# 信息安全数学基础探究报告

——2310558 张藐 密码与科学技术专业

**选题：**

1. 大整数分解问题
2. 离散对数问题
3. 椭圆曲线在密码学中应用

## 大整数分解问题

### 一、数学问题

**背景**：整数分解是数论中的一个经典难题，其核心数学问题是：给定一个足够大的合数 n（通常假设是两个未知大素数 p 和 q 的乘积，即 $n = pq$），如何有效地找到 p 和 q 的值。随着 n 的规模增大(特别是在密码学应用中，常为 1024 位、2048 位甚至更大的整数)，其计算复杂度呈指数级增长。目前并没有一种确定性的、多项式时间复杂度的算法能够解决所有大整数分解问题。例如，对于整数 n=15，很容易分解为 3×5，但当是一个非常大的数（如 1024 位或 2048 位的整数）时，可能的因子组合数量极其庞大，分解就变得极其困难。使得通过穷举法或常规数学方法在合理时间内找到其素因子几乎成为不可能任务。

## 二、大整数分解的算法思想

### 1. 试除法

这是最基本的分解方法。从最小的素数 2 开始，依次用素数去除待分解的大整数 n，如果能整除，则该素数就是 n 的一个因子，然后继续用该素数去除 n 的商，重复这个过程，直到商为素数为止。但对于大整数，这种方法效率极低，因为需要尝试大量的素数，计算量随着的增大而迅速增加。

### 2. 费马分解法

基于费马定理，对于一个奇数 n，尝试将其表示为两个平方数之差，即$n^2 = a^2 - b^2 = (a + b)(a - b)$。通过寻找合适的 a 和 b 来分解。具体做法是从$\sqrt{n}$开始，依次计算$a^2 - n$，判断其是否为完全平方数，如果是，则找到了 a 和 b，从而得到 n 的因子然而，这种方法对于大多数大整数也不实用，因为找到合适的 a 和 b 往往需要大量的计算和尝试)

### 3. 数域筛法

目前对于大整数分解较为有效的算法之一。它将整数分解问题转化为在代数数域上的计算问题，涉及到复杂的代数数论知识。其基本思想是构造一个代数数域，然后在这个数域中寻找合适的元素组合，通过一系列复杂的计算和筛选过程，找到的因子。数域筛法在处理非常大的整数（如 1024 位以上）时相对其他方法更有优势，但计算过程仍然非常复杂，需要大量的计算资源和时间。

## 三、特点

### 1. **计算复杂性**:

随着 **n** 的位数增加，分解 n 的计算量呈指数级增长。目前已知的分解算法在处

理大整数时效率较低，需要耗费大量的计算资源和时间。例如，使用一般的试除法来分解一个 100 位的整数可能需要很长时间，而对于 1024 位或更大的整数，即使使用先进的分解算法（如数域筛法等），计算时间也可能长达数年甚至更长时间。

**2. 分解的不确定性：**

对于一个大整数 **n**，没有一种确定性的、高效的方法可以快速找到其因子。不同的分解算法在不同情况下可能有不同的表现，而且目前还没有找到一种通用的、快速的分解大整数的算法。

## 四、在密码学中的应用

**1. RSA 公钥加密算法**

RSA 算法是基于大整数分解问题的典型应用。密钥生成时，选择两个大素数 p 和 q，计算 **n=pq** 和 **Ψ(n)=(p-1)(q-1)**，选择一个整数 **e**（满足 **1<e<Ψ(n)**且 e 与Ψ(n)互质），计算 d 使得 **ed ≡1 (mod Ψ(n) )**。**(e，n)**是公钥，**(d，n)**是私钥。**加密和解密**：加密时，对于明文 **m**（0≤ m ≤ n），计算密文$\boldsymbol{c = m^e \bmod n}$；解密时，计算明文$\boldsymbol{m = c^d \bmod n}$。由于攻击者难以分解 n 得到 p 和 q，从而无法计算Ψ(n)和 d，所以保证了加密的安全性。只要大整数分解问题仍然困难，RSA 算法就可以提供有效的加密和数字签名功能。如果有一天大整数分解问题被高效解决，那么 RSA 算法的安全性将受到严重威胁。

**2. 数字签名验证**

在 RSA 数字签名方案中，签名者使用私钥 **d** 对消息进行签名，计算 $s = m^d \bmod n$，**s** 就是签名。验证者收到签名 **s** 和消息 **m** 后，使用公钥**(e，n)**验证签名的有效性，

计算$m^1 = s^e \ mod\, n$，如果$m^1 = m$，则签名有效。这里的安全性同样依赖于大整数分解问题的困难性，因为攻击者如果能够分解，就可能获取私钥，从而伪造签名。大整数分解问题保证了只有拥有私钥的签名者才能生成有效的签名，而验证者可以通过公钥验证签名的真实性，防止消息被伪造和篡改。

