通信量为 3 个群元素、验证者计算开销仅为 4 个配对运算的 zk-SNARK (记为 Groth16). 现介绍 Groth16 的主要思路、协议流程并分析复杂度和安全性.

主要思路. Bitansky 等人^[58] 指出 zk-SNARK 均可视为基于 Linear-PCP 和 LIP 实现的, 在此基础上 Groth 给出了一个更为详细的 LIP 定义.

给定形式如下的算法三元组 (**Setup**, **Prove**, **Verify**), 如果其具有完美完备性和统计意义的知识可靠性 (对于只能进行线性/仿射运算的敌手), 则称该算法组是 LIP.

- $(\sigma, \tau) \leftarrow \text{Setup}(1^\lambda, \mathcal{R})$. 参数生成算法以安全参数 λ 的一元表示和 NP 关系 \mathcal{R} 为输入, 生成参考串 σ 和模拟陷门 τ .
- $\pi \leftarrow \text{Prove}(\mathcal{R}, \sigma, x, w)$. 证明生成算法分为两步, 首先运行 $\Pi \leftarrow \text{ProofMatrix}(\mathcal{R}, x, w)$ 生成矩阵 Π , 其中 **ProofMatrix** 为 PPT 的矩阵生成算法, 然后计算证明 $\pi \leftarrow \Pi\sigma$.
- $0/1 \leftarrow \text{Verify}(\mathcal{R}, \sigma, x, \pi)$. 验证算法分为两步, 首先运行确定性多项式算法 $\text{Test}(\mathcal{R}, x)$ 生成函数 $t(\cdot) \leftarrow \text{Test}(\mathcal{R}, x)$, 然后验证者检查 $t(\sigma, \pi) \stackrel{?}{=} 0$, 当且仅当其为 0 时接受, 否则拒绝.

在此基础上, 如果 LIP 具有完美零知识性, 则可利用其构造 zk-SNARK. 其中, 完美零知识性是指存

在 PPT 算法 Sim , 使得对于任意的 \mathcal{R} 及 $(x, w) \in \mathcal{R}$, 对于任意的敌手 \mathcal{A} , 都有

$$\begin{aligned} & \Pr \left[(\sigma, \tau) \leftarrow \text{Setup}(1^\lambda, \mathcal{R}); \pi \leftarrow \text{Prove}(\mathcal{R}, \sigma, x, w) : \mathcal{A}(\mathcal{R}, \sigma, \tau, \pi) = 1 \right] \\ &= \Pr \left[(\sigma, \tau) \leftarrow \text{Setup}(1^\lambda, \mathcal{R}); \pi \leftarrow \text{Sim}(\mathcal{R}, \tau, x) : \mathcal{A}(\mathcal{R}, \sigma, \tau, \pi) = 1 \right]. \end{aligned} \quad (10)$$

现简单介绍利用零知识 LIP 构造针对 QAP 可满足问题的思路. 考虑形如定义 23 的 QAP 串 $\mathcal{Q} = (t(z), \mathcal{U}, \mathcal{W}, \mathcal{Y})$, 一个最直观的想法是在参考串 σ 中存储 $\{s^i\}_{i=0}^{d-1}$ 和 $\{s^i t(s)/\delta\}_{i=0}^{d-2}$, 这样, 可生成证明 $\pi = \Pi\sigma = (A, B, C)$, 其中

$$A = \sum_{i=0}^m c_i u_i(s), \quad B = \sum_{i=0}^m c_i w_i(s), \quad C = \sum_{i=0}^m c_i y_i(s) + h(s)t(s). \quad (11)$$

然后验证者只需验证 $t(\pi) = AB - C \stackrel{?}{=} 0$ 即可. 注意到 A 中 $u_i(s)$ 可表示为 $\sum_{j=0}^{d-1} u_{i,j} s^j$, C 中 $h(s)t(s)$ 可根据 $\{s^i t(s)/\delta\}_{i=0}^{d-2}$ 计算, 则显然有证明 π 与公共参考串 σ 成线性关系. 然而, 上述直观想法无法保障可靠性和零知识性. 在 Groth16 中, 实际的 σ 中引入了 α, β, δ 和 γ 四个新参数, 证明 $\pi = (A, B, C)$ 中还引入了 r_1 和 r_2 两个新参数, 其中 α 和 β 用于保障 A 和 B 使用了相同的系数 c_i , γ 和 δ 有助于证明知识可靠性, r_1 和 r_2 用于保障零知识性. 具体见协议流程.

协议流程 Groth16 的协议流程如协议 2 所示, 其与对应的零知识 LIP 的区别仅在于前者引入了双线性配对用于验证 $t(\sigma, \pi) \stackrel{?}{=} 0$ (即协议 2 第 3 步 \mathcal{V} 检查).

讨论总结. 首先分析安全性. 上述论证可以证明具有完美完备性和完美零知识性, 在敌手只能进行线性/仿射运算的情况下, 该论证可被证明是统计意义知识可靠的. 完美完备性来源于零知识 LIP. 由于 r_1, r_2 的随机性, 真实证明中的 A, B 和模拟器生成的 A, B 是分布一致的, 又 C 由 A, B 确定, 因此完美零知识性得以保障. 现简要分析知识可靠性.

考虑不加随机数 r_1, r_2 的版本, 对于只能进行线性/仿射运算的敌手, 其伪造证据 A' 的形式为

$$\begin{aligned} A' = & A_\alpha \alpha + A_\beta \beta + A_\delta \delta + A_\gamma \gamma + A(s) + \sum_{i=0}^N A_i \frac{\beta u_i(s) + \alpha w_i(s) + y_i(s)}{\gamma} \\ & + \sum_{i=N+1}^m A_i \frac{\beta u_i(s) + \alpha w_i(s) + y_i(s)}{\delta} + A_h(s) \frac{t(s)}{\delta}. \end{aligned} \quad (12)$$

且 B', C' 是类似的形式. 由 Schwartz-Zippel 引理可知, 若将 A', B' 和 C' 视为以 $\alpha, \beta, \delta, \gamma, s$ 为变量的多变量函数, 则由 $C' \leftarrow A'B'$ 缺失 α^2 项可以较大概率推知 $A_\alpha B_\alpha = 0$, 不失一般性, 可设 $B_\alpha = 0$. 同理, 考虑缺失的 $\alpha\beta$ 和 β^2 项, 可知等式 (12) 中敌手构造的 A', B' 前五项实际为

$$A' = \alpha + A_\delta \delta + A_\gamma \gamma + A(s) + \dots, \quad B' = \beta + B_\delta \delta + B_\gamma \gamma + B(s) + \dots. \quad (13)$$

接着, 考虑包含 $1/\delta^2$ 的项, 这说明

$$\left(\sum_{i=N+1}^m A_i (\beta u_i(s) + \alpha w_i(s) + y_i(s)) + A_h(s) t(s) \right) \left(\sum_{i=N+1}^m B_i (\beta u_i(s) + \alpha w_i(s) + y_i(s)) + B_h(s) t(s) \right) = 0, \quad (14)$$

不失一般性, 可假设等式 (14) 中左边的左半部分表达式值为 0. 进一步的, 考虑到 C' 中没有 α 项, 则等式 (14) 中左边的右半部分表达式值为 0. 依此法消去其他项, 最终可提取出证据 $(c_{N+1}, c_{N+2}, \dots, c_m)$. 事实上, 等式 (14) 也体现了引入参数 δ, γ 的作用, 即使得项 $\left\{ \frac{\beta u_i(s) + \alpha w_i(s) + y_i(s)}{\gamma} \right\}_{i=0}^N$ 和 $\left\{ \frac{\beta u_i(s) + \alpha w_i(s) + y_i(s)}{\delta} \right\}_{i=N+1}^m$ 是相互独立的, 从而提出特定项并实现知识可靠性. 此外, α, β 用于保障

A, B 使用了相同的系数, r_1, r_2 通过随机化处理来保障零知识性, 不过在证明知识可靠性时需要引入更多的副本.

其次分析复杂度. Groth16 的参考串包含 $(m + 2d + 2)$ 个 \mathbb{G}_1 群中的元素和 $(d + 3)$ 个 \mathbb{G}_2 群中的元素. 证明者计算开销主要为计算证明 π , 具体为 $O(|C|)$ 次群上运算. 通信复杂度为 2 个 \mathbb{G}_1 群中的元素和 1 个 \mathbb{G}_2 群中的元素. 验证者计算开销为 4 次配对运算.

协议 2 Groth16 [52]

公共输入: 域 \mathbb{F} , 算术电路 $C : \mathbb{F}^{|\mathbf{x}|+|\mathbf{w}|} \rightarrow \mathbb{F}^{|\mathbf{y}|}$. 其中 $(\mathbf{x}, \mathbf{y}) = (c_1, c_2, \dots, c_N)$, $|\mathbf{x}| + |\mathbf{y}| = N$.

证明者秘密输入: $\mathbf{w} = (c_{N+1}, c_{N+2}, \dots, c_m)$, 记集合 $I_{\text{mid}} = \{N + 1, N + 2, \dots, m\}$, 显然有 $|\mathbf{w}| = |I_{\text{mid}}|$.

1. 可信初始化阶段.

- (1) 由可信第三方将算术电路 C 的可满足性问题归约为 QAP 可满足问题. 根据电路 C 构造对应的 QAP 串 $(t(z), \mathcal{U}, \mathcal{W}, \mathcal{Y})$, 其规模为 m , 度为 d .
- (2) 由可信第三方生成相应参数. 生成生成元为 g, h 的群 $\mathbb{G}_1, \mathbb{G}_2$ 及双线性映射群 \mathbb{G}_T . 映射 e 定义为 $e : \mathbb{G}_1 \times \mathbb{G}_2 \rightarrow \mathbb{G}_T$. 记 $[a]_1$ 为 g^a , $[b]_2$ 为 h^b , $[c]_T$ 为 $e(g, h)^c$. 选取随机数 $\alpha, \beta, \gamma, \delta, s \xleftarrow{\$} \mathbb{F}$.
- (3) 由可信第三方生成公共参考字符串 $\sigma = ([\sigma_1]_1, [\sigma_2]_2)$ 和模拟陷门 $\tau = (\alpha, \beta, \delta, \gamma, s)$. 其中

$$\sigma_1 = \left(\begin{array}{c} \alpha, \beta, \delta, \{s^i\}_{i=0}^{d-1}, \left\{ \frac{\beta u_i(s) + \alpha w_i(s) + y_i(s)}{\gamma} \right\}_{i=0}^N \\ \left\{ \frac{\beta u_i(s) + \alpha w_i(s) + y_i(s)}{\delta} \right\}_{i=N+1}^m, \left\{ \frac{s^i t(s)}{\delta} \right\}_{i=0}^{d-2} \end{array} \right), \sigma_2 = (\beta, \gamma, \delta, \{s^i\}_{i=0}^{d-1}).$$

2. 证明者 \mathcal{P} 生成证明. \mathcal{P} 随机挑选 $r_1, r_2 \xleftarrow{\$} \mathbb{F}$ 并计算证明 $\pi = ([A]_1, [C]_1, [B]_2)$, 其中

$$\begin{aligned} A &= \alpha + \sum_{i=0}^m c_i u_i(s) + r_1 \delta, & B &= \beta + \sum_{i=0}^m c_i w_i(s) + r_2 \delta, \\ C &= \frac{\sum_{i=N+1}^m c_i (\beta u_i(s) + \alpha w_i(s) + y_i(s)) + h(s)t(s)}{\delta} + Ar_2 + Br_1 - r_1 r_2 \delta. \end{aligned}$$

3. 验证者 \mathcal{V} 验证证明. \mathcal{V} 检查

$$e([A]_1, [B]_2) \stackrel{?}{=} e([\alpha]_1, [\beta]_2) e \left(\sum_{i=0}^N c_i \left[\frac{\beta u_i(s) + \alpha w_i(s) + y_i(s)}{\gamma} \right]_1, [\gamma]_2 \right) e([C]_1, [\delta]_2).$$

输出: 比特 b , 当且仅当上述检查通过, 输出 $b = 1$; 否则输出 $b = 0$.

5.3.3 GKMM18

GKMM18 由 Groth 等人 [41] 提出, 他们针对 QAP 可满足问题实现了 CRS 全局可更新的 zk-SNARK. 本小节简要介绍 GKMM18 及其 CRS 实现全局和可更新的主要思路.

主要思路. 考虑一个仅包含单项式的 CRS 如 g^x , 参与方 P_1 欲更新时只需随机选取 x_1 将 CRS 更新为 g^{xx_1} 并给出对 x_1 的知识证明 $(g^{x_1}, g^{\alpha x_1})$ 即可, 而该知识证明可基于 q -PKE 假设利用配对完成验证, 即验证 $e(g^{x_1}, g^\alpha) \stackrel{?}{=} e(g^{\alpha x_1}, g)$. 事实上, GKMM18 中的全局 CRS 就是形如 $\{g^{s^i \cdot r^j \cdot q^k}\}$ 的形式, 其中 i, k 均与 QAP 串的度 d 成线性, j 为常数, 因此 GKMM18 中全局 CRS 的规模为 $O(|C_M|^2)$ 级别. 而为更新该 CRS, 只需随机选取 α, β, γ , 并将 CRS 更新为 $\{g^{s^{\alpha i} \cdot r^{\beta j} \cdot q^{\gamma k}}\}$ 即可.

然而, 针对包含多项式的 CRS 实现可更新性是困难的. 考虑对 $g^{f(s)}$ 的更新, 若随机选取 δ 并将旧 CRS 更新为 $g^{f(s)\delta}$, Groth 等人证明了任意可完成上述更新的敌手都可提取出单项式 $(g, g^{s\delta}, \dots, g^{s^n\delta})$, 依据这些单项式, 敌手可破坏 Pinocchio 等 zk-SNARK 的知识可靠性.

针对上述问题, Groth 等人采用了一种与 Pinocchio 及 Groth16 不同的证明方法. 考虑一个 QAP $\mathcal{Q} = (t(z), \mathcal{U}, \mathcal{W}, \mathcal{Y})$, 由于 CRS 中不能包含多项式, 因此无法继续使用之前基于 QAP 的方法, 即通过随机挑选 s 构造与 QAP 对应的多项式编码进而验证正确性. GKMM18 采用了与 BCCGP16 [30]、Bulletproofs [31] 等基于 IPA 的零知识证明 (见第 7 章) 相似的方法, 即通过为 QAP 中等式的每一项分别乘以变量 n 的某次幂从而将 QAP 可满足问题转换为另外一个多项式是否为零多项式的问题. 具体的, 考

虑多项式

$$f(z, n) = h(z)n + \sum_{i=0}^m c_i (y_i(z)n^2 + u_i(z)n^3 + w_i(z)n^4) - n^5 - t(z)n^6, \quad (15)$$

若随机选取 s, r , 计算等式(15)中 $f(s, r) \cdot f(s, r)$ 可知, r^7 项的系数为

$$t(s)h(s) - \sum_{i=0}^m c_i y_i(s) + \left(\sum_{i=0}^m c_i u_i(s) \right) \left(\sum_{i=0}^m c_i w_i(s) \right), \quad (16)$$

其中 $c_0 = 0$, 而这恰为 QAP 可满足问题的形式. 可以证明, 基于 q 阶单项式知识假设 (q -monomial knowledge assumption) 和 q 阶单项式计算假设 (q -monomial computational assumption)^[41], 给定不含 r^7 项的 CRS, 若证明者可生成证明 g^A 且有 $e(g^{f(s,r)}, h^{f(s,r)}) = e(g^A, h)$, 则可说明他拥有 $(c_{N+1}, c_{N+2}, \dots, c_m)$. 类似的, 为说明 $g^{f(s,r)}$ 是正确构造的, 可引入另一个新变量 w 构造一个多项式 $f_2(z, n, w)$ 使得当且仅当 $f(s, r)$ 正确构造时该多项式的某一特定项系数为 0, 然后给定不含该特定项单项式的 CRS, 利用双线性配对验证该多项式特定项系数确实为 0. 在全局 CRS 中, s, r, q 分别对应于变量 z, r, w .

在针对具体关系构造 zk-SNARK 时, 需要对全局 CRS 进行处理从而生成新的针对具体关系的 CRS. 与 Pinocchio 等协议类似, 其生成方式也是构造用于验证 QAP 的系列多项式, 其规模也为 $O(|C|)$ 级别的群元素.

协议流程. 在生成针对具体关系的 CRS 后, GKMM18 的协议流程简述如下.

(1) 在给定 QAP 四元组 $(t(z), \mathcal{U}, \mathcal{W}, \mathcal{Y})$ 及群上生成元 g_1, g_2 后, 证明者 \mathcal{P} 计算

$$h(z) \leftarrow \frac{\left(u_0(z) + \sum_{k=1}^m c_k \cdot u_k(z) \right) \cdot \left(w_0(z) + \sum_{k=1}^m c_k \cdot w_k(z) \right) - \left(y_0(z) + \sum_{k=1}^m c_k \cdot y_k(z) \right)}{t(z)}, \quad (17)$$

随后证明者随机选取 $\delta \xleftarrow{\$} \mathbb{F}$, 并计算证明 $\pi = (A \leftarrow g_1^{f(s,r)}, B \leftarrow g_2^{f(s,r)}, C \leftarrow g_1^{f_2(s,r,q)})$, 其中

$$f(s, r) = h(s)r + \sum_{i=0}^m c_i (y_i(s)r^2 + u_i(s)r^3 + w_i(s)r^4) - r^5 - t(s)r^6, \quad (18)$$

$$f_2(s, r, q) \leftarrow f^2(s, r) + (h(s)r + \delta(r - t(s)r^2) + \sum_{i=\ell+1}^m c_i (y_i(s)r^2 + u_i(s)r^3 + w_i(s)r^4)) \cdot f_3(s, r, q), \quad (19)$$

其中多项式 $f_3(s, r, q)$ 用于辅助验证 $f_2(s, r, q)$ 中某一特定项系数为 0.

(2) 在收到证明 $\pi = (A, B, C)$ 后, 验证者验证

- $e(A, g_2) \stackrel{?}{=} e(g_1, B).$
- $e(A, B) \cdot e(Ag_1^{r^5+t(s)r^6-\sum_{i=0}^{\ell} c_i y_i(s)r^2+u_i(s)r^3+w_i(s)r^4}, g_2^{f_3(r,s,q)}) \stackrel{?}{=} e(C, g_2).$

讨论总结. 首先分析安全性. 基于 q 阶单项式知识假设和 q 阶单项式计算假设^[41], 如果验证者所验证的第二个等式成立, 那么可以得出证明者拥有形式正确的 $f(s, r, q)$ 且他拥有 QAP 可满足问题的解, 即 GKMM18 可被证明具有知识可靠性. 另外, 由于 δ 的均匀随机性, 可以证明 GKMM18 具有完美零知识性.

其次分析复杂度. 如前文所述, GKMM18 的全局 CRS 规模为 $O(|C_M|^2)$ 级别的群元素, 针对具体关系的 CRS 规模为 $O(|C|)$ 级别的群元素. 与 Groth16 类似, GKMM18 的通信复杂度也为 3 个群元素, 证明复杂度也为 $O(|C|)$ 级别的群幂运算, 但验证复杂度比 Groth16 多了一个配对运算.

5.4 总结

zk-SNARK 的通信复杂度为常数个群元素, 验证复杂度也可降低到常数个配对运算, 与区块链低存储、高吞吐的需求相契合, 大部分隐私保护应用如 Zcash、以太坊扩容方案等也都是基于 zk-SNARK 实现的。然而, zk-SNARK 需基于不可证伪的非标准假设, 并且启动阶段的系统参数必须依赖可信第三方生成。为解决这些问题, 出现了 CRS 可更新的零知识证明, 还出现了系列底层假设更通用、启动阶段系统参数可公开生成的零知识证明, 这也被称为透明的 zk-SNARK (transparent zk-SNARK), 本文将在第 6–8 章分别介绍三种启动阶段系统参数可公开生成的零知识证明。

6 基于 DEIP 的零知识证明

本章介绍基于双向高效交互式证明 (DEIP) 的零知识证明。该类协议的核心思路是利用同态承诺、多项式承诺、随机掩藏多项式等技术将已有的针对电路求值问题的 DEIP 转换为简洁 NIZKAO。本章第 6.1 节介绍相关定义及概念, 第 6.2 节介绍背景及主要思路, 第 6.3 节介绍典型协议。

6.1 定义及概念

多项式承诺 (polynomial commitment, PC)^[122] 是对多项式的承诺。除具有隐藏性和绑定性外, 发送者还可向接收者证明多项式 $f(\cdot)$ 在某个由接收者随机选取的点 t 上的取值为 y , 即 $f(t) = y$ 。本章涉及的多项式承诺均是针对多变量多项式的, 且允许交互 (是 Bünnz 等人^[123] 对文献 [122] 中多项式承诺的扩展)。

定义 27 (多项式承诺^[27, 123]) 一个多变量多项式承诺方案 $\text{PC} = (\text{Setup}, \text{Com}, \text{Open}, \text{Eval})$ 由以下 4 个算法组成。

- $\text{pp} \leftarrow \text{Setup}(1^\lambda, \ell, d)$: 参数生成算法。以安全参数 λ 的一元表示、多项式变量参数 ℓ 和度参数 d 为输入, 输出公共参数 pp 。
- $(c, s) \leftarrow \text{Com}(\text{pp}, f(\cdot))$: 多项式承诺算法。以公共参数 pp 和多项式 $f(\cdot)$ 为输入, 输出承诺 c 和打开提示 s 。
- $b \leftarrow \text{Open}(\text{pp}, c, f(\cdot), s)$: 多项式承诺打开算法。给定公共参数 pp 、承诺 c 、多项式 $f(\cdot)$ 和打开提示 s , 打开并验证承诺 c 的正确性, 输出接受 ($b = 1$) 或拒绝 ($b = 0$)。
- $b \leftarrow \text{Eval}(\text{pp}, c, t, y, \ell, d; f(\cdot), s)$: 求值计算协议。 PC.Eval 是一个交互式公开抛币协议, 拥有隐私输入 $f(\cdot)$ 和 s 的证明者向验证者证明 $f(t) = y$ 。 \mathcal{V} 输出接受 ($b = 1$) 或拒绝 ($b = 0$)。

如果一个 PC 的求值计算协议是知识可提取的 (见定义 14), 则称该 PC 是可提取的。如果一个可提取 PC 的求值计算协议是零知识知识论证, 且具有统计意义的证据扩展可仿真性, 则称该 PC 是零知识的, 记为 zk-PC。

在本章中, PC 的主要作用是实现传输多项式的简洁性, 与 PC 相同作用的另一个概念是可验证多项式委托 (verifiable polynomial delegation, VPD)^[124], 其允许用户将多项式 $f(\cdot)$ 在若干点的求值计算外包给服务者, 同时服务者提供一个证明用以证明求值计算的正确性。

定义 28 (可验证多项式委托^[28, 51, 125]) 一个针对多变量多项式 $f(\cdot)$ 的可验证多项式委托 $\text{VPD} = (\text{Setup}, \text{Com}, \text{Eval}, \text{Verify})$ 由以下 4 个算法组成。

- $(\text{pp}, \text{vp}) \leftarrow \text{Setup}(1^\lambda, \ell, d)$: 参数生成算法。以多项式变量参数 λ 的一元表示、多项式变量参数 ℓ 和度参数 d 为输入, 输出公共参数 pp 和验证陷门 vp 。
- $c \leftarrow \text{Com}(\text{pp}, f(\cdot))$: 多项式承诺算法。以公共参数 pp 和多项式 $f(\cdot)$ 为输入, 输出承诺 c 。
- $(y, \pi) \leftarrow \text{Eval}(\text{pp}, f(\cdot), t, r)$: 求值计算算法。以公共参数 pp 、多项式 $f(\cdot)$ 、多项式上的点 t 和随机挑战 r 为输入, 计算 $y = f(t)$, 输出求值 y 及证明 π 。
- $b \leftarrow \text{Verify}(\text{vp}, c, y, t, \pi, r)$: 验证算法。以验证陷门 vp 、承诺 c 、求值 y 、多项式上的点 t 、求值证明 π 和随机挑战 r 为输入, 验证求值计算的正确性, 并输出接受 ($b = 1$) 或拒绝 ($b = 0$)。

如果一个 VPD 的求值计算算法不直接输出 y 而是输出对 y 的承诺 com_y (其能够“零知识地”隐藏 y 且 π 可支持验证 com_y 确实是对 $f(\cdot)$ 的承诺), 则称该 VPD 是零知识的, 记为 zk-VPD。

定义 28 中的 VPD 需要验证陷门 vp , 因此启动阶段系统参数的生成须保持私密, 具体方案见文献 [26, 51]; 也有不需验证陷门的 VPD, 其系统参数可公开生成, 具体方案见文献 [28, 29].

定义 29 (多变量线性扩展) 给定多项式 $f(\cdot) : \{0, 1\}^\ell \rightarrow \mathbb{F}$, $f(\cdot)$ 的多变量线性扩展 (multilinear extension) 是域 \mathbb{F} 上的 ℓ 元多项式 $\tilde{f}(\cdot) : \mathbb{F}^\ell \rightarrow \mathbb{F}$, 其满足对于任意的 $\mathbf{x} \in \{0, 1\}^\ell$, $\tilde{f}(\mathbf{x}) = f(\mathbf{x})$. 其中, 线性的含义是 $\tilde{f}(\cdot)$ 中的每个变量的度最多为 1. 具体的, $f(\cdot)$ 的多变量线性扩展 $\tilde{f}(\cdot)$ 是

$$\begin{aligned}\tilde{f}(\mathbf{x}) &= \tilde{f}(x_1, x_2, \dots, x_\ell) \\ &= \sum_{\mathbf{e} \in \{0, 1\}^\ell} \left(f(\mathbf{e}) \cdot \prod_{i=1}^{\ell} (x_i \cdot e_i + (1 - x_i) \cdot (1 - e_i)) \right) \\ &= \sum_{\mathbf{e} \in \{0, 1\}^\ell} (f(\mathbf{e}) \cdot \tilde{\text{eq}}(\mathbf{x}, \mathbf{e})),\end{aligned}\tag{20}$$

其中, $\tilde{\text{eq}}(\mathbf{x}, \mathbf{e})$ 在 \mathbf{x} 与 \mathbf{e} 每一位相等时值为 1.

定义 30 (sum-check 协议^[63]) 给定域 \mathbb{F} 上的多元多项式 $g(\cdot)$, 不失一般性, 记 $g(\cdot)$ 中所有变量的度均为 m . sum-check 协议是一个如图 6 所示的交互式证明, 其中证明者 \mathcal{P} 可向验证者 \mathcal{V} 证明某公开函数 $g(\cdot)$ 的遍历求和值 $\sum_{x_1, x_2, \dots, x_\ell \in \{0, 1\}} g(x_1, x_2, \dots, x_\ell)$ 为某公开值 T . 该证明的思路是循环地将对 $\sum_{x_1, x_2, \dots, x_\ell \in \{0, 1\}} g(x_1, x_2, \dots, x_\ell)$ 的声明归约为对 $g_i(X_i)$ 的声明, 其中,

$$g_i(X_i) = \sum_{x_{i+1}, x_{i+2}, \dots, x_\ell \in \{0, 1\}} g(r_1, r_2, \dots, r_{i-1}, X_i, x_{i+1}, x_{i+2}, \dots, x_\ell),\tag{21}$$

r_1, r_2, \dots, r_{i-1} 由 \mathcal{V} 在前 $i-1$ 轮分别随机选取, 且可最终归约为对 $g(\mathbf{r}) = g(r_1, r_2, \dots, r_\ell)$ 的声明, 其中 $\mathbf{r} \in \mathbb{F}^\ell$. 当 \mathcal{V} 拥有 $g(\cdot)$ 时, 可自行计算 $g(\mathbf{r})$; 当 $g(\cdot)$ 是秘密时, \mathcal{V} 可通过对谕示 $g(\cdot)$ 的访问请求得到 $g(\mathbf{r})$. 该证明共有 ℓ 轮, 可靠性误差为 $O(m\ell/|\mathbb{F}|)$ (由 Schwartz-Zippel 引理保障), 通信量为 $O(m\ell)$ 个域元素, 验证者计算开销为 $O(m\ell)$ 级别的域上运算, 证明者计算开销为 $O(2^\ell)$ 级别的域上运算^[126]. 相比于平凡计算求和函数的复杂度 $O(2^\ell)$, sum-check 协议的验证复杂度有显著降低.

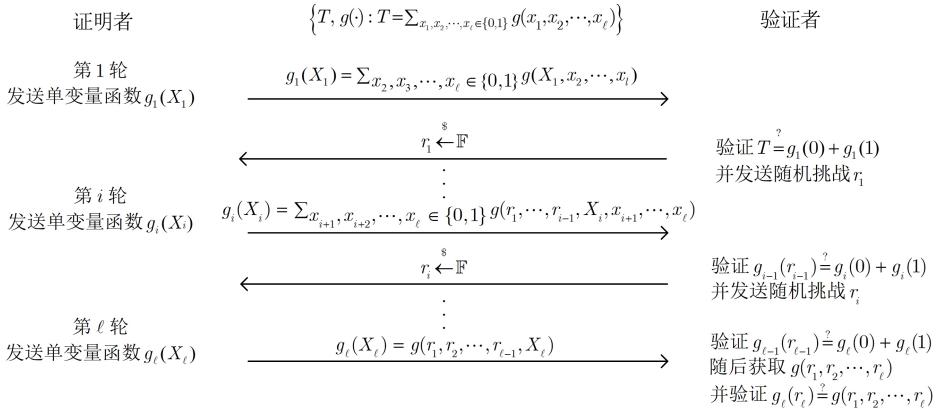


图 6 sum-check 协议流程^[63]
Figure 6 Process of sum-check protocol

6.2 背景及主要思路

6.2.1 背景

针对分层算术电路求值问题的双向高效交互式证明. Goldwasser 等人^[61] 利用 sum-check 协议为核组件, 提出了一个针对分层算术电路求值问题的交互式证明. 由于证明和验证复杂度均较低, 该类协议又被称为双向高效的交互式证明 (DEIP). 之后, Cormode 等人^[60] 利用多变量线性扩展代替了上述协

议中的低度扩展 (low-degree extension), 将上述协议的证明复杂度从 $O(\text{poly } |C|)$ 降低为 $O(|C| \log |C|)$. 后续对证明复杂度的一系列改进大多是在具有特殊结构的分层电路基础上实现的. 对于常规电路 (regular circuit)^[60], 即电路谓词 (wiring predicates, 见第 6.2.2 小节) 在任意点的取值都可在 $O(\text{poly log } |C|)$ 级别时间、 $O(\log |C|)$ 级别空间内可计算的电路 (对数空间均匀电路就是一种常规电路), Thaler^[126] 指出证明复杂度可以降低到 $O(|C|)$; 对于有着许多规模为 $|C'|$ 的相同子电路的数据并行电路 (data parallel circuit), Thaler 同时指出证明复杂度可以降低到 $O(|C| \log |C'|)$, 在此基础上, Wahby 等人^[127] 将证明复杂度继续降低到了 $O(|C| + |C'| \log |C'|)$; 对于有着许多不直接连接且互不相同的子电路的电路, Zhang 等人^[125] 提出了一种证明复杂度为 $O(|C| \log |C'|)$ 的 DEIP. 对于任意的分层算术电路, Xie 等人^[26] 提出了一种证明复杂度为 $O(|C|)$ 的变种 DEIP. 需要注意的是, 上述协议所基于的电路均是分层算术电路.

交互式证明与零知识证明的转换. Cramer 等人^[102] 及 Ben-Or 等人^[128] 给出了利用同态承诺将交互式证明转换为计算零知识证明或完美零知识论证的通用方法. 其核心思路是将交互式证明中证明者发送的信息用同态承诺掩藏. 该方案的安全性支撑有三:

- (1) 由于承诺的隐藏性, 验证者无法推知与被承诺信息相关的其他信息, 进而保障零知识性.
- (2) 由于承诺的绑定性, 证明者无法欺骗, 进而保障可靠性.
- (3) 由于承诺的同态性, 协议可以在保障零知识性的同时完成正确性验证.

6.2.2 主要思路

本小节首先介绍 DEIP, 其次介绍基于 Pedersen 承诺的零知识 sum-check 协议, 最后介绍利用 DEIP 和零知识 sum-check 协议如何构造基于 DEIP 的平凡零知识证明.

DEIP. 一种 DEIP 的主要流程如协议 3 所示. 令 C 是域 \mathbb{F} 上深度为 d 的分层算术电路, 记第 0 层是电路输出线所在层, 第 d 层是电路输入线所在层. DEIP 逐层按轮运行. 在协议第 1 轮, 当验证者 \mathcal{V} 收到证明者 \mathcal{P} 发送的输出 \mathbf{y} 后, \mathcal{P} 和 \mathcal{V} 调用 sum-check 协议将对电路第 0 层的声明转换为对电路第 1 层的声明; 在协议第 i 轮, \mathcal{P} 和 \mathcal{V} 调用 sum-check 协议将对电路第 $i-1$ 层的声明转换为对电路第 i 层的声明; 最终, \mathcal{P} 和 \mathcal{V} 将电路求值转换为对电路第 d 层 (即输入层) 的声明, 由于 \mathcal{V} 拥有电路输入 \mathbf{x} , 故可直接自行计算验证.

协议 3 一种 DEIP^[26]

公共输入: 域 \mathbb{F} , 深度为 d 的分层算术电路 $C : \mathbb{F}^{|\mathbf{x}|} \rightarrow \mathbb{F}^{|\mathbf{y}|}$.

1. \mathcal{P} 向 \mathcal{V} 发送电路输出 \mathbf{y} .
2. \mathcal{V} 计算多项式 $\tilde{V}_0(\cdot)$ 并返回随机挑战 $\mathbf{r}_0 \in \mathbb{F}^{s_0}$.
3. \mathcal{P} 和 \mathcal{V} 调用对 $\tilde{V}_0(\mathbf{r}_0)$ 的 sum-check 协议. 在 sum-check 协议的最后一轮, \mathcal{V} 会收到 $\tilde{V}_1(\mathbf{u}^{(1)})$ 和 $\tilde{V}_1(\mathbf{v}^{(1)})$.
4. **for** $i = 1, 2, \dots, d-1$
 - \mathcal{V} 选择随机挑战 $\alpha^{(i)}, \beta^{(i)} \xleftarrow{\$} \mathbb{F}$ 并发送给 \mathcal{P} ;
 - \mathcal{P} 和 \mathcal{V} 调用对 $\alpha^{(i)}\tilde{V}_i(\mathbf{u}^{(i)}) + \beta^{(i)}\tilde{V}_i(\mathbf{v}^{(i)})$ 的 sum-check 协议;
 - 在 sum-check 协议的最后一轮, \mathcal{P} 发送给 $\tilde{V}_{i+1}(\mathbf{u}^{(i+1)})$ 和 $\tilde{V}_{i+1}(\mathbf{v}^{(i+1)})$ 给 \mathcal{V} ;
 - \mathcal{V} 利用 $\tilde{V}_{i+1}(\mathbf{u}^{(i+1)})$ 和 $\tilde{V}_{i+1}(\mathbf{v}^{(i+1)})$ 参与到第 $i+1$ 轮循环.
- end for**
5. \mathcal{V} 根据第 d 层的输入 \mathbf{x} 自行计算 $\tilde{V}_d(\mathbf{u}^{(d)})$ 和 $\tilde{V}_d(\mathbf{v}^{(d)})$, 进而检查第 $d-1$ 轮的 sum-check 协议的正确性.

输出: 比特 b , $b = 1$ 表示 \mathcal{V} 检查通过, $b = 0$ 表示拒绝.

具体的, 记电路 C 第 i 层的电路门数为 S_i 且 $s_i = \lceil \log_2 S_i \rceil$ (假设 S_i 是 2 的某次幂, 若不是则可对电路作必要填充). 记 $V_i(\cdot) : \{0, 1\}^{s_i} \rightarrow \mathbb{F}$ 是以第 i 层的某个任意电路门的二进制顺序表示字符串 $\{0, 1\}^{s_i}$ 为输入、该门的门值为输出的函数. 记 $\text{add}_i(\cdot), \text{mult}_i(\cdot) : \{0, 1\}^{2s_i+s_{i-1}} \rightarrow \{0, 1\}$ 为电路谓词, 其以第 $i-1$ 层某个电路门 (记为 α_c) 的二进制顺序表示字符串 $\{0, 1\}^{s_{i-1}}$ 、第 i 层的某两个电路门 (记作 α_a, α_b) 的二进制顺序表示字符串 $\{0, 1\}^{s_i}$ 为输入, 当且仅当电路中存在以 α_a, α_b 为输入, α_c 为输出的加法门

(乘法门) 时输出为 1. 基于以上定义, 对于任意的 $\mathbf{c} \in \{0, 1\}^{s_i}$, $\mathbf{a}, \mathbf{b} \in \{0, 1\}^{s_{i+1}}$, $V_i(\mathbf{c})$ 可写为

$$\begin{aligned} V_i(\mathbf{c}) &= \sum_{\mathbf{a}, \mathbf{b} \in \{0, 1\}^{s_{i+1}}} f_i(\mathbf{a}, \mathbf{b}) \\ &= \sum_{\mathbf{a}, \mathbf{b} \in \{0, 1\}^{s_{i+1}}} (\text{add}_{i+1}(\mathbf{a}, \mathbf{b}, \mathbf{c})(V_{i+1}(\mathbf{a}) + V_{i+1}(\mathbf{b})) + \text{mult}_{i+1}(\mathbf{a}, \mathbf{b}, \mathbf{c})(V_{i+1}(\mathbf{a})V_{i+1}(\mathbf{b}))), \end{aligned} \quad (22)$$

这样就可将对第 i 层的声明归约为对第 $i+1$ 层的声明. 由于 $V_i(\cdot)$ 是求和形式, 故可调用 sum-check 协议进行验证. 然而, $V_i(\cdot)$ 的定义域大小为 2, 基于 Schwartz-Zippel 引理难以保障可靠性. 因此, 为调用 sum-check 协议, 需将 $V_i(\cdot)$ 转换为对应的多变量线性扩展多项式 $\tilde{V}_i(\cdot)$.

