### 信息安全数学基础探究报告

学号：2113662 姓名：张丛

### 大整数分解

**大整数分解：**

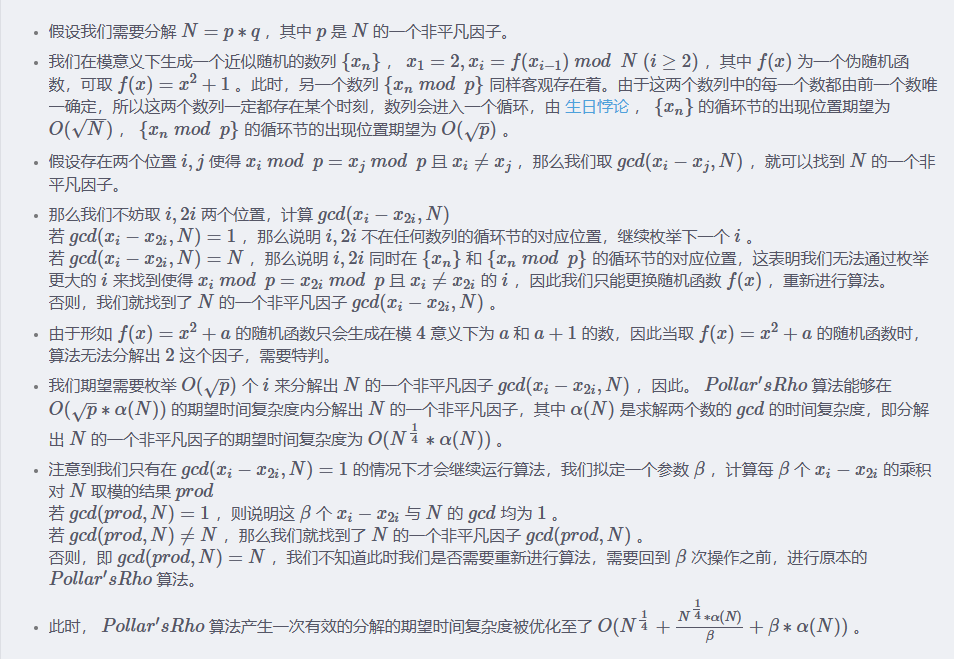
给定一个大整数N，它是两个大素数的乘积，但其因子p和q未知，我们将寻找p和q，使其满足N=p·q的问题叫做大整数分解问题。

大整数分解在密码学中扮演着关键的角色。它的困难性被广泛应用于加密算法、密钥交换协议和数字签名方案的安全性保证。通过困难的大整数分解问题，可以确保加密和通信的保密性和完整性。

**Pollard Rho算法**

算法原理：

生成两个整数a和b，计算p=gcd(a-b,n)，直到p不为1或者a,b出现循环为止；若p=n，则p为质数，否则p为n的一个约数。选取一个小的随机数x1，迭代生成xi=xi-1 ^2+k，一般取k=1，若序列出现循环则退出。计算p=gcd(xi-1 - xi,n)，若p=1，返回上一步，直到p>1为止。若p=n，则n为素数，否则p为n的一个约数并递归分解p和n/p.



**GNFS算法**：

GNFS（General Number Field Sieve）算法是一种用于分解大质数的算法，它是目前已知用于分解较大质数（大于100位）最高效的算法之一。

GNFS算法基于数学原理，通过在有限域（例如有限域上的方程组）中寻找一个适当的解，可以将这个大质数分解为两个较小的质数的乘积。它包含三个主要步骤：线性筛选步骤、平滑步骤和线性代数步骤。

算法思想:

GNFS算法的关键是在一个适当的整数域中进行计算和筛选。我们可以将整数域想象成一个大箱子，里面装满了不同大小的数字。首先，我们选择一个平方根接近整数的数作为起点，然后进行一系列计算和操作，构建出与待分解的数相关的关系方程。接下来使用一种筛选和合并的方法，在整数域中找到满足关系方程的候选整数。我们可以将这个过程想象成在大箱子中筛选和挑选符合一定规则的数字。这些候选整数具有特殊的性质，使得它们可能是待分解的大整数的因子。然后，找到这些候选整数与待分解的数之间的最大公约数。如果我们找到了一个非平凡（非自身或1）的最大公约数，那么这个最大公约数就是我们要找的一个质因数。最后，我们将找到的质因数与原来的大整数进行除法运算，得到另一个整数。然后，我们继续使用相同的方法，在这个新的整数上进行分解，直到无法再继续分解为止。这样，我们就可以逐步找到原始大整数的所有质因数。

总的来说，GNFS算法的思想就是通过在整数域中构建关系方程，筛选候选整数，求解最大公约数，逐步找到大整数的质因数。

算法特点:

高效性。相比传统的试除法和Pollard-Rho算法，GNFS能够在合理的时间内处理大整数。

综合性。基于数论的原理和方法，利用整数域的特性进行多阶段分解，综合了多种算法技巧。

### 离散对数问题

#### **离散对数知识：**

设p为素数，g为p的原根（即g为循环群Z\_p的生成元。对于任意的y∈Z\_p，存在唯一的x（1≤x<p-1）使得g^x≡y(mod p)，则x为在模p下以g为底y的离散对数。

