SIDH 初步研究报告

M0D1

SCNU CS19

2022.06.16

Table of Contents

Table of Contents

困难问题

标准的同源计算问题：给定有限域上的两条椭圆曲线 E_1, E_2，满足 $\#E_1(F_q) = \#E_2(F_q)$，找到 E_1 与 E_2 之间的同源是困难的。

超奇异椭圆曲线

一条椭圆曲线要么是超奇异的，要么是一般的 (ordinary)。
超奇异椭圆曲线间的 isogeny 构成一个非常大而且不交换的自同态环，
而上述的困难问题的计算与自同态环的计算密切相关 [1]。
因此自同态环越大、越复杂，敌手就越难计算出 isogeny。

Difinition

若 E_1, E_2 之间存在 isogeny，则称 E_1, E_2 处在同一个 isogeny class

Fact

处在同一个 isogeny class 的椭圆曲线要么都是超奇异的，要么都是一般的。

以上结论告诉我们，从一条超奇异椭圆曲线出发不断作 isogeny，最后得到的依然是超奇异椭圆曲线。

isogeny

Difinition

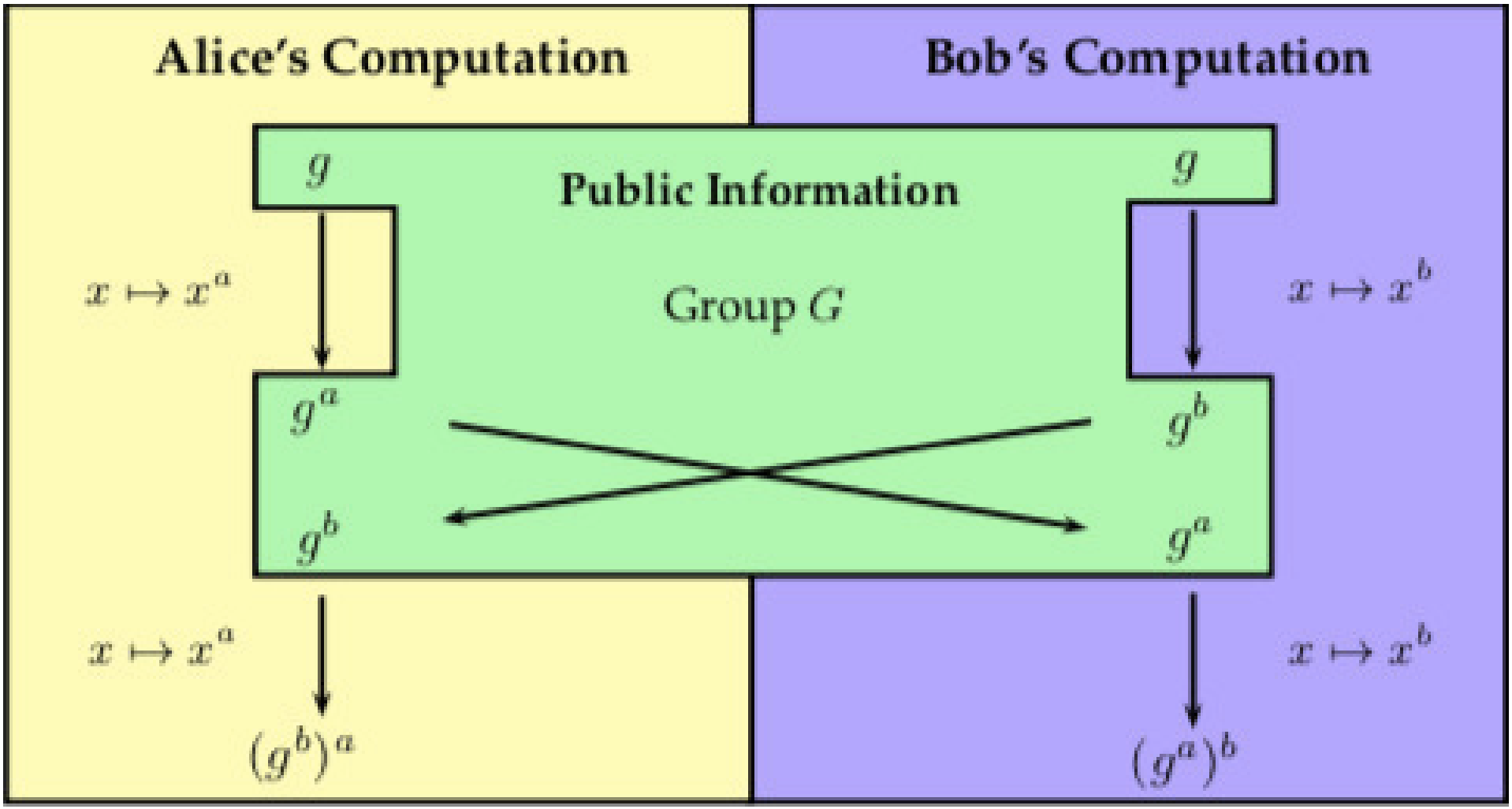
A rational map $\phi : E_1 \to E_2$ is called an *isogeny* if $\phi(O_1) = O_2$

