密码学复杂性注记

A note on the complexity of cryptgraphy*

Gilles Brassard

翻译：李晓峰 (cy_lxf@163.com)‡

V2.0

2024 年 5 月 13 日

摘要

Diffie 和 Hellman 提出了基于单向函数的密码系统, 针对这种系统，我们给出证据，表明证明这类系统计算安全性是困难的，对于密码分析工作，证明这个工作是”NP 完全”(NP-completeness) 的，意味着 NP=CoNP。[1]

I 引言

Diffie 和 Hellman [1] 已经提出在有限域中，指数函数在密码学的应用。这个提议是基于一个猜想，即对数函数是不可计算的。下一个最好的事情是证明这个猜想，证明对数是 **NP** 困难的，在某种意义上，有一个图灵机使用它作为预言机 (oracle)，这个图灵机可以在多项式时间内接受某个 NP-complete 集合[2]。如果是这样的话，我们就会知道，在普遍接受的 $\mathbf{P} \neq \mathbf{NP}$ 的假设下，对数计算确实是困难的 [2，第 10 章]。

*BRASSARD G. A note on the complexity of cryptography (Corresp.)[J/OL]. IEEE Transactions on Information Theory, 1979, 25(2): 232-233. http://dx.doi.org/10.1109/tit.1979.1056010. DOI:10.1109/tit.1979.1056010.

†译者目前为北京联合大学智慧城市学院信息安全老师。

‡译文来自于经典文献翻译项目 https://gitee.com/uisu/InfSecClaT, 欢迎大家加入经典翻译项目，为更多的人能够获取这些经典文献所传递信息做一点贡献。

[1]译者注：这就是在说，单向函数的逆函数的计算难度与 NP-C 计算难度等价，如果能证明了单向函数的逆运算是 NP 问题，也就是证明了 NP=CoNP，而这个问题任然是个未解决的公开问题。

[2]译者注：这意味着对数问题可以 P 规约到 NPC 中的某一个问题，也就是说，对数问题是一个 NP 问题。

这里需要提醒一下。传统的复杂性处理最坏情况下的行为：如果对于计算对数函数的任何算法 A 和任何多项式 P，至少有一个输入 x，A 在该输入 x 上花费超过了 P(n) 步，其中 n 是 x 的长度，则称对数函数是不可计算的。这不是密码学所需要的概念：因为，这将是一个糟糕的密码系统，这个系统允许敌人轻松破译除少数密码外的所有密码[3]。然而，目前的技术甚至难以证明对数计算在最坏情况下的困难性，更不用说证明 Diffie-Hellman 提出的密码系统的计算安全性了。

一个非确定性图灵机 M，如果对于任何输入 x，至少有一条路径的计算会终止，并且每个终止路径在输出带上都以 $f(x)$ 终止，我们称此图灵机 M 计算函数 f。同样，对于任意输入 x，当且仅当 $x \in S$，这里存在一条计算路径终止 (即接受) 时，M 接受集合 S。一个重要的观察是，使用这样的机器可以在多项式时间内计算有限域对数。作为 Miller[3] 定理的应用，对数函数 (定义如下) 的投影 P_{log} 属于 $NP \cap CoNP$。此外，$P_{log} \in P$ 当且仅当对数可以在确定的多项式时间内计算出来。因此，对 Diffie 和 Hellman 猜想的证明意味着 P_{log} 见证 $P \overset{\subset}{\neq} NP \cap CoNP$。此外，如果对数是 NP 困难的，我们可以得出 $NP = CoNP$ 的结论，因为 P_{log} 既是 NP 完全的，又是 CoNP 的成员。下面是更多细节。由于这两个结果 ($P \overset{\subset}{\neq} NP \cap CoNP$ 和 $NP = CoNP$) 都被认为是错误的或很难建立的，因此我们得出结论，对于密码系统的安全性也是如此。

II 构造 (construction)

为了使本节独立，我们将不明确地使用 Miller 的结果 [3]。然而，所使用的技术将与他的相似。我们还将简略地证明对数可以用不确定图灵机在多项式时间内计算出来。