#### **有限域上的离散对数问题：**

设p，q为两个素数，G={gi|0≤i≤q-1，g∈Z\_p }为阶为q的有限域Z\_p上的乘法群。给定一个元素y∈G，找到了一个整数x∈Z\_q，使得y=g^x的问题，称为有限域上的离散对数问题。

#### **两种问题的形式：**

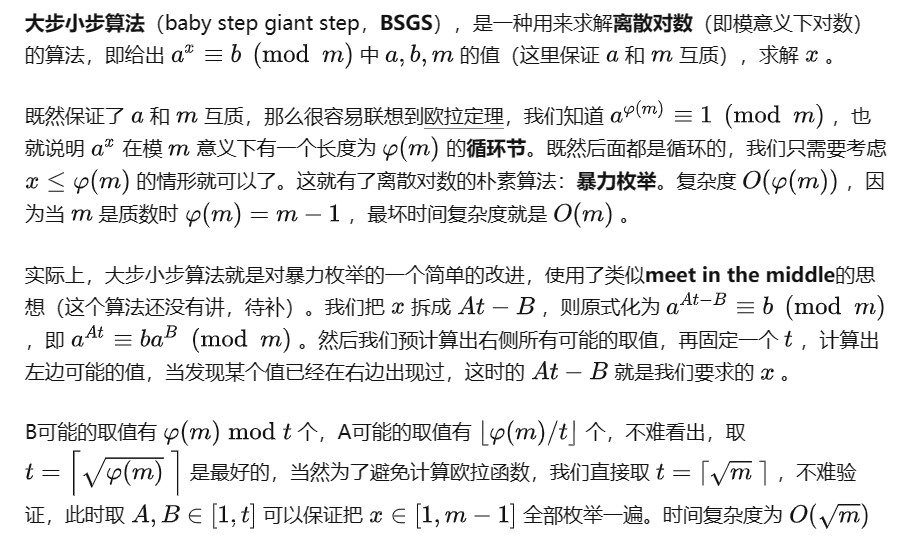
1. 设p为素数，g\_1,g\_2…g\_s为p的s个无关的原根。对于任意的与p互素的整数a可以表示为a≡g\_1^α\_1g\_2^α\_2……g\_s^α\_s(mod p)，则(α\_1,α\_2……α\_s)为a在模p下以g\_1,g\_2…g\_s为底的表示。
2. 设p,q为两个素数，E\_p (a,b):y^2≡x^3+ax+b(mod p)是有限域Z\_q上的椭圆曲线，G是E\_p(a,b)上所有点构成的群的一个循环子群，阶为q，P为G的生成元。给定G中任意点Q，找到一个整数x∈Z\_q，使得Q=xP的问题称为有限域椭圆曲线群上的离散对数问题。

#### **DH问题：**

设p为素数，g为p的原根。任意a，b∈Z\_q，可以表示为a≡g^x (mod p)，b≡g^y (mod p)，x，y均未知，找到一个c使其满足c≡g^xy (mod p)即为Diffie-Hellman问题。