值得注意的是, \mathcal{P} 和 \mathcal{V} 在调用 sum-check 协议验证对电路第 i 层的声明时, \mathcal{V} 需要计算 $\tilde{V}_i(\mathbf{u}^{(i+1)}, \mathbf{v}^{(i+1)})$ ($\mathbf{u}^{(i+1)}, \mathbf{v}^{(i+1)}$ 由 \mathcal{V} 随机选取) 才能验证 sum-check 协议的正确性. 由于 $\text{add}_{i+1}(\cdot)$ 和 $\text{mult}_{i+1}(\cdot)$ 可根据电路结构直接得出, 为计算 $\tilde{V}_i(\mathbf{u}^{(i+1)}, \mathbf{v}^{(i+1)})$, \mathcal{V} 只需获知 $\tilde{V}_{i+1}(\mathbf{u}^{(i+1)})$ 和 $\tilde{V}_{i+1}(\mathbf{v}^{(i+1)})$, 故在 sum-check 协议的最后一轮, \mathcal{P} 不需发送整个多项式而只需发送这两个值. 虽然 $\tilde{V}_{i+1}(\mathbf{u}^{(i+1)})$ 和 $\tilde{V}_{i+1}(\mathbf{v}^{(i+1)})$ 仍是求和形式, 但若采用平凡方法分别对 $\tilde{V}_{i+1}(\mathbf{u}^{(i+1)})$ 和 $\tilde{V}_{i+1}(\mathbf{v}^{(i+1)})$ 再次调用 sum-check 协议, 则验证复杂度至少为 $O(2^d)$. 为减少验证者的计算开销, Goldwasser、Kalai 和 Rothblum [61] 将对 $\tilde{V}_{i+1}(\mathbf{u}^{(i+1)})$ 和 $\tilde{V}_{i+1}(\mathbf{v}^{(i+1)})$ 的 2 个声明转换为 1 个对 $\tilde{V}_{i+1}(\mathbf{w}^{(i+1)})$ 的声明, 其中 $\mathbf{w}^{(i+1)}$ 是与 $\mathbf{u}^{(i+1)}$ 、 $\mathbf{v}^{(i+1)}$ 相关的一个随机取值; Chiesa、Forbes 和 Spooner^[129] 则直接调用 $\tilde{V}_{i+1}(\mathbf{u}^{(i+1)})$ 和 $\tilde{V}_{i+1}(\mathbf{v}^{(i+1)})$ 的随机线性组合参与到下轮 sum-check 协议中, 协议 3 采用的就是第二种方法.

基于文献 [26, 60, 61, 126, 127], 可以证明针对分层算术电路的系列 DEIP 的可靠性误差根据协议的不同从 $O(d \log g / |\mathbb{F}|)$ 到 $O(d \log |C| / |\mathbb{F}|)$ 不等, 证明者计算开销为 $O(|C|)$ 到 $O(|C| \log |C|)$ 级别的域上运算, 实际通信量为 $O(d \log g)$ 到 $O(d \log |C|)$ 级别的域元素. 对于常规电路, 验证者计算开销为 $O(|x| + |y| + d \log g)$ 到 $O(|x| + |y| + d \log |C|)$ 级别的域上运算. 当电路深度 $d = \text{poly}(\log |C|)$ 时, DEIP 系列协议的通信复杂度和验证复杂度均与电路规模成亚线性关系.

基于 Pedersen 承诺的零知识 sum-check 协议. 在 sum-check 协议中, 验证者会获得多项式 $g_i(X_i)$, 当 $g(\cdot)$ 是秘密时, 会泄露 $g(\cdot)$ 的相关信息, 故不是零知识的. 事实上, 基于 Cramer 和 Damgård^[102] 的方法, 利用 Pedersen 承诺容易构造零知识的 sum-check 协议.

在介绍零知识 sum-check 协议之前, 首先给出利用 Pedersen 承诺可实现的两个模块. 给定形如定义 8 的 Pedersen 承诺方案, 则存在以下两个零知识论证.

- (1) 给定公共承诺 $c_1 \leftarrow \text{Com}(x_1; r_1), c_2 \leftarrow \text{Com}(x_2; r_2)$ 且 $x_1 = x_2$, 证明者可向验证者证明 c_1 和 c_2 是对相同消息的承诺, 即 $x_1 = x_2$, 记该论证为 $\mathcal{ZK}_{\text{eq}}(c_1, c_2)$. 具体协议及安全性证明见文献 [130].
- (2) 给定公共承诺 $c_1 \leftarrow \text{Com}(x_1; r_1), c_2 \leftarrow \text{Com}(x_2; r_2), c_3 \leftarrow \text{Com}(x_3; r_3)$ 且 $x_3 = x_1x_2$, 证明者可向验证者证明 c_1 和 c_2 的承诺消息乘积与 c_3 的承诺消息乘积相等, 即 $x_3 = x_1x_2$, 记该论证为 $\mathcal{ZK}_{\text{prod}}(c_1, c_2, c_3)$. 具体协议及安全性证明见文献 [32, 131].

基于上述两个模块, 容易将图 6 中的协议修改为零知识的, 主要思路见协议 4. 需要指出的是, 协议 4 是不考虑 Pedersen 承诺中随机数的简化版本, 详细版本见文献 [51]. 协议 4 与图 6 的主要不同在于:

- (1) 在每一轮, 证明者 \mathcal{P} 不再直接发送 $g_i(X_i)$ 的系数 $(a_{i,0}, a_{i,1}, \dots, a_{i,m})$ (记 $g_i(X_i) = \sum_{j=0}^m a_{i,j} X^j$) 而改为发送对系数 $(a_{i,0}, a_{i,1}, \dots, a_{i,m})$ 的承诺 $\text{Com}(a_{i,0}), \text{Com}(a_{i,1}), \dots, \text{Com}(a_{i,m})$.
- (2) 验证者 \mathcal{V} 验证时不再直接计算 $g_i(0) + g_i(1)$, 而是计算 $\text{Com}(a_{i,0}) \prod_{j=0}^m \text{Com}(a_{i,j})$ 即可得到对 $g_i(0) + g_i(1)$ 的承诺.
- (3) \mathcal{V} 在第 i 轮不用直接验证 $g_{i-1}(r_{i-1}) \stackrel{?}{=} g_i(0) + g_i(1)$, 而需调用 \mathcal{ZK}_{eq} 验证承诺消息相等.

相比于 sum-check 协议, 零知识 sum-check 协议的证明和验证复杂度虽然不变, 但实际证明和验证开销却因引入了大量群上运算尤其是群幂运算而显著增加.

 协议 4 基于 Pedersen 承诺的零知识 sum-check 协议的主要思路 [51]

- 公共输入: 域 \mathbb{F} , $\text{Com}(T), m, \ell$. 证明者秘密输入: $g(\mathbf{x})$, 且有 $T = \sum_{\mathbf{x} \in \{0,1\}^\ell} g(\mathbf{x})$.
1. $i = 1$ 时, 证明者 \mathcal{P} 构造函数 $g_1(X_1)$ 并将对其系数 $(a_{1,0}, a_{1,1}, \dots, a_{1,m})$ 的承诺 $\text{Com}(a_{1,0}), \text{Com}(a_{1,1}), \dots, \text{Com}(a_{1,m})$ (省略了 Com 中的随机数) 发给验证者 \mathcal{V} .
 2. \mathcal{V} 计算 $\text{Com}_1^* \leftarrow \text{Com}(a_{1,0}) \prod_{j=0}^m \text{Com}(a_{1,j})$ 并调用 $\mathcal{ZK}_{eq}(\text{Com}_1^*, \text{Com}(T))$. 若 \mathcal{ZK}_{eq} 验证失败, 输出 $b = 0$; 若验证通过, \mathcal{V} 选取 $r_1 \xleftarrow{\$} \mathbb{F}$ 发送给 \mathcal{P} .
 3. \mathcal{P} 和 \mathcal{V} 共同计算 $\text{Com}_2 \leftarrow \prod_{j=0}^m \text{Com}^{r_1^j}(a_{1,j})$.
 4. **for** $i = 2, 3, \dots, \ell - 1$
 - \mathcal{P} 构造函数 $g_i(X_i)$ 并将对其系数 $(a_{i,0}, a_{i,1}, \dots, a_{i,m})$ 的承诺 $\text{Com}(a_{i,0}), \text{Com}(a_{i,1}), \dots, \text{Com}(a_{i,m})$ 发给验证者 \mathcal{V} ;
 - \mathcal{V} 计算 $\text{Com}_i^* \leftarrow \text{Com}(a_{i,0}) \prod_{j=0}^m \text{Com}(a_{i,j})$ 并调用 $\mathcal{ZK}_{eq}(\text{Com}_i^*, \text{Com}_i)$. 若 \mathcal{ZK}_{eq} 验证失败, 输出 $b = 0$; 若验证通过, \mathcal{V} 选取 $r_i \xleftarrow{\$} \mathbb{F}$ 发送给 \mathcal{P} ;
 - \mathcal{P} 和 \mathcal{V} 共同计算 $\text{Com}_{i+1} \leftarrow \prod_{j=0}^m \text{Com}^{r_i^j}(a_{i,j})$.
 - end for**
 5. \mathcal{P} 构造函数 $g_\ell(X_\ell)$ 并将对其系数 $(a_{\ell,0}, a_{\ell,1}, \dots, a_{\ell,m})$ 的承诺 $\text{Com}(a_{\ell,0}), \text{Com}(a_{\ell,1}), \dots, \text{Com}(a_{\ell,m})$ 发给验证者 \mathcal{V} .
 6. \mathcal{V} 计算 $\text{Com}_\ell^* \leftarrow \text{Com}(a_{\ell,0}) \prod_{j=0}^m \text{Com}(a_{\ell,j})$ 并调用 $\mathcal{ZK}_{eq}(\text{Com}_\ell^*, \text{Com}_\ell)$. 若 \mathcal{ZK}_{eq} 验证失败, 输出 $b = 0$; 若验证通过, \mathcal{V} 选取 $r_\ell \xleftarrow{\$} \mathbb{F}$.
 7. \mathcal{V} 计算 $\text{Com}_{\ell+1} \leftarrow \prod_{j=0}^m \text{Com}^{r_\ell^j}(a_{\ell,j})$. \mathcal{V} 获知 $\text{Com}(g(\mathbf{r}))$ (可通过谕示) 并调用 $\mathcal{ZK}_{prod}(\text{Com}_{\ell+1}, \text{Com}(g(\mathbf{r})))$. 若 \mathcal{ZK}_{prod} 验证失败, 输出 $b = 0$.
- 输出: 比特 b , 若上述验证均通过, 输出 $b = 1$; 否则输出 $b = 0$.
-

基于 DEIP 的平凡零知识证明 利用基于 Pedersen 承诺的零知识的 sum-check 协议可构造基于 DEIP 的平凡零知识证明 (简要流程见图 7), 其与协议 3 主要不同在于:

- (1) 在基于 DEIP 的平凡零知识证明中证明者 \mathcal{P} 发送的消息均为承诺形式.
- (2) 在协议开始前, \mathcal{P} 需分别发送对证据 $(\alpha_1, \alpha_2, \dots, \alpha_{|\mathbf{w}|})$ 及电路输出 \mathbf{y} 的承诺 (对于第 d 层左右输入分别为 α_j 、 α_k 的乘法门还需发送 $\text{Com}(\alpha_j \alpha_k)$).
- (3) 在第 i 层电路 sum-check 协议的最后一轮, 为将对第 i 层的声明归约为对第 $i+1$ 层的声明 \mathcal{V} 需要获知 $\tilde{V}_{i+1}(\mathbf{u}^{(i+1)})$ 、 $\tilde{V}_{i+1}(\mathbf{v}^{(i+1)})$ 和 $\tilde{V}_{i+1}(\mathbf{u}^{(i+1)} \cdot \tilde{V}_{i+1}(\mathbf{v}^{(i+1)}))$, 由于 \mathcal{V} 根据 $\text{Com}(\tilde{V}_{i+1}(\mathbf{u}^{(i+1)}))$ 和 $\text{Com}(\tilde{V}_{i+1}(\mathbf{v}^{(i+1)}))$ 无法自行计算对消息乘积的承诺, 故 \mathcal{P} 需额外发送 $\text{Com}(\tilde{V}_{i+1}(\mathbf{u}^{(i+1)}) \cdot \tilde{V}_{i+1}(\mathbf{v}^{(i+1)}))$ 然后和 \mathcal{V} 调用 \mathcal{ZK}_{prod} 验证承诺消息乘积是否相等.
- (4) 在第 i 轮, \mathcal{V} 在随机挑选 $\alpha^{(i+1)}, \beta^{(i+1)} \xleftarrow{\$} \mathbb{F}$ 后, 需自行计算 $\text{Com}(\alpha^{(i+1)} \tilde{V}_{i+1}(\mathbf{u}^{(i+1)}) + \beta^{(i+1)} \tilde{V}_{i+1}(\mathbf{v}^{(i+1)}))$ 并进行第 $i+1$ 轮的 sum-check 协议.
- (5) 在第 d 轮, \mathcal{V} 根据公共导线值 $(\text{io}_1, \text{io}_2, \dots, \text{io}_{|\mathbf{x}|})$ 和第 (2) 步收到的承诺验证最后一轮 sum-check 协议的正确性.

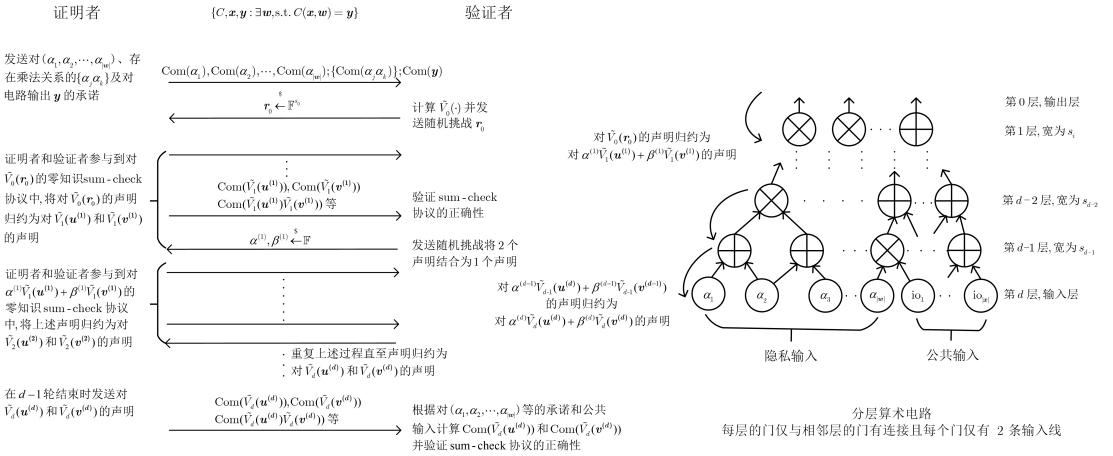
基于 DEIP 的平凡零知识证明可分为 2 部分:

- (1) 在协议第一步, 证明者需对证据 $(\alpha_1, \alpha_2, \dots, \alpha_{|\mathbf{w}|})$ 及 $\{\alpha_j \alpha_k\}$ 作承诺, 其中 $\{\alpha_j \alpha_k\}$ 指所有存在乘法门关系的门值乘积集合; 在协议最后一步, 验证者需根据收到的承诺计算对 $\tilde{V}_d(\mathbf{u}^{(d)})$ 和 $\tilde{V}_d(\mathbf{v}^{(d)})$ 的承诺并验证 sum-check 协议的正确性.
- (2) 调用 d 次零知识的 sum-check 协议.

对于 (2), 由于其只是将域上操作替换为承诺操作及同态运算, 且在普通 DEIP 中不需要第 (1) 步, 故其渐近复杂度与普通 DEIP 是一致的, 但实际证明和验证开销有显著增加.

改进上述零知识证明 基于 DEIP 的平凡零知识证明主要存在如下三点问题:

- (1) 不是简洁的. 通信复杂度为 $(\Theta(|\mathbf{w}|) + O(d \log g))$ 级别的群元素, 其中 $\Theta(|\mathbf{w}|)$ 来自于直接传输证据, $O(d \log g)$ 来自于 d 次零知识 sum-check 协议.
- (2) 实际计算开销大. 由于加性同态承诺中加法运算会转换为乘法运算, 乘法运算会转换成幂运算, 利用加性同态承诺保障零知识性会显著增加实际证明和验证开销.



(3) 待证明陈述表示形式(分层算术电路)不够通用且需要预处理。虽然通过增加电路深度级别的电路门,任意算术电路都可转换为分层算术电路^[32],但是仍需对具体计算问题进行预处理。

近年来的研究在不同程度上解决了上述三点问题,构造了若干基于 DEIP 的简洁 NIZK AoK,其总结列于表 5。表 5 中实现简洁性所采取方案指零知识多项式承诺(zk-PC)及零知识可验证多项式委托(zk-VPD),其用于解决问题(1)。根据多项式承诺或可验证多项式委托方案(列于表 6)的不同,各协议的底层难题假设和可信初始化情况也各不相同。随机掩藏多项式用于解决问题(2),由于基于的 DEIP 不同,各协议支持的电路问题形式也不尽相同。Spartan(针对 R1CS 可满足问题)和 Virgo++(针对任意算术电路)用于解决问题(3)。本章第 6.3 节分别介绍解决上述问题的零知识证明。

6.3 典型协议分析

本节介绍基于 DEIP 的零知识证明典型协议(见图 8),分析各协议的主要思路、协议流程、复杂度及安全性。根据对基于 DEIP 的平凡零知识证明的改进思路及方法的不同,第 6.3.1 小节介绍 ZKvSQL 和 Hyrax,其主要改进是利用 zk-PC 和 zk-VPD 传输证据多项式,从而实现简洁性。第 6.3.2 小节介绍 Libra 和 Virgo,其主要改进是在 ZKvSQL 和 Hyrax 的基础上改用掩藏多项式而不是加性同态承诺实现零知识 sum-check 协议,这减少了大量的群幂运算从而降低了实际证明与验证开销。第 6.3.3 小节介绍 Spartan,其待证明陈述形式是 R1CS 可满足问题,需注意的是严格而言 Spartan 并不是基于 DEIP 构造的,但其与基于 DEIP 的零知识证明思路极为类似,故在本章介绍。第 6.3.4 小节介绍 Virgo++,其在 Libra 和 Virgo 的基础上改进了 DEIP,从而将待证明陈述形式扩展为任意算术电路。

6.3.1 ZKvSQL/Hyrax

主要思路. ZKvSQL 由 Zhang 等人^[51]提出, Hyrax 由 Wahby 等人^[32]提出,它们的主要思路是利用 zk-VPD 和 zk-PC 改进图 7 证明者的第一步操作,从而实现简洁性,其使用的 zk-VPD 及 zk-PC 方案列于表 6。考虑到表 6 中的方案承诺大小 $|c|$ 和通信复杂度均为亚线性,且 d 次零知识 sum-check 协议的通信复杂度为 $O(d \log g)$,因此当 $d = O(\text{poly log } |C|)$ 时,可实现通信复杂度为亚线性的零知识证明。

协议流程. ZKvSQL 与 Hyrax 的协议流程与图 7 基本一致。不同的是,在图 7 证明者的第一步操作中,证明者不再逐个发送对证据的承诺,而是发送对多变量线性多项式 $\tilde{w}(\cdot)$ 的承诺,其由证据 $(\alpha_1, \alpha_2, \dots, \alpha_{|\mathbf{w}|})$ 决定,变量数为 $\log |\mathbf{w}|$ 。协议运行至最后一轮时,证明者不再发送对 $\tilde{V}_d(\mathbf{u}^d)$ 和 $\tilde{V}_d(\mathbf{v}^d)$ 的承诺,而是和验证者运行 zk-PC 中的求值计算协议,他们计算 $\text{Com}(\tilde{V}_d(\mathbf{u}^d))$ 和 $\text{Com}(\tilde{V}_d(\mathbf{v}^d))$ 从而验证 sum-check 协议的正确性。

讨论总结. 现从复杂度和安全性两个方面对 ZKvSQL 和 Hyrax 进行讨论总结。

(1) 现给出利用 zk-PC 实现简洁 NIZK AoK 的复杂度分析,其开销主要分为 d 次零知识 sum-check

表 5 基于 DEIP 的简洁 NIZKAOK 总结

Table 5 Succinct non-interactive zero-knowledge arguments of knowledge based on DEIP

对比项	协议					
	ZKvSQL [51]	Hyrax [32]	Libra [26]	Virgo [28]	Spartan [27]	Virgo++ [29]
待证明陈述表示形式	分层算术电路	分层算术电路	分层算术电路	分层算术电路	一阶约束系统	任意算术电路
电路结构	对数空间均匀电路	数据并行电路	对数空间均匀电路	常规电路	/	常规电路
DEIP	文献 [60]	Gir++ [32, 127]	文献 [26]	文献 [26]	/	文献 [29]
实现简洁性所调用的方案	ZKvSQL-VPD [51]	Hyrax-PC [32]	ZKvSQL-VPD [51]	Virgo-VPD [28]	Hyrax-PC [32] 或 Virgo-VPD [28]	Virgo-VPD [28]
实现非交互基于的模型	ROM	ROM	ROM	ROM	ROM	ROM
启动阶段	私密	公开	私密	公开	公开	公开
可靠性误差 ϵ	$\Theta(\epsilon_{\text{DEIP}} + \epsilon_{\text{VPD}})$	$\Theta(\epsilon_{\text{DEIP}} + \epsilon_{\text{PC}})$	$\Theta(\epsilon_{\text{DEIP}} + \epsilon_{\text{VPD}})$	$\Theta(\epsilon_{\text{DEIP}} + \epsilon_{\text{VPD}})$	$\Theta(\epsilon_{\text{R1CS}} + \epsilon_{\text{PC/VPD}})$	$\Theta(\epsilon_{\text{DEIP}} + \epsilon_{\text{VPD}})$
证明复杂度	$O(C \log g) \mathbb{G}_o$	$O(C + dg \log g) \mathbb{G}_o$	$O(C + \mathbf{in}) \mathbb{F}_o$	$O(\mathbf{in} \log \mathbf{in} + C) \mathbb{F}_o$	$O(n) \mathbb{G}_o$ 或 $O(n \log n) \mathbb{F}_o$	$O(\mathbf{in} \log \mathbf{in} + C) \mathbb{F}_o$
通信复杂度	$O(d \log g + \log \mathbf{w}) \mathbb{G}$	$O(\sqrt{ \mathbf{w} } + d \log Ng) \mathbb{G}$	$O(d \log C) \mathbb{F}$	$O(d \log C + \log^2 \mathbf{in}) \mathbb{F}$	$O(\sqrt{n}) \mathbb{G}$ 或 $O(\log^2 n) \mathbb{F}$	$\min(O(d \log C + d^2), O(C)) \mathbb{F}$
验证复杂度	$O(\mathbf{x} + \mathbf{y} + d \log \log C) \mathbb{G}_o$	$O(\mathbf{x} + \mathbf{y} + d \log(Ng) + dg) \mathbb{G}_o$	$O(\mathbf{x} + \mathbf{y} + d \log C) \mathbb{F}_o$	$O(\mathbf{x} + \mathbf{y} + d \log C + \log^2 \mathbf{in}) \mathbb{F}_o$	$O(\sqrt{n}) \mathbb{G}_o$ 或 $O(\log^2 n) \mathbb{F}_o$	$\min(O(C , O(\mathbf{x} + \mathbf{y} + \log^2 \mathbf{in} + d \log C + d^2)) \mathbb{F}_o$
底层假设	$q\text{-SDH}_{(d, \ell)\text{-EPKE}}$	DLOG	$q\text{-SDH}_{(d, \ell)\text{-EPKE}}$	CRHF	与 zk-PC 及 zk-VPD 有关	CRHF

¹ 由于该类协议是由 DEIP 修改而来, 且 DEIP 与电路每层宽度有关, 故需特意考虑电路输入输出层的宽度.

² \mathbb{G}_o 表示群上操作, \mathbb{G} 表示群元素. \mathbb{F}_o 表示域上操作, \mathbb{F} 表示群元素. $|C|$ 表示电路规模, g 表示电路宽度, d 表示电路深度, $|\mathbf{x}|$ 表示 C-SAT 问题的公共输入长度, $|\mathbf{y}|$ 表示输出长度, $|\mathbf{in}|$ 表示所有输入长度且满足 $|\mathbf{in}| = |\mathbf{x}| + |\mathbf{w}|$. n 表示 R1CS 可满足问题的规模. 数据并行电路 (data-parallel circuit) 是指可以分为 N 份宽度为 g 、深度为 d 的子电路的电路, 且有 $|C| = Ngd$. 对于可靠性误差 ϵ , ϵ_{DEIP} 指 DEIP 的可靠性误差, ϵ_{R1CS} 指针对 R1CS 可满足问题的交互式论证的可靠性误差且有 $\epsilon_{\text{R1CS}} = O(\log n)/|\mathbb{F}|$, 其中 n 指 R1CS 可满足问题的规模, ϵ_{VPD} 和 ϵ_{PC} 分别指 zk-VPD 和 zk-PC 的可靠性误差. 对于底层难题假设, q -SDH 假设见定义 26, (d, ℓ) -EPKE 假设指 (d, ℓ) -extended power knowledge of exponent 假设 (是对 q -PKE 假设的修改). “/” 指当前协议没有该特性值.

³ XXX-PC 是指在对应文献中调用的 zk-PC. ZKvSQL 和 Libra 中的 zk-PC 需要一次性可信初始化, 建立时间为 $O(|\mathbf{in}|)$.

⁴ 对于 Spartan, 根据其调用的 zk-PC 或 zk-VPD 的不同, 协议的复杂度也不一样. 具体的, 若使用 Hyrax-PC, 则证明、通信和验证复杂度分别为 $O(n)$ 、 $O(\sqrt{n})$ 和 $O(\sqrt{n})$; 若使用 Virgo-VPD, 则证明、通信和验证复杂度分别为 $O(n \log n)$ 、 $O(\log^2 n)$ 和 $O(\log^2 n)$.

⁵ 证明、通信和验证复杂度均为协议 1 轮的主要开销, 协议运行轮数及实际证明、验证计算开销和通信量与可靠性误差和目标可靠性误差有关.

表 6 针对多变量线性多项式的零知识可提取多项式承诺方案对比

Table 6 Comparison of several zero-knowledge extractable polynomial commitment schemes for multilinear polynomials

协议方案	启动阶段	\mathcal{P}_{Com}	$\mathcal{P}_{\text{Eval}}$	$ c $	通信复杂度	$\mathcal{V}_{\text{Eval}}$	底层难题假设
ZKvSQL-VPD [51, 125]	私密	$O(n) \mathbb{G}_o$	$O(2^\mu) \mathbb{G}_o$	$O(1) \mathbb{G}$	$O(\mu) \mathbb{G}$	$O(\mu) P$	$q\text{-SDH}_{(d, \ell)\text{-EPKE}}$
Hyrax-PC [32]	公开	$O(\sqrt{2^\mu}) \mathbb{G}_o$	$O(2^\mu) \mathbb{G}_o$	$O(\sqrt{2^\mu}) \mathbb{G}$	$O(\mu) \mathbb{G}$	$O(\sqrt{2^\mu}) \mathbb{G}_o$	DLOG
Virgo-VPD [28]	公开	$O(2^\mu \log(2^\mu)) \mathbb{F}_o$	$O(2^\mu \log(2^\mu)) \mathbb{F}_o$	$O(1) \mathbb{F}$	$O(\mu^2) \mathbb{F}$	$O(\mu^2) \mathbb{F}_o$	CRHF

¹ 由于 PC 与 VPD 在基于 DEIP 的简洁 NIZKAOK 中的作用是一致的, 为表述方便, 本章在下文统一使用 PC 的记法. 事实上, 在分析复杂度时, PC 中的 Com 与 VPD 中的 Com 相对应, PC 中的 $\mathcal{P}_{\text{Eval}}$ 与 VPD 中的 Eval 相对应, PC 中的 c 与 VPD 中的 c 相对应, PC 中的通信复杂度与 VPD 中的 $|\pi|$ 相对应, PC 中的 $\mathcal{V}_{\text{Eval}}$ 与 VPD 中的 Verify 相对应.

² 其中多项式为 μ 元, 共有 n 个单项式. 注, n 与 μ 有关系 $n \leq 2^\mu$. \mathcal{P}_{A} 指在多项式的承诺的 A 阶段 \mathcal{P} 的计算开销 (有 \mathcal{P}_{Com} 和 $\mathcal{P}_{\text{Eval}}$ 两部分). $\mathcal{V}_{\text{Eval}}$ 同理. $|c|$ 指承诺大小. 通信复杂度指 \mathcal{P} 与 \mathcal{V} 在 PC.Eval 协议中的通信开销.

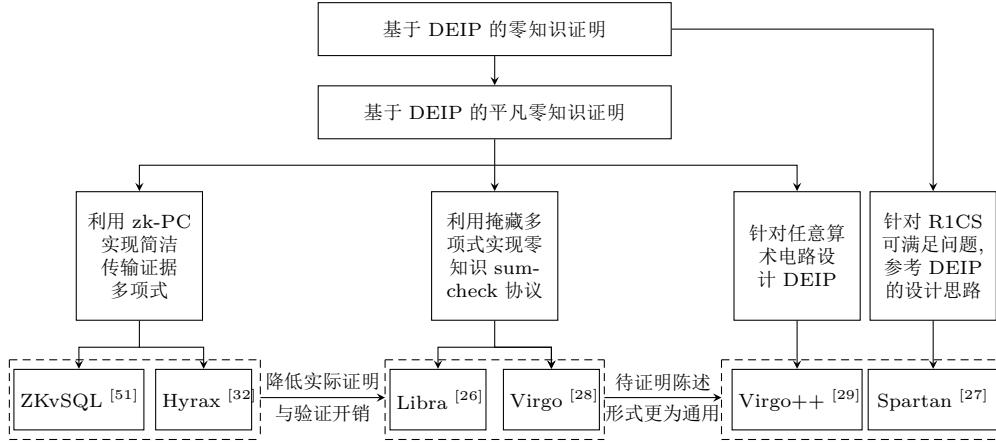


图 8 基于 DEIP 的零知识证明典型协议优化思路

Figure 8 Optimization of typical zero-knowledge proof based on DEIP

协议的开销和参与多项式承诺的开销, 各项开销具体如下.

- 证明复杂度. 证明者需要: 参与 d 次零知识 sum-check 协议, 其与普通 DEIP 的证明复杂度是同级别的, 记该部分计算复杂度为 $\mathcal{P}_{\text{DEIP}}$; 对多项式 $\tilde{w}(\cdot)$ 作承诺及参与求值计算协议, 记该部分计算复杂度为 \mathcal{P}_{PC} .
- 通信复杂度. 通信开销包括: d 次零知识 sum-check 协议所需传输的信息, 其与 DEIP 本身的通信复杂度相同, 记该部分通信复杂度为 π_{DEIP} ; 传输对 $\tilde{w}(\cdot)$ 的承诺带来的通信开销和求值计算的通信开销, 记该部分通信复杂度为 π_{PC} .
- 验证复杂度. 验证者需要: 参与 d 次零知识 sum-check 协议, 其与普通 DEIP 的验证复杂度是同级别的, 记该部分计算复杂度为 $\mathcal{V}_{\text{DEIP}}$; 参与对多项式 $\tilde{w}(\cdot)$ 的求值计算, 记该部分计算复杂度为 \mathcal{V}_{PC} .

根据不同零知识证明所采取的不同 DEIP 和 zk-PC, 基于各参数的关系, 协议的证明、通信和验证复杂度详见表 7.

表 7 利用不同 DEIP 和 zk-PC 实现零知识知识论证的复杂度分析

Table 7 Complexity analysis of zero-knowledge argument of knowledge based on different DEIPs and zk-PC schemes

协议方案	$\mathcal{P}_{\text{DEIP}}$	\mathcal{P}_{PC}	$\mathcal{V}_{\text{DEIP}}$	\mathcal{V}_{PC}	π_{DEIP}	π_{PC}
ZKvSQL [51]	$O(C \log g) \mathbb{G}_o$	$(O(n) + O(w)) \mathbb{G}_o$	$O(x + y) + d \text{poly log } g \mathbb{G}_o$	$O(\log w) P$	$O(d \log g) \mathbb{G}$	$O(\log w) \mathbb{G}$
Hyrax [32]	$O(C + dg \log g) \mathbb{G}_o$	$O(w) \mathbb{G}_o$	$O(x + y + dg + d \log(Ng)) \mathbb{G}_o$	$O(\sqrt{ w }) \mathbb{G}_o$	$O(d \log Ng) \mathbb{G}$	$O(\sqrt{ w }) \mathbb{G}$
Libra [26]	$O(C + \text{in}) \mathbb{F}_o$	$(O(n) + O(w) + O(d)) \mathbb{G}_o$	$O(x + y + d \log C) \mathbb{F}_o$	$(O(\log w) + O(d)) P$	$O(d \log C) \mathbb{F}$	$O(\log w) \mathbb{G}$
Virgo [28]	$O(C) \mathbb{F}_o$	$O(\text{in} \log \text{in}) \mathbb{F}_o$	$O(x + y + d \log C) \mathbb{F}_o$	$O(\log^2 \text{in}) \mathbb{F}_o$	$O(d \log C + \log^2 \text{in}) \mathbb{F}$	$O(1) \mathbb{F}$

¹ 各参数关系有 $\mu = \log |w|$ 和 $n \leq |w| \leq g$.

² 计算 $\mathcal{P}_{\text{DEIP}} + \mathcal{P}_{\text{PC}}$ 即可得各协议的证明复杂度, 其中 $\mathcal{P}_{\text{DEIP}}$ 详见对应文献. 验证复杂度和通信复杂度同理可得.

(2) 安全性分析. ZKvSQL 和 Hyrax 的底层难题假设主要来自于其调用的 zk-PC, 也就是说, ZKvSQL 基于的假设是 q -SDH 假设和 (d, ℓ) -EPKE 假设, Hyrax 基于的假设是 DLOG 假设. 协议的可靠性来自于 zk-PC 的绑定性、DEIP 本身的可靠性和 sum-check 协议中加性同态承诺的绑定性. 协议的知识可靠性来自于 zk-PC 的可提取性. 协议的零知识性来自于 zk-PC 的隐藏性、zk-PC 中求值计算协议的零知识性和加性同态承诺的隐藏性.

6.3.2 Libra/Virgo

主要思路. Libra 由 Xie 等人^[26] 提出, 相比于第 6.2.2 小节中的平凡 DEIP, 在利用 zk-PC 实现简洁性的基础上, Libra 利用随机掩藏多项式降低了零知识 sum-check 协议的证明和验证计算开销. 在此基础上, Zhang 等人^[28] 提出了一种新的多项式承诺 (表 6 中的 Virgo-VPD) 并将其应用于 Libra 中, 实现了一种启动阶段参数可公开生成的 NIZKAOK, 即 Virgo. 本小节主要介绍 Libra 的主要思路和协议流程.

在 sum-check 协议中, 证明者发送给验证者的消息是与电路相关的多项式 $g_i(X_i)$, 这有可能会破坏零知识性. 基于 Pedersen 承诺的零知识 sum-check 协议虽可解决这一问题, 但大量的群幂运算会显著增加证明和验证计算开销. 在此背景下, Chiesa、Forbes 和 Spooner^[129] 指出为 sum-check 协议中的多项式 $g(\cdot)$ 增加随机多项式 $\rho f(\cdot)$ 即可实现零知识性, 其中 ρ 是由验证者选择的随机挑战. 此时证明者和验证者是对多项式 $g(\cdot) + \rho f(\cdot)$ 调用 sum-check 协议. 假设形如定义 27 的 zk-PC 存在, 零知识 sum-check 协议具体见协议 5, 其省略了公共参数 pp、验证陷门 vp 和随机数.