定义对数函数 $log(p, r, n)$ 如下: 如果 p 是素数，r 是 p 的本元根 [4，第 4.5.41 节][4]，且 $0 < n < p$，则 $log(p, r, n)$ 是唯一的 m，使得 $0 < m < p$ 且 $r^m = n \pmod p$，否则 $log(p, r, n)$ 为零。[5] 定义投影 P_{log} 为

$$\{< p, r, n, t > | log(p, r, n) > t\}$$

[3]译者注：因为复杂性是只要有一种情况，计算超过 P(n) 就是不可计算，其考量的是最差情况，而密码中的要求并不是这样，他是要求在符合要求的安全参数选择下，都是不可计算的。

[4]译者注：r 的幂能生成 [1,p-1] 残余类，或 $r^{\phi(p)} = 1 \pmod p \Rightarrow r^{p-1} = 1 \pmod p$

[5]译者注：$m = log(p, r, n)$，即，$r^m = n \pmod p$

，其中 $< p, r, n, t >$ 是 $< p, < r, < n, t >>>$ 的缩写,$< ., . >$ 是一个合适的配对函数。

$P_{log} \in NP$: p 是素数，r 是 p 的本原根当且仅当 $r^{p-1} = 1 \pmod{p}$，且对于每一个 q 是 $p-1$ 的素数因子，$r^{(p-1)/q} \neq 1 \pmod{p}$[4，第 4.5.4 节]。这些条件可以在不确定的多项式时间内通过猜测 $p-1$ 的素数分解以及每个因子的素性证明 [5]，并使用分治策略来计算以 p 为模的指数，来检验 [4，第 4.6.3 节]。一旦知道 p 是素数，r 是原始根，假设 $0 < n < p$，唯一的 m,$0 < m < p$，使得 $r^m = n \pmod{p}$，可以猜出来。如果 $m > t$，$< p, r, n, t >$ 在 P_{log} 中。

$P_{log} \in CoNP$: $< p, r, n, t >$ 不在 P_{log} 中，当且仅当 $log(p, r, n) \leq t$。当且仅当存在 i 和 $j(0 < i < j < p)$, 使得 $r^i = r^j \pmod{p}$，存在 $m \leq t$, 使得 $r^m = n \pmod{p}$，或者不存在 $0 < n < p$ 的情况。这些条件可以在多项式时间内进行不确定性的检验。

如果 $P_{log} \in P$，则对数可以在多项式时间内通过以下分治算法确定地计算出来:

```
function log(p,r,n)
  i :=0
  j :=p−1
  while i<j do
      /*i<=log<=j*/
      k := ⌊(i+j)/2⌋
      if <p,r,n,k> ∈ P_log then
           i :=k+1
      else
           j :=k
      fi
  od
return i
end log
```

最后，如果对数是 NP 困难的，那么 P_{log} 也是 NP 困难的，因为上面的算法表明，在给定 P_{log} 的情况下，可以有效地计算对数。因此，P_{log} 的 Oracle 可以用来模拟对数的 Oracle, 在 NP 中，是 NP 完全的。然而，$P_{log} \in CoNP$。我们得出 $NP = CoNP$ 的结论，否则就不可能存在 NPC

补集属于 NP。我们在附录中给出了这一说法的证明。

III 一般化 (generalization)

以上的推理可以推广到基于单向函数的其他密码系统。如 [1] 中定义的单向函数是一个函数 f，对于 f 的定义域中的任何参数 x，计算相应值 $f(x)$ 是容易的，但是对于 f 的值域中的几乎所有 y 来说，对于任何合适的参数 x, 解方程 y = f(x) 是计算上不可行的。到目前为止，我们已经使用了由 Diffie-Hellman 提出的单向函数的候选者，即有限域中的指数运算。

结果表明，如果满足一些限制条件的任意函数 f 可以被证明是单向的，则 $P \not\supseteq NP \cap CoNP$ 。类似地，如果通过证明 f 的逆是 NP-hard 来证明 f 是单向的，那么 $NP = CoNP$。限制条件是，从 NP 域到 $CoNP$ 值域 f 是一对一的，并且存在一个多项式 P，使得对于 f 值域内的任意 x, $|x| \leq P(|f(x)|)$，其中 $|x|$ 表示 x 的二进制表示的长度。