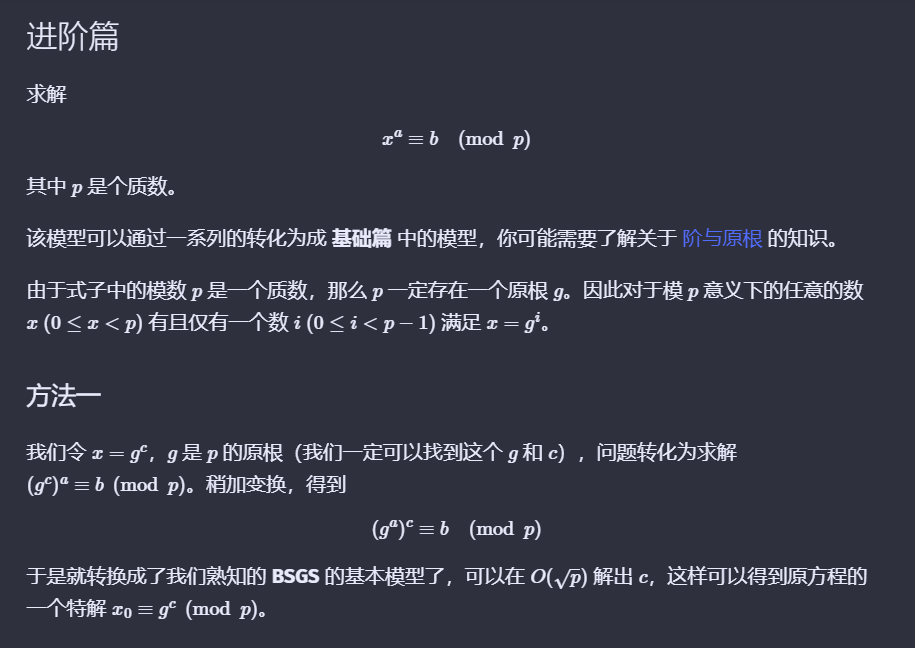
#### **离散对数问题的求解算法**

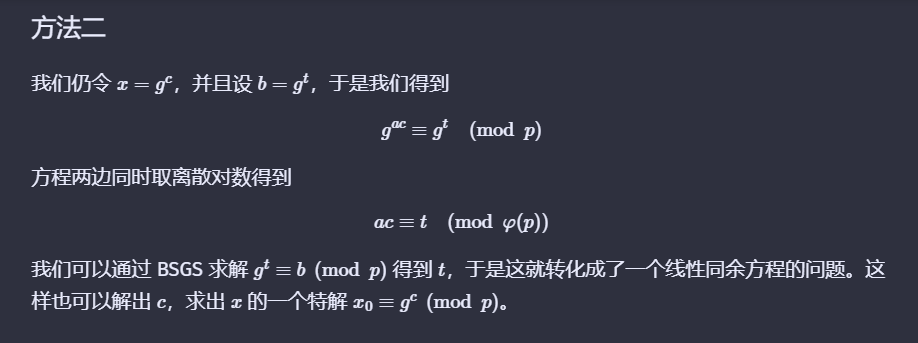
1、商克法：

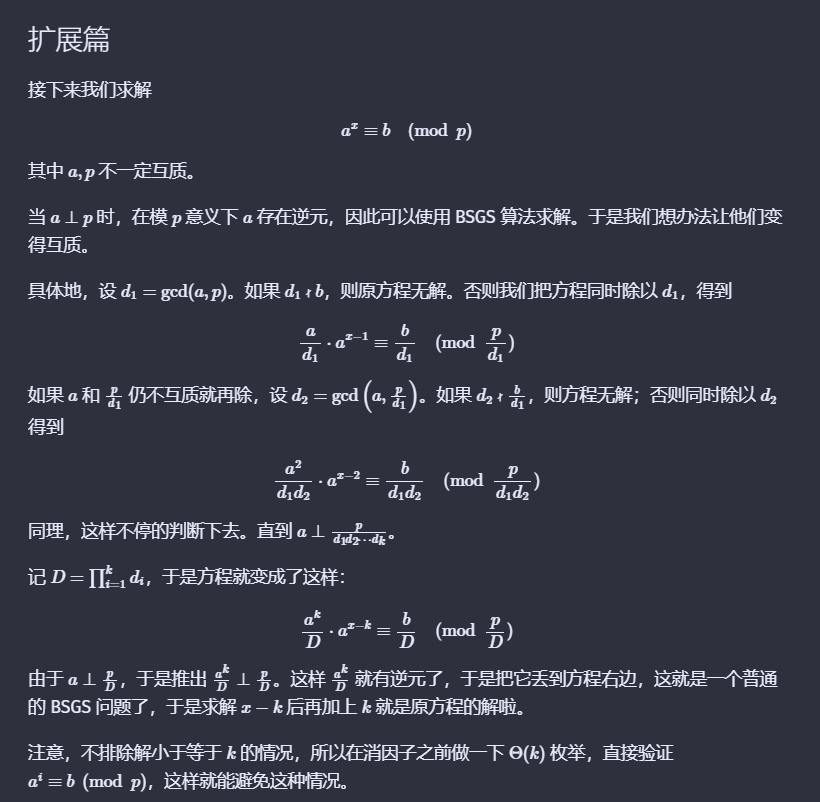


利用了中间相遇攻击的思想

即在小步造表和大步查表过程中发现一组碰撞，也就得到了对应的A和B



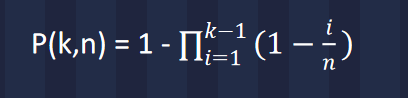


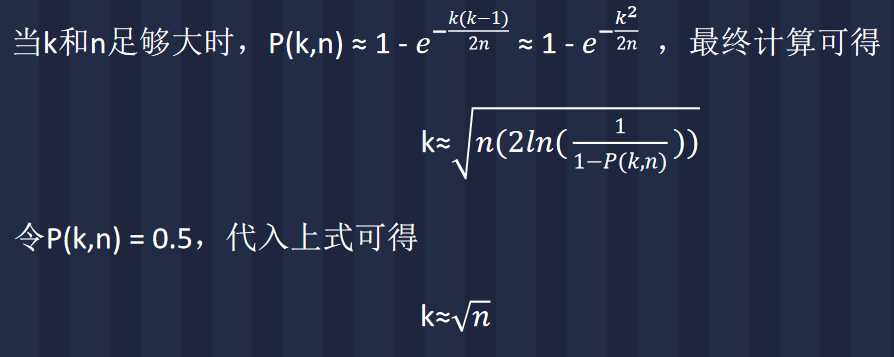


1. Pollard ρ算法：

理论前提：生日悖论（推广）

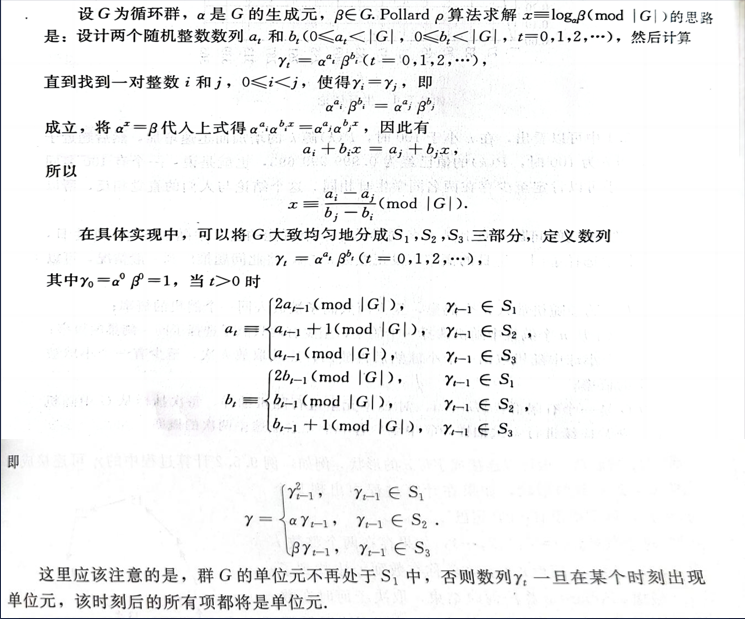
生日悖论推广到一般情况的一种表述：设G是一个有限群，|G|=n，对G中的元素进行随机抽样，每次抽样从G中随机选择一个元素，选择进行k次抽样，G中至少有一个元素被选中两次的概率：