然而, 利用上述零知识 sum-check 协议难以直接构造基于 DEIP 的零知识证明, 这是因为在 DEIP 中第 i 轮 sum-check 协议的最后一轮, \mathcal{V} 会获得 $\tilde{V}_i(\mathbf{u}^{(i)})$ 和 $\tilde{V}_i(\mathbf{v}^{(i)})$ (对应于协议 5 中的 $g(\mathbf{r})$), 这会泄露一定的隐私信息. 为了解决这一问题, Chiesa、Forbes 和 Spooner 进一步指出为 $\tilde{V}_i(\cdot)$ 增加一个随机掩藏低度多项式 $R_i(\cdot)$ 可实现零知识性. 具体来说, 定义多项式

$$\dot{V}_i(\mathbf{c}) = \tilde{V}_i(\mathbf{c}) + Z_i(\mathbf{c}) \sum_{\mathbf{e} \in \{0,1\}^\lambda} R_i(\mathbf{c}, \mathbf{e}), \quad (23)$$

其中 λ 是安全参数, $Z_i(\mathbf{c}) = \prod_{j=1}^{s_i} c_j(1 - c_j)$, 即对于所有的 $\mathbf{c} \in \{0,1\}^{s_i}$, $Z_i(\mathbf{c}) = 0$. 基于此, 等式 (23) 可展开为

$$\begin{aligned} \dot{V}_i(\mathbf{c}) &= \sum_{\mathbf{a}, \mathbf{b} \in \{0,1\}^{s_{i+1}}} \left(\widetilde{\text{mult}}_{i+1}(\mathbf{a}, \mathbf{b}, \mathbf{c}) \left(\dot{V}_{i+1}(\mathbf{a}) \dot{V}_{i+1}(\mathbf{b}) \right) \right. \\ &\quad \left. + \widetilde{\text{add}}_{i+1}(\mathbf{a}, \mathbf{b}, \mathbf{c}) \left(\dot{V}_{i+1}(\mathbf{a}) + \dot{V}_{i+1}(\mathbf{b}) \right) + Z_i(\mathbf{c}) \sum_{\mathbf{e} \in \{0,1\}^\lambda} R_i(\mathbf{c}, \mathbf{e}) \right) \\ &= \sum_{\mathbf{a}, \mathbf{b} \in \{0,1\}^{s_{i+1}}, \mathbf{e} \in \{0,1\}^\lambda} \left(I(\mathbf{0}, \mathbf{e}) \cdot \widetilde{\text{mult}}_{i+1}(\mathbf{a}, \mathbf{b}, \mathbf{c}) \left(\dot{V}_{i+1}(\mathbf{a}) \dot{V}_{i+1}(\mathbf{b}) \right) \right. \\ &\quad \left. + I(\mathbf{0}, \mathbf{e}) \cdot \widetilde{\text{add}}_{i+1}(\mathbf{a}, \mathbf{b}, \mathbf{c}) \left(\dot{V}_{i+1}(\mathbf{a}) + \dot{V}_{i+1}(\mathbf{b}) \right) + I((\mathbf{a}, \mathbf{b}), \mathbf{0}) \cdot Z_i(\mathbf{c}) R_i(\mathbf{c}, \mathbf{e}) \right), \end{aligned} \quad (24)$$

其中当且仅当 $\mathbf{x} = \mathbf{y}$ 时 $I(\mathbf{x}, \mathbf{y}) = 1$, 否则 $I(\mathbf{x}, \mathbf{y}) = 0$.

协议 5 Libra 中的零知识 sum-check 协议^[26]

公共输入: 域 \mathbb{F} , T, m, ℓ .

证明者秘密输入: $g(\mathbf{x})$, 且有 $T = \sum_{\mathbf{x} \in \{0,1\}^\ell} g(\mathbf{x})$.

1. 证明者 \mathcal{P} 随机选取多项式 $f(\mathbf{x})$, 其中每个变量的度也为 m . \mathcal{P} 将 $F = \sum_{\mathbf{x} \in \{0,1\}^\ell} f(\mathbf{x})$ 和对 $f(\cdot)$ 的承诺 $\text{com}_f = \text{Com}(f(\cdot))$ 发给验证者 \mathcal{V} .

2. 验证者 \mathcal{V} 选取 $\rho \xleftarrow{\$} \mathbb{F}$ 发送给 \mathcal{P} 并计算 $T + \rho F$.
3. \mathcal{P} 和 \mathcal{V} 调用对 $T + \rho F$ 的 sum-check 协议.
4. 在 sum-check 协议的最后一轮, \mathcal{V} 收到 $h_\ell(\mathbf{r}) = g(\mathbf{r}) + \rho f(\mathbf{r})$, 然后打开多项式承诺 com_f 获取 $f(\cdot)$ 在点 $\mathbf{r} = (r_1, r_2, \dots, r_\ell)$ 处的值.
5. \mathcal{V} 计算 $h_\ell(\mathbf{r}) - \rho f(\mathbf{r})$ 并验证其与从谕示获得的 $g(\mathbf{r})$ 是否相等.

输出: 比特 b , 若上述验证均通过, 输出 $b = 1$; 否则输出 $b = 0$.

当在 sum-check 协议中调用等式 (24) 时, 由于多项式 $\dot{V}_i(\cdot)$ 被随机多项式掩藏了, 验证者在第 i 层 sum-check 协议的最后一轮收到的信息 $\dot{V}_{i+1}(\mathbf{u}^{(i+1)})$ 和 $\dot{V}_{i+1}(\mathbf{v}^{(i+1)})$ 就不再泄露隐私信息, 其中

$$\mathbf{u}^{(i+1)}, \mathbf{v}^{(i+1)} \xleftarrow{\$} \mathbb{F}^{s_{i+1}}.$$

此外, 为保障证明者不利用 $R_i(\cdot)$ 作恶, 其需事先发送对多项式 $R_i(\cdot)$ 的多项式承诺. 然而, 由于 $R_i(\cdot)$ 是低度多项式而不是多变量线性多项式, 而且有 $s_i + \lambda$ 个变量, 调用表 6 中适配于多变量低度多项式的 ZKvSQL-VPD 会导致指数组级别复杂度的开销.

为解决该问题, Xie 等人^[26] 指出将 $R_i(\cdot)$ 替换为一个变量数为 2、每个变量度为 2 的多项式即可保障零知识性. 在第 i 轮 sum-check 协议的最后一轮, 证明者只需向验证者发送 $\dot{V}_{i+1}(\cdot)$ 的两个值 $\dot{V}_{i+1}(\mathbf{u}^{(i+1)})$ 和 $\dot{V}_{i+1}(\mathbf{v}^{(i+1)})$; 此外, 验证者还需打开承诺获取 $R_i(\mathbf{u}^{(i+1)}, \mathbf{t}^{(i+1)})$ 和 $R_i(\mathbf{v}^{(i+1)}, \mathbf{t}^{(i+1)})$, 其中 $\mathbf{u}^{(i+1)}, \mathbf{v}^{(i+1)} \xleftarrow{\$} \mathbb{F}^{s_{i+1}}, \mathbf{t}^{(i+1)} \xleftarrow{\$} \mathbb{F}^\lambda$. 也就是说, 验证者仅获得了 4 个值. Xie 等人进一步指出只需保障这 4 个值是线性独立的即可保障零知识性, 因此可将等式(23)中的 $Z_i(\mathbf{c}) \sum_{e \in \{0,1\}^\lambda} R_i(\mathbf{c}, e)$ 替换为 $Z_i(\mathbf{c}) \sum_{e \in \{0,1\}} R_i(c_1, e)$, 其中 c_1 是向量 \mathbf{c} 的第一个值. 利用该零知识 sum-check 协议, 结合 ZKvSQL-VPD, Xie 等人构造了 Libra. 在此基础上, Zhang 等人^[28] 基于 RS 码提出了一种新的多项式承诺, 实现了求值计算协议具有对数级别通信复杂度、启动阶段系统参数可公开生成的 zk-PC (列于表 6), 从而构造了 Virgo.

协议流程. Libra 与 Virgo 的协议流程与 ZKvSQL 和 Hyrax 基本一致. 不同的是, Libra 和 Virgo 不再需要使用加性同态承诺, 而是采用随机掩藏多项式 $\rho f(\cdot)$ 和 $R(\cdot)$ 实现零知识性, 其中 $\rho f(\cdot)$ 用于保障 sum-check 协议中证明者发送给验证者的信息是不泄露隐私的, $R(\cdot)$ 用于保障 sum-check 协议最后的谕示也是不泄露隐私的. 不过, 为了防止恶意证明者利用 $R(\cdot)$, 证明者需在协议开始时发送对 $R(\cdot)$ 的多项式承诺.

讨论总结. 现分别从复杂度和安全性两个方面对 Libra 和 Virgo 进行讨论总结.

- (1) Libra 的复杂度. 与第 6.3.1 小节类似, Libra 的复杂度开销也可分为两部分, 即 d 次零知识 sum-check 协议的开销和多项式承诺的开销. Libra 中的零知识 sum-check 协议不再需要使用加性同态承诺, 而是调用 zk-PC 对电路每一层的掩藏多项式进行承诺, 因此不再需要大量的群幂运算. 然而为实现整个协议的零知识性, 需额外对 d 个变量数为 2 且每个变量度为 2 的多项式调用 zk-PC. 因此 Libra 中的 \mathcal{P}_{PC} 和 \mathcal{V}_{PC} 需分别对应增加 $O(d)$, 其具体复杂度列于表 7.
- (2) Virgo 的复杂度. Zhang 等人指出在分析 Virgo 的复杂度时, 可将调用 zk-PC 的开销分为对电路输入层 (变量数为 $\log |\mathbf{in}| = \log(|\mathbf{x}| + |\mathbf{w}|)$) 调用承诺的开销和对电路中间层调用承诺的开销, 他们分别优化了这两者的渐近性质, 并指出相比于 d 次 sum-check 协议的开销, 对电路中间层调用承诺的开销在渐近复杂度上可以忽略. 因此 Virgo 的复杂度开销包括 d 次 sum-check 协议的开销和对电路输入层调用 zk-PC 的开销两部分, 其具体复杂度列于表 7.
- (3) 安全性分析. 同 ZKvSQL 和 Hyrax 类似, Libra 和 Virgo 的底层难题假设主要来自于其调用的 zk-PC, 也就是说, Libra 基于的假设是 q -SDH 假设和 (d, ℓ) -EPKE 假设, Virgo 基于的假设是 CRHF 存在性. 协议的可靠性来自于 zk-PC 的绑定性、DEIP 本身的可靠性. 协议的知识可靠性来自于 zk-PC 的可提取性. 协议的零知识性来自于 zk-PC 的隐藏性、zk-PC 中求值计算协议的零知识性和 sum-check 协议的零知识性.

6.3.3 Spartan

上述协议所针对的待证明陈述表示形式均是分层算术电路, 同时为实现简洁性和亚线性的验证复杂度, 需要对电路有更多的要求, 比如电路深度 d 不能太大, 电路是常规电路、数据并行电路等. 一个改进方向就是设计待证明陈述表示形式更为通用的零知识证明. 其中, Setty^[27] 提出了 Spartan, 一种针对 R1CS 可满足问题的简洁 NIZKAoK, 由于 R1CS 可满足问题是高级语言编译器的常见目标程序^[62, 65] 且任意 C-SAT 问题都可用 R1CS 可满足问题表示^[27], 因此相比于针对分层算术电路可满足问题的零知识证明, Spartan 的待证明陈述更为通用. 事实上, 严格而言 Spartan 并不是基于 DEIP 构造的, 但其与基于 DEIP 的零知识证明思路极为类似, 故在本章介绍. 此外, Zhang 等人^[29] 提出了 Virgo++, 其是直接针对任意算术电路可满足问题的简洁 NIZKAoK. 本小节介绍 Spartan, 第 6.3.4 小节介绍 Virgo++.

主要思路. Spartan 的设计思路如图 9 所示.

首先介绍编码 R1CS 可满足问题的思路. 为借鉴之前利用 sum-check 协议构造零知识证明的思路, 需

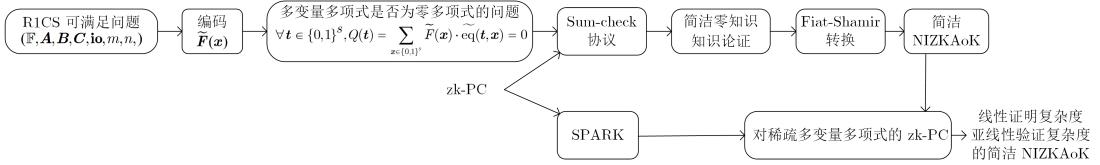


图 9 Spartan 的主要思路 [27]

Figure 9 Main idea of Spartan

将 R1CS 可满足问题编码为低度多项式. 考虑形如定义 3 的 R1CS 组, 令 $s = \lceil \log_2 m \rceil$, 则矩阵 $\mathbf{A}, \mathbf{B}, \mathbf{C}$ 可视为函数 $A(\cdot, \cdot), B(\cdot, \cdot), C(\cdot, \cdot) : \{0, 1\}^s \times \{0, 1\}^s \rightarrow \mathbb{F}$. 类似的, 令 $\mathbf{z}^T = (\mathbf{io}, 1, \mathbf{w})$, 则 \mathbf{z}^T 也可以视为函数 $Z(\cdot) : \{0, 1\}^s \rightarrow \mathbb{F}$. 定义函数 $F(\mathbf{x}) : \{0, 1\}^s \rightarrow \mathbb{F}$

$$F(\mathbf{x}) = \left(\sum_{\mathbf{y} \in \{0,1\}^s} A(\mathbf{x}, \mathbf{y}) \cdot Z(\mathbf{y}) \right) \cdot \left(\sum_{\mathbf{y} \in \{0,1\}^s} B(\mathbf{x}, \mathbf{y}) \cdot Z(\mathbf{y}) \right) - \sum_{\mathbf{y} \in \{0,1\}^s} C(\mathbf{x}, \mathbf{y}) \cdot Z(\mathbf{y}). \quad (25)$$

考虑到当 $\mathbf{x} = 0^s$ 时, $\sum_{\mathbf{y} \in \{0,1\}^s} A(\mathbf{x}, \mathbf{y}) \cdot Z(\mathbf{y})$ 就是 \mathbf{Az} 的第一个元素, 以此类推, 则可推知上述 R1CS 组是可满足的当且仅当对于任意的 $\mathbf{x} \in \{0, 1\}^s$, $F(\mathbf{x}) = 0$. 与 DEIP 类似, 可将 $F(\mathbf{x})$ 转换为多变量扩展形式 $\tilde{F}(\mathbf{x}) : \mathbb{F}^s \rightarrow \mathbb{F}$, 其具体为

$$\tilde{F}(\mathbf{x}) = \left(\sum_{\mathbf{y} \in \{0,1\}^s} \tilde{A}(\mathbf{x}, \mathbf{y}) \cdot \tilde{Z}(\mathbf{y}) \right) \cdot \left(\sum_{\mathbf{y} \in \{0,1\}^s} \tilde{B}(\mathbf{x}, \mathbf{y}) \cdot \tilde{Z}(\mathbf{y}) \right) - \sum_{\mathbf{y} \in \{0,1\}^s} \tilde{C}(\mathbf{x}, \mathbf{y}) \cdot \tilde{Z}(\mathbf{y}). \quad (26)$$

则有上述 R1CS 组是可满足的当且仅当对于任意的 $\mathbf{x} \in \{0, 1\}^s$, $\tilde{F}(\mathbf{x}) = 0$.

由于 $\tilde{F}(\cdot)$ 是低度多项式, 证明者和验证者可通过 sum-check 协议验证 $\sum_{\mathbf{x} \in \{0,1\}^s} \tilde{F}(\mathbf{x}) \stackrel{?}{=} 0$. 然而, 上式成立并不足以说明对于任意的 $\mathbf{x} \in \{0, 1\}^s$, $\tilde{F}(\mathbf{x}) = 0$. 为此, Spartan 构造了多项式 $Q(\mathbf{t}) : \mathbb{F}^s \rightarrow \mathbb{F}$

$$Q(\mathbf{t}) = \sum_{\mathbf{x} \in \{0,1\}^s} \tilde{F}(\mathbf{x}) \cdot \tilde{\text{eq}}(\mathbf{t}, \mathbf{x}), \quad (27)$$

其中 $\tilde{\text{eq}}(\mathbf{t}, \mathbf{x}) = \prod_{i=1}^s (t_i \cdot x_i + (1 - t_i) \cdot (1 - x_i))$, t_i 和 x_i 分别表示向量 \mathbf{t} 和 \mathbf{x} 的第 i 位.

由于对于任意的 $\mathbf{t} \in \{0, 1\}^s$, 有 $Q(\mathbf{t}) = \tilde{F}(\mathbf{t})$, 因此上述 R1CS 组是可满足的当且仅当对于任意的 $\mathbf{t} \in \mathbb{F}^s$, 有 $Q(\mathbf{t}) = 0$, 即 $Q(\cdot)$ 是零多项式. 由 Schwartz-Zippel 引理, 对于随机挑战 $\tau \stackrel{\$}{\leftarrow} \mathbb{F}^s$, 若 $Q(\tau) = 0$, 则有较大的概率说明 $Q(\cdot)$ 是零多项式, 且有 $s = O(\log m) = O(\log n)$. 又 $Q(\tau)$ 为求和形式, 故可调用 sum-check 协议验证 $Q(\tau) \stackrel{?}{=} 0$.

至此, R1CS 组就可编码为对应的多变量多项式, 且 R1CS 问题是可满足的当该多变量多项式是零多项式. 借鉴之前基于 DEIP 构造零知识证明的思路, 利用表 6 中的 Hyrax-PC 和 Virgo-VPD, 容易构造针对 R1CS 可满足问题的简洁零知识知识论证. 其中, 分别需要对证据多项式 $\tilde{w}(\cdot)$ 和 $\tilde{A}(\cdot, \cdot), \tilde{B}(\cdot, \cdot), \tilde{C}(\cdot, \cdot)$ 作多项式承诺. 随后利用 ROM 下的 Fiat-Shamir 启发式, 即可将上述论证转换为简洁 NIZK AoK.

然而, 平凡调用 zk-PC 会导致平方级别的证明复杂度. 这是因为 $\tilde{A}(\cdot, \cdot), \tilde{B}(\cdot, \cdot), \tilde{C}(\cdot, \cdot)$ 的变量个数为 $2 \log m$, 代入表 6 可得证明复杂度起码为平方级别. 为优化证明复杂度, Setty 指出 R1CS 可满足问题编码后的多项式是稀疏的, 其变量个数实际上远达不到 $2 \log m$, 故仍有可能实现线性级别的证明复杂度. Setty 基于此设计了 SPARK, 一种将针对多变量线性多项式的 zk-PC 高效转化为针对稀疏多变量线性多项式的 zk-PC 的编译器, 并最终实现了线性证明复杂度、亚线性通信和验证复杂度的简洁 NIZK AoK.

协议流程. Spartan 的协议流程与基于 DEIP 的零知识证明是类似的, 现简要介绍 Spartan 中的简洁交互式知识论证. 首先证明者 \mathcal{P} 将对多项式 $\tilde{A}(\cdot, \cdot), \tilde{B}(\cdot, \cdot), \tilde{C}(\cdot, \cdot)$ 及 $\tilde{w}(\cdot)$ 的承诺发送给验证者 \mathcal{V} . 随后 \mathcal{V} 选取随机挑战 $\mathbf{t} \stackrel{\$}{\leftarrow} \mathbb{F}^s$, 然后 \mathcal{P} 和 \mathcal{V} 调用对 $Q(\mathbf{t})$ 的 sum-check 协议, 并将验证 $Q(\mathbf{t}) \stackrel{?}{=} 0$ 归约为验证

$\tilde{F}(\mathbf{r}_x) \stackrel{?}{=} 0$, 其中 $\mathbf{r}_x \stackrel{\$}{\leftarrow} \mathbb{F}^s$ 是 \mathcal{V} 在 sum-check 协议中选取的随机挑战. 为计算 $\tilde{F}(\mathbf{r}_x)$, 记

$$\dot{A}(\mathbf{r}_x) = \sum_{\mathbf{y} \in \{0,1\}^s} \tilde{A}(\mathbf{r}_x, \mathbf{y}) \cdot Z(\mathbf{y}), \quad \dot{B}(\mathbf{r}_x) = \sum_{\mathbf{y} \in \{0,1\}^s} \tilde{B}(\mathbf{r}_x, \mathbf{y}) \cdot Z(\mathbf{y}), \quad \dot{C}(\mathbf{r}_x) = \sum_{\mathbf{y} \in \{0,1\}^s} \tilde{C}(\mathbf{r}_x, \mathbf{y}) \cdot Z(\mathbf{y}), \quad (28)$$

$$v_A = \dot{A}(\mathbf{r}_x), \quad v_B = \dot{B}(\mathbf{r}_x), \quad v_C = \dot{C}(\mathbf{r}_x), \quad (29)$$

\mathcal{P} 将 v_A, v_B, v_C 发送给 \mathcal{V} , 随后 \mathcal{P} 和 \mathcal{V} 还需再次调用 sum-check 协议验证等式 (29) 是否成立, 并将验证这三个等式归约为验证

$$\tilde{A}(\mathbf{r}_x, \mathbf{r}_y) \tilde{Z}(\mathbf{r}_y) \stackrel{?}{=} v_A, \quad \tilde{B}(\mathbf{r}_x, \mathbf{r}_y) \tilde{Z}(\mathbf{r}_y) \stackrel{?}{=} v_B, \quad \tilde{C}(\mathbf{r}_x, \mathbf{r}_y) \tilde{Z}(\mathbf{r}_y) \stackrel{?}{=} v_C, \quad (30)$$

其中 $\mathbf{r}_y \stackrel{\$}{\leftarrow} \mathbb{F}^s$ 是 \mathcal{V} 在此轮 sum-check 协议中选取的随机挑战. 最后, 为计算 $\tilde{A}(\mathbf{r}_x, \mathbf{r}_y) \tilde{Z}(\mathbf{r}_y)$ 、 $\tilde{B}(\mathbf{r}_x, \mathbf{r}_y) \tilde{Z}(\mathbf{r}_y)$ 和 $\tilde{C}(\mathbf{r}_x, \mathbf{r}_y) \tilde{Z}(\mathbf{r}_y)$, \mathcal{V} 需打开对多项式 $\tilde{A}(\cdot, \cdot), \tilde{B}(\cdot, \cdot), \tilde{C}(\cdot, \cdot)$ 及 $\tilde{w}(\cdot)$ 的承诺.

同基于 DEIP 的零知识证明类似, 利用基于 Pedersen 的零知识 sum-check 协议, 可将上述论证转换为简洁零知识知识论证.

讨论总结. 在协议复杂度方面, 根据调用 zk-PC 的不同, Spartan 的各项性能也不一样. 若调用 Hyrax-PC, Spartan 的证明复杂度为 $O(n)$, 验证复杂度和通信复杂度均为 $O(\sqrt{n})$; 若调用 Vrigo-VPD, 证明复杂度为 $O(n \log n)$, 验证复杂度和通信复杂度均为 $O(\log^2 n)$. 在安全性方面, Spartan 的(知识)可靠性来自于针对 R1CS 可满足问题的交互式论证的可靠性、zk-PC 的绑定性和 zk-PC 中求值计算协议的(知识)可靠性. Spartan 的零知识性来自于 zk-PC 的隐藏性和求值计算协议的零知识性.

6.3.4 Virgo++

Virgo++ 由 Zhang 等人^[29] 提出, 他们的主要贡献是提出了一种针对任意算术电路的 DEIP 并将其应用于 Virgo 中, 进而得到了一种针对任意算术电路可满足问题的简洁 NIZK AoK. Virgo++ 的协议流程和安全性分析均与 Virgo 类似, 本小节仅简要介绍 Virgo++ 中的 DEIP, 并给出简单的复杂度分析.

主要思路. Zhang 等人指出制约系列 DEIP 效率的最主要因素之一就是 DEIP 只能应用于分层算术电路. 在渐近级别上, 将任意电路转化为分层算术电路会将电路规模从 $O(|C|)$ 增加到 $O(d|C|)$; 在实际工程应用中, 将任意电路转化为分层算术电路会增加证明者 1-2 个数量级的计算开销^[29]. 此外, 将任意电路转换为 R1CS 也不够高效. 基于以上背景, Zhang 等人提出了一种针对通用算术电路且不额外增加证明者计算开销的 DEIP, 并采用 Virgo 中的方法将 DEIP 转换为简洁 NIZK AoK.

Virgo++ 中 DEIP 构造的主要思路如下. 虽然通用电路没有严格的分层结构, 但仍可以进行逻辑意义上的分层, 即一个电路门的输入门的层级一定比该电路门高. 考虑到第 i 层的任意电路门 α 一定至少有一条输入导线位于 $i+1$ 层(否则该门就不可能位于第 i 层), 而另一条输入导线可能位于 $(i+1) \sim d$ 的任意一层, 因此电路层 i 的多项式规模为 $O((d-i)g)$. 基于此思想可构造证明复杂度为 $O(d|C|)$ 的 DEIP, 这是因为证明者调用 sum-check 协议的复杂度为 $O(dg + (d-1)g + \dots + g) = O(d^2g) = O(d|C|)$. 此外, 在电路层 i , \mathcal{P} 还需在调用 sum-check 协议完毕后将对 $O((d-i)g)$ 级别多项式取值的声明转换为 1 个声明, 该阶段的复杂度也起码为 $O(d|C|)$. 上述平凡方法与将任意电路转换为分层算术电路的渐近复杂度是至少一样的, 也就没有实际意义. 针对上述两个问题, Zhang 等人分别提出了两个解决方案. 对于前一个问题, 他们指出由于每层电路门的输入门最多为 $2g$, 因此调用 sum-check 协议的复杂度可降低至 $O(dg)$. 对于后一个问题, 他们指出可将转换声明的计算归约为运行一个针对特定分层算术电路的普通 DEIP, 由于归约后协议轮数为 $O(|C|)$, 因此该阶段的证明复杂度也为 $O(|C|)$. 基于以上两点改进, Zhang 等人实现了针对通用算术电路且证明复杂度为 $O(|C|)$ 的 DEIP, 并最终将其转换为了简洁 NIZK AoK.

协议流程. Virgo++ 的协议流程与 Virgo 的协议流程基本一致, 即在 DEIP 的基础上, 利用多项式承诺实现简洁性, 利用随机掩藏多项式和 zk-PC 保障零知识性. 不同的是, Virgo++ 中 DEIP 所针对的电路可以是任意的算术电路.

讨论总结. Zhang 等人提出的 DEIP 的性能表现分析如下. 对于规模为 $|C|$ 、输入长度为 $|x|$ 、深度为 d 的通用电路, 该协议可靠性误差为 $O(d \log |C|/\mathbb{F})$, 证明复杂度为 $O(|C|)$, 通信复杂度为 $\min\{O(d \log |C| + d^2), O(|C|)\}$. 当电路为常规电路时, 验证复杂度为 $\min\{O(|x| + |y| + d \log |C| + d^2), O(|C|)\}$. 基于该 DEIP, 结合 Virgo 的复杂度分析方法, 可以得出 Virgo++ 的各项复杂度, 其列于表 5.

6.4 总结

基于 DEIP 的零知识证明是本文涉及的唯一一种可实现证明复杂度与电路规模成线性关系、通信和验证复杂度与电路规模成亚线性关系的系统参数可公开生成的零知识证明, 但也存在若干限制. 第一, 由于 DEIP 按层计算的特性, 只有针对深度较低的电路该类零知识证明的通信复杂度才与电路规模成亚线性. 因此, 能否及如何削弱电路深度的线性因子影响是未来的一个研究方向. 第二, 由于 DEIP 对于特殊结构的电路, 比如对数空间均匀电路或者数据并行电路才能快速验证的特性, 只有针对特殊结构的电路, 验证复杂度才与电路规模成亚线性.

7 基于 IPA 的零知识证明

本章介绍基于内积论证 (IPA) 的零知识证明. 该类零知识证明的底层难题假设均基于离散对数假设 (见定义 32), 故也被称为基于离散对数假设的零知识证明. 然而为便于与其他基于离散对数假设的零知识证明加以区分 (如 Hyrax, 见第 6.3.1 小节), 且由于内积论证是实现该类协议对数级别通信复杂度的关键技术, 故本文将该类协议称为基于 IPA 的零知识证明. 该类协议的核心思路是将电路中的所有约束归约转化为内积形式的陈述, 然后调用内积论证实现对数级别通信复杂度的简洁 NIZKAoK. 本章第 7.1 节介绍相关定义及概念, 第 7.2 节介绍该类协议的背景及主要思路, 第 7.3 节介绍典型协议.

7.1 定义及概念

给定群 \mathbb{G} , 记其生成元为 g . 记双线性配对群的生成元为 $g_1, g_2, g_T = e(g_1, g_2)$. 为表述方便, 固定群的生成元 g 后, 记 g^r 为 $[r]$, 令 $n \in \mathbb{N}$, 记 $(g^{r_1}, g^{r_2}, \dots, g^{r_n})$ 为 $[\mathbf{r}]$, 对于配对运算, 记 $e([r]_1, [s]_2) = [rs]_T$, 记 $[\mathbf{a} \cdot \mathbf{r}] = \prod_{i=1}^n g^{a_i r_i} = g^{\sum_{i=1}^n a_i r_i}$. 特别的, 对于 n 为偶数的向量 $\mathbf{r} = (r_1, r_2, \dots, r_n)$ (不是偶数可适量填充), 记 $\mathbf{r}_{\frac{1}{2}} = (r_1, r_2, \dots, r_{n/2}), \mathbf{r}_{\frac{2}{2}} = (r_{n/2+1}, r_{n/2+2}, \dots, r_n)$.

定义 31 (向量分布^[54]) 向量分布 (vector distribution) \mathcal{D}_n 用于描述一个 n 长向量的分布情况. 记向量分布 \mathcal{U}_n 为 $\mathbf{x} = (x_1, x_2, \dots, x_n)$, 向量分布 \mathcal{ML}_{2^n} 为 $\bar{\mathbf{x}} = (1, x_1, x_2 x_1, \dots, x_v x_{v-1} \dots x_1)$, 其生成方式可首先设 $\bar{\mathbf{x}}$ 为 1, 然后由递归 $\{\bar{\mathbf{x}} \leftarrow (\bar{\mathbf{x}}, x_i \bar{\mathbf{x}})\}_{i \in [v]}$ 生成.

定义 32 (离散对数假设) 令 $\text{Gen}(\cdot)$ 为参数生成算法, λ 为安全参数. 离散对数假设 (discrete logarithm assumption, DLOG 假设) 是指对于任意的 PPT 敌手 \mathcal{A} , 都有:

$$\Pr \left[(q, \mathbb{G}, g) \leftarrow \text{Gen}(1^\lambda); r \xleftarrow{\$} \mathbb{Z}_q; r' \leftarrow \mathcal{A}(q, \mathbb{G}, [r]) : r = r' \right] \leq \text{negl}(\lambda). \quad (31)$$

DLOG 假设有若干自然扩展^[54]. 在 n -DLOG 假设中敌手 \mathcal{A} 的输入为 $([1], [r], \dots, [r^n])$, 在非对称 DLOG 假设 (asymmetric DLOG assumption, A-DLOG 假设) 中敌手 \mathcal{A} 的输入为 $([r]_1, [r]_2)$, 在非对称 n -DLOG 假设 (n -A-DLOG 假设) 中敌手 \mathcal{A} 的输入为 $([1]_1, [r]_1 \dots, [r^n]_1, [1]_2, [r]_2 \dots, [r^n]_2)$. 在以上情形中, \mathcal{A} 的目标均为获取 r .

对于定义 31 中的向量分布, DLOG 假设有变种 \mathcal{D}_n -Find-Rep 假设^[54], 即对于任意的 PPT 敌手 \mathcal{A} , 都有

$$\Pr \left[(q, \mathbb{G}, g) \leftarrow \text{Gen}(1^\lambda); \mathbf{r} \xleftarrow{\$} \mathcal{D}_n; \mathbf{a} \leftarrow \mathcal{A}(q, \mathbb{G}, [\mathbf{r}]) : [\mathbf{a} \cdot \mathbf{r}] = [0] \wedge \mathbf{a} \neq \mathbf{0} \right] \leq \text{negl}(\lambda). \quad (32)$$

其中, \mathcal{U}_n -Find-Rep 假设可归约为 DLOG 假设, 离散对数关系假设^[30] 与 \mathcal{U}_n -Find-Rep 假设等价. 此外, 可以证明 \mathcal{ML}_{2^n} -Find-Rep 假设可归约为 A-DLOG 假设^[54].

7.2 背景及主要思路

内积论证提出之前的相关工作 1998 年, Cramer 和 Damgård^[102] 利用 Pedersen 承诺实现了一种针对 C-SAT 问题的证明、通信和验证复杂度均为 $O(|C|)$ 的零知识论证。该协议的零知识性由承诺隐藏性保障, 协议的正确性由 Pedersen 承诺的同态性质保障。基于该思想, Groth^[132] 和 Seo^[133] 在保持证明和验证复杂度不变的同时, 将协议的通信复杂度降低为 $O(\sqrt{|C_M|})$, 其中 $|C_M|$ 为电路中乘法门的个数。

基于 Linear-PCP 的零知识证明虽然实现了较低的通信和验证复杂度, 但具有若干缺点, 其中之一就是底层难题假设均为不可证伪假设。Groth 和 Seo 提出的协议虽然仅基于离散对数假设并具有更高的安全性, 但通信复杂度仍较高。在此基础上, Bootle 等人^[30] 提出了内积论证并利用其构造了对数级别通信复杂度的 NIZKAO (记为 BCCGP16)。证明者通过内积论证可利用循环递归的方式证明他拥有两个公开向量承诺的消息, 且这两个消息的内积等于某个公开值。对于长度为 n 的消息向量, 内积论证的通信复杂度为 $O(\log n)$ 。本小节接下来介绍内积论证的主要思路及其优化。

内积论证. 在 BCCGP16 的内积论证中, 证明者 \mathcal{P} 可向验证者 \mathcal{V} 证明对于公共输入 $A, B \in \mathbb{G}, \mathbf{g}, \mathbf{h} \in \mathbb{G}^n$ 和公开标量 $z \in \mathbb{Z}_q$, \mathcal{P} 拥有向量 \mathbf{a}, \mathbf{b} , 满足 $A = \mathbf{g}^\mathbf{a}$ 、 $B = \mathbf{h}^\mathbf{b}$ 和 $\mathbf{a} \cdot \mathbf{b} = z$ 。该陈述可记为

$$\left\{ (\mathbf{g}, \mathbf{h}, A, B, z; \mathbf{a}, \mathbf{b}) : A = \mathbf{g}^\mathbf{a} \wedge B = \mathbf{h}^\mathbf{b} \wedge \mathbf{a} \cdot \mathbf{b} = z \right\}, \quad (33)$$

其中分号前后分别表示公共输入和证据。若 \mathbf{g}, \mathbf{h} 的生成方式为 $\mathbf{g} \leftarrow [r], \mathbf{h} \leftarrow [s]$, 则陈述 (33) 可改记为

$$\{([r], [s], A, B, z; \mathbf{a}, \mathbf{b}) : A = [\mathbf{a} \cdot r] \wedge B = [\mathbf{b} \cdot s] \wedge \mathbf{a} \cdot \mathbf{b} = z\}. \quad (34)$$

内积论证的核心思想是将针对 n 长向量的陈述根据 \mathcal{V} 的随机挑战归约为对 $n/2$ 长向量的等价陈述, 在向量不断缩减至为标量后, \mathcal{P} 只需要直接发送标量即可。首先基于 \mathcal{V} 的随机挑战 c 构造长度一半于原密钥长度的承诺密钥, 即 $[r'] \leftarrow [c^{-1} \mathbf{r}_{\frac{n}{2}} + c^{-2} \mathbf{r}_{\frac{n}{2}}]$ 。其次, 为防止 \mathcal{P} 利用新的承诺密钥 $[r']$ 作恶, \mathcal{P} 需在挑战阶段之前发送部分承诺值 $A_{-1} = [\mathbf{a}_{\frac{n}{2}} \cdot \mathbf{r}_{\frac{n}{2}}]$ 和 $A_1 = [\mathbf{a}_{\frac{n}{2}} \cdot \mathbf{r}_{\frac{n}{2}}]$ 。此时新证据为 $\mathbf{a}' = c\mathbf{a}_{\frac{n}{2}} + c^2\mathbf{a}_{\frac{n}{2}}$ 。最后, \mathcal{P} 和 \mathcal{V} 计算新承诺

$$A' \leftarrow [\mathbf{a}' \cdot r'] = [(c\mathbf{a}_{\frac{n}{2}} + c^2\mathbf{a}_{\frac{n}{2}}) \cdot (c^{-1}\mathbf{r}_{\frac{n}{2}} + c^{-2}\mathbf{r}_{\frac{n}{2}})] = AA_{-1}^{c^{-1}} A_1^c. \quad (35)$$

对于承诺密钥 $[s]$ 、承诺 B 和秘密输入 \mathbf{b} , 利用挑战的逆 c^{-1} 构造对应的承诺密钥 $[s']$ 、新证据 \mathbf{b}' 和承诺值 B' 即可, 具体的,

$$[s'] \leftarrow [cs_{\frac{n}{2}} + c^2s_{\frac{n}{2}}], \quad \mathbf{b}' \leftarrow c^{-1}\mathbf{b}_{\frac{n}{2}} + c^{-2}\mathbf{b}_{\frac{n}{2}}, \quad B' \leftarrow [\mathbf{b}' \cdot s'] = B_{-1}^c BB_1^{c^{-1}}. \quad (36)$$

对于 z , \mathcal{P} 需在挑战阶段前构造 $z_{-1} \leftarrow \mathbf{a}_{\frac{n}{2}} \cdot \mathbf{b}_{\frac{n}{2}}$ 和 $z_1 \leftarrow \mathbf{a}_{\frac{n}{2}} \cdot \mathbf{b}_{\frac{n}{2}}$, 此时更新后的 $z' \leftarrow \mathbf{a}' \cdot \mathbf{b}' = z_{-1}c + z + z_1c^{-1}$ 。至此, 归约后的新陈述为 $\{([r'], [s'], A', B', z'; \mathbf{a}', \mathbf{b}') : A' = [\mathbf{a}' \cdot r'] \wedge B' = [\mathbf{b}' \cdot s'] \wedge \mathbf{a}' \cdot \mathbf{b}' = z'\}$ 。

BCCGP16 中的内积论证具体如协议 6 所示, 协议共有 $\log_2 n$ 轮, 每轮需发送 4 个 \mathbb{G} 上的群元素、2 个 \mathbb{Z}_q 上的域元素; 最后一轮还需额外发送 a 和 b 共 2 个域元素, 故总通信量为 $(6 \log_2 n + 2)$ 个元素。证明者的主要计算开销为计算 A_k, B_k, z_k 和计算新承诺密钥 $[r'], [s']$, 而这需要 $O(n)$ 级别的群上运算和域上运算; 验证者的主要计算开销为计算 $A', B', [r'], [s']$ 及 z' , 这需要 $O(n)$ 级别的群上运算和 $O(\log n)$ 级别的域上运算。对于安全性, 可以证明, 基于 \mathcal{U}_n -Find-Rep 假设, 该内积论证具有完美完备性和统计意义的证据扩展可仿真性, 若有 $O(n^2)$ 个对于不同随机挑战 c 的接受副本, 则会以 $1 - \text{negl}(\lambda)$ 的概率提取出证据 \mathbf{a}, \mathbf{b} 。

Bünz 等人^[31] 指出将陈述 (33) 中的承诺 A 和 B 结合为一个承诺 $P = \mathbf{g}^\mathbf{a} \mathbf{h}^\mathbf{b}$ 就足以用于证明很多问题 (范围证明、C-SAT 可满足性问题等)。具体的, 修改后的陈述可记为

$$\{([r], [s], z, P; \mathbf{a}, \mathbf{b}) : P = [\mathbf{a} \cdot r][\mathbf{b} \cdot s] \wedge z = \mathbf{a} \cdot \mathbf{b}\}. \quad (37)$$

 协议 6 BCCGP16 中的内积论证^[30]

公共输入: $(\mathbb{G}, q, g, [r], [s], A, B, z)$.
证明者秘密输入: \mathbf{a}, \mathbf{b} .