这就引出了最后一个例子。Rivest、Shamir 和 Adleman[6] 基于函数乘法在质数范围内是单向的这一信念，提出了一种公钥密码系统。用更传统的术语来说，两个素数相乘是很容易计算的，但我们还不知道如何有效地找到一个大数的素数因数，即使我们预先知道它是两个素数的乘积。这里再次证明，他们的密码系统的计算安全性证明将意味着 $P \not\supseteq NP \cap CoNP$ ，而密码分析工作的 NP-hardness 证明将意味着 $NP = CoNP$ 。对于这些结果的直接证明，模拟第 2 节用 S 替换 P_{log} 的证明，我们定义

$$\{< x, y > | x\text{有比 y 小的素数因子}\}.$$

如第 2 节所述，证明了 $S \in NP \cap CoNP$ 和 $S \in P$，且仅当素数分解是可计算的。

承认 (acknowledgment)

Leonard Adleman、Ronald rives 和 Gary Miller 也独立发现了类似的结果 [7]。

附录 (appendix)

我们将证明如果一个集合 A 是 $NP-C$ 的，并且如果是 $A \in CoNP$，那么 $NP = CoNP$。这对于 Karp 的 NP 完备性概念 [8] 来说是显而易见的，但需要对 Cook 的概念 [9] 进行证明，我们在本文中一直使用了 Cook 的概念 [9]。设 $M+$ 和 $M-$ 分别为接受 A 及其补的不确定多项式有界图灵机。

对于任何 $L \in NP$，存在一个确定性多项式图灵机 M，给定 A 的一个神谕 (Oracle),M 接受 L。考虑不确定性图灵机 M'，对于任何输入，它都是确定性地一步一步地模拟 M，除非 M 向 oracle 提出问题。在这种情况下，M' 猜测答案应该是“是”还是“否”，并根据这个猜测不确定地模拟 M+ 或 M-。如果 M' 达到 M+ 或 M-的接受状态，它将知道它在正确的轨道上，并将在适当的 Oracle 回答状态下恢复 M 的确定性模拟。如果 M' 达到 M+ 或 M-的拒绝状态，它将什么都不知道，并立即拒绝原始输入。最后，当 M 达到最终状态时，M' 接受，当且仅当 M 拒绝。

很明显，M' 在多项式时间内运行，并且它接受 L 的补，因此 $L \in CoNP$。这证明了 $NP \subseteq CoNP$。现在，给定任何 $L \in CoNP$，它的补是在 NP 中，根据 $CoNP$ 的定义，因此在 $CoNP$ 中，因为 $NP \subseteq CoNP$。$L \in NP$ 又是由 $CoNP$ 定义的。这证明了 $CoNP \subseteq NP$，完成了 $NP = CoNP$ 的证明。

参考文献

[1]W. Diffie and M. E. Hellman, “New directions in cryptography,” IEEE Tram. Inform. Theoty, vol. IT-22, pp. 644654, Nov. 1976.

[2]A. V. Aho, J. E. Hopcroft, and J. D. Ullman, The Design and Analysis of Computer Algorithms. Reading, MA: Addison-Wesley, 1974.

[3]G. L. Miller, “Riemann’s hypothesis and tests for primality,” J. Compur. Syst. Sci., vol. 13, pp. 300-317, 1976.

[4]D. E. Knuth, The Art of Computer Programming, vol. 2, Semi-Numerical Algorithms. Reading, MA: Addison-Wesley, 1969.

[5]V. Pratt, “Every prime has a succinct certificate,” SIAM J. Corn@., vol. 4, pp. 214-220, 1975.

[6]R. Rivest, A. Shamir, and L. Adleman, “A method of obtaining digital signatures and public-key cryptosystem,” Commun Am. Conput.

Mach., vol. 21, pp. 120-126, Feb. 1978.

[7]L. Adleman, Private communication, 1978.

[8]R. M. Karp, "Reducibility among combinatorial problems,"in Complexity of Compurer Computations, R. E. Miller and J. W. Thatcher, Eds. NY: Plenum, 1972.

[9]S. A. Cook, "The complexity of theorem proving procedures," in Proc. 3rd Ann. ACM Symp. Theory of Computing, 1971, pp. 151-158.