综上，在有限群G中，对G中元素进行随机抽样，每次抽样从G中随机选择一个元素，如果n足够大，抽样次数大于√n，基本可以使G中至少有一个元素被选中两次的概率超过0.5

算法流程：

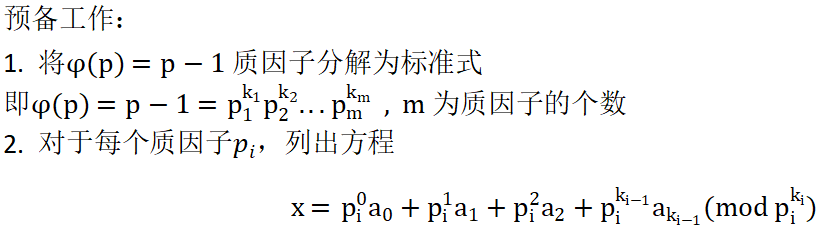


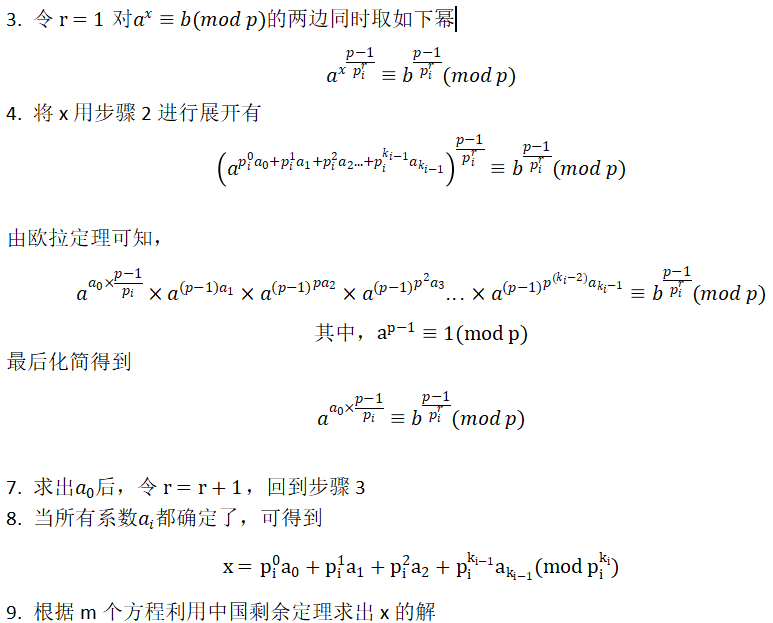
算法改进：

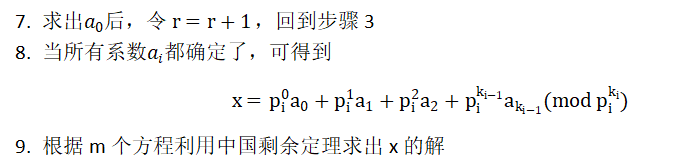
改进主要是使用了一个碰撞流，如果γ\_μ+λ=γ\_μ是第一个碰撞，那么可以有一个迭代式：对于所有非负整数s都有γ\_μ+λ+s=γ\_μ+s。这便可以帮助我们生成连续的碰撞流。上式易证，可以用a\_t，b\_t结合γ\_t来证明此迭代式的正确性。若定义s = λ - μ，则有碰撞γ\_2λ=γ\_λ。我们通过计算两个数列γ\_t,γ\_2t(t=0,1,2,...)便可以寻找碰撞γ\_2λ=γ\_λ。