1. \mathcal{P} 向 \mathcal{V} 发送 $A_{-1}, B_{-1}, z_{-1}, , A_1, B_1, z_1$, 具体有

$$\begin{aligned} A_{-1} &\leftarrow [\mathbf{a}_{\frac{1}{2}} \cdot \mathbf{r}_{\frac{1}{2}}], & B_{-1} &\leftarrow [\mathbf{b}_{\frac{1}{2}} \cdot \mathbf{s}_{\frac{1}{2}}], & z_{-1} &\leftarrow \mathbf{a}_{\frac{1}{2}} \cdot \mathbf{b}_{\frac{1}{2}}, \\ A_1 &\leftarrow [\mathbf{a}_{\frac{1}{2}} \cdot \mathbf{r}_{\frac{1}{2}}], & B_1 &\leftarrow [\mathbf{b}_{\frac{1}{2}} \cdot \mathbf{s}_{\frac{1}{2}}], & z_1 &\leftarrow \mathbf{a}_{\frac{1}{2}} \cdot \mathbf{b}_{\frac{1}{2}}. \end{aligned}$$

2. \mathcal{V} 向 \mathcal{P} 发送随机挑战 $c \xleftarrow{\$} \mathbb{Z}_q^*$.

3. \mathcal{P} 和 \mathcal{V} 共同计算新的承诺密钥 $[r'], [s']$ 和新承诺 A', B' 及 z'

$$\begin{aligned} [r'] &\leftarrow [c^{-1}\mathbf{r}_{\frac{1}{2}} + c^{-2}\mathbf{r}_{\frac{1}{2}}], & A' &\leftarrow AA_{-1}^{c^{-1}}A_1^c, \\ [s'] &\leftarrow [c\mathbf{s}_{\frac{1}{2}} + c^2\mathbf{s}_{\frac{1}{2}}], & B' &\leftarrow BB_{-1}^cBB_1^{c^{-1}}, & z' &\leftarrow z_{-1}c + z + z_1c^{-1}. \end{aligned}$$

4. \mathcal{P} 计算下一轮的新证据 $\mathbf{a}' \leftarrow c\mathbf{a}_{\frac{1}{2}} + c^2\mathbf{a}_{\frac{1}{2}}$ 和 $\mathbf{b}' \leftarrow c^{-1}\mathbf{b}_{\frac{1}{2}} + c^{-2}\mathbf{b}_{\frac{1}{2}}$ 并参与到下一轮循环中, 此时待证明陈述被归约为 $\{([r'], [s'], A', B', z'; \mathbf{a}', \mathbf{b}') : A' = [\mathbf{a}' \cdot \mathbf{r}'] \wedge B' = [\mathbf{b}' \cdot \mathbf{s}'] \wedge \mathbf{a}' \cdot \mathbf{b}' = z'\}$.

5. 上述循环归约过程共重复 $t = \log_2 n$ 次, 直至 \mathbf{a} 和 \mathbf{b} 缩减为标量, 此时 \mathcal{P} 只需直接将 a 和 b 发给 \mathcal{V} 然后 \mathcal{V} 自行验证如下等式是否成立即可

$$A_t \stackrel{?}{=} [a \cdot r_t], \quad B_t \stackrel{?}{=} [b \cdot s_t], \quad z_t \stackrel{?}{=} a \cdot b,$$

其中 A_t, B_t 表示第 t 轮中的承诺, $[r_t]$ 和 $[s_t]$ 表示第 t 轮的承诺密钥.

为证明陈述 (37), Bulletproofs 首先证明了如下形式的陈述

$$\{([r], [s], u, P; \mathbf{a}, \mathbf{b}) : P = [\mathbf{a} \cdot \mathbf{r}][\mathbf{b} \cdot \mathbf{s}]u^{\mathbf{a} \cdot \mathbf{b}}\}, \quad (38)$$

然后证明了陈述 (37). 证明陈述 (38) 的主要思路也是在每一轮将对 n 长向量的陈述递归为对 $n/2$ 长向量的陈述, 具体见协议 7. 在该内积论证中, 每轮需要传输的群元素仅为 2 个 (L 和 R), 且最后一轮需额外发送 a 和 b 共 2 个域元素, 故总通信量为 $(2\log_2 n + 2)$ 个元素. 证明者计算开销与 BCCGP16 类似, 为 $O(n)$ 次群上运算; 但验证者每轮不再需要计算 $[r']$ 、 $[s']$ 而只需计算 A' 、 B' , 但需额外验证 2 个配对等式, 故为 $O(\log n)$ 次群上运算和配对运算. 对于安全性, 与 BCCGP16 类似, Bulletproofs 中的内积论证也具有完美完备性和统计意义的证据扩展可仿真性.

 协议 7 Bulletproofs 中的内积论证^[31]

公共输入: $(\mathbb{G}, q, [r], [s], P, u)$, 其中 $P, u \in \mathbb{G}$.
证明者秘密输入: \mathbf{a}, \mathbf{b} , 其满足 $P = [\mathbf{a} \cdot \mathbf{r}][\mathbf{b} \cdot \mathbf{s}]u^{\mathbf{a} \cdot \mathbf{b}}$.

1. \mathcal{P} 计算 $L, R \in \mathbb{G}$ 并向 \mathcal{V} 发送 L, R , 其中

$$L \leftarrow [\mathbf{a}_{\frac{1}{2}} \cdot \mathbf{r}_{\frac{1}{2}}][\mathbf{b}_{\frac{1}{2}} \cdot \mathbf{s}_{\frac{1}{2}}]u^{\mathbf{a}_{\frac{1}{2}} \cdot \mathbf{b}_{\frac{1}{2}}}, \quad R \leftarrow [\mathbf{a}_{\frac{1}{2}} \cdot \mathbf{r}_{\frac{1}{2}}][\mathbf{b}_{\frac{1}{2}} \cdot \mathbf{s}_{\frac{1}{2}}]u^{\mathbf{a}_{\frac{1}{2}} \cdot \mathbf{b}_{\frac{1}{2}}}.$$

2. \mathcal{V} 向 \mathcal{P} 发送随机挑战 $c \xleftarrow{\$} \mathbb{Z}_q^*$.

3. \mathcal{P} 和 \mathcal{V} 共同计算新的承诺密钥 $[r'], [s']$ 和新承诺 P' , 其中

$$[r'] \leftarrow [c^{-1}\mathbf{r}_{\frac{1}{2}} + cr_{\frac{1}{2}}], \quad [s'] \leftarrow [c\mathbf{s}_{\frac{1}{2}} + c^{-1}\mathbf{s}_{\frac{1}{2}}], \quad P' \leftarrow L^{c^2} \cdot P \cdot R^{c^{-2}}.$$

4. \mathcal{P} 计算下一轮的证据 $\mathbf{a}' \leftarrow c\mathbf{a}_{\frac{1}{2}} + c^{-1}\mathbf{a}_{\frac{1}{2}}$ 和 $\mathbf{b}' \leftarrow c^{-1}\mathbf{b}_{\frac{1}{2}} + cb_{\frac{1}{2}}$ 并参与到下一轮循环中. 此时新承诺密钥为 $[r']$ 和 $[s']$, 归约后的陈述为 $\{(\mathbb{G}, q, [r'], [s'], P', u; \mathbf{a}', \mathbf{b}') : P' = [\mathbf{a}' \cdot \mathbf{r}'][\mathbf{b}' \cdot \mathbf{s}']u^{\mathbf{a}' \cdot \mathbf{b}'}\}$.

5. 协议共重复 $t = \log_2 n$ 轮直至 \mathbf{a} 和 \mathbf{b} 缩减为标量, 此时 \mathcal{P} 只需直接将 a 和 b 发给 \mathcal{V} 然后 \mathcal{V} 自行验证本轮的 P_t 是否满足 $P_t = [a \cdot r_t][b \cdot s_t]u^{ab}$ 即可.
-

Zhang 等人^[134] 指出相比于 BCCGP16 中的内积论证, 虽然 Bulletproofs 中的内积论证具有较

低的总通信量, 但修改陈述会使得其应用条件相对苛刻。针对此, Zhang 等人在不修改陈述的前提下将 BCCGP16 中内积论证的通信复杂度降低了约 2/3。相比于 Bulletproofs, Zhang 等人的内积论证虽然总通信量较高, 但其应用条件和 BCCGP16 一样宽松。

DRZ20^[54] 指出在承诺密钥 \mathbf{g} 和 \mathbf{h} 结构化, 即 \mathbf{r} 和 \mathbf{s} 具有特殊分布的情况下, 可将协议 6 的验证复杂度降低到对数级别。然而为保障协议的可靠性, 密钥需由可信第三方生成, 也就是说该内积论证需要 CRS。

具体的, 在 BCCGP16 的内积论证中, 验证者的主要计算开销为计算新承诺密钥 $[\mathbf{r}'] \leftarrow [c^{-1}\mathbf{r}_{\frac{1}{2}} + c^{-2}\mathbf{r}_{\frac{2}{2}}] \text{ 和 } [\mathbf{s}'] \leftarrow [c\mathbf{s}_{\frac{1}{2}} + c^2\mathbf{s}_{\frac{2}{2}}]$, 该过程共需要 $2n + n + \dots + 4 = O(n)$ 次群幂操作。然而, 若承诺密钥分布为 $\mathcal{ML}_{n=2^v}$, 仅考虑 $[\mathbf{r}]$, 由于 $[\mathbf{r}_{\frac{1}{2}}] = [x_v \mathbf{r}_{\frac{1}{2}}]$, 新承诺密钥满足 $[\mathbf{r}'] = [(c^{-1} + x_v c^{-2}) \mathbf{r}_{\frac{1}{2}}]$ 。一直递归, 可得最终的承诺密钥为标量, 表示为

$$[r'] = \left[\prod_{i=1}^v (c_i^{-1} + x_{v-i+1} c_i^{-2}) \right]. \quad (39)$$

故对于拥有 (x_1, x_2, \dots, x_v) 的特定验证者, 其不需在每轮计算新的承诺密钥而只需验证等式(39)是否成立即可, 且验证开销为对数级别的配对运算。

对于公开验证者, 由于验证者不知道 x_1, x_2, \dots, x_v , 其无法直接验证等式(39), 因此需引入 CRS。考虑协议第一轮, 验证者需验证 $[r']_1 \stackrel{?}{=} [(c^{-1} + x_v c^{-2}) \mathbf{r}_{\frac{1}{2}}]_1$, 即对于所有的 $j \leq 2^{v-1}$, 验证 $[r'_j]_1 \stackrel{?}{=} [c^{-1} r_j + c^{-2} x_v r_j]_1$ 。其他轮类似。这意味着 CRS 中起码应存储 $\{[\mathbf{r}]_1, [\mathbf{x}]_2 = ([x_1]_2, [x_2]_2, \dots, [x_v]_2)\}$ 。然而, 直接验证这一系列等式需要线性级别的配对操作。针对该问题, DRZ20 进一步指出在每一轮验证者只需验证一个等式即可保障证据扩展可仿真性^[54]。例如, 在协议第 1 轮验证者只需验证 $[r'_1]_1 \stackrel{?}{=} [c^{-1} r_1 + c^{-2} x_v r_1]_1$, 即验证

$$e([r'_1 - c^{-1} r_1]_1, [1]_2) \stackrel{?}{=} e([c^{-2} r_1]_1, [x_v]_2), \quad (40)$$

其他轮类似。此时验证者在每轮只需进行常数级别的配对运算, 故验证复杂度是对数级别的。此外, 每轮验证形如等式(40)的等式也意味着相比于 BCCGP16 每轮 \mathcal{P} 需额外发送 $[r'_1]_1, [s'_1]_1$ 共 2 个元素, 故通信复杂度为 $(8 \log_2 n + 2)$ 个元素。与 BCCGP16 类似, 证明者计算开销也为 $O(n)$ 级别的群上和域上运算, DRZ20 中的内积论证也具有完美完备性和统计意义的证据扩展可仿真性。

7.3 典型协议分析

基于内积论证的零知识证明的主要构造思路为: 首先将电路中的乘法门约束和乘法门之间的线性约束利用 Schwartz-Zippel 引理归约为一个多项式的某一特定项系数为零的问题, 然后将该问题转化为内积论证的陈述表示形式, 最后调用内积论证实现零知识证明。事实上, 由于上述过程中前两个步骤是较为简单的, 对该类零知识证明的改进大都与内积论证的改进紧密相关, 相关协议总结列于表 8, 各协议优化思路见图 10。基于第 7.2 节中涉及到的三种内积论证, 本章第 7.3.1 小节介绍 BCCGP16, 第 7.3.2 小节介绍 Bulletproofs 和 HKR19, 第 7.3.3 小节介绍 DRZ20。

7.3.1 BCCGP16

主要思路与协议流程 BCCGP16 中交互式简洁零知识知识论证的主要思路与协议流程如图 11 所示, 说明如下。

(1) 在协议交互之前, 证明者 \mathcal{P} 将算术电路所有的门约束分为乘法门约束和不同乘法门之间的线性约束。对于乘法门约束, \mathcal{P} 将电路中所有乘法门的左输入 \mathbf{a} 、右输入 \mathbf{b} 和输出 \mathbf{c} 排布为 $m \times n$ 的矩阵 $\mathbf{A}, \mathbf{B}, \mathbf{C}$, 其中左输入矩阵 \mathbf{A} 的每一行分别记为 $(\mathbf{a}_1 = (a_{1,1}, a_{1,2}, \dots, a_{1,n}), \dots, \mathbf{a}_m = (a_{m,1}, a_{m,2}, \dots, a_{m,n}))$, 矩阵 \mathbf{B}, \mathbf{C} 同理。这样, 电路中的乘法门约束就可记为 $\mathbf{A} \odot \mathbf{B} = \mathbf{C}$, 共有 $mn = |\mathcal{C}_M|$ 个等式。对于不同乘法门之间的线性约束, 其可记为对于 $q \in [Q]$, 有

$$\sum_{i=1}^m \mathbf{a}_i \cdot \mathbf{w}_{q,a,i} + \sum_{i=1}^m \mathbf{b}_i \cdot \mathbf{w}_{q,b,i} + \sum_{i=1}^m \mathbf{c}_i \cdot \mathbf{w}_{q,c,i} = K_q, \quad (41)$$

表 8 基于 IPA 的简洁 NIZK AoK 总结
Table 8 Succinct non-interactive zero-knowledge arguments of knowledge based on IPA

对比项	协议			
	BCCGP16 [30]	Bulletproofs [31]	HKR19 [53]	DRZ20 [54]
待证明陈述表示形式	算术电路	算术电路	算术电路	算术电路
调用的内积论证	BCCGP16-IPA [30]	Bulletproofs-IPA [31]	Bulletproofs-IPA	DRZ20-IPA [54]
对应 IPA 的证明复杂度	$8n E$ $6n M$	$8n + 2 \log_2 n E$ $6n M$	$8n + 2 \log_2 n E$ $6n M$	$8n E_1$ $6n M$
对应 IPA 的验证复杂度	$(4n + 4 \log_2 n) E$ $2 \log_2 n M$	$(4n + 2 \log_2 n) E$	$(4n + 2 \log_2 n) E$	$8 \log_2 n E_1$ $4 \log_2 n P$ $2 \log_2 n M$
对应 IPA 的通信复杂度	$4 \log_2 n \mathbb{G}$ $2 \log_2 n \mathbb{F}$	$2 \log_2 n \mathbb{G}$	$2 \log_2 n \mathbb{G}$	$6 \log_2 n \mathbb{G}_1$ $2 \log_2 n \mathbb{F}$
实现非交互基于的模型	ROM	ROM	ROM	ROM
启动阶段	公开	公开	公开	私密
论证的证明复杂度	$O(C) \mathbb{F}_o$ $O(C_M) \mathbb{G}_o$	$O(C) \mathbb{F}_o$ $O(C_M) \mathbb{G}_o$	$O(C) \mathbb{F}_o$ $O(C_M) \mathbb{G}_o$	$O(C) \mathbb{F}_o$ $O(C_M) \mathbb{G}_o$
论证的验证复杂度	$O(C_M) \mathbb{G}_o$	$O(C_M) \mathbb{G}_o$	$O(C_M) \mathbb{G}_o$	$O(\log C_M) \mathbb{G}_o$ $O(\log C_M) P$
论证的通信复杂度	$(4 \log_2 C_M + 7) \mathbb{G}$ $(2 \log_2 C_M + 6) \mathbb{F}$	$(2 \log_2 C_M + 8) \mathbb{G}$ $5 \mathbb{F}$	$(2 \log_2 [C_M + 2] + 3) \mathbb{G}$ $2 \mathbb{F}$	$O(\log C_M) \mathbb{G}$ $O(\log C_M) \mathbb{F}$
可靠性误差 ϵ	$\Theta(\epsilon_{\text{PR}} + \epsilon_{\text{IPA}})$	$\Theta(\epsilon_{\text{PR}} + \epsilon_{\text{IPA}})$	$\Theta(\epsilon_{\text{PR}} + \epsilon_{\text{IPA}})$	$\Theta(\epsilon_{\text{PR}} + \epsilon_{\text{IPA}})$
底层难题假设	$\mathcal{U}_{ C }$ -Find-Rep 假设	$\mathcal{U}_{ C }$ -Find-Rep 假设	$\mathcal{U}_{ C }$ -Find-Rep 假设	$\mathcal{ML}_{ C }$ -Find-Rep 假设

¹ 其中 xxx-IPA 表示在 xxx 协议中首次提出的内积论证, n 表示内积论证中陈述的向量长度。 E 、 M 、 P 分别指群幂运算、域上乘法运算和非对称群中的配对运算, E_1 指双线性映射中群 \mathbb{G}_1 上的群幂运算, \mathbb{G} 、 \mathbb{F} 分别指对应群和域中的元素, \mathbb{G}_o 、 \mathbb{F}_o 分别指对应群和域中的运算。DRZ20 中的可信初始化用于确保承诺密钥结构化的正确性。对于可靠性误差, $\Theta(\epsilon_{\text{PR}})$ 指将电路中所有约束归约为一个多项式约束的可靠性误差, 其具体数值可由 Schwartz-Zippel 引理计算得出, $\Theta(\epsilon_{\text{IPA}})$ 指内积论证本身的知识可靠性误差。底层难题假设的具体定义见定义 32, 其中 $\mathcal{U}_{|C|}$ -Find-Rep 假设与离散对数关系假设 [30, §2.1] 等价。

² 证明、通信和验证复杂度均为协议 1 轮开销, 协议实际运行轮数及实际证明、验证计算开销和通信量与可靠性误差和目标可靠性误差有关。

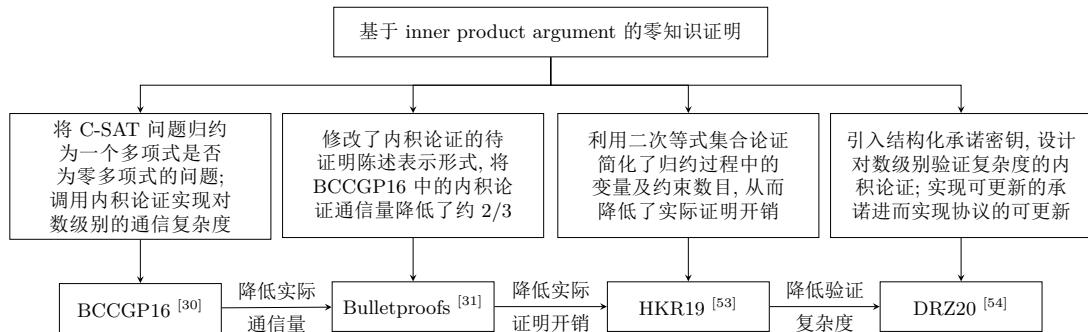


图 10 基于 IPA 的零知识证明典型协议优化思路
Figure 10 Optimization of typical zero-knowledge proof based on inner product argument

其中 $\mathbf{w}_{q,a,i}, \mathbf{w}_{q,b,i}, \mathbf{w}_{q,c,i}$ 为常向量, K_q 为常标量. 例如, 假设电路中仅有一个加法门且门的左输入、右输入、输出分别为 $2a_{1,1} + a_{1,2} + b_{1,1}$, 则此时 $Q = 1, m = 1, \mathbf{w}_{1,a,1} = (2, 1, 0, \dots, 0), \mathbf{w}_{1,b,1} = (-1, 0, \dots, 0)$, $K_1 = 0$, 等式(41)等价于 $2a_{1,1} + a_{1,2} - b_{1,1} = 0$. 由于每个乘法门最多有两个输入, 因此线性约束等式最多有 $Q \leq 2mn$ 个.

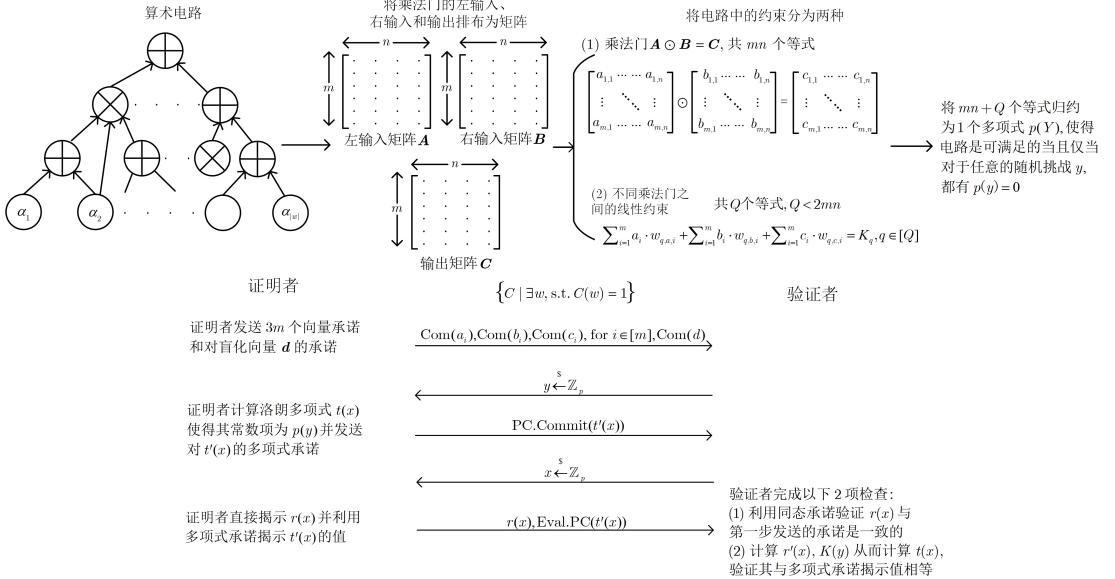


图 11 BCCGP16 的主要思路与协议流程 [30]

Figure 11 Main idea and process of BCCGP16

为同时验证 $mn + Q$ 个等式, \mathcal{P} 将这些等式归约为一个多项式 $p(Y)$, 其思路是为每个等式逐个乘 Y 的 j 次幂, $j \in [mn + Q]$. 具体的, 构造

$$p_M(Y) = \sum_{i=1}^m \sum_{j=1}^n (a_{i,j} b_{i,j} - c_{i,j}) Y^{i+(j-1)m}, \quad (42)$$

用于验证乘法门约束, 构造

$$p_L(Y) = \sum_{q=1}^Q \left(\sum_{i=1}^m a_i \cdot \mathbf{w}_{q,a,i} + \sum_{i=1}^m b_i \cdot \mathbf{w}_{q,b,i} + \sum_{i=1}^m c_i \cdot \mathbf{w}_{q,c,i} \right) Y^{mn+q} - \sum_{q=1}^Q K_q Y^{mn+q}, \quad (43)$$

用于验证线性约束. 此时 C-SAT 问题被归约为 $p(Y) = p_M(Y) + p_L(Y)$ 是否为零多项式的问题. 由 Schwartz-Zippel 引理, 验证 $p(Y) = p_M(Y) + p_L(Y)$ 是否为零多项式可通过选取随机挑战 $y \stackrel{\$}{\leftarrow} \mathbb{Z}_p$ 然后验证 $p(y) \stackrel{?}{=} 0$ 实现, 其可靠性误差约为 $(mn + Q)/|\mathbb{F}|$.

(2) 在协议交互阶段, 首先 \mathcal{P} 发送对证据向量 $\{\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i\}_{i \in [m]}$ 的承诺. 其次, 在收到第一次随机挑战 y 后, \mathcal{P} 构造洛朗多项式 $t(X)$ 使得 $t(X)$ 的常数项为 $p(y)$, 并构造对去除 $t(X)$ 常数项的多项式 $t'(X)$ 的多项式承诺. 其中 $t(X) = r(X) \cdot r'(X) - 2K(y)$ 为内积形式. 一个 $t(X)$ 的例子见本小节 BCCGP16 的平凡改进部分. 具体而言, $r(X)$ 可由 $\{\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i\}_{i \in [m]}$ 计算得出, $\text{Com}(r(X))$ 也可由 $\text{Com}(\mathbf{a}_i), \text{Com}(\mathbf{b}_i), \text{Com}(\mathbf{c}_i)$ 计算得出; $r'(X) = r(X) \odot (y^m, y^{2m}, \dots, y^{nm}) + 2s(X), s(X)$ 和 $K(y)$ 均由电路结构和随机挑战决定. 也就是说, 只需给出 $r(x)$ 验证者就可自己构造 $r'(x), K(y)$. 最后, 在 \mathcal{V} 发送第二次随机挑战 x 后, \mathcal{P} 将 $r(x)$ 发送给 \mathcal{V} 并利用多项式承诺揭示 $t'(x)$ 的值. 由于直接发送 $r(x)$ 可能泄露隐私信息, 为保障零知识性, 需引入盲化向量 d 对 $r(X)$ 进行盲化, 故第一轮还需发送对盲化向量 d 的承诺.

(3) 在检查阶段, \mathcal{V} 首先利用承诺的同态属性, 验证 $\mathbf{r}(x)$ 与利用第一轮收到承诺构造的 $\mathbf{r}(x)$ 是一致的; 然后根据电路结构和 y 自行计算 $\mathbf{s}(x), K(y)$, 并计算 $t(x) \leftarrow \mathbf{r}(x) \cdot \mathbf{r}'(x) - 2K(y)$, 最后验证 $t'(x) \stackrel{?}{=} t(x)$. 若验证通过, 则由 Schwarz-Zippel 引理可知左式以极高的概率等于右式, 又右式与左式之差即为 $t(X)$ 的常数项 $p(y)$, 故可说明 $p(y)$ 有极高的概率为 0, 因此协议的可靠性得以保障.

上述协议最后需直接发送向量 $\mathbf{r}(x)$. BCCGP16 指出可不直接发送 $\mathbf{r}(x)$ 而发送对 $\mathbf{r}(x), \mathbf{r}'(x)$ 的承诺并调用内积论证验证 $t'(x) \stackrel{?}{=} \mathbf{r}(x) \cdot \mathbf{r}'(x) - 2K(y)$, 进而降低通信复杂度. 事实上, 对 $\mathbf{r}(x), \mathbf{r}'(x)$ 的承诺也可由 \mathcal{V} 根据对 $\{\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i\}_{i \in [m]}, \mathbf{d}$ 的承诺、电路结构及随机挑战自行算出. 故此时的通信开销仅包含利用多项式承诺揭示 $t'(x)$ 及内积论证所需要的通信量, 分析如下.

讨论总结. 首先分析复杂度. 对于图 11, \mathcal{P} 的计算开销包括构造多项式 $p(Y)$, 具体为 $O(|C|)$ 级别的域上运算; 计算洛朗多项式 $t(X)$, 具体为 $O(mn) = O(|C_M|)$ 级别的域上运算; 承诺向量 $\{\mathbf{a}_i, \mathbf{b}_i, \mathbf{c}_i\}_{i \in [m]}$ 和 \mathbf{d} 和承诺多项式 $t(X)$, 具体都为 $O(|C_M|)$ 次群上运算. \mathcal{V} 的计算开销主要分为打开对多项式 $t'(x)$ 的承诺和基于承诺的同态属性验证 $\mathbf{r}(x)$ 的一致性和计算对 $\mathbf{r}'(x), \mathbf{s}(x)$ 的承诺, 具体为 $O(mn) = O(|C_M|)$ 次群上运算. 通信开销为发送 $3m$ 个对 n 长向量的向量承诺, 具体为 $O(m)$; 打开长为 n 的向量多项式 $\mathbf{r}(X)$ 在 x 的取值 $\mathbf{r}(x)$, 具体为 $O(n)$; 打开多项式承诺 $t'(x)$, 具体为 $O(\sqrt{m})$; 总计为 $O(m) + O(n)$. 考虑到 $mn = |C_M|$, 因此设置 $m \approx n$ 可实现 $O(\sqrt{|C_M|})$ 级别的通信复杂度.

如果调用内积论证, 协议会增加 $\log_2 n$ 轮. 此时 \mathcal{P} 的计算开销会因为内积论证增加 $O(|C_M|)$ 级别的群上运算. \mathcal{V} 的计算开销主要分为打开对多项式 $t'(x)$ 的承诺, 基于承诺的同态属性计算对 $\mathbf{r}(x)$ 、 $\mathbf{r}'(x)$ 和 $\mathbf{s}(x)$ 的承诺和参与内积论证, 而这均需要 $O(|C_M|)$ 级别的群上运算. 而对于通信复杂度, 在满足 $mn = |C_M|$ 的条件下, 将 m 设置为 2、 n 设置为 $|C_M|/2$ 时通信量可达到最低, 为 $(6 \log |C_M| + 13)$ 个元素 (包括群元素和域元素).

其次分析安全性. 基于 DLOG 假设, BCCGP16 是具有完美完备性、完美特殊诚实验证者零知识性和统计意义的证据扩展可仿真性的零知识论证. 对于完美完备性, 若算术电路所有的门约束都是可满足的, 则多项式 $p(Y)$ 是零多项式, 进而可得洛朗多项式 $t(X)$ 没有常数项, 故最终验证者会接受证明. 对于完美特殊诚实验证者零知识性, 可构造一个模拟器 \mathcal{S} , 其输入公开陈述和验证者的挑战, 随机生成证明中的部分群或域元素, 并利用验证方程计算证明中的其余元素, 输出模拟证明, 使得该证明与诚实证明者生成的证明具有完美不可区分的分布. 该属性由承诺的隐藏性和盲化向量 \mathbf{d} 保障. 对于统计意义的证据扩展可仿真性, 可构造一个提取器 \mathcal{X} , 其根据数量多项式于安全参数的可接受副本提取出有效的证据. 该属性由承诺的绑定性和内积论证的证据扩展可仿真性保障.

BCCGP16 的平凡改进. 事实上, BCCGP16 中将证据向量 \mathbf{a}, \mathbf{b} 和 \mathbf{c} 排布为矩阵是非必要的, 后续研究^[31, 53, 54] 等也均是直接对 $\mathbf{a} \odot \mathbf{b} = \mathbf{c}$ 进行约束转化的. 在这种设置下, 等式(41)–(43)可相应消除与 m 相关的子式. 此时, 等式(42)可改记为

$$p'_M(Y) = \sum_{i=1}^{|C_M|} (\mathbf{a}_i \mathbf{b}_i - \mathbf{c}_i) Y^i, \quad (44)$$

等式(41)可改记为

$$\mathbf{a} \cdot \mathbf{w}_{q,a} + \mathbf{b} \cdot \mathbf{w}_{q,b} + \mathbf{c} \cdot \mathbf{w}_{q,c} = K_q, q \in [Q], \quad (45)$$

等式(43)可改记为

$$p'_L(Y) = \sum_{q=1}^Q (\mathbf{a} \cdot \mathbf{w}_{q,a} + \mathbf{b} \cdot \mathbf{w}_{q,b} + \mathbf{c} \cdot \mathbf{w}_{q,c}) Y^{|C_M|+q} - \sum_{q=1}^Q K_q Y^{|C_M|+q}. \quad (46)$$

对于 $t(X)$, 有

$$\mathbf{r}(X) = \mathbf{a}X + \mathbf{b}X^{-1} + \mathbf{c}X^2 + \mathbf{d}X^3, \quad (47)$$

$$s(X) = \sum_{q=1}^Q \mathbf{w}_{q,a} y^{|C_M|+q} X^{-1} + \sum_{q=1}^Q \mathbf{w}_{q,b} y^{|C_M|+q} X + \left((y, y^2, \dots, y^{|C_M|}) + \sum_{q=1}^Q \mathbf{w}_{q,c} y^{|C_M|+q} \right) X^{-2}, \quad (48)$$

$$t(X) = r(X) \cdot \left(r(X) \odot (y, y^2, \dots, y^{|C_M|}) + 2s(X) \right) - 2 \sum_{q=1}^Q K_q Y^{|C_M|+q}. \quad (49)$$

容易验证, $t(X)$ 的常数项为 $2p(y) = 2(p'_M(Y) + p'_L(Y))$. 此外, $t(X)$ 的度为 6, 故对其承诺不再需要使用多项式承诺而只需分别对每一项系数进行承诺, 这有助于降低证明者的计算开销.

7.3.2 Bulletproofs/HKR19

主要思路. Bulletproofs 的主要思路如图 12 所示. 与 BCCGP16 类似, Bulletproofs 的主要思路也是先将 $n = |C_M|$ 个乘法门约束和 Q 个线性约束归约为一个多项式约束, 即 $p(Y, Z)$ 是否为零多项式; 然后将该多项式约束转换为内积形式陈述, 即 $t(X) = \sum_{i=1}^6 t_i X^i = \mathbf{L}(X) \cdot \mathbf{R}(X)$ 中 $t_2 = p(y, z)$ 是否为 0; 随后就可构造 $\text{Com}(t(x))$ 和缺失 t_2 项的 $\text{Com}(t'(x))$ 并验证上述两个承诺相等从而说明 $p(Y, Z)$ 是零多项式, 而内积论证可保障承诺是正确构造的. 与 BCCGP16 不同的是, Bulletproofs 中向量多项式 $t(X)$ 的度为常数, 这与对 BCCGP16 的平凡改进思路一致.

