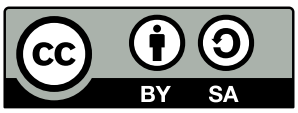
使用哈希函数构造密码学证明

Alessandro Chiesa

Eylon Yogev

译者：Kurt Pan

May 27, 2024



Contents

前言

简洁证明系统是一种非凡的工具。我们用一个简单的例子来简要解释它们是什么。假设爱丽丝有一个做了一些有趣事情的公共程序 P，比如会依次输出 π 的十进制数字，首先是 3，然后是 1，然后是 4，依此类推。爱丽丝在她的笔记本电脑上运行这个程序，程序输出了 π 的第一百万位小数，这正好是 5。她非常兴奋，于是告诉了她所有的朋友。爱丽丝的一个朋友鲍勃有点怀疑。他检查了程序 P 的代码，确认它确实是在正确地计算 π 的数字。但他担心爱丽丝只是猜测了第一百万位数字是什么而并没有实际运行程序。鲍勃决定自己验证一下这个计算。他设置好他的笔记本电脑运行 P ，并等待它重新确认爱丽丝的发现。

当鲍勃等待他的笔记本电脑从头开始重新运行整个计算时，他开始觉得这毫无意义。如果他们的朋友卡罗尔也不信任他们中的任何一个，她也将不得不重新运行整个计算以说服自己 P 确实正确运行了。如果大卫不信任他们三个人中的任何一个，他也将不得不重新运行整个计算。这是大量的重复劳动。他们所有人都在一遍又一遍地重做相同的计算。

有没有更好的方法？ 简洁证明系统提供了一个非凡的解决方案。它使得爱丽丝能够计算出程序 P 正确运行并且其输出如所声称的那样的证明。这个证明是**简洁的**，意味着它很短且验证速度很快。在实践中，简洁证明只有几千字节长，验证只需几毫秒。这里应当停下来对此感到惊叹：无论爱丽丝计算的是 π 的第一百万位还是第一十亿位数字，证明她计算是正确的证明总是只有几千字节长且只需几毫秒就能验证。

令人惊奇的是，这适用于任意程序 P，即使它需要爱丽丝提供辅助输入。爱丽丝可以运行程序并发布其输出以及程序正确运行的简洁证明。然后，只需几毫秒的工作，鲍勃、卡罗尔、大卫以及其他任何人都可以检查这个证明，并确信爱丽丝正确地运行了这个程序。他们无需重新运行计算。这个简洁证明甚至可以是**零知识**的，意思是证明不会透露爱丽丝的辅助输入的任何信息，但它仍然可以让所有人相信程序确实在这个输入上正确运行了。

近年来，简洁证明系统的研究已经发展成为一个庞大的研究领域，许多研究者都贡献了许多美妙的想法。推动这一快速进展的原因之一是这些工具的商业化应用，例如对去中心化系统扩容。另一个原因是这一领域为新的技术创新提供了肥沃的土壤。有多种构造简洁证明的方法，给研究者提供了许多探索的方向。同时，工程师们也有强烈的需求希望构造编程框架，使得非专家也能轻松使用这些工具。总体而言，这是一个由研究者和开发者组成的充满活力的社区，这一领域的工作既令人兴奋又充满乐趣。

本书涵盖了一种特定的简洁证明系统构造方法，称为**基于哈希的证明**。这些证明系统在概念上是最简单的，不需要复杂的数学知识，因此对广泛的读者群体都具有吸引力。此外，这些证明系统的安全性依赖于哈希函数的相对简单的性质。对于标准的密码学哈希函数，这些性质被认为即使在敌手拥有大规模量子计算机的情况下也能成立。因此，这些证明系统是**后量子**的：即使在构建出大规模量子计算机之后，它们仍将保持安全。