1. Pohlig-Hellman算法：

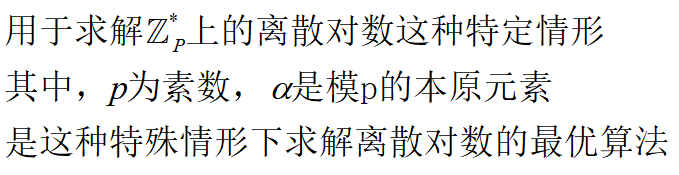
Pohlig-Hellman算法基于中国剩余定理，它只适用于求解合数阶循环群上且模数为素数的离散对数问题，而且必须已知循环群的阶的所有素因子

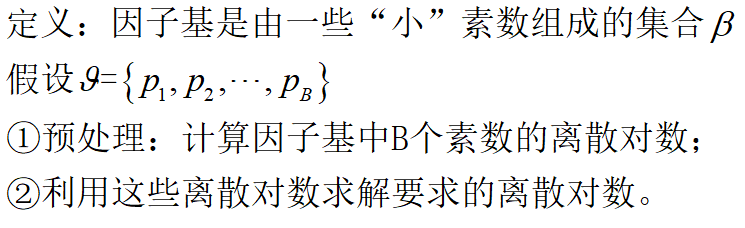


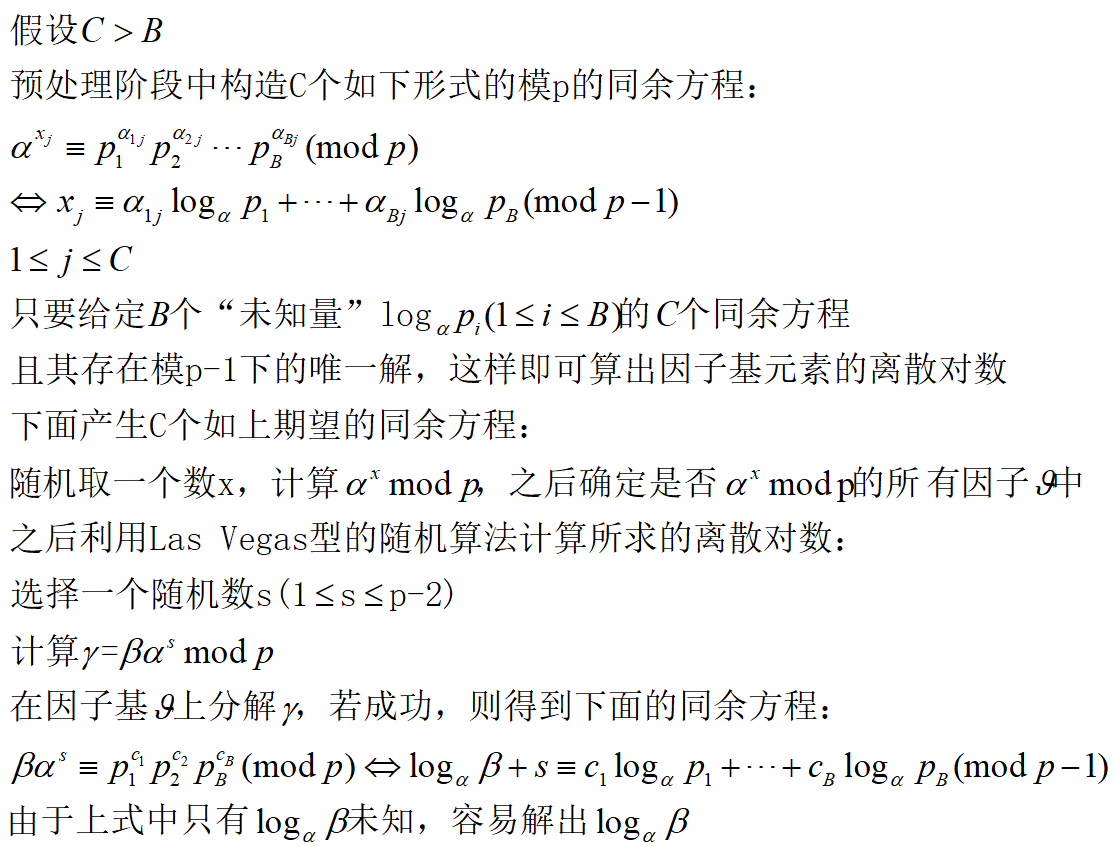




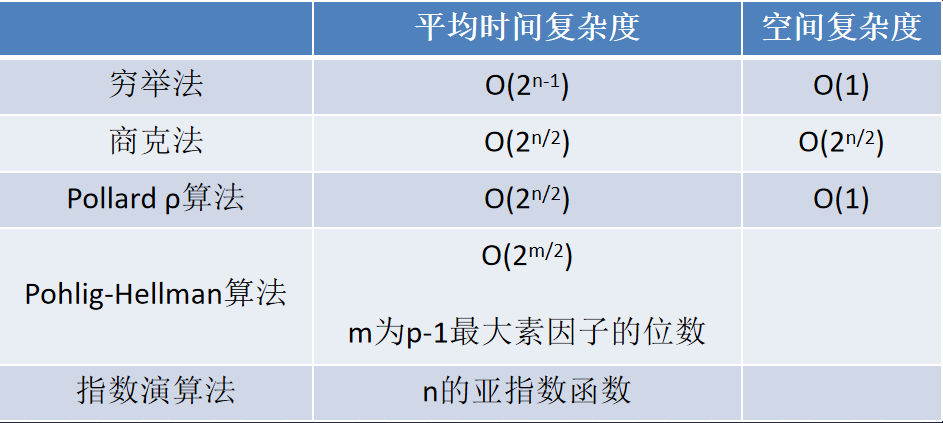
4、指数演算法：







#### **常见离散对数求解算法性能对比：**



#### **密码学中的应用：**

离散对数是密码学中的一种重要数论问题，它在很多公钥密码系统中应用广泛，其中最著名的就是RSA算法和椭圆曲线密码。

RSA算法使用一个公开的指数和幂模运算来进行加密和解密操作，并基于离散对数问题，使得只有私钥持有者能够有效地执行这些操作。具体来说，RSA算法构建在大素数分解和欧拉函数的难题上，利用了模幂运算和扩展欧几里得算法等数学工具。

椭圆曲线密码则利用了在定义在椭圆曲线上的离散对数问题。椭圆曲线密码相比于传统的RSA、DSA或Elgamal等加密方法，具有更高的安全性、小体积和较低算法运算成本的优点，因此在物联网、移动设备安全和数字认证领域得到了广泛应用。

总之，离散对数问题已经成为现代密码学中必不可少的数学基础之一，其在各种加密算法和密钥交换协议中扮演着关键的角色。