**3. 密钥交换协议中的安全性保障**（部分协议）

一些密钥交换协议可能会间接利用大整数分解问题的困难性来增强安全性。例如，在某些混合密钥交换方案中，可能会使用 RSA 算法生成一些辅助密钥或参数，这些密钥或参数的安全性基于大整数分解问题。通过这种方式，即使在密钥交换过程中的其他部分可能存在一定风险，但由于大整数分解问题的保护，整体密钥交换的安全性得到了提升。攻击者难以通过分解相关大整数来获取关键信息，从而保证了密钥交换过程中共享密钥的保密性和完整性，使得通信双方能够安全地协商出用于后续通信的密钥。

**4. 密码分析与密码系统安全性评估**

在密码学研究中，大整数分解问题是评估密码系统安全性的重要指标之一。密码分析人员会研究大整数分解算法的进展，以此来判断基于大整数分解问题的密码系统（如 RSA）的安全性。如果发现新的分解算法或计算技术使得分解大整数变得更容易，就需要对现有的密码系统进行重新评估和改进。例如，如果某一天出现了一种新的算法能够在较短时间内分解 1024 位的整数，那么现有的使用 1024 位密钥的 RSA 系统就可能面临安全风险，需要升级到更长的密钥长度（如 2048 位或更高）来保持安全性。同时，对大整数分解问题的研究也有助于设计新的密码系统，通过避免或利用与大整数分解相关的数学结构来提高安全性

# 离散对数问题

## 一、离散对数问题的数学问题

离散对数问题是在特定的数学结构中定义的难题，给定一个有限循环群 G（群的元素个数为 n），一个生成元 g（**g∈G** 满足 **G={$g^0$,$g^1$,$g^2$,⋯,$g^{n-1}$}**）和群中的一个元素 **h**（**h∈G**）。找到一个整数 x（0≤x≤n-1）使得$g^x = h$。例如，在一个简单的整数模 **p** 乘法群 **Zp**（p 为素数）中，**g** 是 **Zp** 的一个生成元，对于给定的 **h∈Zp**，要找到 **x** 使得 $g^x \equiv h \ mod \ p$。

## 二、算法思想

### 1. 暴力搜索法

最直接的算法思想就是从开始 $x = 0$，依次计算$g^x$，并与 **h** 进行比较，直到找到满足$g^x = h$的 x。例如，在一个简单的群 G={$g^0$,$g^1$,$g^2$,⋯,g7}（假设 n=8）中，给定 g=2（假设 2 是生成元）和，从 x=0 开始计算$2^0 = 1 \neq 4;\ 2^1 = 2 \neq 4;\ 2^2 = 4$，此时找到 x=2。但在实际应用中，当群的规模 n 很大时（如在密码学中使用的大素数 p 对应的$Z_p$群），这种方法的计算量呈指数级增长，效率极低。

### 2. 大步小步算法（Baby - Step Giant - Step Algorithm）

该算法的思想是将搜索范围划分为两个部分来减少计算量。

**小步部分**：设 $m = \sqrt{n}$，计算并存储$g^i$（$i = 0,1,2,...m-1$）的值

**大步部分**：计算$h(g^{-m})^j$（$j = 0,1,2,...m-1$）的值。并检查是否与小步部分计算的值相等。如果相等，设 i 和 j 是满足$g^i = h(g^{-m})^j$的指数，则 $x = im + j$ 这种方法相比于暴力搜索法在一定程度上减少了计算量，但仍然面临计算复杂度较高的问题，尤其是当 n 非常大时。

**3. 指数计算法（Index Calculus Method）**

该方法基于对群中元素的分解和对数关系的计算。首先选择一个因子基，然后尝试将群中的元素表示为因子基中元素的乘积。对于给定的 g 和 h，通过计算一些与因子基相关的对数关系，逐步构建离散对数表，最终尝试从表中找到 x 使得 $g^x = h$。这在某些情况下比前两种方法更有效，但计算过程复杂，并且对于一些特殊结构的群可能效果不佳，同时在处理大整数规模的群时仍然面临计算困难。

## 二、特点

- **计算困难性**：当群的规模较大时（例如是一个很大的数），通过穷举所有可能的值来计算离散对数是非常困难的，计算量呈指数级增长。例如，对于一个 1024 位的素数，中的元素数量巨大，直接计算离散对数几乎不可能在合理时间内完成。
- **单向性**：已知 g、x 计算 $g^x$ 相对容易（可以通过快速指数算法等高效计算），但从 $g^x$ 反推 x 却很困难，这就是离散对数问题的单向性。这种单向性使得它在密码学中具有重要应用价值。
- **与群结构相关**：离散对数问题的难度与群的结构密切相关。不同类型的群（如椭圆曲线群、整数模乘法群等）其离散对数问题的计算难度不同。例如，椭圆曲线群上的离散对数问题在某些情况下比整数模乘法群上的离散对数问题更难计算，这使得椭圆曲线密码系统在相同安全强度下可以使用更短的密钥，提高了计算效率和存储效率。同时，群的生成元选择也会影响离散对数问题的难度，合适的生成元选择可以增加问题的复杂性，提高密码系统的安全性。