基于哈希的证明系统的理论已经相当成熟且被透彻理解。然而，直到现在，还没有一本书提供所有的详细信息。这就是本书的目的。本书开始部分精确定义了构成简洁证明系统的元素。接下来，本书转向从几个抽象的信息论对象构建简洁证明系统。一个广泛使用的例子是从一个重要对象——**交互式预言机证明（Interactive Oracle Proof，简称 IOP）**构造简洁证明系统。该构造完全依赖于哈希函数。书中逐步构造必要的工具，然后清晰描述了该构造及其安全性证明。

本书是对证明系统领域的重大贡献。它适合喜欢对材料进行清晰且精确处理的读者。我相信，一旦你读了这本书，你会想要了解更多关于这个领域的知识。本书是进入简洁证明世界的一个很好的起点，带你踏上一段有趣的旅程。

Dan Boneh
2024 年 4 月
斯坦福大学

序言

> “证明对我们的生活至关重要，而对于所有基本的事物，我们应该有所预料，对证明是什么的回答将永远是一个发展中的过程。“
>
> Silvio Micali, 计算可靠证明

本书讨论的是**密码学证明**，即用于证明和验证计算的正确执行的简洁且零知识的（稍后将详细介绍）协议。密码学证明是计算机科学中的一个基本对象，处于密码学和计算复杂性理论的交汇之处。由于实际应用的推动，在学术界和工业界受到了极大的关注。密码学证明结合了计算理论中一些最优美的思想和最强大的工具，是深奥的数学思想在新技术中起关键作用的一个显著例子。

本书提供了一个基于（理想）哈希函数构造的密码学证明的严格介绍，既是学生自学的教材，也是从业者的参考书。这包括了基于（理想）哈希函数的简洁非交互论证（SNARGs）的著名构造。例如，STARKs（可扩展透明知识论证）就是此类 SNARGs 的一个例子。

最新版本 本书的最新版本可以在以下网站找到：

`https://hash-based-snargs-book.github.io/`

该网站还链接了本书源代码的 Git 库，包括最新的更正和补充内容。我们非常欢迎对本书的评论（无论正面或负面的！），以及任何的更正或建议。您可以直接在 GitHub 上的库里提交 issues 或 PR，或者直接通过电子邮件联系我们。

动机

重新执行问题 计算机科学的一个基本目标是**计算完整性**：

一方如何验证另一方正确地执行了一个给定的计算？

实现这一目标的一种自然方法是**重新执行计算**，并检查这得到的结果是否与声称的结果相同。

例如，考虑一个计算任务：计算 1 亿以内的质数个数。Alice 通过质数计数算法（例如埃拉托色尼筛法）确定答案是 50,847,534，并发布计算声明“1 亿以内的质数个数是 50,847,534”。Bob 如何检查 Alice 的计算声明是否正确？ 一个选择是 Bob 重新运行 Alice 使用的算法（或其他质数计数算法），并检查 1 亿以内的质数是否确实是 50,847,534。

然而，在许多情况下，重新执行**不是一个选项**。

- 一方可能无法承担（或没有足够的资源）重新执行与计算声明相关的计算。例如，使用最先进的质数计数算法在 2022 年计算 10^{29} 以内的质数个数要花费 90 多核年和超过 1TB 的峰值内存使用；[1]这样的计算不易重新执行。
- 此外，计算可能涉及无法与他人共享的私有输入，因此无法重新执行计算。以下是一个涉及 RSA 密钥对的例子。回忆一下，RSA 密钥对包括一个包含两个质数（从合适的分布中抽样）的私钥和一个包含它们乘积的公钥；这样的密钥对用于加密方案和基于 RSA 假设的数字签名方案。考虑声明“我知道两个质数，它们的乘积是 RSA 公钥 N”。揭示这两个质数可以让其他人轻松验证该声明。然而，这样做会泄露与公钥对应的私匙。

于是我们面临重新执行问题：

如何在不重新执行的情况下确保计算完整性?