所有的 isogeny **都是满射**。
此外，isogeny **是一种群同态**，**即** $\phi(P + Q) = \phi(P) + \phi(Q)$.

$\phi(P) = \phi((x, y)) = (\phi_1(x), \phi_2(y))$
所谓的有理映射，也就是说 ϕ_1, ϕ_2 是一些代数分数，例如

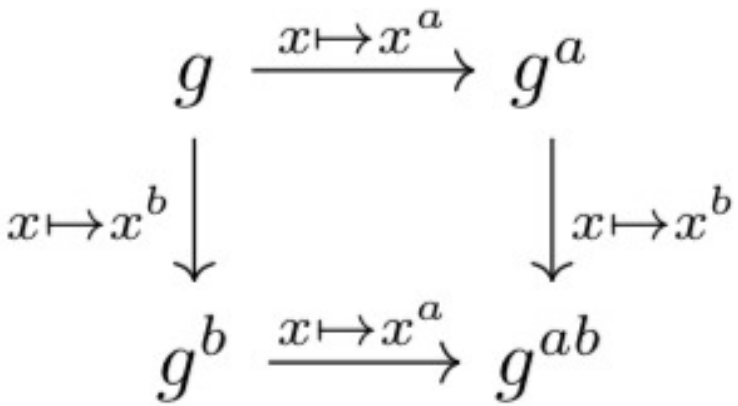
$$\frac{x^3 + 4x + 1}{2x + 5}$$

DH 的一般过程


Alice's Computation
Bob's Computation
Public Information
Group G
g
g
$x \mapsto x^a$
$x \mapsto x^b$
g^a
g^b
g^b
g^a
$x \mapsto x^a$
$x \mapsto x^b$
$(g^b)^a$
$(g^a)^b$

DH 的高度抽象

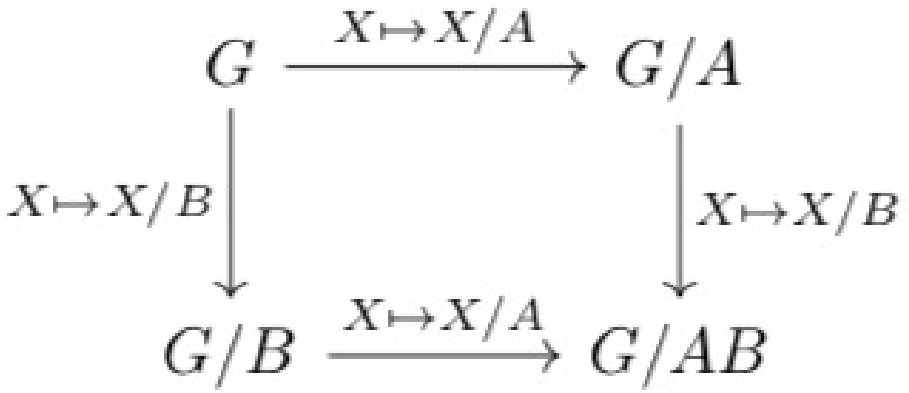
DH 的关键在于：不同的计算路径可以得到相同的计算结果。这暗示了负责运算的映射组成了一个交换环。