## 三、在密码学中的应用

### 1. 密钥交换协议（如 Diffie - Hellman 密钥交换）：

Diffie - Hellman 密钥交换协议基于离散对数问题的困难性。通信双方（如 Alice 和 Bob）可以在不安全的信道上共同协商出一个共享密钥。

例如，他们选择一个大素数 p 和 Zp 的一个生成元 g。Alice 选择一个秘密整数 a，计算 $A=g^a \bmod p$ 并发送给 Bob；Bob 选择一个秘密整数 b，计算 $B=g^b \bmod p$ 并发送给 Alice。然后 Alice 计算$\boldsymbol{K = B^a \bmod p}$，Bob 计算$\boldsymbol{K = A^b \bmod p}$，双方得到相同的共享密钥。由于攻击者难以计算离散对数，所以无法从 A 和 g 计算出 a，也无法从 B 和 g 计算出 b，从而保证了密钥交换的安全性。

### 2. 数字签名算法（如 ElGamal 数字签名）：

ElGamal 数字签名算法利用离散对数问题。签名者选择一个大素数 p 和生成元 g，以及私钥 x，计算公钥$\mathbf{y = g^x \bmod p}$。签名时，对于消息 m，选择一个随机数 k，计算$\mathbf{r = g^k \bmod p}$ 和$\mathbf{s = (m - xr)k^{-1} \bmod p}$（其中$k^{-1}$是 k 在模 p-1 下的逆元），(r，s)就是签名。验证者可以利用公钥和签名以及消息来验证签名的有效性。由于离散对数问题的困难性，攻击者难以伪造签名。

# 椭圆曲线在密码学中的应用

**背景**：椭圆曲线密码学在现代密码应用中具有诸多优势，其基于的椭圆曲线上的离散对数问题比传统的离散对数问题在计算上更困难，在相同安全强度下所需的密钥长度更短，从而提高了计算效率、降低了存储和传输成本，在保障信息安全方面发挥着重要作用。

## 一、密码原语

### 1. 点加运算

给定椭圆曲线上的两个点 P 和 Q（椭圆曲线方程一般形式为 y2=x3+ax+b，点 P=(x1，y1)，Q=(x2，y2) )，点加运算的结果是椭圆曲线上的另一个点 R。计算方法如下:

若 $P=Q$，则$\boldsymbol{k}=\frac{3x_1^2+a}{2y_1}$（这里是椭圆曲线方程中的系数)。

若 $P\neq Q$，则$\boldsymbol{k}=\frac{y_2-y_1}{x_2-x_1}$。

然后计算$x_3=k^2-x_1-x_2$，$y_3=k(x_1-x_3)-y_1$，得到$R=(x_3-y_3)$ 。

点加运算满足结合律，即(P+Q)+R=P+(Q+R)，这是椭圆曲线密码学中许多算法的基础运算之一。

### 2. 标量乘运算

对于椭圆曲线上的一个点 P 和一个整数 k（标量)，标量乘运算 kP 表示将点自身相加 k 次。计算标量乘运算可以通过反复使用点加运算来实现，但为了提高效率，通常使用一些优化算法，如二进制展开法等。例如，计算 5P，可以写成 P+P+P+P+P，但通过二进制展开 5=101，可以计算 5P=2P+P，先计算 $P_1$=2P，

然后再计算 $P=2P_1+P$，这样可以减少点加运算的次数。标量乘运算在椭圆曲线密码学中用于密钥生成、加密和解密等操作。

## 二、椭圆曲线在密码学中的具体应用

### 1. 密钥交换

**原理**：基于椭圆曲线的 Diffie - Hellman 密钥交换（ECDHE）利用椭圆曲线上点的运算性质。通信双方（如 Alice 和 Bob）选择一个椭圆曲线 E 和曲线上的一个基点 G。Alice 选择一个私钥α，计算 A=αG（A 是椭圆曲线上的一个点）并发送给 Bob；Bob 选择一个私钥 b，计算 B=bG 并发送给 Alice。然后 Alice 计算 αbG（通过计算 bA），Bob 计算αbG（通过计算αB），双方得到相同的共享密钥αbG。