### 零知识证明

#### **零知识证明(Zero-Knowledge Proof, ZKP):**

指的是证明者能够在不向验证者提供任何有用的信息的情况下，使验证者相信某个论断是正确的，它实质上是一种涉及两方或更多方的协议，即两方或更多方完成一项任务所需采取的一系列步骤。证明者向验证者证明并使其相信自己知道或拥有某一消息，但证明过程不能向验证者泄漏任何关于被证明消息的信息。

#### **举例来通俗理解：**

1.阿里巴巴与四十大盗

有一天，阿里巴巴被强盗抓住了，强盗向他索要开启山洞大门的咒语。

但此时阿里巴巴面临一个两难的问题，如果把密码告诉强盗，自己就没有利用价值了，最后肯定会被杀。

如果不告诉强盗咒语，强盗以为自己不知道咒语，自己还是会被杀。

怎么能做到让他们相信自己确实知道咒语，但是还不能让他们知道咒语是什么。

这确实是一个很难的问题，但是阿里巴巴想出了一个好办法。

他对强盗头领提议说，你们离我30米远，然后用弓箭指着我，当你举起双手后，我就念咒语开启山洞大门；当你把双手放下后，我就念咒语关上山洞大门，如果我要是逃跑，你们就用弓箭射死我。

对于强盗头领来说，这显然是个好主意，于是照办。

强盗头领先是举起双手，看到阿里巴巴动了动嘴皮子，门就开了，然后放下双手，阿里巴巴又动了动嘴皮子，门就关了。

显然，强盗相信了阿里巴巴。

这样，阿里巴巴在没有告诉强盗头领开门咒语的情况下，又向强盗证明了，自己是知道开门咒语的。

2.地图着色问题

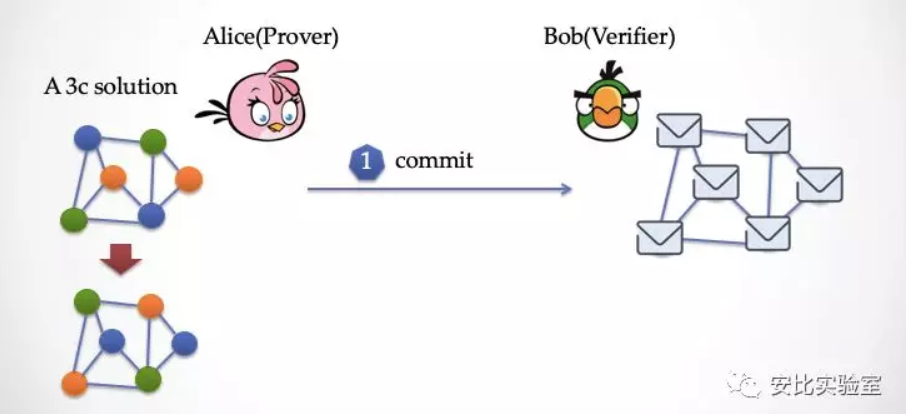
如何用三种颜色染色一个地图，保证任意两个相邻的地区都是不同的颜色。我们把这个「地图三染色问题」转变成一个「连通图的顶点三染色问题」。假设每个地区都有一个首府（节点），然后把相邻的节点连接起来，这样地图染色问题可以变成一个连通图的顶点染色问题。