DH 还要求计算是容易的，即 $x \to x^a$ 是容易的。实际上，这里并没有要求**直接**计算它是容易的，而是把它看作一系列群操作的复合，只要每次群操作都是容易的，那么得到最终结果就是容易的。这暗示了映射是可分的。

SIDH 的抽象

联想刚才的 DH，可以得到一种新的密钥协商方案：



其中 G 是一个乘法群，A 和 B 是 G 的正规子群，映射即是作商群。最后得到结果 G/AB，AB 理解为 $A \cap B$，必须承认这个方案并不完善，因为要得到 $A \cap B$ 就必须共享 A 和 B，也就是说 Alice 和 Bob 都不能隐藏自己的秘密。不过，数学家的理想就是要构造出这样一幅交换图，且满足以下两种属性：

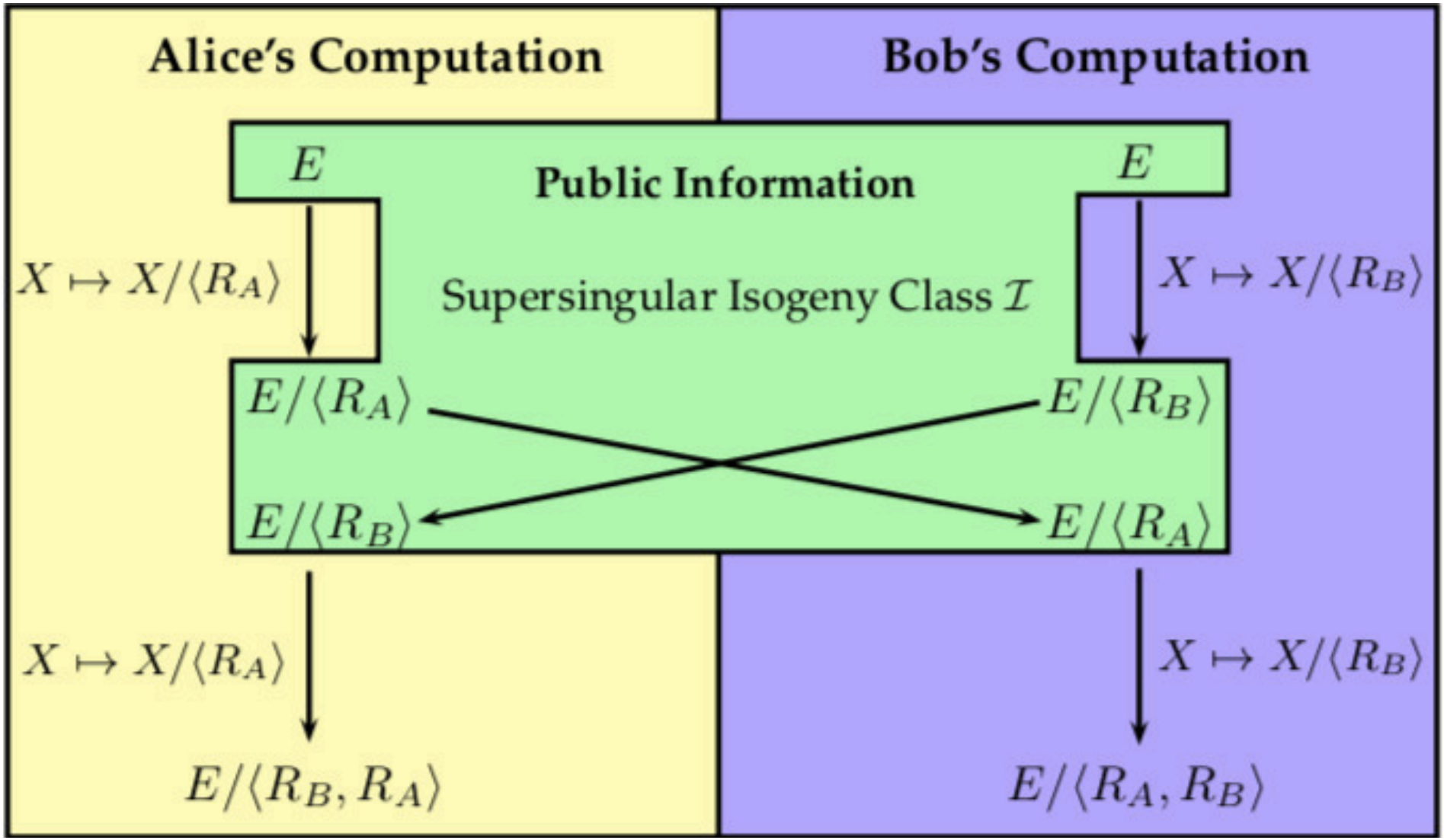
- 给定 G、G/A 和 B，容易计算 G/AB；给定 G，G/B 和 A，容易计算 G/AB
- 给定 G、G/A（或 G/B），求 A（或 B）是困难问题，即不存在量子多项式时间算法。