**优势**：相比传统的 Diffie - Hellman 密钥交换基于离散对数问题（在有限域乘法群中），椭圆曲线 Diffie - Hellman 密钥交换基于椭圆曲线上的离散对数问题，在相同的安全强度下，椭圆曲线密码系统使用的密钥长度更短。例如，160 位的椭圆曲线密钥提供的安全性相当于 1024 位的 RSA 密钥，这使得计算量更小、加密和解密速度更快、占用更少的带宽和存储空间。

### 2. 数字签名

ECDSA（椭圆曲线数字签名算法）：

**签名过程**：签名者选择一个椭圆曲线 **E** 和基点 **G**，以及私钥 **d**，计算公钥 **Q=dG**。对于要签名的消息 **m**，先计算消息的哈希值 **h(m)**，然后选择一个随机数 **k**（k

与椭圆曲线的阶 n 互质)，计算**(x1,y1)=kG**，**r=x1 mod n**，**$s=k^{-1}( h(m) + dr )$mod n**（其中 $k^{-1}$是在模下 n 的逆元），签名为**(r，s)**。

验证过程：验证者收到消息 **m**、签名**(r，s)**和公钥 **Q** 后，

计算 **$w=s^{-1}$mod n**，**$u_1$=h(m)w mod n**，**$u_2$=rw mod n**，

然后计算（**$x_0$，$y_0$)=$u_1$G+$u_2$Q**，如果 **r=$x_0$ mod n**，则签名有效。

椭圆曲线数字签名算法在保证安全性的同时，由于密钥长度较短，签名和验证速度相对较快，适用于资源受限的环境，如移动设备等。

### 3. 椭圆曲线加密，解密（ECC)：

**加密过程**：发送方选择一个椭圆曲线 **E**、基点 **G** 和接收方的公钥 **Q**（接收方选择私钥并计算 **Q=dG**)。

对于明文 **m**，将其编码为椭圆曲线上的一个点 **Pm**（如果 m 不能直接编码为点，可能需要使用一些编码技巧）

然后选择一个随机数 **k**，计算 **C1=kG** 和 **C2=Pm+kQ**，密文为**(C1，C2)**。

**解密过程**：接收方使用私钥 **d** 解密，计算 **Pm=C2-d C1**，然后将点 **Pm** 解码得到明文 **m**。椭圆曲线加密在相同安全级别的情况下，相比传统的 RSA 等加密算法，计算复杂度更低，加密后的密文长度更短，提高了加密效率和传输效率。

### 4. 身份认证

椭圆曲线密码系统可以用于构建身份认证方案。例如，在一个网络系统中，用户可以使用基于椭圆曲线的私钥对特定信息进行签名，服务器通过验证签名（使用

用户的公钥）来确认用户的身份。由于椭圆曲线数字签名的安全性和高效性，这种身份认证方式可以在保证安全的前提下快速完成认证过程，防止身份伪造和欺骗。同时，椭圆曲线密码系统的密钥管理相对简单，因为密钥长度较短，便于存储和传输，适合在各种需要身份认证的场景中应用，如电子商务、在线银行等领域。

## 三、椭圆曲线在密码学中包含的数学问题：

### 1. 离散对数问题（ECDLP）

给定椭圆曲线上的一个基点 G 和另一个点 Q（Q 是 G 的若干倍，即 Q=kG，k 为整数），椭圆曲线上的离散对数问题就是要找到 k 的值。当椭圆曲线的参数选择合适（如曲线的阶较大等）时，这个问题在计算上是非常困难的。目前已知的求解椭圆曲线上离散对数问题的算法在计算复杂度上很高，对于较大规模的椭圆曲线，通过暴力计算或其他常规方法几乎不可能在合理时间内找到的值。椭圆曲线密码学中的密钥交换、数字签名等算法的安全性都依赖于椭圆曲线上离散对数问题的困难性。如果这个问题能够被轻易解决，那么基于椭圆曲线的密码系统将被攻破。

### 2. 椭圆曲线的点群结构问题

椭圆曲线上的点在特定的运算（点加运算）下构成一个群，研究椭圆曲线的点群结构对于理解椭圆曲线密码学的安全性和性能至关重要。例如，需要确定椭圆曲线的阶（即点群中元素的个数），以及点群是否满足某些数学性质（如循环群性

质等）。**一些攻击方法可能会利用椭圆曲线点群结构的弱点来试图破解密码系统。**在选择椭圆曲线用于密码学应用时，需要确保所选择的曲线具有良好的点群结构，使得基于该曲线的密码算法能够抵抗各种已知和潜在的攻击。例如，超奇异椭圆曲线在某些情况下可能存在安全性弱点，因为其点群结构具有一些特殊性质，容易被攻击者利用，所以在实际应用中通常会**避免使用超奇异椭圆曲线**，而选择更安全的普通椭圆曲线。同时，对于椭圆曲线的参数选择（如曲线方程中的系数a、b 等）也会影响点群结构，需要谨慎选择以保证密码系统的安全性。