下面我们设计一个交互协议：

「证明者」Alice

「验证者」 Bob

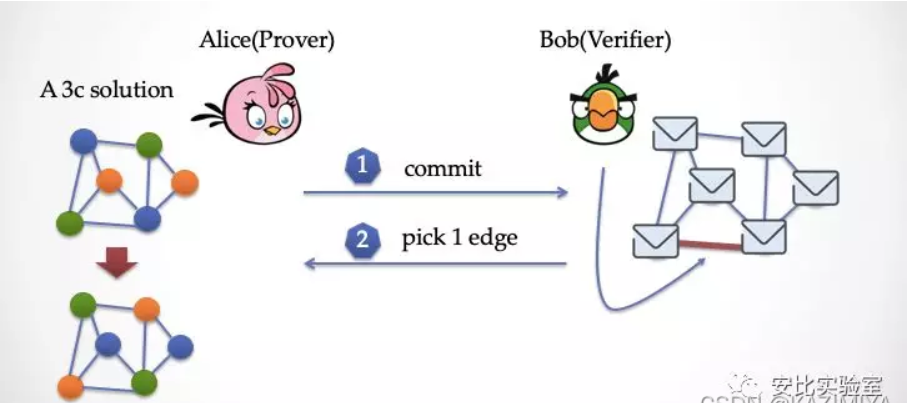
Alice 手里有一个地图三染色的答案，请见下图。这个图总共有 6 个顶点，9 条边。



现在 Alice 想证明给 Bob 她有答案，但是又不想让 Bob 知道这个答案。Alice 要怎么做呢？

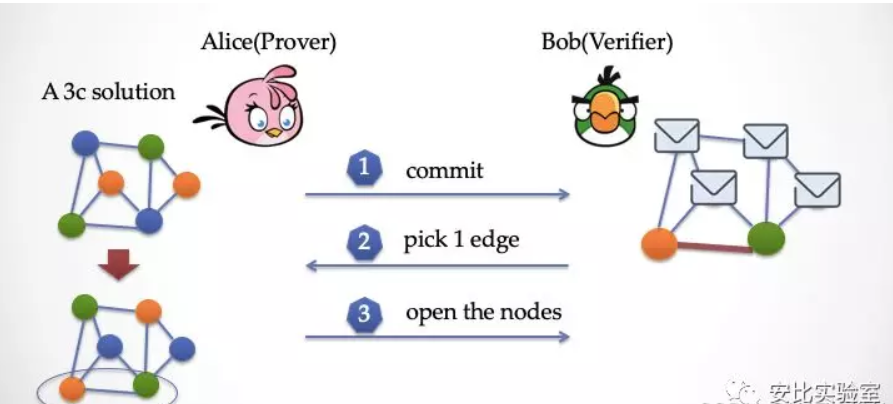
Alice 先要对染过色的图进行一些「变换」，把颜色做一次大挪移，例如把所有的绿色变成橙色，把所有的蓝色变成绿色，把所有的绿色变成橙色。然后 Alice 得到了一个新的染色答案，这时候她把新的图的每一个顶点都用纸片盖上，然后出示给 Bob 看。

看下图，这时候 Bob 要出手了（请见下图），他要随机挑选一条「边」，注意是随机，不让 Alice 提前预测到的随机数。

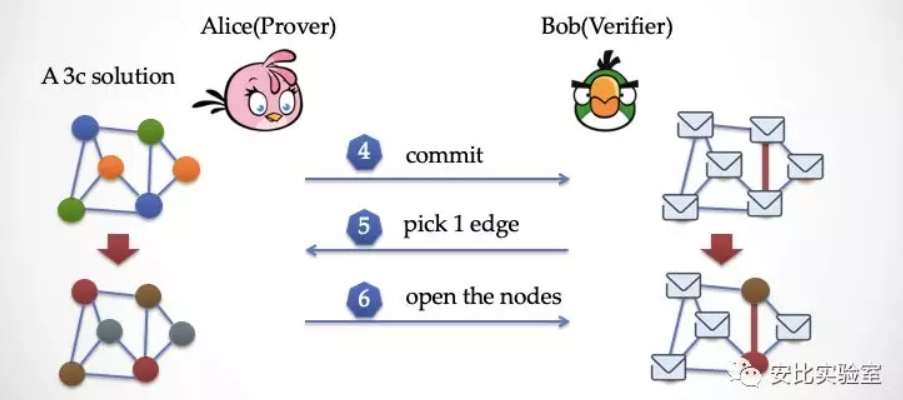


假设 Bob 挑选的是最下面的一条边，然后告诉 Alice。

这时候 Alice 揭开这条边两端的纸片，让 Bob 检查，Bob 发现这两个顶点的颜色是不同的，那么 Bob 认为这次检验同构。这时候，Bob 只看到了图的局部，能被说服剩下的图顶点的染色都没问题吗？你肯定觉得这远远不够，也许恰好 Alice 蒙对了呢？其它没暴露的顶点可能是胡乱染色的。