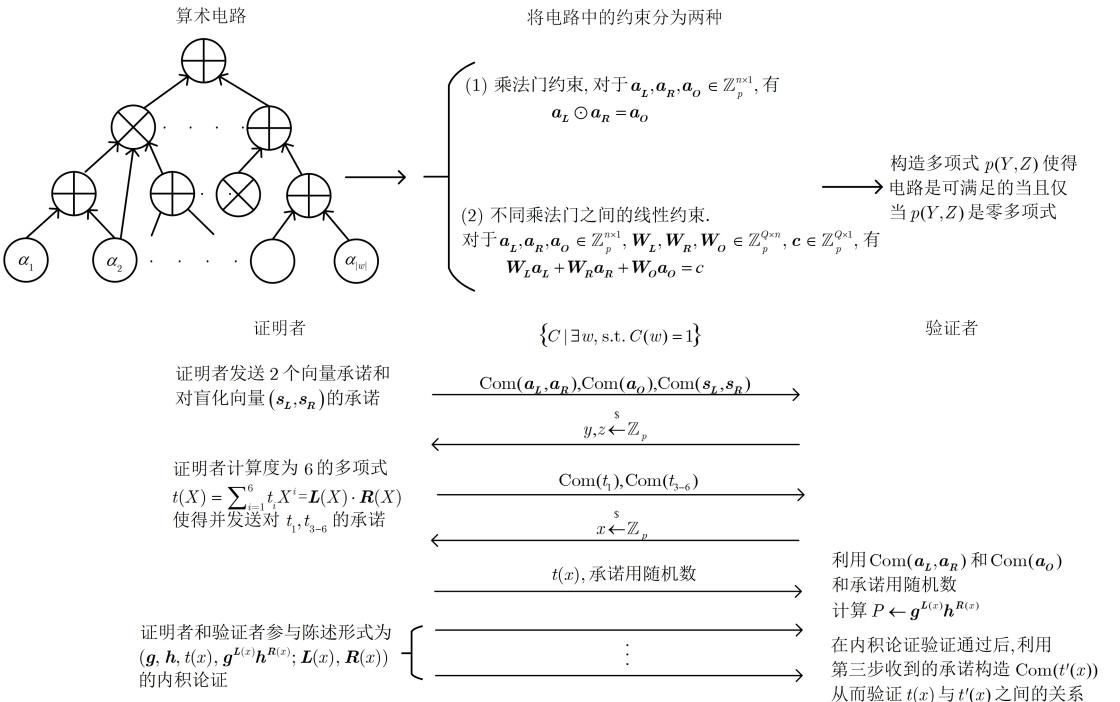


图 12 Bulletproofs 的主要思路与协议流程 [31]

Figure 12 Main idea and process of Bulletproofs

具体而言, $p(Y, Z)$ 及 $t(X)$ 的构造方法如下. 对于乘法门约束, 记电路中所有乘法门的左输入、右输入和输出分别为 \mathbf{a}_L 、 \mathbf{a}_R 和 \mathbf{a}_O , 其中 $\mathbf{a}_L, \mathbf{a}_R, \mathbf{a}_O \in \mathbb{Z}_p^{n \times 1}$, 记线性约束矩阵 $\mathbf{W}_L, \mathbf{W}_R, \mathbf{W}_O \in \mathbb{Z}_p^{Q \times n}$, 则电路中的约束可写为

$$\mathbf{a}_L \odot \mathbf{a}_R = \mathbf{a}_O, \quad \mathbf{W}_L \mathbf{a}_L + \mathbf{W}_R \mathbf{a}_R + \mathbf{W}_O \mathbf{a}_O = \mathbf{c}. \quad (50)$$

其中 $\mathbf{c} \in \mathbb{Z}_p^{Q \times 1}$. 对于随机挑战 $y, z \xleftarrow{\$} \mathbb{Z}_p$, 证明者 \mathcal{P} 和验证者 \mathcal{V} 可以构造

$$\mathbf{y}^n \leftarrow (1, y, y^2, \dots, y^{n-1}) \in \mathbb{Z}_p^{n \times 1}, \quad \mathbf{z}^Q \leftarrow (z, z^2, \dots, z^Q) \in \mathbb{Z}_p^{Q \times 1}, \quad k(y, z) = (\mathbf{y}^{-n} \odot (\mathbf{z}^Q \mathbf{W}_R)) \cdot (\mathbf{z}^Q \mathbf{W}_L). \quad (51)$$

在此基础上, 电路是可满足的必要条件为

$$p(y, z) = \mathbf{a}_L \cdot (\mathbf{a}_R \odot \mathbf{y}^n) - \mathbf{a}_O \cdot \mathbf{y}^n + \mathbf{z}^Q \cdot (\mathbf{W}_L \mathbf{a}_L + \mathbf{W}_R \mathbf{a}_R + \mathbf{W}_O \mathbf{a}_O) - \mathbf{z}^Q \cdot \mathbf{c} = 0, \quad (52)$$

注意到拥有证据 \mathbf{a}_L 、 \mathbf{a}_R 和 \mathbf{a}_O 的 \mathcal{P} 可以构造

$$\mathbf{L}(X) = \mathbf{a}_L X + \mathbf{a}_O X^2 + (\mathbf{y}^{-n} \odot (\mathbf{z}^Q \mathbf{W}_R)) X, \quad \mathbf{R}(X) = (\mathbf{y}^n \odot \mathbf{a}_R) X - \mathbf{y}^n + \mathbf{z}^Q (\mathbf{W}_L X + \mathbf{W}_O), \quad (53)$$

则有 $t(X) = \sum_{i=1}^6 t_i X^i \leftarrow \mathbf{L}(X) \cdot \mathbf{R}(X)$ 的 X^2 项系数 (即 t_2) 为

$$\mathbf{a}_L \cdot (\mathbf{a}_R \odot \mathbf{y}^n) - \mathbf{a}_O \cdot \mathbf{y}^n + \mathbf{z}^Q \cdot (\mathbf{W}_L \mathbf{a}_L + \mathbf{W}_R \mathbf{a}_R + \mathbf{W}_O \mathbf{a}_O) + k(y, z) = p(y, z) + \mathbf{z}^Q \cdot \mathbf{c} + k(y, z), \quad (54)$$

注意到等式(54)中 $\mathbf{z}^Q \cdot \mathbf{c}$ 和 $k(y, z)$ 仅与电路结构有关, 因此可由 \mathcal{V} 自行算出. 也就是说, \mathcal{V} 可通过验证 $t(x)$ 和缺失 x^2 项的 $t'(x)$ 之间的关系, 即 $t(x) \stackrel{?}{=} t'(x) + \mathbf{z}^Q \cdot \mathbf{c} + k(y, z)$ 从而验证电路是否可满足. 除验证该项外, \mathcal{V} 还需验证 $t(x)$ 的结构正确性和 $\mathbf{L}(x)$ 、 $\mathbf{R}(x)$ 的正确性, 即 $t(x)$ 满足内积关系 $t(x) = \mathbf{L}(x) \cdot \mathbf{R}(x)$ 且 $\mathbf{L}(x)$ 、 $\mathbf{R}(x)$ 是形如等式(53)的形式. 若采用平凡方法验证上述约束, \mathcal{P} 需将向量 $\mathbf{L}(x)$ 、 $\mathbf{R}(x)$ 直接发送给 \mathcal{V} , 这会导致线性级别的通信复杂度. 注意到 \mathcal{V} 需验证的两个约束恰巧为内积论证所证明的陈述, 并且结合第一步收到的 $\text{Com}(\mathbf{a}_L, \mathbf{a}_R)$ 和 $\text{Com}(\mathbf{a}_O)$, 依据等式(53) \mathcal{V} 可直接自行计算 $\mathbf{g}^{\mathbf{L}(x)} \mathbf{h}^{\mathbf{R}(x)}$, 因此可调用内积论证在保障完备性的同时实现对数级别的通信复杂度.

此外, 上述过程并不是零知识的, 这是因为在调用内积论证时 \mathcal{V} 可能会获得部分秘密信息 (\mathcal{V} 起码会在内积论证的最后一轮获得秘密的组合). 为实现零知识性, 可引入随机向量 \mathbf{s}_L 、 \mathbf{s}_R 并在 $\mathbf{L}(X)$ 、 $\mathbf{R}(X)$ 分别增加 $\mathbf{s}_L X^3$ 项和 $(\mathbf{y}^n \odot \mathbf{s}_R) X^3$ 项, 考虑到引入随机向量不会改变 t_2 , 因此其可在不影响完备性的同时保障零知识性.

协议流程 基于以上主要思路, Bulletproofs 协议流程 (见图 12 所示) 简要描述如下, 其中 $\text{Com}(\mathbf{a}, \mathbf{b})$ 表示形如 $\mathbf{g}^{\mathbf{a}} \mathbf{h}^{\mathbf{b}} h^r$ 的承诺, 其中 $r \xleftarrow{\$} \mathbb{F}$.

- (1) 证明者 \mathcal{P} 构造对 $(\mathbf{a}_L, \mathbf{a}_R)$ 、 \mathbf{a}_O 和盲化向量 $(\mathbf{s}_L, \mathbf{s}_R)$ 的承诺.
- (2) 在验证者 \mathcal{V} 发送随机挑战 y, z 后, \mathcal{P} 构造多项式 $t(X)$ 并将对除 t_2 外的其他项 $t_1, t_3, t_4, \dots, t_6$ 的承诺发给 \mathcal{V} .
- (3) \mathcal{V} 发送随机挑战 x .
- (4) \mathcal{P} 将 $t(x)$ 和承诺用随机数 (帮助 \mathcal{V} 构造 $P \leftarrow \mathbf{g}^{\mathbf{L}(x)} \mathbf{h}^{\mathbf{R}(x)}$ 和对 $t(x)$ 的承诺) 发给 \mathcal{V} .
- (5) \mathcal{V} 构造 $P \leftarrow \mathbf{g}^{\mathbf{L}(x)} \mathbf{h}^{\mathbf{R}(x)}$, 最后和 \mathcal{P} 参与陈述形式如下的内积论证

$$\left\{ (\mathbf{g}, \mathbf{h}, t(x), P; \mathbf{L}(x), \mathbf{R}(x)) : P = \mathbf{g}^{\mathbf{L}(x)} \mathbf{h}^{\mathbf{R}(x)} \wedge t(x) = \mathbf{L}(x) \cdot \mathbf{R}(x) \right\}. \quad (55)$$

- (6) \mathcal{V} 进行验证. \mathcal{V} 通过内积论证验证 $t(x) \stackrel{?}{=} \mathbf{L}(x) \cdot \mathbf{R}(x)$; 随后, \mathcal{V} 构造对缺失 x^2 项的多项式 $t'(x)$ 的承诺 $\text{Com}(t'(x))$ 和对多项式 $t(x)$ 的承诺 $\text{Com}(t(x))$, \mathcal{V} 通过验证 $\text{Com}(t(x)) \stackrel{?}{=} \text{Com}(t'(x))$ 从而验证 $t(x) \stackrel{?}{=} t'(x)$. \mathcal{V} 选择接受当且仅当内积论证和上述检查均通过.

讨论总结. 首先分析复杂度. \mathcal{P} 的计算开销包括构造多项式 $p(Y, Z)$ 和 $t(X)$, 具体为 $O(|C|)$ 级别的域上运算; 承诺向量 \mathbf{a}_L 、 \mathbf{a}_R 等, 具体为 $O(|C_M|)$ 级别的群上运算. \mathcal{V} 的计算开销包括利用承诺构造 P 和构造承诺 $\text{Com}(t'(x))$ 、 $\text{Com}(t(x))$, 具体为 $O(|C_M|)$ 级别的群上运算. 相比于 BCCGP16, Bulletproofs 的总通信量可降低到 $(2 \log_2 |C_M| + 13)$ 个元素⁴.

⁴其中, 调用内积论证需传输的 $(2 \log |C_M| + 5)$ 个元素 (确定陈述需 3 个, 内积论证过程需 $(2 \log |C_M| + 2)$ 个), 协议流

然后分析安全性. 与 BCCGP16 类似, Bulletproofs 具有完美完备性、完美特殊诚实验证者零知识性和计算意义的证据扩展可仿真性. 对于完美完备性, 若算术电路所有的门约束都是可满足的, 则多项式 $p(Y, Z)$ 是零多项式, 进而可得多项式 $t(X)$ 的二次项系数 t_2 为 0, 故最终验证者会接受证明. 对于完美特殊诚实验证者零知识性, 可构造一个模拟器 \mathcal{S} , 其输入公开陈述和验证者的挑战, 随机生成证明中的部分群或域元素, 并利用验证方程计算证明中的其余元素, 输出模拟证明, 使得该证明与诚实证明者生成的证明具有完美不可区分的分布. 该属性由承诺的隐藏性和盲化向量 $\mathbf{s}_L, \mathbf{s}_R$ 保障. 对于计算意义的证据扩展可仿真性, 可构造一个提取器 \mathcal{X} , 其根据数量多项式于安全参数的可接受副本, 以不可忽略的概率提取出有效的证据. 该属性由承诺的绑定性和内积论证的证据扩展可仿真性保障.

HKR19 Hoffmann, Kloß 和 Rupp^[53] 指出 Bulletproofs 中乘法门约束和线性约束为 R1CS 可满足问题中的形式, 而用二次等式 (quadratic equations) 表达约束可以降低证明者的计算开销. 具体而言, 二次等式集合论证 (quadratic equation set argument) 是证明: 给定矩阵 $\Gamma \in \mathbb{F}^{n \times n}$ 及对 \mathbf{w} 的承诺, 证明拥有 \mathbf{w} 使得对于任意的 i , 有 $\mathbf{w} \cdot \Gamma \mathbf{w} = 0$. 此外, 为降低验证时间, 可将 i 个约束转化为 1 个约束, 即证明 $\mathbf{w} \cdot \Gamma \mathbf{w} = 0$, 其中 $\Gamma = \sum_i r_i \Gamma_i$, $r_i \xleftarrow{\$} \mathbb{F}$.

HKR19 指出 R1CS 是二次等式约束的一种特例, 并且相比于用二次等式表达电路, 利用 R1CS 表达电路需要更多的中间变量和等式, 因此在一定程度上会增加证明者的计算开销^[53].

基于以上发现, HKR19 首先借鉴 BCCGP16 和 Bulletproofs 中的内积论证构造了零知识的内积论证, 然后利用零知识内积论证构造了零知识的二次等式集合论证, 并沿用 Bulletproofs 的主要思路构造了针对 C-SAT 问题的简洁 NIZK AoK. 实验仿真表明, HKR19 中的零知识论证的实际通信量和实际验证计算开销与 Bulletproofs 基本相同, 实际证明计算开销比 Bulletproofs 少约 1/4.

7.3.3 DRZ20

主要思路. DRZ20^[54] 实现了 CRS 可更新的证明复杂度为线性、通信和验证复杂度均为对数级别的简洁 NIZK AoK, 其验证复杂度在渐近级别上的突破主要依赖于结构化承诺密钥, 而 CRS 用于保障结构化承诺密钥的私密性.

对于验证复杂度, DRZ20 指出 BCCGP16 中验证者 \mathcal{V} 的复杂度成线性的原因有三:

- (1) 在内积论证中 \mathcal{V} 需要每轮更新密钥, 即计算 $[\mathbf{r}'] \leftarrow [c^{-1} \mathbf{r}_{\frac{1}{2}} + c^{-2} \mathbf{r}_{\frac{2}{2}}]$ 和 $[\mathbf{s}'] \leftarrow [c \mathbf{s}_{\frac{1}{2}} + c^2 \mathbf{s}_{\frac{2}{2}}]$, 这需要线性级别的群幂运算.
- (2) 在最后的验证阶段, \mathcal{V} 需利用图 11 中第五步收到的 $\mathbf{r}(x)$ 构造 $t(x)$, 而计算 $\mathbf{r}'(x) \leftarrow \mathbf{r}(x) \odot \mathbf{y}^{|C_M|}$ 需要线性级别的域乘运算.
- (3) 在最后的验证阶段, \mathcal{V} 构造 $\mathbf{s}(X)$ 并计算 $\mathbf{s}(x)$ 需要线性级别运算^[30].

对于 (1), 可调用对数级别验证复杂度的内积论证将群幂运算量降低到对数级别. 对于 (2), 记 $n = |C_M|$, DRZ20 指出给定以服从 \mathcal{ML}_n 分布的 $[\mathbf{r}^n]$ 为承诺密钥的对向量 \mathbf{x} 的承诺, 可在对数时间内计算出以 $[\mathbf{r}^n \odot \mathbf{y}^{-n}]$ 为承诺密钥的对向量 $\mathbf{x} \odot \mathbf{y}^n$ 的承诺^[54, §4.1], 若令更新前后的密钥恰巧是协议 6 中内积论证陈述的 \mathbf{g} 和 \mathbf{h} , 则 $\mathbf{r}(x) \odot \mathbf{y}^{|C_M|}$ 也可在对数时间内计算得出. 对于 (3), DRZ20 将构造 $\mathbf{s}(X)$ 和计算 $\mathbf{s}(x)$ 的任务委托给证明者并实现了计算开销的转移.

对于可更新性, DRZ20 是利用可更新的 Pedersen 承诺实现的, 其能够保障参与方可更新承诺密钥 \mathbf{g} ; 且只要接收者是诚实的, 则更新后承诺的绑定性就是安全 (基于 \mathcal{ML}_n -Find-Rep 假设) 的. 由于整个简洁 NIZK AoK 的可靠性均是由 Pedersen 承诺的绑定性保障的, 因此仅需可更新承诺就足以保障整个协议是可更新的. 针对呈 \mathcal{ML}_n 分布的承诺密钥 $[\mathbf{r}]_1 = [1, x_1, x_2, x_2 x_1, \dots, x_v \dots x_1]$, 即向量承诺为 $[\mathbf{x}]_2 = [\mathbf{r} \cdot \mathbf{r}]_2$, 该可更新承诺需要引入验证密钥用于说明 $[\mathbf{r}]$ 的正确性. 具体的, 验证密钥为 $[\mathbf{x}]_2 = ([r_{2^0}]_2, [r_{2^1}]_2, \dots, [r_{2^v}]_2)$, 验证正确性时需计算对于 $1 \leq i \leq v, 1 \leq j \leq 2^{i-1}$, $e([r_{2^{i-1}+j}]_1, [1]_2) = e([r_j]_1, [x_i]_2)$. 为更新承诺, 更新发起方只需随机挑选 $\mathbf{y} \xleftarrow{\$} \mathbb{Z}_q^v$ 并计算 $[\mathbf{r}'] \leftarrow [\bar{\mathbf{y}} \odot \mathbf{r}]_1, [\mathbf{x}']_2 \leftarrow [\mathbf{y} \odot \mathbf{x}]_2$ (其中 $\bar{\mathbf{y}}$ 表示服从 $\mathcal{ML}_{n=2^v}$ 分布的向量) 即可得到更新后的密钥 $([\mathbf{r}']_1, [\mathbf{x}']_2)$, 此外参与方还需生成 NIZK 证明 π 用于在不泄露 \mathbf{y} 的情况下证明 \mathbf{y} 满足关系对于 $1 \leq i \leq v$, 有 $[x'_i]_2 = [y_i x_i]_2$ (π 可调用 Bulletproofs^[31] 生成, 实际通信量为 $O(\log \log |C_M|)$).

程的第 (1) 步需传输 3 个承诺值, 第 (2) 步需传输 5 个承诺值.

协议流程. DRZ20 的协议流程与 BCCGP16 的协议流程是类似的, 简要描述如下.

- (1) 将 C-SAT 问题归约为 $p(Y) = p_M(Y) + p_L(Y)$ 为是否为零多项式的问题. 其形式如第 7.3.1 小节的等式(44) 和等式(45).
- (2) 证明者 \mathcal{P} 将对证据 a, b, c 和盲化向量 d 的承诺发送给验证者 \mathcal{V} . 与 BCCGP16 不用的是, 此时的承诺为可更新的 Pedersen 向量承诺, 为实现可更新性, 需在 CRS 中相应引入验证密钥.
- (3) 在 \mathcal{V} 发送随机挑战 y 后, \mathcal{P} 构造多项式 $t(X)$ 使得 $t(X)$ 的常数项为 $p(y)$, $t(X)$ 形式如等式(49). \mathcal{P} 构造缺失常数项的多项式 $t'(X)$ 并发送对其的多项式承诺.
- (4) 在 \mathcal{V} 发送随机挑战 x 后, \mathcal{P} 通过多项式承诺揭示 $t'(x)$ 的值, 然后 \mathcal{P} 和 \mathcal{V} 调用对 $t(x)$ 的内积论证. 与 BCCGP16 不同的是, 此时的内积论证为基于结构化承诺密钥的对数级别验证复杂度的内积论证. 为保障可靠性, 需在 CRS 中相应引入验证密钥. 此外, 为实现对数级别的验证复杂度, 需将计算 $s(x)$ 的任务委托给证明者, 故证明者在该步还需给出对 $s(x)$ 的承诺并附带一个零知识证明, 其用于证明 $s(x)$ 是正确构造的.

讨论总结. 基于 A-DLOG 和 q -A-DLOG 假设, DRZ20 具有完美完备性、完美特殊诚实验证者零知识性和统计意义的证据扩展可仿真性. 相比于其他可更新的零知识证明方案^[23, 40, 42, 120], DRZ20 不再需要系列知识假设和代数群模型 (algebraic group model), 但在 CRS 长度、通信和验证复杂度上有所牺牲 (特别的, 通信复杂度从常数群元素级别提升至对数级别). 具体的, DRZ20 中可更新的简洁 NIZK AoK 的证明开销为 $(22+10M)n'E_1$, 通信量为 $(12\log n'E_1 + 8\log n'P)$, 验证开销为 $(12\log n'\mathbb{G}_1 + 4\log n'\mathbb{F})$, 其中 m 指电路中的导线数目, M 是描述预处理电路输入输出导线数目上限的参数 (预处理电路由 \mathcal{P} 构造用于委托计算 $s(X)$), n' 是预处理电路的规模, 其满足 $n' \leq n + (2m/M - 1)$. 在 Sonic^[40] 中, $n' = 3|C_M|$, $M = 3$. 此外, E 、 P 分别指群幂运算和配对运算, E_1 指非对称群中群 \mathbb{G}_1 上的群幂运算, \mathbb{G} 、 \mathbb{F} 分别指对应群和域中的元素. 对于实际性能, 由于引入了双线性群, 相比于其他不需配对的同类零知识证明, DRZ20 会带来一定的性能损失.

7.4 总结

基于 IPA 的零知识证明具有以下优点:

- (1) 底层难题假设更为通用. 均基于 DLOG 假设及其变种, 属于标准假设.
- (2) 应用场景多元. 除实现对 C-SAT 问题的证明外, 基于 IPA 还可构造低实际通信量的范围证明^[31] (range proof)、洗牌正确性证明 (proof of correctness of a shuffle)^[53, 135]、向量置换证明^[54] (vector permutation proof) 等, 在区块链密码货币场景下应用广泛.

然而, 该类零知识证明的验证复杂度为线性, 验证时由于引入了大量的群幂运算也会导致实际验证开销较大. 值得注意的是, DRZ20 虽然实现了验证复杂度在渐近级别上的突破, 但需要承诺密钥结构化并引入 CRS 用于保障密钥分布的正确性.

8 基于 MPC-in-the-Head 的零知识证明

本章介绍基于 MPC-in-the-Head 的零知识证明. 该类协议的构造思路是证明者在脑海中模拟运行一个针对零知识函数的安全多方计算协议, 然后将协议运行过程中的视图发送给验证者, 验证者验证视图正确性, 而协议的零知识性由 MPC 协议的隐私性保障. 本章第 8.1 节介绍相关定义及概念, 第 8.2 节介绍该类协议的背景及主要思路, 第 8.3 节介绍典型协议.

8.1 定义及概念

定义 33 (安全多方计算^[136]) 安全多方计算 (secure multiparty computation, MPC) 可使独立参与方在不信任彼此及第三方的情况下, 基于各自的秘密输入共同计算某个目标联合函数, 且计算期间不泄露除计算结果外的其他额外信息. 记安全多方计算协议为 Π_f , 目标联合函数为 $f(x, w_1, r_1, \dots, w_n, r_n)$, 其中公共输入为 x , 参与方 P_1, P_2, \dots, P_n 的秘密输入分别为 w_1, w_2, \dots, w_n 、随机输入分别为 r_1, r_2, \dots, r_n . 参与方 P_i 在协议运行第 $j+1$ 轮时发送的消息可由消息确定函数 $\Pi(i, x, w_i, r_i, (\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_j))$ 决定, $(\mathbf{m}_1, \mathbf{m}_2, \dots, \mathbf{m}_j)$ 分别代表参与方 P_i 前 j 轮收到的消息向量. 若消息向量包含 k 个不同参与方的消息 (包括 P_i), 就称消息确定函数 Π 为 k 元. 记参与方 P_i 的视图为 V_i , 其包含 w_i, r_i 及 P_i

在协议运行过程中收到的所有消息. 参与方 P_i 的本地输出可由其视图 V_i 确定, 记为 $f_i(x, V_i)$. 在本章中, 只考虑各参与方的本地输出与目标联合函数相等的情况, 即

$$\forall i \in [n], \quad f_i(x, V_i) = f(x, w_1, r_1, \dots, w_n, r_n). \quad (56)$$

安全多方计算的敌手模型分为半诚实敌手模型和恶意敌手模型. 半诚实敌手会诚实地运行协议, 但会通过分析其他参与方的消息试图获得与诚实参与方秘密输入相关的信息; 而恶意敌手可以在协议运行过程中实现任意高效攻击, 如控制参与方发送消息、拒绝其他参与方的消息、篡改消息等. 本章中出现的 MPC 协议均处于半诚实敌手模型下.

一个安全多方计算协议 Π_f 的完美正确性 (perfect correctness) 是指随机数 r_1, r_2, \dots, r_n 的选取不会影响目标函数计算结果的正确性, 即

$$\forall i \in [n], \forall x, w_1, w_2, \dots, w_n, \forall r_1, r_2, \dots, r_n, \Pr[f_i(x, V_i) \neq f(x, w_1, r_1, \dots, w_n, r_n)] = 0. \quad (57)$$

一个安全多方计算协议 Π_f 的 t -隐私性 (t -privacy) 是指 t 个半诚实敌手无法获得与诚实参与方秘密输入相关的其他信息. 具体的, 称安全多方计算协议 Π_f 具有 t -隐私性. 如果对于任意输入 $(x, w_1, r_1, \dots, w_n, r_n)$ 及任意腐化参与方集合 T (满足 $|T| \leq t$), 都存在一个 PPT 模拟器 \mathcal{S} , 使得腐化参与方的联合视图分布与模拟器生成的分布相同, 即

$$\{x, V_i\}_{i \in T} = \{\mathcal{S}(T, x, (w_i, f_i(x, V_i)))\}_{i \in T}. \quad (58)$$

一个安全多方计算协议的 r -鲁棒性 (r -robustness) 是指如果 $\mathcal{R}(x, w) \neq 1$, 那么即使 r 个恶意敌手也无法使诚实参与方输出接受.

一个安全多方计算协议的视图一致性 [98] (consistency of view) 是指对于任意的视图对 (V_i, V_j) , x 和 V_i 所确定的由参与方 P_i 发给 P_j 的消息与视图 V_j 所表明的此消息是一致的. 反之亦然.

定义 34 (交织里德-所罗门码 [35]) 对正整数 n 和 k , 域 \mathbb{F} 及向量 $\xi = (\xi_1, \xi_2, \dots, \xi_n) \in \mathbb{F}^n$, RS 码 $L = \text{RS}_{\mathbb{F}, n, k, \xi}$ 是 $[n, k, d]$ 线性码, 其形式为 $(p(\xi_1), p(\xi_2), \dots, p(\xi_n))$, 其中 $p(\cdot)$ 是度小于 k 的多项式, 记为码多项式. 由于任意两个度小于 k 的域上多项式最多有 $k - 1$ 个交点, 因此码距 d 最小为 $n - k + 1$.

若 $L \subset \mathbb{F}^n$ 是 $[n, k, d]$ 线性码, 则交织码 (interleaved code) L^m 是定义在 $\mathbb{F}^{m \times n}$ 上的 $[n, mk, d]$ 线性码. 具体的, 该码可排列成 $m \times n$ 的码矩阵 \mathbf{U} , 该码矩阵的每一行 \mathbf{u}_i 满足 $\mathbf{u}_i \in L$, 即第 i 行的表示形式为 $(p_i(\xi_1), p_i(\xi_2), \dots, p_i(\xi_n))$, 其中 $p_i(\cdot)$ 是度小于 k 的多项式. 基于交织码, 交织里德-所罗门码的定义自然可得, 简记为 IRS 码.

在本章中, 由于 Ligero 系列协议 [35, 36, 48] 中码矩阵 \mathbf{U} 可被视为信息论安全证明中的谕示, 因此也被称为谕示矩阵.

定义 35 (利用 IRS 码加密消息 [35]) 给定 $\eta = (\eta_1, \eta_2, \dots, \eta_\ell)$ ($\ell \leq k$) 及 RS 码 $L = \text{RS}_{\mathbb{F}, n, k, \xi}$, 对长为 ℓ 的消息向量 $\mathbf{x} = (x_1, x_2, \dots, x_\ell)$ 的加密即为码 $(p_{\mathbf{u}}(\xi_1), p_{\mathbf{u}}(\xi_2), \dots, p_{\mathbf{u}}(\xi_n))$, 其中 $p_{\mathbf{u}}(\cdot)$ 是码 \mathbf{u} 对应的码多项式, 且满足对于任意的 $i \in [\ell]$, $p_{\mathbf{u}}(\eta_i) = x_i$. 给定码矩阵 \mathbf{U} 的每一行 $\mathbf{u}_1, \mathbf{u}_2, \dots, \mathbf{u}_m$, 对消息向量 $\mathbf{x} = (x_{11}, x_{12}, \dots, x_{1\ell}, \dots, x_{m1}, x_{m2}, \dots, x_{m\ell})$ 的加密即为码 $(p_{\mathbf{u}_1}(\xi_1), p_{\mathbf{u}_1}(\xi_2), \dots, p_{\mathbf{u}_1}(\xi_n), \dots, p_{\mathbf{u}_m}(\xi_1), p_{\mathbf{u}_m}(\xi_2), \dots, p_{\mathbf{u}_m}(\xi_n))$.

8.2 背景及主要思路

8.2.1 背景

零知识证明与 MPC 在多个层面存在紧密联系. 在协议内涵层面, 零知识证明可以视为恶意敌手模型下的一种两方特殊 MPC 协议 [136]. 在该协议中, 证明者和验证者共同拥有问题的陈述 x , 证明者还拥有证据 w , 证明者试图在不泄露证据的同时证明 $\mathcal{R}(x, w) = 1$. 该协议中目标联合函数 $f(x, w, r_1, r_2) = \mathcal{R}(x, w)$, 验证者是敌手, 其目标是获取与证据 w 相关的其他隐私信息.

在协议构造层面, 零知识证明可用于构造 MPC 协议, MPC 也可用于构造零知识证明. 针对前者,

Goldreich、Micali 和 Wigderson^[137] 利用零知识证明给出了一种在不更改目标联合函数的同时将半诚实敌手模型下的 MPC 协议转换为恶意敌手模型下的 MPC 协议的通用方法。针对后者，又分为基于混淆电路 (garbled circuits) 的零知识证明和基于 MPC-in-the-Head 的零知识证明。

Jawurek、Kerschbaum 和 Orlandi^[138] 于 2013 年提出了一种基于混淆电路的零知识证明，该类证明不需要底层的混淆电路具有隐私性而只需其具有可认证性 (authenticity) 和可验证性 (verifiability)，因此 Frederiksen、Nielsen 和 Orlandi^[139] 将该类协议称为免隐私的混淆方案 (privacy-free garbling scheme)。后续工作如文献 [140, 141] 从通信复杂度、底层难题假设等方面优化了上述零知识证明。由于该类证明的通信复杂度均与陈述规模和证据大小成线性，故不是简洁的，本章不再详述。

Ishai 等人^[98] 于 2007 年提出了一种基于 MPC-in-the-Head 的零知识证明，后续工作具体实现了该协议^[33]，并从通信复杂度、可靠性等角度进行了改进^[34–36, 47]。该类零知识证明只需对称密钥操作、不需要可信初始化、通信复杂度可达到亚线性级别、实际证明速率较快，具有较高的理论价值和较为广泛的应用前景。

8.2.2 主要思路

Ishai 等人提出的零知识证明主要思路如协议 8 所示。证明者首先在脑海中模拟一个 MPC 协议的运行，得到每个参与方的视图，证明者随后对每个视图作承诺，并将这些承诺发给验证者；然后验证者挑选 2 个随机挑战；其次证明者打开这 2 个承诺；最后验证者验证一致性和正确性。该协议具有完美完备性，其源于 Π_f 的完美正确性；具有可靠性，其源于 Π_f 的完美正确性和敌手模型的半诚实；具有零知识性，其源于 2-隐私性。对于零知识性，由于验证者 \mathcal{V} 拥有 2 个参与方的视图，其本质上相当于一个腐化了 2 个参与方并试图获取其他隐私信息的半诚实敌手，这与 2-隐私性的内涵是一致的。事实上，该证明的模拟器就是调用 2-隐私性的模拟器构造的。

协议 8 IKOS07 协议^[98]

公共输入：陈述 x 和 MPC 协议 Π_f ，其中 Π_f 具有完美正确性和 2-隐私性。给定一个多项式时间内可判定的二

元关系 \mathcal{R} 及其对应的 NP 语言 $\mathcal{L}(\mathcal{R})$ ，目标联合函数 f 与 $\mathcal{L}(\mathcal{R})$ 满足条件：对于任意的 x 、任意的

$w = w_1 \oplus w_2 \oplus \dots \oplus w_n$ 和 r_1, r_2, \dots, r_n ，有 $f(x, w_1, r_1, \dots, w_n, r_n) = \mathcal{R}(x, w_1 \oplus w_2 \oplus \dots \oplus w_n)$ 。

证明者输入： w ，满足 $\mathcal{R}(x, w) = 1$ 。

1. \mathcal{P} 将证据 w 随机分为 n 份 w_1, w_2, \dots, w_n ，随后在脑海中模拟以 x 为公共输入、以 w_1, w_2, \dots, w_n 为各参与方隐私输入、以 r_1, r_2, \dots, r_n 为各参与方随机输入的 MPC 协议 Π_f 。协议运行完毕后， \mathcal{P} 会得到 n 个参与方的视图 V_1, V_2, \dots, V_n ，他分别承诺这 n 个视图，即生成随机数 s_1, s_2, \dots, s_n ，然后计算 $\text{Com}(V_1; s_1), \text{Com}(V_2; s_2), \dots, \text{Com}(V_n; s_n)$ ，并将这 n 个承诺发送给 \mathcal{V} 。
 2. \mathcal{V} 随机挑选两个参与方 $i, j \xleftarrow{\$} [n]$ ，并将 i, j 发送给 \mathcal{P} 。
 3. \mathcal{P} 将 V_i, s_i, V_j, s_j 发送给验证者。
 4. \mathcal{V} 验证如下三项并输出比特 b ：(1) 第 3 步收到的消息与第 1 步收到的承诺是一致的。(2) 参与方 P_i 和 P_j 的本地输出 $f_i(x, V_i)$ 与 $f_j(x, V_j)$ 皆为 1。(3) 参与方 P_i 和 P_j 的视图 V_i 和 V_j 是一致的。如果上述三项未均通过，则选择拒绝。
- 输出：**比特 b ， $b = 1$ 代表 \mathcal{V} 接受， $b = 0$ 代表 \mathcal{V} 拒绝。
-

基于 MPC-in-the-Head 的零知识证明运行 MPC 协议的效率与运行普通 MPC 协议的效率存在一定的差别。在 MPC-in-the-Head 环境下，证明者可以免费利用 OT 信道^[98](bblivious transfer channel) 及任意的二元确定函数^[33](特定条件下的 n 元确定函数也可以^[47])，这是因为证明者只需脑海中模拟，而无需真正运行 MPC 协议，这也就省去了部分计算和通信开销。因此，虽然该类零知识证明的证明复杂度为线性或准线性级别，但实际证明开销通常并不大。

对于证明复杂度以外的性能表现，针对实现非交互的方式，该类零知识证明是 Σ 协议，故可通过 ROM 下的 Fiat-Shamir 启发式转换为非交互零知识证明；针对是否抗量子及需要公钥加密，该类零知识证明是基于 PCP、IPCP 或 IOP 构造的，调用合适的 MPC 协议^[137, 142, 143] 即可实现只依赖对称密钥操作且抗量子的零知识证明。事实上对该类零知识证明的优化主要集中在降低通信复杂度，且根据通信复杂度渐近级别的不同，该类零知识证明可分为两类，协议总结如表 9 所示，具体描述如下。

第一种是通信复杂度为 $O(|C|)$ 的零知识证明，包括表 9 中的 ZKBoo^[33]、ZKB++^[34]、KKW18^[47] 和 Limbo^[49]。协议 8 的通信量为 $(n \cdot |\text{Com}| + t(|\mathbb{F}| + |V| + |s|))$ ，其中 $|\text{Com}|$ 指承诺的大小， t 为打开视图数目， $|\mathbb{F}|$ 指域元素的大小， $|V|$ 指视图规模， $|s|$ 指随机数的大小，因此通信复杂度主要由参与方数目和

表 9 基于 MPC-in-the-Head 的(简洁) NIZKAOK 总结

Table 9 (Succinct) non-interactive zero-knowledge arguments of knowledge based on MPC-in-the-Head

对比项	协议						
	ZKBoo [33]	ZKB++ [34]	KKW18 [47]	Ligero [35]	Ligero++ [36]	BooLigero [48]	Limbo [49]
待证明陈述表示形式	算术/布尔	算术/布尔	布尔	算术/布尔	算术/布尔	布尔	算术/布尔
MPC 模型	/	/	预处理模型	Client-Server	Client-Server	Client-Server	Client-Server
MPC 协议	GMW [137]	GMW	文献 [143]	/	/	/	文献 [144–146]
MPC 协议参与方数目	$n = 3$	$n = 3$	n	$O(\sqrt{ C })$	$O(\frac{ \mathcal{C} }{\log \mathcal{C} })$	$O(\frac{\sqrt{ C }}{\sqrt{\log \mathcal{F} }})$	n
消息确定函数	2 元	2 元	n 元	1 元	1 元	1 元	n 元
通信复杂度	$O(C)\mathbb{F}$	$O(C)\mathbb{F}$	$O(C + n\lambda_s)\mathbb{F}$	$O(\sqrt{ C })\mathbb{F}$	$O(\log C)$	$\frac{O(\sqrt{ C })}{\sqrt{\log \mathcal{F} }} \mathbb{F} \sim \frac{O(\sqrt{ C })}{(\log \mathcal{F})^{1/4}} \mathbb{F}$	$O(C)\mathbb{F}$
可靠性误差 ϵ	$\frac{2}{3}$	$\frac{2}{3}$	$\max\left\{\frac{1}{m}, \frac{1}{n}\right\}$	$\frac{\epsilon+6}{ \mathcal{F} } + 5\left(\frac{\epsilon+2k}{n}\right)^t + (1 - \frac{\epsilon}{n})^t$	$\frac{d+2}{ \mathcal{F} } + 2(e+2k)^t + (1 - \frac{\epsilon}{n})^t + 3\epsilon_i$	$\frac{\epsilon+6}{ \mathcal{F} } + 5\left(\frac{\epsilon+2k}{n}\right)^t + (1 - \frac{\epsilon}{n})^t + \frac{1}{2\lambda_t}$	$\frac{1}{n} + (1 - \frac{1}{n})\epsilon_r$
证明复杂度	$O(C)\mathbb{F}_o$	$O(C)\mathbb{F}_o$	$O(C)\mathbb{F}_o$	$O(C \log^2 C)\mathbb{F}_o$	$O(C \log^2 C)\mathbb{F}_o$	$O(C \log^2 C)\mathbb{F}_o$	$O(C)\mathbb{F}_o$
验证复杂度	$O(C)\mathbb{F}_o$	$O(C)\mathbb{F}_o$	$O(C)\mathbb{F}_o$	$O(C)\mathbb{F}_o$	$O(C)\mathbb{F}_o$	$O(C)\mathbb{F}_o$	$O(C)\mathbb{F}_o$

¹ MPC 模型指协议构造 MPC 协议所基于的模型, MPC 协议指 MPC-in-the-Head 调用的具体 MPC 协议 Π_f . 预处理模型 (preprocessing model) 详见第 8.3.2 小节, Client-Server 模型详见第 8.3.5 小节.