在 SIDH 中，G 定义为超奇异椭圆曲线群
计算 G/AB，其实就是计算 $(E/\langle R_A\rangle)/\phi_A(\langle R_B\rangle)$ 与 $(E/\langle R_B\rangle)/\phi_B(\langle R_A\rangle)$ (可以证明这两者同构)。上文提到，DH 所用的映射必须是可交换的，但出于安全性的考虑，超奇异 isogeny 并不可交换。这表明想要计算 $(E/\langle R_A\rangle)/\phi_A(\langle R_B\rangle)$，必须要 Alice 和 Bob 提供额外信息，好在 isogeny 是一种群同态，具体过程如下。

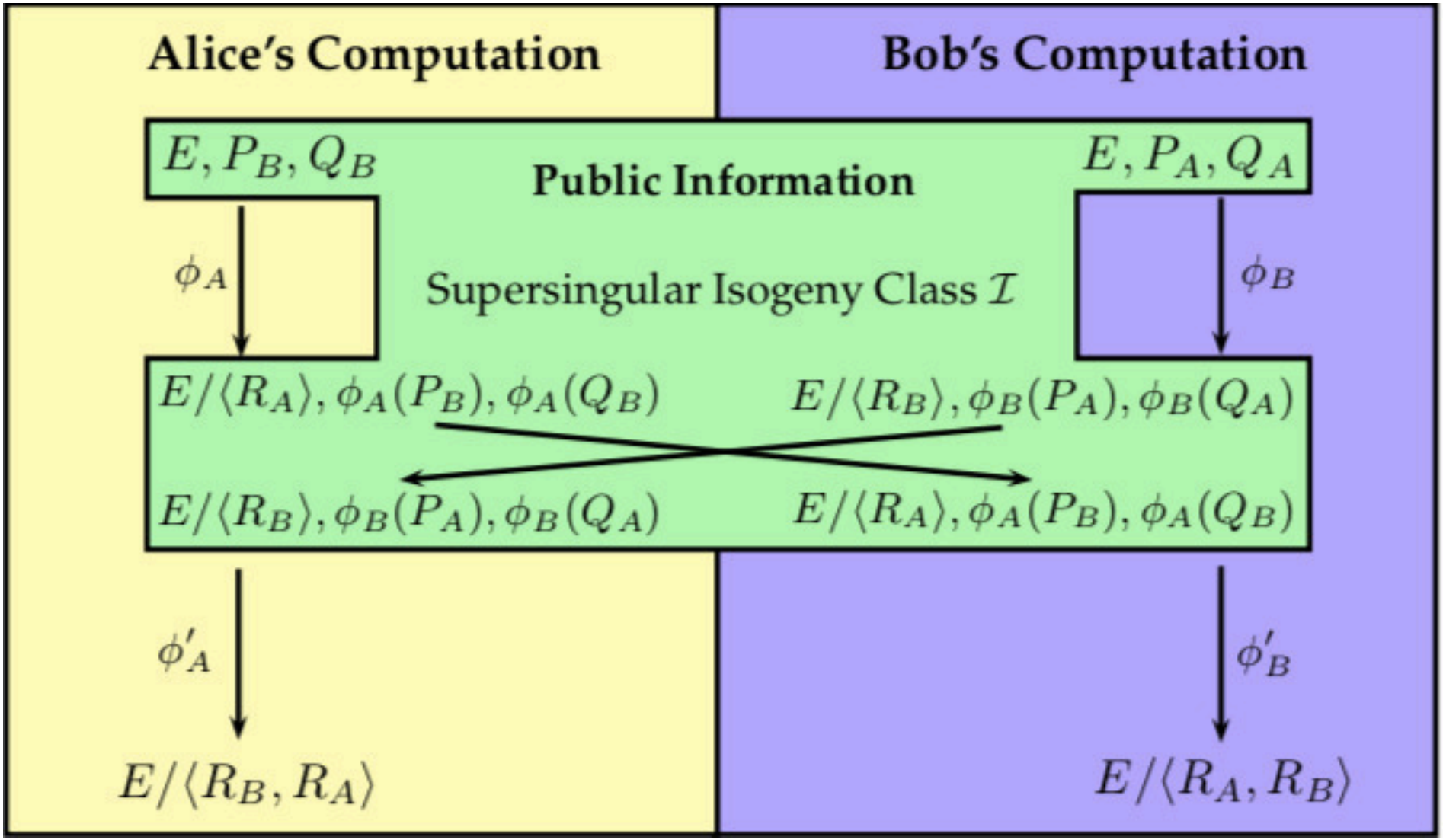
E 所在域

SIDH 选用的椭圆曲线均落在域 F_{p^2} 上，其中 $p = l_A^{e_A} l_B^{e_B} f \pm 1$,p 是素数,$l_A, l_B$ 也是素数,e_A, e_B, f 是正整数，$l_A^{e_A} \approx l_B^{e_B}$。在实践中常取 $l_A = 2, l_B = 3, f = 1$。p 这样选取保证了在 F_{p^2} 中可以计算 isogeny。
域 F_{p^2} 中的元素长这样子：$u + iv$, 其中 $u, v \in F_p, i^2 = -1$
E 的阶是 $(l_A^{e_A} l_B^{e_B} f)^2$，需要注意的是，$E/\langle R_A \rangle$ 其实并不是一个椭圆曲线群，但鉴于它与 isogeny class 中某个椭圆曲线群同构（第一同构定理 [4]），我们可以把它视为一个椭圆曲线群，记作 E_A。

SIDH 简化过程图


Alice's Computation