乍一看这似乎是不可能的。计算复杂性中的分层定理指出，一般来说，没有办法减少确定性计算输出所使用的时间或空间。[2] 在不重新执行的情况下确保计算完整性似乎是一个难以实现的目标。

避免重新执行 令人惊讶的是，通过要求提出计算声明的一方同时提供该声明的计算完整性的**密码学证明**，就可以避免重新执行。（特别地，这规避了前面提到的限制，以及在此尚未提及的其他限制。）非正式地，一方可以通过**证明者算法**生成密码学证明来认证计算的正确性。随后，任何人都可以通过**验证算法**检查密码学证明，而非重新执行原始计算。安全性适用于资源适当受限的敌手（稍后会详细介绍）。

但是，密码学证明的验证如何“优于”重新执行呢？ 这是由于密码学证明可以实现的优良性质。

- **简洁性。**验证算法可以比重新执行原始计算**快得多**。这种特性，称为简洁性，使得计算能力弱的一方可以验证计算复杂的计算的正确性。
- **零知识。**密码学证明不会泄露任何计算中使用的私有输入的信息。这种特性称为零知识，使得其他人可以验证涉及秘密的计算的正确性。（这些秘密只有执行计算并生成密码学证明的一方知道。）

满足（某种形式的）简洁性的密码学证明通常称为**简洁证明**，如果是非交互式的，则称为 **SNARG**。一个流行的概念是 zkSNARK，即**零知识简洁非交互知识论证**。这些概念将在本书中引入并定义。

应用 密码学证明通常用于学术研究，设计密码学原语、协议和系统。更令人兴奋的是，密码学证明已经在学术界之外找到了应用：它们在许多无法重新执行的场景中得到了广泛的实际应用。截至写作时，它们是应用密码学中最热门的话题之一。

例如，密码学证明在设计安全分布式系统（如区块链）中起着关键作用。它们被用于实现可扩展性和隐私的目标，简要介绍如下。

- **可扩展性。**简洁性性质通过将昂贵的计算移出网络节点，从而增加对等网络中的交易吞吐量。

  智能合约系统（例如，以太坊）中的计算既慢又昂贵：每个计算步骤都由网络中的每个节点重新执行，因此网络受限于最慢的节点，用户为每个计算步骤支付高昂的价格。

  一类称为“基于证明的 rollups”的架构将计算移出链外，用户将其计算请求发送给聚合者，聚合者定期生成关于用户计算批次的密码学证明；然后智能合约系统验证密码学证明并授权一个

[1]参见 https://oeis.org/A006880 和 https://www.mersenneforum.org/showthread.php?t=20473

[2]例如，确定性多带图灵机分层定理指出，对于每个时间可构造函数 $t(n)$，复杂性类 DTIME$(t(n))$ 严格大于复杂性类 DTIME$(o(\frac{t(n)}{\log t(n)}))$。换句话说，有些计算任务的输出可以在时间 $t(n)$ 内确定，但不能在时间 $o(\frac{t(n)}{\log t(n)})$ 内确定。其他资源和计算模型也存在其他分层定理。

反映批次中所有计算的状态转换。简洁性确保检查密码学证明比检查它所证明的计算便宜得多。因此，让网络中的每个节点仅检查密码学证明要便宜得多，并且使系统能够扩展到更多的用户和更大的计算。

- **隐私。**零知识性质允许设计具有强用户隐私的对等网络。非正式地，零知识性质使得网络中的一方能够广播关于涉及其私有输入的计算的密码学证明。例如，计算可能授权从一方向另一方转移数字资产，但不透露这些方的身份或数字资产的类型和数量。这种隐私保护功能对密码货币和其他去中心化金融应用至关重要，因为交易通常涉及高度敏感的信息，不应向整个网络公开。

这些及其他应用在学术界、工业界和政府的多个社区中引起了对密码学证明的极大兴趣。这种更广泛的兴趣需要关于密码学证明的材料更易于访问和系统化，以有效地满足超出少数专家的受众的需求。