² n 表示参与方个数, $|C|$ 表示电路规模; m 表示 KKW 协议中预处理阶段的份数; \mathbb{F} 表示域元素, $|\mathcal{F}|$ 表示域上运算, $|\mathcal{F}|$ 表示 Ligero 等协议中域的大小, t 表示 Ligero 等协议中打开的视图数目; k, ℓ, e, d 均为 RS 码的参数, 其中 k 表示编码多项式的度, ℓ 表示原码消息长度, d 表示该编码的码距, e 满足 $e < d/4$; λ_s 表示伪随机生成器的种子长度, λ_t 表示比特约束检查协议的重复次数; ϵ 表示可靠性误差, ϵ_r 表示 MPC 协议的鲁棒性误差, ϵ_i 表示内积论证的可靠性误差.

³ 证明、通信和验证复杂度均为协议 1 轮开销, 协议实际运行轮数及实际证明、验证计算开销和通信量与可靠性误差和目标可靠性误差有关.

视图规模决定. 考虑视图规模, 对于一个目标联合函数为电路求值且基于加性秘密分享的 MPC 协议, 各参与方分别持有每条电路输入导线的秘密分享份额. 为了保障 MPC 协议的正确性和安全性, 每个参与方视图在每个电路门处都需要存储一个秘密分享份额, 否则要么无法正确得到电路计算结果, 要么最多 $n-1$ 个半诚实敌手就可破坏隐私性, 这就意味着每个参与方的视图规模至少为 $O(|C|)$.

事实上, 第一个可实现的基于 MPC-in-the-Head 的零知识证明 ZKBoo [33] 的通信复杂度为 $O((n-1)^2|C|)$. 这是因为 ZKBoo 的底层 MPC 协议是 GMW 协议 [137], 其需要参与方两两交互, 每个视图规模为 $O((n-1)|C|)$, 且协议共需打开 $n-1$ 个视图. 记目标可靠性误差为 $2^{-\sigma}$, 则 ZKBoo 的通信复杂度约为 $\sigma \cdot |C|(n-1)^2 / (\log_2 n - 1)$, 其中 $n \geq 3$. 由于该函数是递增函数, 因此 $n=3$ 时通信复杂度最低, 此时可靠性误差为 $2/3$, 故协议在实际运行过程中需要重复较多轮数. ZKB++ [34] 虽然将 ZKBoo 的实际通信量降低了约一半, 然而其仍与 n 有关, 可靠性误差也为 $2/3$. 一个改进方向就是设计通信复杂度与 n 无关的 MPC 底层协议, 这样就可以通过增加参与方个数 n 来降低可靠性误差进而降低总通信量. 基于此, KKW18 [47] 设计了一个针对布尔电路、消息确定函数为 n 元、视图打开个数为 $n-1$ 的 MPC 协议. 在 KKW18 中, 对于打开的 $n-1$ 个视图, 每个视图的规模与 n 无关, 此时可靠性误差为 $1/n$. 当然, 为了验证协议的正确性与一致性, 第 n 个参与方的广播消息仍需要由证明者发给验证者, 此消息级别为 $O(|C|)$, 故协议的通信复杂度仍为 $O(|C|)$. 与上述思路不同的是, Limbo [49] 基于 Client-Server 模型构造了一种较为适合 MPC-in-the-Head 的 MPC 协议, 实现了通信复杂度虽为 $O(|C|)$ 、但实际性能良好的 NIZKAOK.

通信复杂度为 $O(|C|)$ 的工作有两个共同特征, 第一是均调用了基于加性秘密分享的 MPC 协议, 每个参与方的视图规模与电路规模成线性关系; 第二是除加性秘密分享带来的约束之外, 参与方的视图之间仅有电路约束关系.

第二种是通信复杂度为亚线性的零知识证明, 包括表 9 中的 Ligero [35]、Ligero++ [36] 和 Boo-

Ligero^[48]. 与通信复杂度为 $O(|C|)$ 的工作不同的是, 在 Ligero^[35] 系列协议中, 虽然参与方的总视图规模是线性于电路规模的, 但每个参与方的视图是亚线性的; 且参与方视图之间除电路约束关系外, 还具有多项式约束 (由 RS 码决定). Ligero^[35] 的主要思路是将电路线值排列为一个 $m \times \ell$ 的谕示矩阵, 故有 $ml = O(|C|)$, 此外矩阵的每一行都是 RS 码. 在证明过程中证明者仅需发送给验证者 $O(m + t\ell)$ 个值, 其中 t 为打开视图的个数且可设置为与安全参数成线性关系, 由基本不等式, 可将 m 和 ℓ 设置为 $O(\sqrt{|C|})$, 从而实现 $O(\sqrt{|C|})$ 级别的通信复杂度. Ligero++^[36] 利用内积论证优化了 Ligero 中的一致性检查, 将通信复杂度进一步降低到了 $O(\log |C|)$ 级别. 针对 Ligero 的布尔电路版本, BooLigero^[48] 优化了布尔电路中的数据存储方式从而压缩了参与方视图规模, 进而降低了通信复杂度.

8.3 典型协议分析

本节介绍基于 MPC-in-the-Head 的零知识证明典型协议, 分析各协议的构造思路、协议流程、复杂度及安全性, 各典型协议优化思路如图 13 所示. 本节第 8.3.1 小节介绍 ZKBoo 和 ZKB++, 第 8.3.2 小节介绍 KKW18, 第 8.3.3 小节介绍 Ligero 和 Ligero++, 第 8.3.4 小节介绍 BooLigero, 第 8.3.5 小节介绍 Limbo.

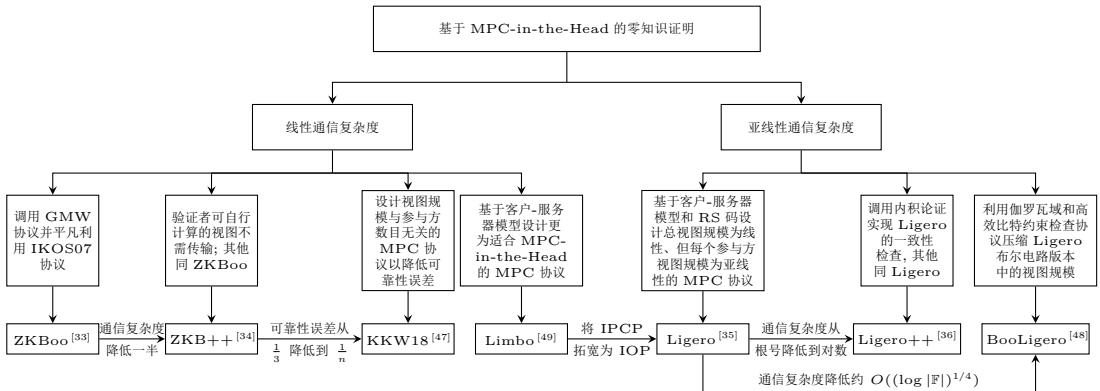


图 13 基于 MPC-in-the-Head 的零知识证明典型协议优化思路

Figure 13 Optimization of typical zero-knowledge proof based on MPC-in-the-Head

8.3.1 ZKBoo/ZKB++

ZKBoo 由 Giacomelli、Madsen 和 Orlandi^[33] 提出, 是第一个可实现的基于 MPC-in-the-Head 的 NIZKAOK. ZKBoo 的底层 MPC 协议是 GMW 协议^[137], 其消息确定函数为 2 元、消息传播方式点对点、视图打开个数为 $n - 1$. 可以证明参与方数目 $n = 3$ 时 ZKBoo 的实际通信量最低.

主要思路. ZKBoo 设计了一个 $(2, 3)$ -函数拆解方案 ($(2, 3)$ -function decomposition), 即对于计算电路的某个目标函数 $f : \mathbb{F}^{|\mathbf{w}|} \rightarrow \mathbb{F}$, 可将其拆解为 3 个计算分支, 并且任意揭示 2 个计算分支不会泄露与秘密输入 \mathbf{w} 相关的其他信息. 此函数拆解方案本质上是 GMW 协议^[137] 在算术电路下的变种, 即一种基于加性秘密分享的 2-隐私性 MPC 协议.

ZKBoo 的拆解方案如图 14 所示, 在该方案中证明者 \mathcal{P} 的步骤主要包括:

- (1) 将证据 \mathbf{w} 利用加性秘密分享 (图 14 中的 Share) 随机分为 $\mathbf{w}_1, \mathbf{w}_2, \mathbf{w}_3$ 共 3 份, 并分发给相应参与方.
- (2) 在脑海中模拟各参与方利用消息确定函数 $\phi_i^{(j)}$ 计算下一个电路门的份额值并写入视图 \mathbf{V}_i 的过程 (图 14 中的虚线框内部分), 直至计算出本地电路输出的份额值.
- (3) 在脑海中模拟各参与方将本地电路输出份额值广播, 并在获取其他方的份额后恢复电路最终输出值的过程 (图 14 中的 Recover).

由于 \mathbf{w}_i 是随机生成的, 且消息确定函数 ϕ_i^j 是二元函数, 故获取其中 2 个计算分支既能检查正确性和一致性, 又不会泄露第 3 个参与方的秘密输入信息. 具体的, 对于输入为第 x 个门、第 y 个门、输出为

第 z 个门的加法门, 任意的 $i \in [3]$, 下式成立

$$\mathbf{V}_i[z] = \phi_i^{(z)}(\mathbf{V}_i[x], \mathbf{V}_i[y]) = \mathbf{V}_i[x] + \mathbf{V}_i[y].$$

对于输入为 x, y , 输出为 z 的乘法门, 任意的 $i \in [3]$, 下式成立

$$\begin{aligned} \mathbf{V}_i[z] &= \phi_i^{(z)}(\mathbf{V}_i[x, y], \mathbf{V}_{i+1}[x, y], s_i, s_{i+1}) \\ &= \mathbf{V}_i[x] \cdot \mathbf{V}_i[y] + \mathbf{V}_{i+1}[x] \cdot \mathbf{V}_i[y] + \mathbf{V}_i[x] \cdot \mathbf{V}_{i+1}[y] + R(s_i, z) - R(s_{i+1}, z), \end{aligned} \quad (59)$$

其中 s_i 表示第 i 个参与方的伪随机种子, $R(\cdot, \cdot)$ 表示以对应种子和对应电路门为输入生成的随机数, i 和 $i+1$ 均为模 3 剩余.

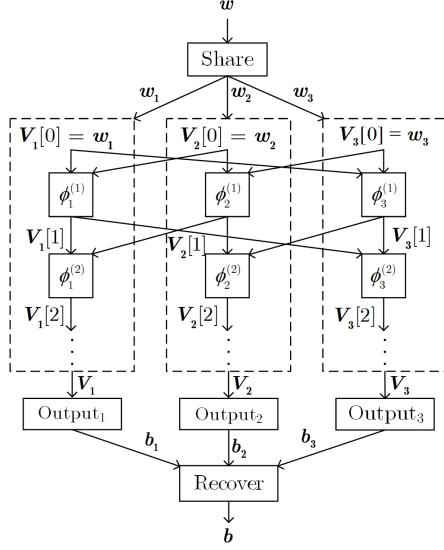


图 14 ZKBoo [33](2,3)-函数拆解方案⁵
Figure 14 (2,3)-function decomposition of ZKBoo

协议流程 ZKBoo 的协议流程如图 15 所示, 其分为证明者生成承诺、验证者挑战、证明者响应、验证者检查四个阶段, 具体过程如下.

- (1) 证明者生成承诺. 证明者在脑海中运行 MPC 协议, 得到视图 $\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3$ 和本地电路输出份额 b_1, b_2, b_3 . 然后 \mathcal{P} 生成三个承诺, 即对于任意的 $i \in [3]$, $c_i \leftarrow \text{Com}(\mathbf{V}_i; s_i)$, 最后 \mathcal{P} 将 $\mathbf{a} = (b_1, b_2, b_3, c_1, c_2, c_3)$ 发送给验证者 \mathcal{V} .
- (2) 验证者发送挑战. \mathcal{V} 选择随机挑战 $e \xleftarrow{\$} [3]$ 发送给 \mathcal{P} .

- (3) 证明者打开承诺回复响应. \mathcal{P} 打开对应承诺 c_e, c_{e+1} 并将 $\mathbf{z} = (\mathbf{V}_e, s_e, \mathbf{V}_{e+1}, s_{e+1})$ 发送给 \mathcal{V} .

验证者 \mathcal{V} 进行如下两项检查. ①正确性检查: 对任意的 $i \in [3]$, b_i 确实可由 w_i 和 \mathbf{V}_i 正确生成, 且 b_1, b_2, b_3 确实可以恢复 b . ②一致性检查: 对任意的 $j \in [|C|]$, 检查有 $\mathbf{V}_e[j] \stackrel{?}{=} \phi_e^{(j)}(\mathbf{V}_e, \mathbf{V}_{e+1}, s_e, s_{e+1})$. 当且仅当以上检查均通过, 验证者接受.

ZKB++ 由 Chase 等人^[34] 指出, 由于给定 \mathbf{V}_{e+1} 和 w_e 后 \mathbf{V}_e 可自行算出, 而且 \mathcal{V} 在验证过程中本身就需要重新计算一遍电路, 因此 \mathcal{P} 可以在响应阶段不发送 \mathbf{V}_e . 基于上述思想, 结合若干其他的优化措施, ZKB++ 在不增加 \mathcal{P} 和 \mathcal{V} 的计算复杂度的同时, 将 ZKBoo 的通信复杂度降低了约一半.

⁵图中一个虚线框表示一个参与方的视图, 对于域上电路 $C : \mathbb{F}^{|\mathbf{w}|} \rightarrow \mathbb{F}$, $\mathbf{V}_i(i \in [3])$ 是一个长为 $|C| + 1$ 的向量 (分别记为 $\mathbf{V}_i[j], j \in 0 \cup [|C|]$), $|C|$ 表示电路规模, $\mathbf{V}_i[0]$ 表示第 i 个参与方的输入 w_i (长为 $|\mathbf{w}| \cdot |\mathbb{F}|$ 个比特), $\mathbf{V}_i[j](j \neq 0)$ 表示参与方 i 在第 j 个电路门处的份额. $\phi_i^{(j)}$ 表示参与方 i 在第 j 个电路门处的消息确定函数. 其中 $\phi_i^{(j)}$ 由 $\mathbf{V}_i, \mathbf{V}_{i+1}$ 确定, 与 \mathbf{V}_{i+2} 无关.

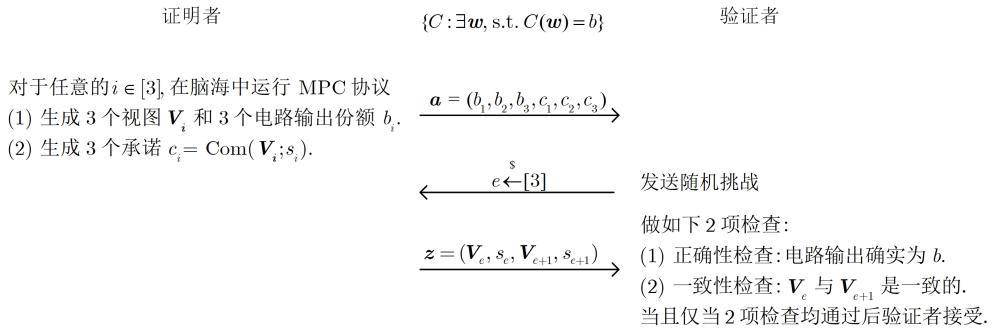


图 15 ZKBoo 流程图 [33]

Figure 15 Process of ZKBoo

讨论总结. 现分别从可靠性、零知识性、复杂度和改进方向对 ZKBoo 和 ZKB++ 进行讨论总结.

- (1) 可靠性. 在 ZKBoo 中, 一个恶意的证明者可以通过伪造参与方视图或者破坏视图一致性来欺骗验证者. 考虑伪造证据 \mathbf{w}' 且 $\mathcal{R}(\mathbf{x}, \mathbf{w}') = 0$, 若恶意证明者不破坏视图一致性, 则由 MPC 协议的完美正确性, 验证者一定会发现错误. 若恶意证明者破坏视图一致性, 考虑最有利于欺骗成功的情况, 即恶意证明者仅破坏一对视图的一致性, 此时可靠性为 $1/3$, 可靠性误差为 $2/3$. 因此 ZKBoo 的可靠性误差为 $2/3$.
 - (2) 零知识性. 考虑 ZKBoo 逐门更新视图和计算电路的过程, 参与方的隐私信息就是每个电路门值的秘密分享份额, 只要不泄露该份额即可满足零知识性. 对于加法门, 各参与方可以本地更新视图, 故零知识性可自然保障. 对于乘法门, 计算参与方 e 的视图 \mathbf{V}_e 时需要参与方 $e+1$ 的视图 \mathbf{V}_{e+1} , 而 \mathbf{V}_{e+1} 又与 \mathbf{V}_{e+2} 相关, 因此验证者获取视图 \mathbf{V}_e 和 \mathbf{V}_{e+1} 时可能会获取与 \mathbf{V}_{e+2} 相关的隐私信息, 这违背了隐私性, 从而破坏了零知识性. ZKBoo 解决该问题的方法是为消息确定函数增加随机输入 (等式(59)中的 $R(\cdot)$), 该随机输入能够在保障正确性的同时实现视图的均匀分布, 进而实现特殊诚实验证者零知识性.
 - (3) 知识可靠性. ZKBoo 是一个 3 轮 Σ 协议, 并且具有 3-特殊可靠性. 给定 3 个接受副本 $(\mathbf{a}, 1, z_1), (\mathbf{a}, 2, z_2), (\mathbf{a}, 3, z_3)$, 由于承诺的绑定性, z_1 和 z_3 中包含的视图 \mathbf{V}_1 是一致的. 同理, z_1 和 z_2 中包含的视图 \mathbf{V}_2 , z_2 和 z_3 中包含的视图 \mathbf{V}_3 也是一致的. 在拥有 $\mathbf{V}_1, \mathbf{V}_2, \mathbf{V}_3$ 后, 可分别计算 $w_1 \leftarrow \mathbf{V}_1[0], w_2 \leftarrow \mathbf{V}_2[0], w_3 \leftarrow \mathbf{V}_3[0]$, 然后恢复 $\mathbf{w} \leftarrow w_1 + w_2 + w_3$, 从而提取出证据 \mathbf{w} .
 - (4) 复杂度. 对于证明复杂度, 证明者需在脑海中模拟电路的计算过程并调用 $O(|C|)$ 次消息确定函数, 故证明复杂度为 $O(|C|)$; 对于验证复杂度, 验证者需在验证过程中重新计算电路, 故验证复杂度也为 $O(|C|)$; 对于通信复杂度, 当 n 确定时, 通信量主要由视图 \mathbf{V}_e 和 \mathbf{V}_{e+1} 决定, 而视图规模均为 $O(|C|)$, 故通信复杂度也为 $O(|C|)$.
 - (5) 改进方向. 对于通信复杂度, ZKBoo 和 ZKB++ 在采用加性秘密分享计算电路时会生成 n 个大小为 $O(|C|)$ 副本, 若视图打开数目为 $n-1$, 则通信复杂度为 $O((n-1)|C|)$, 其与参与方数目 n 和电路规模 $|C|$ 有关, 而 n 又与可靠性误差有关. 在 ZKBoo 和 ZKB++ 中, 为实现最低的通信复杂度, 需取 $n=3$, 而这会使得可靠性误差为 $2/3$. 因此, 相比于其他零知识证明, 为达到同样的目标可靠性误差, ZKBoo 和 ZKB++ 需重复运行较多的轮数, 导致较高的实际通信量. 一个优化方向是采用合适的 MPC 协议将生成视图的过程交与验证者, 使通信复杂度中的 $|C|$ 项与 n 无关, 从而通过降低可靠性误差来降低运行轮数进而优化通信复杂度, 而这恰好就是 KKW18 的主要思想.

8.3.2 KKW18

KKW18 由 Katz、Kolesnikov 和 Wang^[47] 提出，其底层 MPC 协议出自文献 [143]，该协议消息确定函数为 n 元、视图打开个数为 $n - 1$ 。

主要思路. KKW18 虽然也是逐门计算电路, 但是在该协议中各参与方的份额本质上是均匀分布的随机数, 可由伪随机生成器生成, 故对于打开的 $n - 1$ 个视图, 可以仅发送对应的随机种子 (长度仅与安全参数有关) 即可检查正确性和一致性. 该思想与 ZKB++ 内涵是一致的, 即可由验证者计算出的视图不需要发送. 需要注意的是, 采用 n 元消息确定函数并不是自然的, 因为检查正确性时需要第 n 个参与方的相关信息, 而获取该相关信息又可能违背 n -隐私性进而破坏零知识性. 针对这一问题, KKW18 采用了预处理模型下的 MPC 协议 (MPC protocol in the preprocessing model), 使份额的生成与电路的计算相互独立, 进而保障了零知识性.

底层 MPC 协议 KKW18 中的 MPC 模型为预处理模型, 可分为预处理阶段和在线阶段, 其具体流程如图 16 所示. 在给定具体描述前, 首先介绍加性秘密分享的参数记法. 记 $[\cdot]$ 表示加性秘密分享的份额, $[\cdot]_i$ 特指参与方 P_i 的份额. 对于电路中每条导线, n 个参与方拥有一个 (n, n) 的随机秘密分享掩藏份额和一个公开的掩藏值. 记 z_α 为布尔电路 C 在导线 α 处的值, \hat{z}_α 为公开掩藏值, $[\lambda_\alpha]$ 为秘密分享掩藏份额, 其恢复值为 λ_α , 则 z_α 满足 $z_\alpha = \hat{z}_\alpha \oplus \lambda_\alpha$.

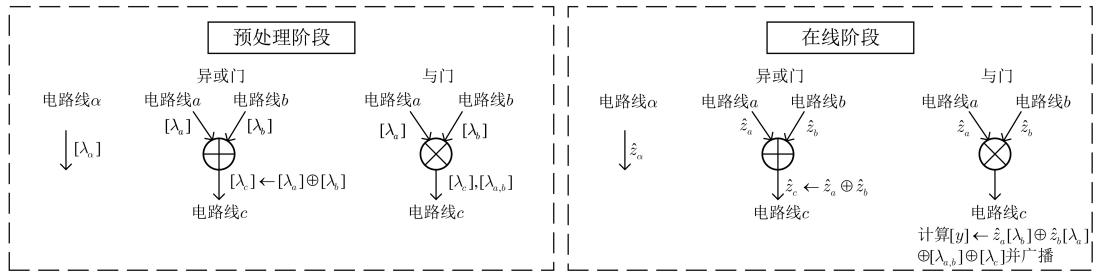


图 16 KKW18 底层 MPC 协议流程图 [47]
Figure 16 Process of MPC protocol in KKW18

- (1) 预处理阶段. 预处理阶段各参与方以伪随机种子和辅助输入为输入, 生成各输入导线值及电路门值的秘密分享掩藏份额. 对于输入导线 α , 各参与方获得份额 $[\lambda_\alpha]$. 对于输入导线为 a, b , 输出导线为 c 的异或门, 各参与方获得秘密掩藏份额 $[\lambda_a], [\lambda_b]$ 并自行计算生成 $[\lambda_c] \leftarrow [\lambda_a] \oplus [\lambda_b]$. 对于输入导线分别为 a 和 b , 输出导线为 c 的与门, 各参与方获得份额 $[\lambda_{a,b}]$, 其满足 $\lambda_{a,b} = \lambda_a \cdot \lambda_b$. 在预处理阶段, 份额 $[\lambda]$ 是均匀随机的, 因此可由伪随机种子 $s \in \{0, 1\}^{k_s}$ 生成. 需要注意的是, 前 $n - 1$ 个参与方的秘密掩藏份额 $[\lambda_{a,b}]$ 虽然也可由该法生成, 但第 n 个参与方 P_n 的 $\lambda_{a,b}$ 需要长为 $|C|$ 的辅助输入 aux 才能保证 $[\lambda_{a,b}] = \lambda_a \cdot \lambda_b$ 成立.
- (2) 在线阶段. 在线阶段各参与方以输入导线的公开掩藏值 \hat{z} 为输入, 结合输入导线和各自持有的秘密分享掩藏份额, 计算电路中所有导线的公开掩藏值. 对于异或门, 参与方可本地计算公开掩藏值 \hat{z} . 对于与门, 参与方需本地计算 $[y]$ 并将其广播, 之后每个参与方才能得到该与门的公开掩藏值 \hat{z}_c . 各参与方逐门更新, 在计算得到输出导线的公开掩藏值 \hat{z} 之后, 他们就广播各自输出导线的秘密分享掩藏份额 $[\lambda]$ 并最终恢复输出导线值.

现对上述 MPC 协议进行总结. 首先, 若证明者诚实运行协议, 由于秘密分享掩藏份额是在公开掩藏值之前生成的, 因此它们是互相独立的. 其次, 上述 MPC 协议仅需在计算与门和恢复电路最终输出导线值时广播本地计算值 $[y]$, 且各参与方计算广播值时不需要交互, 故在整个协议中一个参与方最多需要传输 $|C_A| + 1$ 个比特, 其中 $|C_A|$ 表示电路中与门的个数. 最后, 在拥有参与方 P_i 的伪随机种子 s_i 及可能需要的辅助输入 (仅针对 P_n) 后, 该参与方的广播值可自行算出.

协议流程 在零知识证明中调用上述 MPC 协议时, 每条输入导线值和每个电路值的秘密分享掩藏份额 $[\lambda]$ 就是各参与方持有的隐私信息, 其在预处理阶段生成, 本质上是均匀分布的随机数; 公开掩藏值 \hat{z} 于在线阶段生成, 是每个参与方的公共输入. 若协议诚实运行, 由于 $[\lambda]$ 在 \hat{z} 之前生成, $[\lambda]$ 与 \hat{z} 是独立的.

与一般 3 轮 Σ 协议不同的是, KKW18 协议为 5 轮. 事实上, 前 3 轮和后 3 轮可分别视为 2 个 Σ 协议, 且第 3 轮为重用轮. 在前 3 轮, 证明者生成 m 份预处理信息, 验证者只选取其中 1 份用以完成后续证

明; 在后 3 轮, 验证者打开 n 个参与方中的 $n - 1$ 份视图以验证正确性和一致性。具体流程如图 17 所示(省略了承诺中的随机数), 简述如下。

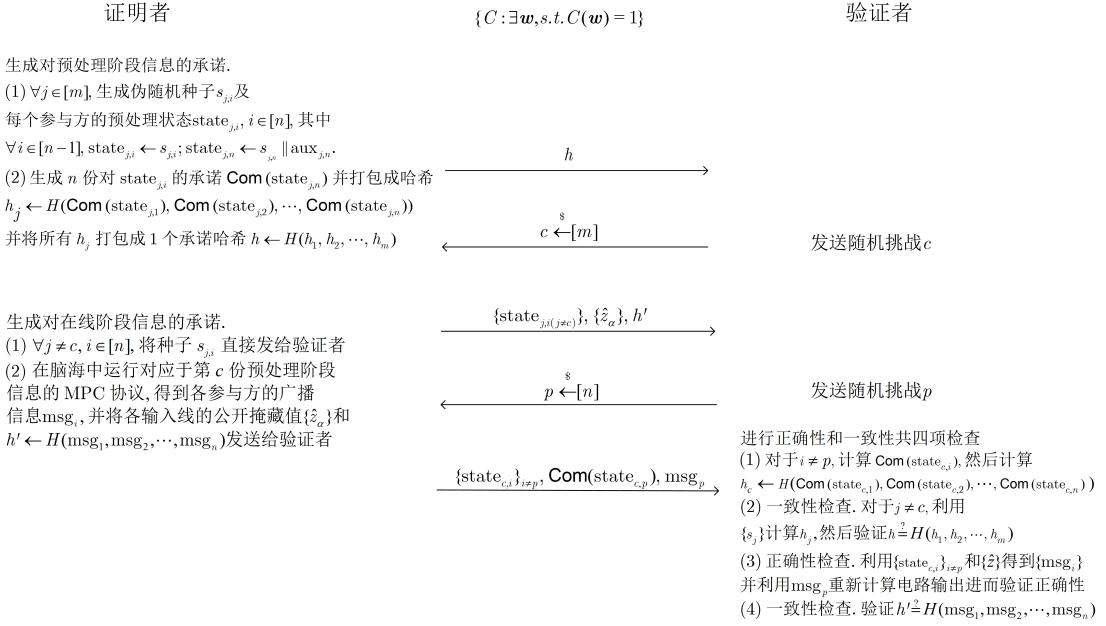


图 17 KKW18 协议流程图 [47]

Figure 17 Process of KKW18

- (1) \mathcal{P} 生成 m 份对预处理状态的承诺, 并计算对该 m 份承诺的哈希.
- (2) \mathcal{V} 发送第一次挑战, 即对预处理状态承诺的随机挑战.
- (3) \mathcal{P} 响应第一次挑战, 打开 $m - 1$ 份承诺. \mathcal{P} 在脑海中模拟与未打开的预处理阶段适配的在线阶段, 生成 n 份对广播信息的承诺, 并计算对该 n 份承诺的哈希.
- (4) \mathcal{V} 发送第二次挑战, 即对在线阶段广播信息承诺的随机挑战.
- (5) \mathcal{P} 响应第二次挑战, 将随机挑战值对应的预处理状态承诺和广播信息及剩余 $n - 1$ 组的预处理信息发送给 \mathcal{V} .

验证者进行如下三项检查, 当且仅当检查全部通过后接受. ①利用步骤 (5) 中 $n - 1$ 组预处理状态生成承诺, 结合步骤 (5) 中收到的预处理状态承诺构造 h_c , 验证与步骤 (1) 中哈希 h 的一致性. ②利用步骤 (5) 中 $\{\text{state}_{c,i}\}_{i \neq p}$ 计算 $\{\text{msg}_{i,i \neq p}\}$, 结合 msg_p 和 $\{\hat{z}_a\}$ 重新计算电路输出并验证正确性. ③利用广播信息 $\{\text{msg}_i\}_{i \in [n]}$ 计算哈希, 验证广播信息与步骤 (3) 中哈希 h' 的一致性.

讨论总结. 现分别从知识可靠性、零知识性和复杂度对 KKW18 进行讨论总结.

- (1) 知识可靠性. 假定抗碰撞哈希函数存在和承诺具有绑定性, 给定对于挑战对 (c, p) 、 $(c', *)$ 和 (c, p') 的接受副本, 其中 * 为任意值, 且 $c \neq c', p \neq p'$, 则可提取出有效证据 \mathbf{w} , 其可靠性误差为 $\max\{1/m, 1/n\}$ [47].
- (2) 零知识性. 与 ZKBoo 及 ZKB++ 类似, KKW18 的零知识性也是由秘密分享掩藏份额的随机性保障的. 不同的是, KKW18 中的消息确定函数为 n 元, 打开 $n - 1$ 个视图是无法验证协议正确性的, 为验证正确性还需要第 n 个参与方的广播信息 msg_n , 而该信息可能泄露隐私. 解决该问题的方法是在预处理模型下分阶段运行 MPC 协议, 从而保证秘密分享掩藏份额分布均匀且与公开掩藏值保持独立. 具体的, 对于某个输入导线为 a, b , 输出导线为 c 的与门, 由于公共掩藏值 \hat{z}_a 、 \hat{z}_b 与秘密分享掩藏份额 $[\lambda]_a, [\lambda]_b$ 是独立的, 因此即使获取参与方 P_p 的广播信息 $[y]_p$ (见图 16)

也难以推知导线 a, b 的秘密分享掩藏份额. 而参与方 P_p 的广播信息 msg_p 恰好就是其在所有与门处的广播信息 $\{[y]_p\}$, 因此 KKW18 的零知识性得以保障. 为确保证明者生成的秘密分享掩藏份额是均匀随机且与公开掩藏值是独立的, 需要在原有 Σ 协议的基础上增加两轮对预处理阶段的检查 (图 17 中的前两步), 故协议共为 5 轮.

- (3) 复杂度. 该协议的证明和验证复杂度与 ZKBoo 及 ZKB++ 类似, 均为 $O(|C|)$. 对于通信复杂度, 该协议的通信开销主要集中于协议流程的步骤 (5), 其中处理与门信息的通信量期望大小为 $(n - 1) \cdot (|C|/n + \kappa_s)$ 比特, 广播通信规模至多为 $|C_A| + 1$ 比特, $|C_A|$ 为电路中与门的个数. 因此, 协议的单轮通信复杂度为 $O(|C| + n\kappa_s)$. 相比于 ZKB++ 和 Ligero (见第 8.3.3 小节), 针对与门数为 300-100 000 的布尔电路可满足问题, KKW18 实现了证明的最低实际通信量.

8.3.3 Ligero/Ligero++

Ligero 由 Ames 等人^[35] 提出, 其底层 MPC 协议的消息确定函数为 1 元、消息传播方式为广播、视图打开个数为 t (可调参数)、通信复杂度为 $O(\sqrt{|C|})$. Ligero++ 由 Bhaduria 等人^[36] 提出, 他们利用内积论证改进了 Ligero 协议验证一致性的过程, 进而实现了 $O(\log |C|)$ 的通信复杂度. 由于 Ligero++ 仅在一致性检查方法上与 Ligero 有所不同, 本小节主要介绍 Ligero.