Bob's Computation
E
Public Information
E
$X \mapsto X/\langle R_A \rangle$
Supersingular Isogeny Class $\mathcal{I}$
$X \mapsto X/\langle R_B \rangle$
$E/\langle R_A \rangle$
$E/\langle R_B \rangle$
$E/\langle R_B \rangle$
$E/\langle R_A \rangle$
$X \mapsto X/\langle R_A \rangle$
$X \mapsto X/\langle R_B \rangle$
$E/\langle R_B, R_A \rangle$
$E/\langle R_A, R_B \rangle$

SIDH 细化过程图


Alice's Computation
Bob's Computation
E, P_B, Q_B
Public Information
E, P_A, Q_A
ϕ_A
Supersingular Isogeny Class $\mathcal{I}$
ϕ_B
$E/\langle R_A \rangle, \phi_A(P_B), \phi_A(Q_B)$
$E/\langle R_B \rangle, \phi_B(P_A), \phi_B(Q_A)$
$E/\langle R_B \rangle, \phi_B(P_A), \phi_B(Q_A)$
$E/\langle R_A \rangle, \phi_A(P_B), \phi_A(Q_B)$
ϕ'_A
ϕ'_B
$E/\langle R_B, R_A \rangle$
$E/\langle R_A, R_B \rangle$