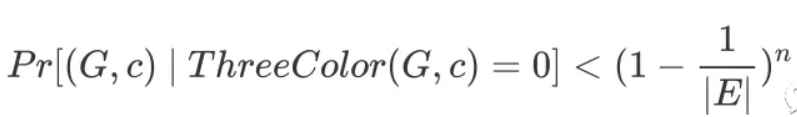


没关系，Bob 可以要求 Alice 再来一遍，看下图：



Alice 再次把颜色做一次变换，把蓝色改成紫色，改绿色改成棕色，把橙色改成灰色，然后把所有的顶点盖上纸片。然后 Bob 再挑选一条边，比如像上图一样，选择的是一条竖着的边，然后让 Alice 揭开纸片看看，如果这时候 Bob 再次发现这条边两端的顶点颜色不同，那么 Bob 这时候已经有点动摇了，可能 Alice 真的有这个染色答案。可是，两次仍然不够，Bob 还想再多来几遍。

那么经过反复多次重复这三个步骤，可以让 Alice 作弊并能成功骗过 Bob 的概率会以指数级的方式减小。假设经过 n 轮之后，Alice 作弊的概率为



这里 |E| 是图中所有边的个数， 如果 n 足够大，这个概率 Pr 会变得非常非常小，变得「微不足道」。

可是，Bob 每次看到的局部染色情况都是 Alice 变换过后的结果，无论 Bob 看多少次，都不能拼出一个完整的三染色答案出来。实际上，Bob 在这个过程中，虽然获得了很多「信息」，但是却没有获得真正的「知识」。

### **零知识证明的特点：**

根据零知识证明的定义和有关例子，可以得出零知识证明具有以下三个特点：

完备性。如果证明方和验证方都是诚实的，并遵循证明过程的每一步，进行正确的计算，那么这个证明一定是成功的，验证方一定能够接受证明方。

合理性。没有人能够假冒证明方，使这个证明成功。

零知识性。证明过程执行完之后，验证方只获得了“证明方拥有这个知识”这条信息，而没有获得关于这个知识本身的任何一点信息。

### **零知识证明的应用：**

1. 区块链

区块链是一种分布式账本技术，其核心特点是去中心化和不可篡改性。在一些使用场景中，区块链需要保证交易的隐私性和匿名性，零知识证明可以很好地实现这一点。例如，某些区块链系统中，零知识证明可用于隐私保护和货币流通证明等方面。

2. 数字身份验证

零知识证明技术可用于数字身份验证，可以提供高度的安全性和隐私性，还能有效地避免重放攻击等安全问题。用户可以使用零知识证明来证明自己的身份，例如在一些数字身份验证应用中，用户通过提供零知识证明来证明自己的年龄、身份或状态等信息。

3. 金融交易

在金融交易中，零知识证明可用于保护交易的隐私性和安全性，例如在保护用户银行账户信息和密码方面。用户可以使用零知识证明来证明自己的身份、资产和可信度等信息，同时也可以保护他们的个人隐私。

4. 数据共享

零知识证明技术可用于数据共享安全和隐私保护，使得数据可以在不泄露原始数据的情况下进行共享和使用。例如，在健康保险领域，零知识证明可用于验证用户的健康状况或疾病风险等信息，同时又不会泄露用户的个人隐私。

### **零知识证明中包含的数学问题：**

1. 离散对数问题

离散对数问题是指在有限域内选择一个数（一般是质数），如何寻找一个指数，使得这个数的指数次方等于另外一个给定数。例如，在大质数模下，某数的离散对数问题就可以表示为：解决一个模数质数p和底数g的幂等于给定数x的离散对数，即找到一个整数k，使得g^k ≡ x (mod p)。

离散对数问题是零知识证明算法的核心数学问题之一。在零知识证明中，证明者需要通过离散对数问题来证明交互双方之间共享的密钥，而同时并不需要将这个密钥直接传递给验证者。

2. 哈希碰撞问题

哈希碰撞问题是指寻找一对不同的输入，使得它们通过哈希函数映射后得到相同的输出。例如，有一组输入 (m1, m2)，通过哈希函数 H 映射后得到相同的输出值 H(m1) = H(m2)。

哈希碰撞问题是零知识证明算法的另一个核心问题。在零知识证明中，证明者需要通过哈希碰撞来证明某个信息的知晓，而同时并不需要直接透露该信息。

离散对数和哈希碰撞都是非常困难的数学问题，它们的难度使得零知识证明的算法具有了很高的安全性，使得证明者不需要透露信息，而验证者也无法猜到信息内容。尽管目前针对这些数学问题的攻击手段正在不断发展，但无疑，零知识证明算法在信息安全方面已经具有了非常强大的防护力。