主要思路. Ligero 的主要思路如图 18 所示. 证明者 \mathcal{P} 首先将秘密输入 (x, y, z, w_e) 排列为矩阵, 然后利用 RS 码编码矩阵的每一行得到谕示矩阵, 接着利用默克尔树对谕示矩阵的每一列进行承诺, 最后 \mathcal{P} 与 \mathcal{V} 参与交互式协议用以证明电路是可满足的. 其中, RS 码保证了 \mathcal{V} 在获取谕示矩阵每行的部分值后也不能恢复整个多项式所以也无法得到电路其他位置的导线值, 而 RS 码的线性纠错性质保障了不符合电路约束的导线值会以较高的概率被 \mathcal{V} 发现.

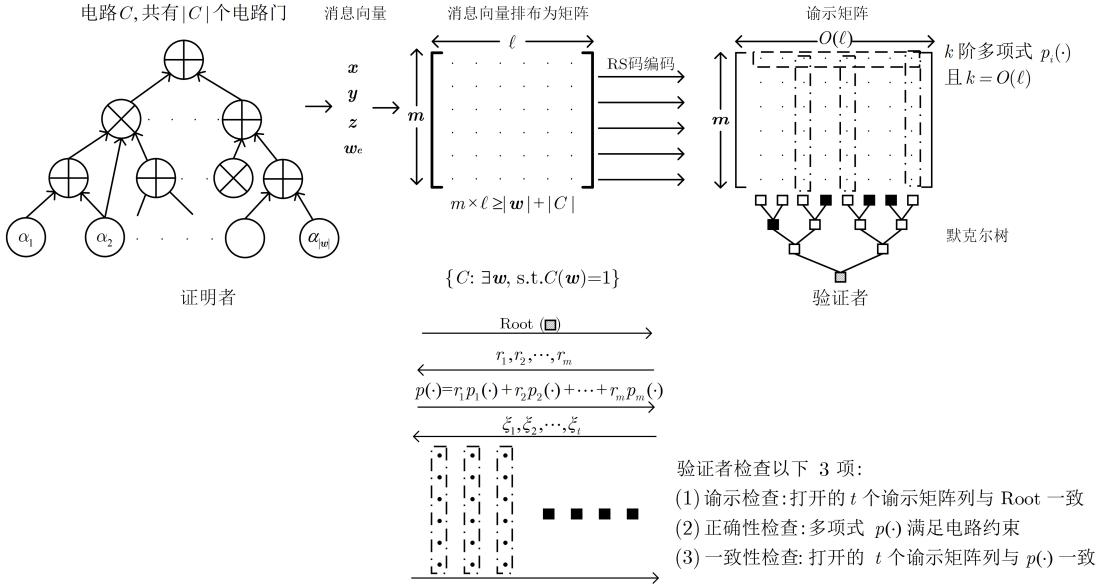


图 18 Ligero 主要思路及协议流程图^[35]

Figure 18 Main idea and process of Ligero

协议流程 在交互式协议中, 证明者和验证者需调用若干测试模块. 测试模块分为:

- (1) 对 IRS 码的检查, 该模块可用于检查谕示矩阵 \mathbf{U} 与其对应的 IRS 码 L^m 的距离是否小于 e , 其中 e 与 IRS 码的码距 d 有关, 在 Ligero 中, 有 $e < d/3$.
- (2) 对 IRS 码中线性约束的检查. 在模块 (1) 成立的情况下, 该模块可验证某 IRS 码 \mathbf{U} 加密的消息向量 $\mathbf{x} \in \mathbb{F}^{ml \times 1}$ 是否满足某线性关系 $\mathbf{Ax} = \mathbf{b}$, 其中 $\mathbf{A} \in \mathbb{F}^{ml \times ml}$, $\mathbf{b} \in \mathbb{F}^{ml \times 1}$.
- (3) 对 IRS 码中二次约束的检查. 在模块 (1) 成立的情况下, 该模块可验证 IRS 码 $\mathbf{U}^x, \mathbf{U}^y, \mathbf{U}^z$ 加密的消息向量 $\mathbf{x}, \mathbf{y}, \mathbf{z}$ 是否满足二次关系 $\mathbf{x} \odot \mathbf{y} + \mathbf{a} \odot \mathbf{z} = \mathbf{b}$, 其中 $\mathbf{a}, \mathbf{b} \in \mathbb{F}^{ml \times 1}$.

上述三个测试模块的主要思路也列于图 18. 首先证明者 \mathcal{P} 发送对谕示矩阵的承诺 (用默克尔树实现, 图 18 中的 Root), 其次验证者 \mathcal{V} 发送随机挑战 r_1, r_2, \dots, r_m , 接着 \mathcal{P} 返回谕示矩阵行多项式与挑战值的线性组合多项式 $p(\cdot)$, 然后 \mathcal{V} 发送第二次随机挑战 $\xi_1, \xi_2, \dots, \xi_t$, 最后 \mathcal{P} 将谕示矩阵的这 t 列及其对应的默克尔树路径发送给 \mathcal{V} . 此时 \mathcal{V} 验证以下三项:

- (1) 结合默克尔树路径, 验证打开的 t 列与承诺 Root 是一致的.
- (2) 验证组合多项式 $p(\cdot)$ 在点 $\eta_{i,i \in [t]}$ 处是否满足电路约束, 即验证 IRS 码的对应消息向量满足电路约束.
- (3) 验证组合多项式 $p(\cdot)$ 与谕示矩阵中随机选取的 t 列是一致的.

需要指出的是, 上述测试模块的可靠性均是由 RS 码的线性纠错性质保障的. 同时, 这些测试模块的检查过程本质上是一种 IPCP. 其中, 谕示矩阵是谕示, 随机挑战是访问请求.

借助上述测试模块, Ligero 的交互式论证简要流程如下.

- (1) 输入阶段. 记算术电路为 $C : \mathbb{F}^{|\mathbf{w}|} \rightarrow \mathbb{F}$, 证明者 \mathcal{P} 拥有秘密输入 $\mathbf{w} = (\alpha_1, \alpha_2, \dots, \alpha_{|\mathbf{w}|})$ 使得 $C(\mathbf{w}) = 1$. 记电路中门数为 $|C|$, 其门值分别记为 $\beta_1, \beta_2, \dots, \beta_{|C|}$.
- (2) 码矩阵生成阶段. 记 m, ℓ 满足 $m \cdot \ell \geq |\mathbf{w}| + |C|$, \mathcal{P} 首先生成扩展证据 $\mathbf{w}_e \in \mathbb{F}^{m\ell \times 1}$, \mathbf{w}_e 满足前 $|\mathbf{w}| + |C|$ 个值为 $(\alpha_1, \alpha_2, \dots, \alpha_{|\mathbf{w}|}, \beta_1, \beta_2, \dots, \beta_{|C|})$. 其次 \mathcal{P} 构造向量 $\mathbf{x}, \mathbf{y}, \mathbf{z} \in \mathbb{F}^{m\ell \times 1}$ 且其满足第 j 个值分别对应第 j 个乘法门的左输入、右输入及输出. 再次 \mathcal{P} 构造二次约束检查矩阵 $\mathbf{P}_x, \mathbf{P}_y$ 和 $\mathbf{P}_z \in \mathbb{F}^{m\ell \times m\ell}$, 其满足

$$\mathbf{P}_x \mathbf{w}_e = \mathbf{x}, \quad \mathbf{P}_y \mathbf{w}_e = \mathbf{y}, \quad \mathbf{P}_z \mathbf{w}_e = \mathbf{z}. \quad (60)$$

然后 \mathcal{P} 构造线性约束检查矩阵 \mathbf{P}_{add} , 其满足 $\mathbf{P}_{\text{add}} \mathbf{w}_e$ 的第 j 位置与第 j 个加法门的左输入、右输入及输出分别对应相等. 最后 \mathcal{P} 对消息向量 $\mathbf{w}_e, \mathbf{x}, \mathbf{y}$ 和 \mathbf{z} 进行加密得到 4 个码矩阵 $\mathbf{U}^{\mathbf{w}_e}, \mathbf{U}^{\mathbf{x}}, \mathbf{U}^{\mathbf{y}}, \mathbf{U}^{\mathbf{z}} \in L^m$, 记码矩阵 $\mathbf{U} \in L^{4m}$ 为 4 个码矩阵纵向排列后的码矩阵. 需要注意的是, 上述检查矩阵中元素全为常数, 且由于电路结构是公开的, 故检查矩阵可由验证者自行计算得出.

- (3) 交互阶段. \mathcal{P} 和 \mathcal{V} 运行如下 3 个模块协议: ①调用模块 (1) 检查 \mathbf{U} 与 L^{4m} 的码距是否小于 e . ②调用模块 (2) 检查线性约束的正确性, 即利用码矩阵 $\mathbf{U}^{\mathbf{w}_e}$ 检查 \mathbf{w} 是否满足 $\mathbf{P}_{\text{add}} \mathbf{w}_e = \mathbf{0}$. ③调用模块 (2) 和模块 (3) 检查二次约束的正确性, 即先调用模块 (2) 利用码矩阵 $\mathbf{U}^{\mathbf{w}}, \mathbf{U}^{\mathbf{x}}, \mathbf{U}^{\mathbf{y}}, \mathbf{U}^{\mathbf{z}}$ 检查对于任意的 $\mathbf{a} \in \mathbf{x}, \mathbf{y}, \mathbf{z}$, 有

$$[\mathbf{I}] - \mathbf{P}_a \begin{bmatrix} \mathbf{U}^{\mathbf{a}} \\ \mathbf{U}^{\mathbf{w}_e} \end{bmatrix} \stackrel{?}{=} \mathbf{0}, \quad (61)$$

其用于验证 $\mathbf{x}, \mathbf{y}, \mathbf{z}$ 是正确构造的, 再调用模块 (3) 利用码矩阵 $\mathbf{U}^{\mathbf{x}}, \mathbf{U}^{\mathbf{y}}, \mathbf{U}^{\mathbf{z}}$ 检查 $\mathbf{x} \odot \mathbf{y} - \mathbf{z} \stackrel{?}{=} \mathbf{0}$, 其用于验证 $\mathbf{x}, \mathbf{y}, \mathbf{z}$ 满足乘法门约束.

可以证明, 协议的可靠性误差为 $(e+6)/|\mathbb{F}| + (1-e/n)^t + 5((e+2k)/n)^t$.

Ligero++ 指出, 由于在 Ligero 中检查码矩阵与谕示多项式一致性的过程相当于验证一个内积关系, 即验证对于任意的 $\xi_{i,i \in [t]}, p(\xi_i) \stackrel{?}{=} \mathbf{r} \cdot \mathbf{p}(\xi_i)$, 其中 $\mathbf{r} = (r_1, r_2, \dots, r_m)$, $\mathbf{p}(\xi_i) = (p_1(\xi_i), p_2(\xi_i), \dots, p_m(\xi_i))$. 因此, 可以调用内积论证 (出自 Virgo^[28]) 实现上述检查. 若内积论证中的向量规模为 n , 则其通信复杂度为 $O(\log n)$. 也就是说, 可通过调节 m, ℓ 进一步降低通信复杂度, 具体描述见讨论总结.

讨论总结. 现分别从底层 MPC 协议、零知识性、复杂度和布尔电路版本对上述协议进行讨论分析.

- (1) 底层 MPC 协议. 与 ZKBoo、ZKB++ 和 KKW18 不同的是, Ligero 系列协议的底层 MPC 协议较为简单. 在 Ligero 中, 参与者的数目与谕示矩阵的列相等, 且谕示矩阵的每一列即是每个参与方的隐私信息. 由于该 MPC 协议的目标联合函数是计算谕示矩阵每一行与某个随机向量的线性组合, 每个参与方的计算任务就是在获取该随机向量后计算持有列与随机向量的线性组合并广

播. 该 MPC 协议本质上属于客户 - 服务器模型 (client - server model, 见第 8.3.5 小节), 基于该 MPC 模型, Ligero、Ligero++ 和 BooLigero 结合 IRS 码实现了通信复杂度亚线性级别的突破; Limbo^[49] 实现了通信复杂度虽为 $O(|C|)$ 、但中等规模电路下 (少于 500 000 个乘法门) 实际通信量最低的 NIZKAOK.

- (2) 零知识性. 上述协议并不是零知识的, 这是因为验证者获取的两种信息可能破坏零知识性, 第一是秘密输入与挑战值的线性组合, 第二是谕示中的 t 列. 针对第一点, 一个有效的方法是为各谕示矩阵增加一行随机编码且令其不影响原线性组合在 ℓ 个点的取值, 这保障了线性组合也不会泄露隐私信息. 针对第二点, 基于 IRS 码加密消息的随机性, 只需满足验证者可接触的多项式点值数目 (获取 ℓ 个点值以验证电路约束关系、 t 个点值以验证谕示的一致性) 小于多项式的度大小即可保障零知识性, 即满足 $k > \ell + t$.
- (3) 复杂度. 对于验证复杂度, 验证者需要验证正确性和一致性, 即分别计算 m 个多项式 $p_i(\cdot)$ 在 $(\ell + t)$ 个点的取值, 故大致需要 $m(\ell + t) = |C| + t\sqrt{|C|}$ 次运算 (t 可为常数), 复杂度为 $O(|C|)$. 对于证明复杂度, \mathcal{P} 的主要开销为构造 m 个度为 $k = O(\ell)$ 的多项式, 利用快速傅里叶变换^[147] 构造多项式的复杂度为 $O(mk \log^2 k) < O(|C| \log^2 |C|)$. 对于通信复杂度, 通信开销主要集中在在线性组合阶段发送的多项式 (规模为 $O(k + \ell)$) 和打开谕示矩阵阶段发送的 t 个矩阵列 (规模为 $O(t \cdot m)$), 又 m 和 ℓ 需满足 $O(m \cdot \ell) = O(|C|)$, 故实际参数设置中可设置 $k = O(\lambda), \ell = m = O(\sqrt{|C|})$ 从而实现 $O(\sqrt{|C|})$ 级别的通信复杂度. 需要指出的是, Ligero 可通过调整 m, ℓ 之间的关系调整证明复杂度和通信复杂度之间的关系. 在 Ligero++ 中, 如果调用内积论证检查一致性, 则检查 t 个规模为 m 的码矩阵的一致性仅需发送 $t \log m$ 个元素, 因此可重新设置参数 m, ℓ 为 $m = O(|C| / \log |C|), \ell = O(\log |C|)$ 从而实现 $O(\log |C|)$ 的通信复杂度.

8.3.4 BooLigero

BooLigero 由 Gvili、Scheffler 和 Varia^[48] 提出, 其优化了 Ligero 布尔电路版本的通信复杂度. BooLigero 的协议流程与 Ligero 是类似的, 本小节只讨论 BooLigero 优化通信复杂度的方法.

主要思路. BooLigero 的主要思路分为四部分, 分别是增加布尔约束、用伽罗瓦域存储比特、利用伽罗瓦域乘法实现按位与和高效比特约束检查协议, 简要描述如下.

- (1) 增加布尔约束. Ligero 本身虽然是针对算术电路的, 但也可适用于布尔电路. 对于布尔电路, 首先需要为每个电路门值增加布尔约束限制其为 0 或 1, 即增加约束 $\alpha^2 - \alpha = 0$. 此外还要增加对异或门和与门约束的算术版本. 给定布尔值 α_1, α_2 , 考虑约束 $\alpha_1 + \alpha_2 = r_0 + 2 \cdot r_1$, 若规定上述约束中值均为 0 或 1, 则 r_0 是 α_1 和 α_2 的异或, 则 r_1 是 α_1 和 α_2 的与, 因此可通过增加辅助输入比特 d 独立验证异或门或与门约束. 例如, 对于与门约束 $\alpha_1 \cdot \alpha_2 = \alpha_3$, 可增加辅助输入比特 d 验证线性约束 $\alpha_1 + \alpha_2 \stackrel{?}{=} d + 2 \cdot \alpha_3$. 验证异或门的线性约束方法同理可得. 值得注意的是, 为保障可靠性误差足够小, Ligero 中的域往往需取足够大. 但是 BooLigero 只需利用域中 0 和 1 共两个元素, 这会造成约 $(\log_2 |\mathbb{F}| - 1)$ 比特的存储浪费, 并增加通信开销. 一个自然的想法是引入能用一个域元素存储多个布尔值的域, 比如伽罗瓦域, 从而充分利用存储空间.
- (2) 用伽罗瓦域存储比特. 为解决存储浪费问题, BooLigero 指出可利用伽罗瓦域存储比特值, 从而实现一个域元素存储多个比特. 考虑伽罗瓦域 \mathbb{F}_{2^γ} , 若每一位代表一个比特值, 则一个域元素一次性可以存储 γ 个比特. 基于此, Ligero 中谕示矩阵的元素数目会降低为原来的 $1/O(\log |\mathbb{F}|)$, 通信复杂度会降低为原来的 $1/O(\log |\mathbb{F}|^{1/2})$. 然而, 运行 Ligero 中的电路约束检查模块不仅需要布尔域到伽罗瓦域的存储转化, 还需要布尔域到伽罗瓦域的电路计算转化, 而这直观上是难以实现的.
- (3) 利用伽罗瓦域乘法实现按位与. 虽然伽罗瓦域上的加法等同于布尔域上的按位异或, 但是伽罗瓦域上的乘法不同于布尔域上的按位与. 为解决该问题, 针对与门, BooLigero 提出了一种利用伽罗瓦域乘法实现比特的按位与的方法, 但该方法一方面需要为每个与门增加变量, 另一方面还需要增加新的变量约束检查. 具体的, 为验证长为 γ 的两个向量之间的与关系, 需在谕示矩阵中增加 $\sqrt{\gamma}$ 个新变量, 也就需增加 $\sqrt{\gamma}$ 个对新变量的约束检查.
- (4) 高效比特约束检查协议. 在 BooLigero 中, 对新变量的约束具有高度规律性, 其形式都是一组向

量的“某位置是 0”或者一组向量的“某个位置等于目标向量的某个位置”。为了验证某变量满足某约束关系，BooLigero 提出了一种零知识的高效比特约束检查协议来实现批量化比特关系验证。该协议基于 cut-and-choose 思想，可以在不揭露秘密输入信息 x 的同时证明其满足关系 $\mathbf{T}x = \mathbf{0}$ （其中 \mathbf{T} 为某个公开矩阵）。同时，该协议的通信复杂度仅与安全参数有关，与隐私信息的长度无关。

讨论总结。根据电路结构的不同，相比于 Ligero 的布尔电路版本，BooLigero 可将通信复杂度降低为原来的 $1/O(\log |\mathbb{F}|^{1/2}) \sim 1/O(\log |\mathbb{F}|^{1/4})$ 。具体的，若电路全是异或门，谕示矩阵元素数目会减少至原来的 $1/O(\log |\mathbb{F}|)$ ，通信复杂度会降低到原来的 $1/O(\log |\mathbb{F}|^{1/2})$ ；若电路全是与门，谕示矩阵会因实现伽罗瓦域按位与运算增加至原来变量数的 $O(\log |\mathbb{F}|^{1/2})$ 倍，故矩阵元素数目会减少至原来的 $1/O(\log |\mathbb{F}|^{1/2})$ ，通信复杂度会降低至原来的 $1/O(\log |\mathbb{F}|^{1/4})$ 。

8.3.5 Limbo

Limbo 由 Guilhem、Orsini 和 Tanguy^[49] 提出，其拓展了 Ligero 的 MPC 协议模型并基于 IOP 实现了通信复杂度虽然为 $O(|C|)$ 、但实际性能良好的 NIZKAoK。Limbo 适用于算术电路和布尔电路，对于布尔电路，Limbo 的实际通信量相比于 KKW18 有显著降低；对于算术电路，相比于其他基于 MPC-in-the-Head 的零知识证明，Limbo 实现了针对中等规模电路（乘法门少于 500 000 个）可满足性问题的当前最优实际性能。

主要思路。构造基于 MPC-in-the-Head 的高效零知识证明的一个自然思路是利用高效的 MPC 协议，而这也恰好是 KKW18 的主要思路。与之不同的是，Limbo 的主要思路是利用最适合 MPC-in-the-Head 的 MPC 模型可能会使零知识证明更高效。具体的，Limbo 中的 ρ 轮通用 MPC 模型如图 19 所示，其是客户-服务器模型（client-server model），即参与者可分为 1 个客户（发送方） P_S 、 n 个计算服务器 P_1, P_2, \dots, P_n 和 1 个接收方 P_R 。在该模型中， P_S 拥有整个计算的输入，且在每一轮的开始阶段 P_S 最多发送 1 次消息。在第 j 轮 ($j \in [2, \rho - 1]$)，计算服务器调取公共抛币函数获得随机串 R^j 并根据本轮及之前收到的信息进行本地计算。在第 ρ 轮，计算服务器获得随机串 R^ρ 经过本地计算后向 P_R 发送消息。在上述模型中，计算服务器之间不需进行交互。

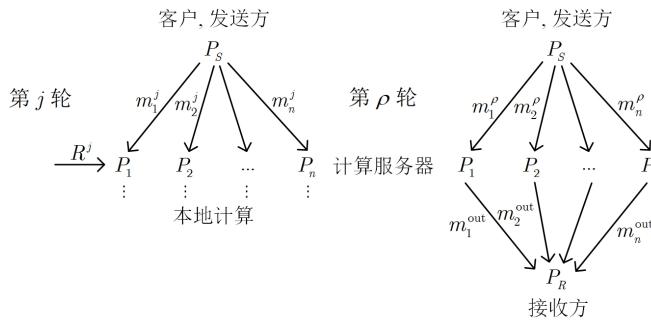


图 19 Limbo 的 MPC 模型^[49](client-server 模型)

Figure 19 MPC model of Limbo (client-server model)

上述模型略加修改即可构造基于 MPC-in-the-Head 的零知识证明。此时，证明者 \mathcal{P} 在脑海中模拟上述 MPC 协议的运行，此时随机串来源于验证者 \mathcal{V} 的随机挑战，且在每轮结束后 \mathcal{P} 利用本轮发送的信息构造一个谕示。相比于调用其他 MPC 模型，该证明中计算服务器之间不需交互，考虑到交互会带来视图规模的增大和运行 MPC 协议开销的提升，因此基于该模型构造的零知识证明的实际通信量和计算开销均较低。事实上，Ligero 就是 $\rho = 1$ 时的特例，并且该模型本质上属于 IOP。可以证明，若 MPC 协议具有 $(P_R, n - 1)$ -隐私性和 $(P_S, 0)$ -鲁棒性时，基于该模型的交互式零知识论证可靠性误差为 $\epsilon = 1/n + \delta(1 - 1/n)$ ，其中， $(P_R, n - 1)$ -隐私性指半诚实模型下敌手腐蚀 P_R 和 $n - 1$ 个计算服务器仍可保障隐私； $(P_S, 0)$ -鲁棒性指在恶意模型下敌手腐蚀 P_S 仍可保障鲁棒性， δ 为鲁棒性误差并取决于具体 MPC 协议。此外，将上述 MPC 模型中的发送方数目增加为 τ 个并调用相同的公共抛币函数可将可靠性误差降低为 $\epsilon = 1/n^\tau + \delta(1 - 1/n^\tau)$ 。

底层 MPC 协议. 基于上述思路, Limbo 采用一个简单的 MPC 协议 (见文献 [144–146]) 构造了交互式零知识知识论证, 并利用 Fiat-Shamir 启发式转换为了 NIZKAO. 对于基于加性秘密分享的 MPC 协议, 加法门的结果可根据自身的秘密掩藏份额自行计算和更新, 因此只需考虑乘法门. 具体的, 该 MPC 协议用于证明 m 个乘法门约束成立, 即证明给定 m 个三元组 $\{x_\ell, y_\ell, z_\ell\}_{\ell \in [m]}$, 对于任意的 $\ell \in [m]$, 都有 $x_\ell \cdot y_\ell = z_\ell$, 该协议的主要思路如协议 9 所示.

协议 9 Limbo 的底层 MPC 协议主要思路 [49]

公共输入: 域 \mathbb{F} , ℓ, n , 电路 C .

证明者秘密输入: $\{[x_\ell]_i, [y_\ell]_i, [z_\ell]_i\}$, 其中 $\ell \in [m], i \in [n]$, $[x_\ell]_i$ 表示第 i 个计算服务器在第 ℓ 个乘法门处左输入的份额, $[y_\ell]_i, [z_\ell]_i$ 分别表示在第 ℓ 个乘法门处右输入、输出的份额.

1. 对于所有的 $\ell \in [m], i \in [n]$, 发送方 P_s 将份额 $[x_\ell]_i, [y_\ell]_i, [z_\ell]_i$ 发送给 P_i . 此外, 发送方 P_s 随机选取满足关系 $\mathbf{a} \cdot \mathbf{b} = c$ 的向量份额 $[\mathbf{a}]_i, [\mathbf{b}]_i, [\mathbf{c}]_i$ 也发送给 P_i .
2. 各计算服务器调取公共抛币函数获取随机数 R, s .
3. 对于所有的 $i \in [n]$, 计算服务器 P_i 计算

$$[\mathbf{x}]_i \leftarrow ([x_1]_i, R \cdot [x_2]_i, \dots, R^{m-1} \cdot [x_m]_i), [\mathbf{y}]_i \leftarrow ([y_1]_i, [y_2]_i, \dots, [y_m]_i), [z]_i \leftarrow \sum_{\ell \in [m]} R^{\ell-1} \cdot [z_\ell]_i.$$

然后计算服务器 P_i 计算 $[\boldsymbol{\sigma}]_i \leftarrow s \cdot [\mathbf{x}]_i - [\mathbf{a}]_i$ 和 $[\boldsymbol{\rho}]_i \leftarrow [\mathbf{y}]_i - [\mathbf{b}]_i$, 随后广播 $[\boldsymbol{\sigma}]_i$ 和 $[\boldsymbol{\rho}]_i$.

4. 各计算服务器恢复 $\boldsymbol{\sigma}$ 和 $\boldsymbol{\rho}$.
5. 对于所有的 $i \in [n]$, 计算服务器 P_i 计算

$$[v]_i \leftarrow s \cdot [z]_i - [\mathbf{c}]_i - [\mathbf{b}]_i \cdot \boldsymbol{\sigma} - [\mathbf{a}]_i \cdot \boldsymbol{\rho} - \boldsymbol{\rho} \cdot \boldsymbol{\sigma}.$$

然后 P_i 将 $[v]_i$ 发送给接收方 P_R .

6. P_R 恢复 v .

输出: 比特 b , 若 $v = 0$, 输出 $b = 1$; 否则输出 $b = 0$.

现分析该 MPC 协议的可靠性. 若至少一个乘法门三元组是错误的, 则由 Schwartz-Zippel 引理和 R 的随机性可知, 若 $[\mathbf{x}]_i \cdot [\mathbf{y}]_i - [z]_i \neq 0$, 则验证者通过的概率为 $(m-1)/|\mathbb{F}|$. 基于此, 如果 $[\mathbf{x}]_i \cdot [\mathbf{y}]_i - [z]_i$ 和 $[\mathbf{a}] \cdot [\mathbf{b}] - [\mathbf{c}]$ 至少有一个不为 0, 由 s 的随机性可知, 验证者通过的概率为 $2/|\mathbb{F}|$. 因此, 该协议的可靠性为 $(m-1)/|\mathbb{F}| + (1-(m-1)/|\mathbb{F}|) \cdot 2/|\mathbb{F}|$.

利用上述 MPC 协议可自然构造对应的零知识证明, 此时协议轮数为 3. 然而在上述 MPC 协议中, 广播的消息规模为 $2mn$, 因此对应零知识证明的通信复杂度至少为 $O(n|C_M|)$. 为进一步降低通信复杂度, Guilhem、Orsini 和 Tanguy 提出了一种通过压缩内积规模降低通信复杂度的方法, 不过其会增加 $\log_k m$ 轮交互. 利用该方法, 对于每个乘法门, 只需传输 1 个域元素即可完成证明, 此时协议的通信复杂度可降低为 $O(|C|)$.

讨论总结. 基于 client-server 模型和 IOP, Limbo 实现了针对中等规模算术电路 (乘法门少于 500 000 个) 可满足性问题的良好实际性能. 除此之外, 与 Ligero 类似, Limbo 也具有一定的灵活性 (flexibility). 例如, 在相同的目标可靠性误差下, 增加 n 的数目可有效降低轮数从而降低实际通信量, 然而这会增加证明者的计算开销, 因此可通过调节 n, τ 等参数的大小从而调节证明者计算开销和通信量之间的关系.

8.4 总结

基于 MPC-in-the-Head 的零知识证明具有以下优点:

- (1) 底层难题假设更为通用. 该类证明只需假设 CRHF 存在, 且仅有对称密钥操作, 故是抗量子的, 并可用于构造高效抗量子数字签名 (如 Picnic^[34, 47, 49]⁶).
- (2) 具有一定的灵活性. 可通过调整参数控制证明者的计算开销和通信量之间的关系.
- (3) 可堆叠性 (stackable). Goel 等人^[148] 指出 Ligero 和 KKW18 具有可堆叠性, 即给定某个针对 NP 语言 $\mathcal{L}(\mathcal{R})$ 的 Σ 协议, 证明陈述 $(x_1 \in \mathcal{L}(\mathcal{R})) \vee (x_2 \in \mathcal{L}(\mathcal{R})) \vee \dots \vee (x_\ell \in \mathcal{L}(\mathcal{R}))$ 的通信复杂度可为 $O(CC(\Sigma) + \lambda \log \ell)$, 其中 $CC(\Sigma)$ 为 Σ 协议的通信复杂度, λ 为安全参数, 而证明上述陈述的平凡通信复杂度为 $O(\ell \cdot CC(\Sigma) + \lambda)$.

⁶<https://microsoft.github.io/Picnic>

然而, 该类协议的通信复杂度虽已达到对数级别, 但是实际通信量通常较高 [27, §9.2], 如何进一步降低该类零知识证明的实际通信量是存在的一个问题。此外, 目前在理论研究和实际性能层面均未找到最适合 MPC-in-the-Head 的 MPC 协议, 因此, 如何选取高效的 MPC 协议是可能是未来的一个研究方向。

9 未来研究方向

近年来, 虽然针对简洁非交互零知识证明的研究取得了较大进展, 并且简洁非交互零知识证明逐渐在区块链、隐私计算等领域得到了较为广泛的应用, 但是仍然存在很多尚未解决的理论和技术问题。以下列出未来的主要发展趋势。

- (1) 通用构造方法层面。进一步总结完善构造简洁非交互零知识证明的通用构造方法, 如拓宽 IPCP、IOP, 探讨是否存在可用于更高效构造简洁非交互零知识证明的信息论安全证明, 讨论基于 IPA 的零知识证明能否也存在底层信息论安全证明等; 基于某一特定信息论安全证明, 优化改进密码编译器的具体技术, 实现更好的性能。
- (2) 性能层面。从复杂度角度, 进一步降低简洁非交互零知识的证明、通信和验证复杂度, 如削弱基于 DEIP 的零知识证明中电路深度 d 的影响; 从实际性能角度, 优化算法性能, 如利用批量验证的方式降低基于 IPA 的零知识证明的实际验证开销、通过证明者在脑海中一次性模拟多个 MPC 运行的方式降低基于 MPC-in-the-Head 的零知识证明的实际证明开销、选取或设计更适合 MPC-in-the-Head 的 MPC 协议等。
- (3) 初始化层面。虽然基于 QAP 的零知识证明 (zk-SNARK) 可实现常数级别的群元素的通信复杂度和仅与公共输入输出长度成线性关系的验证复杂度, 但在区块链应用中, 如何保障可信初始化的安全、优化可信初始化的性能仍是目前的一个难点。因此, 探讨高效可更新 CRS 的构造方式, 研究 CRS 长度更短、生成更快的 zk-SNARK 是未来一个可能的发展方向。
- (4) 安全层面。研究基于标准假设如何构造简洁非交互零知识证明, 提高协议的安全性; 基于更为通用的难题假设构造 zk-SNARK 及系列可更新零知识证明; 探讨在系统参数可公开生成的零知识证明中利用基于 ROM 的 Fiat-Shamir 启发式实现非交互是否及会如何降低协议性能。

参考文献

- [1] GOLDWASSER S, MICALI S, RACKOFF C. The knowledge complexity of interactive proof-systems (extended abstract)[C]. In: Proceedings of the Seventeenth Annual ACM Symposium on Theory of Computing (STOC 1985). ACM, 1985: 291–304. [DOI: 10.1145/22145.22178]
- [2] SAHAI A. Non-malleable non-interactive zero knowledge and adaptive chosen-ciphertext security[C]. In: Proceedings of 40th Annual Symposium on Foundations of Computer Science (FOCS 1999). IEEE, 1999: 543–553. [DOI: 10.1109/SFFCS.1999.814628]
- [3] GUILLOU L C, QUISQUATER J. A “paradoxical” identity-based signature scheme resulting from zero-knowledge[C]. In: Advances in Cryptology—CRYPTO ’88. Springer Berlin Heidelberg, 1990: 216–231. [DOI: 10.1007/0-387-34799-2_16]
- [4] FEIGE U, FIAT A, SHAMIR A. Zero-knowledge proofs of identity[J]. Journal of Cryptology, 1988, 1(2): 77–94. [DOI: 10.1007/BF02351717]
- [5] MA S L, DENG Y, HE D B, et al. An efficient NIZK scheme for privacy-preserving transactions over account-model blockchain[J/OL]. IEEE Transactions on Dependable Secure Computing, 2021, 18(2): 641–651. [DOI: 10.1109/TDSC.2020.2969418]
- [6] CHASE M, GANESH C, MOHASSEL P. Efficient zero-knowledge proof of algebraic and non-algebraic statements with applications to privacy preserving credentials[C]. In: Advances in Cryptology—CRYPTO 2016, Part III. Springer Berlin Heidelberg, 2016: 499–530. [DOI: 10.1007/978-3-662-53015-3_18]
- [7] BEN-SASSON E, CHIESA A, GARMAN C, et al. Zerocash: Decentralized anonymous payments from bitcoin[C]. In: Proceedings of 2014 IEEE Symposium on Security and Privacy (S&P 2014). IEEE, 2014: 459–474. [DOI: 10.1109/SP.2014.36]
- [8] ZHANG Z F, YANG K, HU X X, et al. Practical anonymous password authentication and TLS with anonymous client authentication[C]. In: Proceedings of the 2016 ACM SIGSAC Conference on Computer and Communications Security (CCS 2016). ACM, 2016: 1179–1191. [DOI: 10.1145/2976749.2978354]

- [9] LI Z P, WANG D, MORAIS E. Quantum-safe round-optimal password authentication for mobile devices[J]. *IEEE Transactions on Dependable and Secure Computing*, 2022, 19(3): 1885–1899. [DOI: 10.1109/TDSC.2020.3040776]
- [10] WANG Q X, WANG D, CHENG C, et al. Quantum2FA: Efficient quantum-resistant two-factor authentication scheme for mobile devices[J]. *IEEE Transactions on Dependable and Secure Computing*, 2021: 1–1. [DOI: 10.1109/TDSC.2021.3129512]
- [11] GOLDRICH O, MICALI S, WIGDERSON A. How to prove all NP-statements in zero-knowledge, and a methodology of cryptographic protocol design[C]. In: *Advances in Cryptology—CRYPTO '86*. Springer Berlin Heidelberg, 1987: 171–185. [DOI: 10.1007/3-540-47721-7_11]
- [12] BLUM M, FELDMAN P, MICALI S. Non-interactive zero-knowledge and its applications (extended abstract)[C]. In: *Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing (STOC 1988)*. ACM, 1988: 103–112. [DOI: 10.1145/62212.62222]
- [13] BELLARE M, ROGAWAY P. Random oracles are practical: A paradigm for designing efficient protocols[C]. In: *Proceedings of the 1st ACM Conference on Computer and Communications Security (CCS 1993)*. ACM, 1993: 62–73. [DOI: 10.1145/168588.168596]
- [14] CANETTI R, GOLDRICH O, HALEVI S. The random oracle methodology, revisited (preliminary version)[C]. In: *Proceedings of the Thirtieth Annual ACM Symposium on Theory of Computing (STOC 1998)*. ACM, 1998: 209–218. [DOI: 10.1145/276698.276741]
- [15] KILIAN J. A note on efficient zero-knowledge proofs and arguments (extended abstract)[C]. In: *Proceedings of the Twenty-fourth Annual ACM Symposium on Theory of Computing (STOC 1992)*. ACM, 1992: 723–732. [DOI: 10.1145/129712.129782]
- [16] BABAI L, FORTNOW L, LEVIN L A, et al. Checking computations in polylogarithmic time[C]. In: *Proceedings of the Twenty-third Annual ACM Symposium on Theory of Computing (STOC 1991)*. ACM, 1991: 21–31. [DOI: 10.1145/103418.103428]
- [17] ARORA S, SAFRA S. Probabilistic checking of proofs: A new characterization of NP[J]. *Journal of the ACM*, 1998, 45(1): 70–122. [DOI: 10.1145/273865.273901]
- [18] MICALI S. CS proofs (extended abstracts)[C]. In: *Proceedings of 35th Annual Symposium on Foundations of Computer Science (FOCS 1994)*. IEEE, 1994: 436–453. [DOI: 10.1109/SFCS.1994.365746]
- [19] GENNARO R, GENTRY C, PARNO B, et al. Quadratic span programs and succinct NIZKs without PCPs[C]. In: *Advances in Cryptology—EUROCRYPT 2013*. Springer Berlin Heidelberg, 2013: 626–645. [DOI: 10.1007/978-3-642-38348-9_37]
- [20] PARNO B, HOWELL J, GENTRY C, et al. Pinocchio: Nearly practical verifiable computation[C]. In: *Proceedings of 2013 IEEE Symposium on Security and Privacy (S&P 2013)*. IEEE, 2013: 238–252. [DOI: 10.1109/SP.2013.47]
- [21] DANEZIS G, FOURNET C, KOHLWEISS M, et al. Pinocchio coin: Building zerocoin from a succinct pairing-based proof system[C]. In: *Proceedings of the First ACM Workshop on Language Support for Privacy-enhancing Technologies (PETShop 2013)*. ACM, 2013: 27–30. [DOI: 10.1145/2517872.2517878]
- [22] BEN-SASSON E, CHIESA A, TROMER E, et al. Succinct non-interactive zero knowledge for a von Neumann architecture[C]. In: *Proceedings of the 23rd USENIX Conference on Security Symposium (SEC '14)*. USENIX, 2014: 781–796. [DOI: 10.5555/2671225.2671275]
- [23] GABIZON A, WILLIAMSON Z J, CIOBOTARU O. PLONK: Permutations over Lagrange-bases for oecumenical noninteractive arguments of knowledge[J/OL]. *IACR Cryptology ePrint Archive*, 2019: 2019/953. <https://eprint.iacr.org/2019/953.pdf>
- [24] NAOR M. On cryptographic assumptions and challenges[C]. In: *Advances in Cryptology—CRYPTO 2003*. Springer Berlin Heidelberg, 2003: 96–109. [DOI: 10.1007/978-3-540-45146-4_6]
- [25] GENTRY C, WICHS D. Separating succinct non-interactive arguments from all falsifiable assumptions[C]. In: *Proceedings of the Forty-third Annual ACM Symposium on Theory of Computing (STOC 2011)*. ACM, 2011: 99–108. [DOI: 10.1145/1993636.1993651]
- [26] XIE T C, ZHANG J H, ZHANG Y P, et al. Libra: Succinct zero-knowledge proofs with optimal prover computation[C]. In: *Advances in Cryptology—CRYPTO 2019, Part III*. Springer Cham, 2019: 733–764. [DOI: 10.1007/978-3-030-26954-8_24]
- [27] SETTY S. Spartan: Efficient and general-purpose zkSNARKs without trusted setup[C]. In: *Advances in Cryptology—CRYPTO 2020, Part III*. Springer Cham, 2020: 704–737. [DOI: 10.1007/978-3-030-56877-1_25]
- [28] ZHANG J H, XIE T C, ZHANG Y P, et al. Transparent polynomial delegation and its applications to zero knowledge proof[C]. In: *Proceedings of 2020 IEEE Symposium on Security and Privacy (S&P 2020)*. IEEE, 2020: 859–876. [DOI: 10.1109/SP40000.2020.00052]

- [29] ZHANG J H, LIU T Y, WANG W, et al. Doubly efficient interactive proofs for general arithmetic circuits with linear prover time[C]. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS 2021). ACM, 2021: 159–177. [DOI: 10.1145/3460120.3484767]
- [30] BOOTLE J, CERULLI A, CHAIDOS P, et al. Efficient zero-knowledge arguments for arithmetic circuits in the discrete log setting[C]. In: Advances in Cryptology—EUROCRYPT 2016, Part II. Springer Berlin Heidelberg, 2016: 327–357. [DOI: 10.1007/978-3-662-49896-5_12]
- [31] BÜNZ B, BOOTLE J, BONEH D, et al. Bulletproofs: Short proofs for confidential transactions and more[C]. In: Proceedings of 2018 IEEE Symposium on Security and Privacy (S&P 2018). IEEE, 2018: 315–334. [DOI: 10.1109/SP.2018.00020]
- [32] WAHBY R S, TZIALLA I, SHELAT A, et al. Doubly-efficient zkSNARKs without trusted setup[C]. In: Proceedings of 2018 IEEE Symposium on Security and Privacy (S&P 2018). IEEE, 2018: 926–943. [DOI: 10.1109/SP.2018.00060]
- [33] GIACOMELLI I, MADSEN J, ORLANDI C. ZKBoo: Faster zero-knowledge for Boolean circuits[C]. In: Proceedings of the 25th USENIX Conference on Security Symposium (SEC ’16). USENIX, 2016: 1069–1083. [DOI: 10.5555/3241094.3241177]
- [34] CHASE M, DERLER D, GOLDFEDER S, et al. Post-quantum zero-knowledge and signatures from symmetric-key primitives[C]. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS 2017). ACM, 2017: 1825–1842. [DOI: 10.1145/3133956.3133997]
- [35] AMES S, HAZAY C, ISHAI Y, et al. Ligero: Lightweight sublinear arguments without a trusted setup[C]. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS 2017). ACM, 2017: 2087–2104. [DOI: 10.1145/3133956.3134104]
- [36] BHADAURIA R, FANG Z Y, HAZAY C, et al. Ligero++: A new optimized sublinear IOP[C]. In: Proceedings of the 2020 ACM SIGSAC Conference on Computer and Communications Security (CCS 2020). ACM, 2020: 2025–2038. [DOI: 10.1145/3372297.3417893]
- [37] BELLARE M, FUCHSBAUER G, SCAFURO A. NIZKs with an untrusted CRS: Security in the face of parameter subversion[C]. In: Advances in Cryptology—ASIACRYPT 2016, Part II. Springer Berlin Heidelberg, 2016: 777–804. [DOI: 10.1007/978-3-662-53890-6_26]
- [38] ABDOLMALEKI B, BAGHERY K, LIPMAA H, et al. A subversion-resistant SNARK[C]. In: Advances in Cryptology—ASIACRYPT 2017. Part III. Springer Cham, 2017: 3–33. [DOI: 10.1007/978-3-319-70700-6_1]
- [39] FUCHSBAUER G. Subversion-zero-knowledge SNARKs[C]. In: Public-Key Cryptography—PKC 2018, Part I. Springer Cham, 2018: 315–347. [DOI: 10.1007/978-3-319-76578-5_11]
- [40] MALLER M, BOWE S, KOHLWEISS M, et al. Sonic: Zero-knowledge SNARKs from linear-size universal and updatable structured reference strings[C]. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS 2019). ACM, 2019: 2111–2128. [DOI: 10.1145/3319535.33339817]
- [41] GROTH J, KOHLWEISS M, MALLER M, et al. Updatable and universal common reference strings with applications to zk-SNARKs[C]. In: Advances in Cryptology—CRYPTO 2018. Part III. Springer Cham, 2018: 698–728. [DOI: 10.1007/978-3-319-96878-0_24]
- [42] CHIESA A, HU Y, MALLER M, et al. Marlin: Preprocessing zkSNARKs with universal and updatable SRS[C]. In: Advances in Cryptology—EUROCRYPT 2020, Part I. Springer Cham, 2020: 738–768. [DOI: 10.1007/978-3-030-45721-1_26]
- [43] BEN-SASSON E, BENTOV I, HORESH Y, et al. Scalable zero knowledge with no trusted setup[C]. In: Advances in Cryptology—CRYPTO 2019, Part III. Springer Cham, 2019: 701–732. [DOI: 10.1007/978-3-030-26954-8_23]
- [44] GOLDREICH O. Zero-knowledge twenty years after its invention[J/OL]. IACR Cryptology ePrint Archive, 2002: 2002/186. <https://eprint.iacr.org/2002/186.pdf>
- [45] LI F, McMILLIN B M. A survey on zero-knowledge proofs[J]. Advances in Computers, 2014, 94: 25–69. [DOI: 10.1016/B978-0-12-800161-5.00002-5]
- [46] MOHR A. A survey on zero-knowledge proofs with applications to cryptography[EB/OL]. 2007. <http://www.austinmohr.com/work/files/zkp.pdf>
- [47] KATZ J, KOLESNIKOV V, WANG X. Improved non-interactive zero knowledge with applications to post-quantum signatures[C]. In: Proceedings of the 2018 ACM SIGSAC Conference on Computer and Communications Security (CCS 2018). ACM, 2018: 525–537. [DOI: 10.1145/3243734.3243805]
- [48] GIVILI Y, SCHEFFLER S, VARIA M. BooLigero: Improved sublinear zero knowledge proofs for Boolean circuits[C]. In: Financial Cryptography and Data Security—FC 2021, Part I. Springer Berlin Heidelberg, 2021: 476–496. [DOI: 10.1007/978-3-662-64322-8_23]
- [49] DE SAINT GUILHEM C D, ORSINI E, TANGUY T. Limbo: Efficient zero-knowledge MPCitH-based argu-

- ments[C]. In: Proceedings of the 2021 ACM SIGSAC Conference on Computer and Communications Security (CCS 2021). ACM, 2021: 3022–3036. [DOI: 10.1145/3460120.3484595]
- [50] BEN-SASSON E, CHIESA A, RIABZEV M, et al. Aurora: Transparent succinct arguments for R1CS[C]. In: Advances in Cryptology—EUROCRYPT 2019, Part I. Springer Cham, 2019: 103–128. [DOI: 10.1007/978-3-030-17653-2_4]
- [51] ZHANG Y, GENKIN D, KATZ J, et al. A zero-knowledge version of vSQL[J/OL]. IACR Cryptology ePrint Archive, 2017: 2017/1146. <https://eprint.iacr.org/2017/1146.pdf>
- [52] GROTH J. On the size of pairing-based non-interactive arguments[C]. In: Advances in Cryptology—EUROCRYPT 2016, Part II. Springer Berlin Heidelberg, 2016: 305–326. [DOI: 10.1007/978-3-662-49896-5_11]
- [53] HOFFMANN M, KLOOSS M, RUPP A. Efficient zero-knowledge arguments in the discrete log setting, revisited[C]. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS 2019). ACM, 2019: 2093–2110. [DOI: 10.1145/3319535.3354251]
- [54] DAZA V, RÀFOLS C, ZACHARAKIS A. Updateable inner product argument with logarithmic verifier and applications[C]. In: Public-Key Cryptography—PKC 2020, Part I. Springer Cham, 2020: 527–557. [DOI: 10.1007/978-3-030-45374-9_18]
- [55] NITUDESCU A. zk-SNARKs: A gentle introduction[EB/OL]. 2020. <https://www.di.ens.fr/~nitudesc/files/Survey-SNARKs.pdf>
- [56] MORAIS E, KOENS T, VAN WIJK C, et al. A survey on zero knowledge range proofs and applications[J]. SN Applied Sciences, 2019, 1(8): 946. [DOI: 10.1007/s42452-019-0989-z]
- [57] SUN X Q, YU F R, ZHANG P, et al. A survey on zero-knowledge proof in blockchain[J]. IEEE Network, 2021, 35(4): 198–205. [DOI: 10.1109/MNET.011.2000473]
- [58] BITANSKY N, CHIESA A, ISHAI Y, et al. Succinct non-interactive arguments via linear interactive proofs[C]. In: Theory of Cryptography—TCC 2013. Springer Berlin Heidelberg, 2013: 315–333. [DOI: 10.1007/978-3-642-36594-2_18]
- [59] ARORA S, LUND C, MOTWANI R, et al. Proof verification and the hardness of approximation problems[J]. Journal of the ACM, 1998, 45(3): 501–555. [DOI: 10.1145/278298.278306]
- [60] CORMODE G, MITZENMACHER M, THALER J. Practical verified computation with streaming interactive proofs[C]. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS 2012). ACM, 2012: 90–112. [DOI: 10.1145/2090236.2090245]
- [61] GOLDWASSER S, KALAI Y T, ROTHBLUM G N. Delegating computation: Interactive proofs for muggles[J]. Journal of the ACM, 2015, 62(4): 27:1–27:64. [DOI: 10.1145/2699436]
- [62] SETTY S T V, BRAUN B, VU V, et al. Resolving the conflict between generality and plausibility in verified computation[C]. In: Proceedings of the 8th ACM European Conference on Computer Systems (EuroSys 2013). ACM, 2013: 71–84. [DOI: 10.1145/2465351.2465359]
- [63] LUND C, FORTNOW L, KARLOFF H J, et al. Algebraic methods for interactive proof systems[J]. Journal of the ACM, 1992, 39(4): 859–868. [DOI: 10.1145/146585.146605]
- [64] KALAI Y T, RAZ R. Interactive PCP[C]. In: Automata, Languages and Programming—ICALP 2008, Part II. Springer Berlin Heidelberg, 2008: 536–547. [DOI: 10.1007/978-3-540-70583-3_44]
- [65] WAHBY R S, SETTY S T V, REN Z C, et al. Efficient RAM and control flow in verifiable outsourced computation[C]. In: Proceedings of 2015 Network and Distributed System Security Symposium (NDSS 2015). NDSS, 2015. [DOI: 10.14722/ndss.2015.23097]
- [66] BEN-SASSON E, BENTOV I, CHIESA A, et al. Computational integrity with a public random string from quasi-linear PCPs[C]. In: Advances in Cryptology—EUROCRYPT 2017, Part III. Springer Cham, 2017: 551–579. [DOI: 10.1007/978-3-319-56617-7_19]
- [67] REINGOLD O, ROTHBLUM G N, ROTHBLUM R D. Constant-round interactive proofs for delegating computation[J]. SIAM Journal on Computing, 2021, 50(3): STOC16-255-STOC16-340. [DOI: 10.1137/16M1096773]
- [68] YAO A C. Protocols for secure computations (extended abstract)[C]. In: Proceedings of 23rd Annual Symposium on Foundations of Computer Science (FOCS 1982). IEEE, 1982: 160–164. [DOI: 10.1109/SFCS.1982.38]
- [69] REED I S, SOLOMON G. Polynomial codes over certain finite fields[J]. Journal of the Society for Industrial and Applied Mathematics, 1960, 8(2): 300–304. [DOI: 10.1137/0108018]
- [70] BITANSKY N, CANETTI R, CHIESA A, et al. From extractable collision resistance to succinct non-interactive arguments of knowledge, and back again[C]. In: Proceedings of the 3rd Innovations in Theoretical Computer Science Conference (ITCS 2012). ACM, 2012: 326–349. [DOI: 10.1145/2090236.2090263]
- [71] PEDERSEN T P. Non-interactive and information-theoretic secure verifiable secret sharing[C]. In: Advances in Cryptology—CRYPTO '91. Springer Berlin Heidelberg, 1992: 129–140. [DOI: 10.1007/3-540-46766-1_9]

- [72] GOLDREICH O. The Foundations of Cryptography—Volume 1: Basic Techniques[M]. Cambridge, UK: Cambridge University Press, 2001.
- [73] BRASSARD G, CHAUM D, CRÉPEAU C. Minimum disclosure proofs of knowledge[J]. Journal of Computer and System Sciences, 1988, 37(2): 156–189. [DOI: 10.1016/0022-0000(88)90005-0]
- [74] BELLARE M, GOLDREICH O. On defining proofs of knowledge[C]. In: Advances in Cryptology—CRYPTO '92. Springer Berlin Heidelberg, 1993: 390–420. [DOI: 10.1007/3-540-48071-4_28]
- [75] GROTH J, ISHAI Y. Sub-linear zero-knowledge argument for correctness of a shuffle[C]. In: Advances in Cryptology—EUROCRYPT 2008. Springer Berlin Heidelberg, 2008: 379–396. [DOI: 10.1007/978-3-540-78967-3_22]
- [76] GOLDREICH O, HÄSTAD J. On the complexity of interactive proofs with bounded communication[J]. Information Processing Letters, 1998, 67(4): 205–214. [DOI: 10.1016/S0020-0190(98)00116-1]
- [77] CRAMER R, DAMGÅRD I, SCHOENMAKERS B. Proofs of partial knowledge and simplified design of witness hiding protocols[C]. In: Advances in Cryptology—CRYPTO '94. Springer Berlin Heidelberg, 1994: 174–187. [DOI: 10.1007/3-540-48658-5_19]
- [78] GOLDREICH O, OREN Y. Definitions and properties of zero-knowledge proof systems[J]. Journal of Cryptology, 1994, 7(1): 1–32. [DOI: 10.1007/BF00195207]
- [79] BEN-SASSON E, CHIESA A, GREEN M, et al. Secure sampling of public parameters for succinct zero knowledge proofs[C]. In: Proceedings of 2015 IEEE Symposium on Security and Privacy (S&P 2015). IEEE, 2015: 287–304. [DOI: 10.1109/SP.2015.25]
- [80] BOWE S, GABIZON A, GREEN M D. A multi-party protocol for constructing the public parameters of the Pinocchio zk-SNARK[C]. In: Financial Cryptography and Data Security—FC 2018. Springer Berlin Heidelberg, 2018: 64–77. [DOI: 10.1007/978-3-662-58820-8_5]
- [81] BEN-SASSON E, CHIESA A, GENKIN D, et al. SNARKs for C: Verifying program executions succinctly and in zero knowledge[C]. In: Advances in Cryptology—CRYPTO 2013, Part II. Springer Berlin Heidelberg, 2013: 90–108. [DOI: 10.1007/978-3-642-40084-1_6]
- [82] FEIGE U, LAPIDOT D, SHAMIR A. Multiple non-interactive zero knowledge proofs based on a single random string (extended abstract)[C]. In: Proceedings 31st Annual Symposium on Foundations of Computer Science (FOCS 1990). IEEE, 1990: 308–317. [DOI: 10.1109/FSCS.1990.89549]
- [83] BERNHARD D, PEREIRA O, WARINSCHI B. How not to prove yourself: Pitfalls of the Fiat-Shamir heuristic and applications to Helios[C]. In: Advances in Cryptology—ASIACRYPT 2012. Springer Berlin Heidelberg, 2012: 626–643. [DOI: 10.1007/978-3-642-34961-4_38]
- [84] BITANSKY N, DACHMAN-SOLED D, GARG S, et al. Why “Fiat-Shamir for proofs” lacks a proof[C]. In: Theory of Cryptography—TCC 2013. Springer Berlin Heidelberg, 2013: 182–201. [DOI: 10.1007/978-3-642-36594-2_11]
- [85] CHEN Y, LOMBARDI A, MA F, et al. Does Fiat-Shamir require a cryptographic hash function?[C]. In: Advances in Cryptology—CRYPTO 2021, Part IV. Springer Cham, 2021: 334–363. [DOI: 10.1007/978-3-030-84259-8_12]
- [86] SCHWARTZ J T. Fast probabilistic algorithms for verification of polynomial identities[J]. Journal of the ACM, 1980, 27(4): 701–717. [DOI: 10.1145/322217.322225]
- [87] ZIPPEL R. Probabilistic algorithms for sparse polynomials[C]. In: Symbolic and Algebraic Computation—EUROSAM 1979. Springer Berlin Heidelberg, 1979: 216–226. [DOI: 10.1007/3-540-09519-5_73]
- [88] MEIKLEJOHN S, ERWAY C C, KÜPCÜ A, et al. ZKPDL: A language-based system for efficient zero-knowledge proofs and electronic cash[C]. In: Proceedings of 19th USENIX Security Symposium. USENIX, 2010: 193–206. [DOI: 10.13140/RG.2.1.2367.9767]
- [89] VIRZA M. On Deploying Succinct Zero-knowledge Proofs[D/OL]. Cambridge, MA, USA: Department of Electrical Engineering and Computer Science, Massachusetts Institute of Technology, 2017.
<https://hdl.handle.net/1721.1/113986>
- [90] BEN-SASSON E, CHIESA A, SPOONER N. Interactive oracle proofs[C]. In: Theory of Cryptography—TCC 2016, Part II. Springer Berlin Heidelberg, 2016: 31–60. [DOI: 10.1007/978-3-662-53644-5_2]
- [91] GROTH J. Short pairing-based non-interactive zero-knowledge arguments[C]. In: Advances in Cryptology—ASIACRYPT 2010. Springer Berlin Heidelberg, 2010: 321–340. [DOI: 10.1007/978-3-642-17373-8_19]
- [92] MERKLE R C. A digital signature based on a conventional encryption function[C]. In: Advances in Cryptology—CRYPTO '87. Springer Berlin Heidelberg, 1988: 369–378. [DOI: 10.1007/3-540-48184-2_32]
- [93] VALIANT P. Incrementally verifiable computation or proofs of knowledge imply time/space efficiency[C]. In: Theory of Cryptography—TCC 2008. Springer Berlin Heidelberg, 2008: 1–18. [DOI: 10.1007/978-3-540-78524-8_1]

- [94] CRESCENZO G D, LIPMAA H. Succinct NP proofs from an extractability assumption[C]. In: Logic and Theory of Algorithms—CiE 2008. Springer Berlin Heidelberg, 2008: 175–185. [DOI: 10.1007/978-3-540-69407-6_21]
- [95] CHIESA A, YOGEV E. Subquadratic SNARGs in the random oracle model[C]. In: Advances in Cryptology—CRYPTO 2021, Part I. Springer Cham, 2021: 711–741. [DOI: 10.1007/978-3-030-84242-0_25]
- [96] CHIESA A, YOGEV E. Tight security bounds for Micali’s SNARGs[C]. In: Theory of Cryptography—TCC 2021, Part I. Springer Cham, 2021: 401–434. [DOI: 10.1007/978-3-030-90459-3_14]
- [97] ISHAI Y, KUSHILEVITZ E, OSTROVSKY R. Efficient arguments without short PCPs[C]. In: Twenty-second Annual IEEE Conference on Computational Complexity (CCC 2007). IEEE, 2007: 278–291. [DOI: 10.1109/CCC.2007.10]
- [98] ISHAI Y, KUSHILEVITZ E, OSTROVSKY R, et al. Zero-knowledge from secure multiparty computation[C]. In: Proceedings of the Thirty-ninth Annual ACM Symposium on Theory of Computing (STOC 2007). ACM, 2007: 21–30. [DOI: 10.1145/1250790.1250794]
- [99] HÅSTAD J, KHOT S. Query efficient PCPs with perfect completeness[J]. Theory of Computing, 2005, 1(1): 119–148. [DOI: 10.4086/toc.2005.v001a007]
- [100] BEN-SASSON E, CHIESA A, GENKIN D, et al. On the concrete efficiency of probabilistically-checkable proofs[C]. In: Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing (STOC 2013). ACM, 2013: 585–594. [DOI: 10.1145/2488608.2488681]
- [101] BEN-SASSON E, CHIESA A, GABIZON A, et al. Quasi-linear size zero knowledge from linear-algebraic PCPs[C]. In: Theory of Cryptography—TCC 2016, Part II. Springer Berlin Heidelberg, 2016: 33–64. [DOI: 10.1007/978-3-662-49099-0_2]
- [102] CRAMER R, DAMGÅRD I. Zero-knowledge proofs for finite field arithmetic; or: Can zero-knowledge be for free?[C]. In: Advances in Cryptology—CRYPTO ’98. Springer Berlin Heidelberg, 1998: 424–441. [DOI: 10.1007/BFb0055745]
- [103] BEN-SASSON E, SUDAN M. Short PCPs with polylog query complexity[J/OL]. SIAM Journal on Computing, 2008, 38(2): 551–607. [DOI: 10.1137/050646445]
- [104] LIPMAA H. Progression-free sets and sublinear pairing-based non-interactive zero-knowledge arguments[C]. In: Theory of Cryptography—TCC 2012. Springer Berlin Heidelberg, 2012: 169–189. [DOI: 10.1007/978-3-642-28914-9_10]
- [105] LIPMAA H. Succinct non-interactive zero knowledge arguments from span programs and linear error-correcting codes[C]. In: Advances in Cryptology—ASIACRYPT 2013, Part I. Springer Berlin Heidelberg, 2013: 41–60. [DOI: 10.1007/978-3-642-42033-7_3]
- [106] DANEZIS G, FOURNET C, GROTH J, et al. Square span programs with applications to succinct NIZK arguments[C]. In: Advances in Cryptology—ASIACRYPT 2014, Part I. Springer Berlin Heidelberg, 2014: 532–550. [DOI: 10.1007/978-3-662-45611-8_28]
- [107] KOSBA A E, PAPADOPOULOS D, PAPAMANTHOU C, et al. TRUESET: Faster verifiable set computations[C]. In: Proceedings of the 23rd USENIX Conference on Security Symposium (SEC ’14). USENIX, 2014: 765–780. [DOI: 10.5555/2671225.2671274]
- [108] BACKES M, BARBOSA M, FIORE D, et al. ADSNARK: Nearly practical and privacy-preserving proofs on authenticated data[C]. In: Proceedings of 2015 IEEE Symposium on Security and Privacy (S&P 2015). IEEE, 2015: 271–286. [DOI: 10.1109/SP.2015.24]
- [109] COSTELLO C, FOURNET C, HOWELL J, et al. Geppetto: Versatile verifiable computation[C]. In: Proceedings of 2015 IEEE Symposium on Security and Privacy (S&P 2015). IEEE, 2015: 253–270. [DOI: 10.1109/SP.2015.23]
- [110] GROTH J, MALLER M. Snarky signatures: Minimal signatures of knowledge from simulation-extractable SNARKs[C]. In: Advances in Cryptology—CRYPTO 2017, Part II. Springer Cham, 2017: 581–612. [DOI: 10.1007/978-3-319-63715-0_20]
- [111] BITANSKY N, CANETTI R, CHIESA A, et al. Recursive composition and bootstrapping for SNARKS and proof-carrying data[C]. In: Proceedings of the Forty-fifth Annual ACM Symposium on Theory of Computing (STOC 2013). ACM, 2013: 111–120. [DOI: 10.1145/2488608.2488623]
- [112] KARCHMER M, WIGDERSON A. On span programs[C]. In: Proceedings of the Eighth Annual Structure in Complexity Theory Conference (CoCo 1993). IEEE, 1993: 102–111. [DOI: 10.1109/SCT.1993.336536]
- [113] DAMGÅRD I. Towards practical public key systems secure against chosen ciphertext attacks[C]. In: Advances in Cryptology—CRYPTO ’91. Springer Berlin Heidelberg, 1992: 445–456. [DOI: 10.1007/3-540-46766-1_36]
- [114] BONEH D, BOYEN X. Short signatures without random oracles[C]. In: Advances in Cryptology—EUROCRYPT 2004. Springer Berlin Heidelberg, 2004: 56–73. [DOI: 10.1007/978-3-540-24676-3_4]
- [115] GROTH J, OSTROVSKY R, SAHAI A. Non-interactive zaps and new techniques for NIZK[C]. In: Advances in

- Cryptology—CRYPTO 2006. Springer Berlin Heidelberg, 2006: 97–111. [DOI: 10.1007/11818175_6]
- [116] BRAUN B, FELDMAN A J, REN Z C, et al. Verifying computations with state[C]. In: Proceedings of the Twenty-fourth ACM Symposium on Operating Systems Principles (SOSP 2013). ACM, 2013: 341–357. [DOI: 10.1145/2517349.2522733]
- [117] CHIESA A, TROMER E, VIRZA M. Cluster computing in zero knowledge[C]. In: Advances in Cryptology—EUROCRYPT 2015, Part II. Springer Berlin Heidelberg, 2015: 371–403. [DOI: 10.1007/978-3-662-46803-6_13]
- [118] SHOUP V. Lower bounds for discrete logarithms and related problems[C]. In: Advances in Cryptology—EUROCRYPT ’97. Springer Berlin Heidelberg, 1997: 256–266. [DOI: 10.1007/3-540-69053-0_18]
- [119] FUCHSBAUER G, KILTZ E, LOSS J. The algebraic group model and its applications[C]. In: Advances in Cryptology—CRYPTO 2018, Part II. Springer Cham, 2018: 33–62. [DOI: 10.1007/978-3-319-96881-0_2]
- [120] GABIZON A. AuroraLight: Improved prover efficiency and SRS size in a sonic-like system[J/OL]. IACR Cryptology ePrint Archive, 2019: 2019/601. <https://eprint.iacr.org/2019/601.pdf>
- [121] VON ZUR GATHEN J, GERHARD J. Modern Computer Algebra[M]. 3rd ed. Cambridge, UK: Cambridge University Press, 2013.
- [122] KATE A, ZAVERUCHA G M, GOLDBERG I. Constant-size commitments to polynomials and their applications[C]. In: Advances in Cryptology—ASIACRYPT 2010. Springer Berlin Heidelberg, 2010: 177–194. [DOI: 10.1007/978-3-642-17373-8_11]
- [123] BÜNZ B, FISCH B, SZEPENIEC A. Transparent SNARKs from DARK compilers[C]. In: Advances in Cryptology—EUROCRYPT 2020, Part I. Springer Cham, 2020: 677–706. [DOI: 10.1007/978-3-030-45721-1_24]
- [124] BENABBAS S, GENNARO R, VAHLIS Y. Verifiable delegation of computation over large datasets[C]. In: Advances in Cryptology—CRYPTO 2011. Springer Berlin Heidelberg, 2011: 111–131. [DOI: 10.1007/978-3-642-22792-9_7]
- [125] ZHANG Y, GENKIN D, KATZ J, et al. vSQL: Verifying arbitrary SQL queries over dynamic outsourced databases[C]. In: Proceedings of 2017 IEEE Symposium on Security and Privacy (S&P 2017). IEEE, 2017: 863–880. [DOI: 10.1109/SP.2017.43]
- [126] THALER J. Time-optimal interactive proofs for circuit evaluation[C]. In: Advances in Cryptology—CRYPTO 2013, Part II. Springer Berlin Heidelberg, 2013: 71–89. [DOI: 10.1007/978-3-642-40084-1_5]
- [127] WAHBY R S, JI Y, BLUMBERG A J, et al. Full accounting for verifiable outsourcing[C]. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS 2017). ACM, 2017: 2071–2086. [DOI: 10.1145/3133956.3133984]
- [128] BEN-OR M, GOLDREICH O, GOLDWASSER S, et al. Everything provable is provable in zero-knowledge[C]. In: Advances in Cryptology—CRYPTO ’88. Springer Berlin Heidelberg, 1990: 37–56. [DOI: 10.1007/0-387-34799-2_4]
- [129] CHIESA A, FORBES M A, SPOONER N. A zero knowledge sumcheck and its applications[J/OL]. IACR Cryptology ePrint Archive, 2017: 2017/305. <https://eprint.iacr.org/2017/305.pdf>
- [130] SCHNORR C. Efficient signature generation by smart cards[J]. Journal of Cryptology, 1991, 4(3): 161–174. [DOI: 10.1007/BF00196725]
- [131] MAURER U M. Unifying zero-knowledge proofs of knowledge[C]. In: Progress in Cryptology—AFRICACRYPT 2009. Springer Berlin Heidelberg, 2009: 272–286. [DOI: 10.1007/978-3-642-02384-2_17]
- [132] GROTH J. Linear algebra with sub-linear zero-knowledge arguments[C]. In: Advances in Cryptology—CRYPTO 2009. Springer Berlin Heidelberg, 2009: 192–208. [DOI: 10.1007/978-3-642-03356-8_12]
- [133] SEO J H. Round-efficient sub-linear zero-knowledge arguments for linear algebra[C]. In: Public Key Cryptography—PKC 2011. Springer Berlin Heidelberg, 2011: 387–402. [DOI: 10.1007/978-3-642-19379-8_24]
- [134] ZHANG Z Y, ZHOU Z B, LI W H, et al. An optimized inner product argument with more application scenarios[C]. In: Information and Communications Security—ICICS 2021, Part II. Springer Cham, 2021: 341–357. [DOI: 10.1007/978-3-030-88052-1_20]
- [135] BAYER S, GROTH J. Efficient zero-knowledge argument for correctness of a shuffle[C]. In: Advances in Cryptology—EUROCRYPT 2012. Springer Berlin Heidelberg, 2012: 263–280. [DOI: 10.1007/978-3-642-29011-4_17]
- [136] EVANS D, KOLESNIKOV V, ROSULEK M. A pragmatic introduction to secure multi-party computation[J]. Foundations and Trends in Privacy and Security, 2018, 2(2–3): 70–246. [DOI: 10.1561/3300000019]
- [137] GOLDREICH O, MICALI S, WIGDERSON A. How to play any mental game or A completeness theorem for protocols with honest majority[C]. In: Proceedings of the Nineteenth Annual ACM Symposium on Theory of Computing (STOC 1987). ACM, 1987: 218–229. [DOI: 10.1145/28395.28420]
- [138] JAWUREK M, KERSCHBAUM F, ORLANDI C. Zero-knowledge using garbled circuits: How to prove nonal-

- gebraic statements efficiently[C]. In: Proceedings of the 2013 ACM SIGSAC Conference on Computer Communications Security (CCS 2013). ACM, 2013: 955–966. [DOI: 10.1145/2508859.2516662]
- [139] FREDERIKSEN T K, NIELSEN J B, ORLANDI C. Privacy-free garbled circuits with applications to efficient zero-knowledge[C]. In: Advances in Cryptology—EUROCRYPT 2015, Part II. Springer Berlin Heidelberg, 2015: 191–219. [DOI: 10.1007/978-3-662-46803-6_7]
- [140] ZAHUR S, ROSULEK M, EVANS D. Two halves make a whole—Reducing data transfer in garbled circuits using half gates[C]. In: Advances in Cryptology—EUROCRYPT 2015, Part II. Springer Berlin Heidelberg, 2015: 220–250. [DOI: 10.1007/978-3-662-46803-6_8]
- [141] KONDI Y, PATRA A. Privacy-free garbled circuits for formulas: Size zero and information-theoretic[C]. In: Advances in Cryptology—CRYPTO 2017, Part I. Springer Cham, 2017: 188–222. [DOI: 10.1007/978-3-319-63688-7_7]
- [142] BEN-OR M, GOLDWASSER S, WIGDERSON A. Completeness theorems for non-cryptographic fault-tolerant distributed computation (extended abstract)[C]. In: Proceedings of the Twentieth Annual ACM Symposium on Theory of Computing (STOC 1988). ACM, 1988: 1–10. [DOI: 10.1145/62212.62213]
- [143] WANG X, RANELLUCCI S, KATZ J. Authenticated garbling and efficient maliciously secure two-party computation[C]. In: Proceedings of the 2017 ACM SIGSAC Conference on Computer and Communications Security (CCS 2017). ACM, 2017: 21–37. [DOI: 10.1145/3133956.3134053]
- [144] BONEH D, BOYLE E, CORRIGAN-GIBBS H, et al. Zero-knowledge proofs on secret-shared data via fully linear PCPs[C]. In: Advances in Cryptology—CRYPTO 2019, Part III. Springer Cham, 2019: 67–97. [DOI: 10.1007/978-3-030-26954-8_3]
- [145] BOYLE E, GILBOA N, ISHAI Y, et al. Practical fully secure three-party computation via sublinear distributed zero-knowledge proofs[C]. In: Proceedings of the 2019 ACM SIGSAC Conference on Computer and Communications Security (CCS 2019). ACM, 2019: 869–886. [DOI: 10.1145/3319535.3363227]
- [146] GOYAL V, SONG Y. Malicious security comes free in honest-majority MPC[J/OL]. IACR Cryptology ePrint Archive, 2020: 2020/134. <https://eprint.iacr.org/2020/134.pdf>
- [147] LIN S J, CHUNG W H, HAN Y S. Novel polynomial basis and its application to Reed-Solomon erasure codes[C]. In: Proceedings of 2014 IEEE 55th Annual Symposium on Foundations of Computer Science (FOCS 2014). IEEE, 2014: 316–325. [DOI: 10.1109/FOCS.2014.41]
- [148] GOEL A, GREEN M, HALL-ANDERSEN M, et al. Stacking Sigmas: A framework to compose-protocols for disjunctions[J/OL]. IACR Cryptology ePrint Archive, 2021: 2021/422. <https://eprint.iacr.org/2021/422.pdf>

作者信息



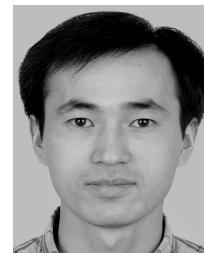
李威翰 (1997–), 山东烟台人, 博士生。主要研究领域为零知识证明和区块链。
leeweihan@buaa.edu.cn



张宗洋 (1984–), 山东菏泽人, 博士, 副教授。主要研究领域为密码学和区块链。
zongyangzhang@buaa.edu.cn



周子博 (1999–), 山东菏泽人, 博士生。主要研究领域为零知识证明和区块链。
zbzhou@buaa.edu.cn



邓燚 (1977–), 湖南望城人, 博士, 研究员。主要研究领域为零知识证明及其在金融科技中的应用。
deng@iie.ac.cn