SIDH 具体步骤

以 Alice 为例

1. **公共参数**:$E = l_A^{e_A} l_B^{e_B} f \pm 1$, **线性无关的两点** P_A, Q_A
2. **私钥**: **一个群阶为** $l_A^{e_A}$ **的循环子群，表示为** $\langle R_A \rangle$, **其中** $R_A = m_A P + n_A Q$ **是生成元**； isogeny:$\phi_A : E \to E/\langle R_A \rangle$
3. **计算**： **为 Bob 计算** $\phi_A(P_B), \phi_A(Q_B)$
4. **交换**： **从 Bob 拿到** $E/\langle R_B \rangle, \phi_B(P_A), \phi_B(Q_A)$
5. **计算**： $\phi_B(R_A) = \phi_B(m_A P + n_A Q) = m_A \phi_B(P_A) + n_A \phi_B(Q_A)$
6. **得到共享曲线**： isogeny:$\phi_{BA} : E/\langle R_B \rangle \to (E/\langle R_B \rangle)/\langle \phi_B(R_A) \rangle$， **即** E_{BA}
7. **计算** j-invariant$((E/\langle R_B \rangle)/\langle \phi_B(R_A) \rangle)$, **即** j-invariant$(E_{BA})$

注： **两条椭圆曲线的** j-invariant **相等当且仅当它们同构。**

Table of Contents

倍点运算

为了计算上方便，点坐标多采取三维坐标。SIDH 中涉及的倍点运算主要包括计算 R = mP + nQ 和 IR。对于计算 R，一般转为计算 $P + m^{-1}nQ$。在 [3] 中介绍了利用 weil pairing 得到 P 和 Q 的方法（以 P_A, Q_A 为例）：

生成公钥 P，Q

不断作如下操作：随机生成点 P 并乘以 $(l_B^{e_B} f)^2$ 得到点 P'——直到 P' 的阶为 $l_A^{e_A}$，令 $P_A = P'$；依此法不断生成点 Q'——直到 $e_{l^e}(P_A, Q')$ 的阶为 $l_A^{e_A}$，令 $Q_A = Q'$。

其中 $e_{l^e}(P_A, Q')$ 代表 weil pairing。当然还有不使用 pairing 生成公钥的方法 [5]

mP + nQ 的计算

kernel 由生成元 R = mP + nQ 生成，而 R 的计算一般转为计算 $R' = P + m^{-1}nQ$, 其中 $m * m^{-1} = 1 \ mod \ l^e$。可以证明 [4]，这两者生成同一个循环群。而计算 R' 就是一次椭圆曲线的倍点运算和一次点加，显然比计算 R 快（大概率）。

isogeny 的计算

isogeny 由其 kernel 唯一确定。在知道 kernel 后可以利用 Vélu 公式得到 isogeny。
isogeny 的计算十分复杂，甚至因此衍生出了 isogeny 计算策略的计算。在 SIDH 中，需要考虑的是 $/^e$-isogeny 的计算，核心思想就是将它分为 e 个 l-isogeny 的复合。De Feo 等人 [3] 提出了 3 种基本方法：基于同源的方法、基于标量乘的方法和最优策略方法。其中，基于同源的方法是在复合过程用计算同源的方式去计算每次同源的核生成点；而基于标量乘的方法是在复合过程中用基于标量乘的方式计算每次同源的核生成点；最优策略算法是结合前两种方法的优势提出的一种新的方法，通过比较每次复合中标量乘计算和同源计算的耗费量，并结合最优策略的性质，即一个策略是最优的当且仅当其分解为两个最优子策略，得到最优路径，利用该路径来计算每次同源的核生点.3 种方法相比较而言，第 3 种方法的计算效率是最优的。

isogeny 计算策略

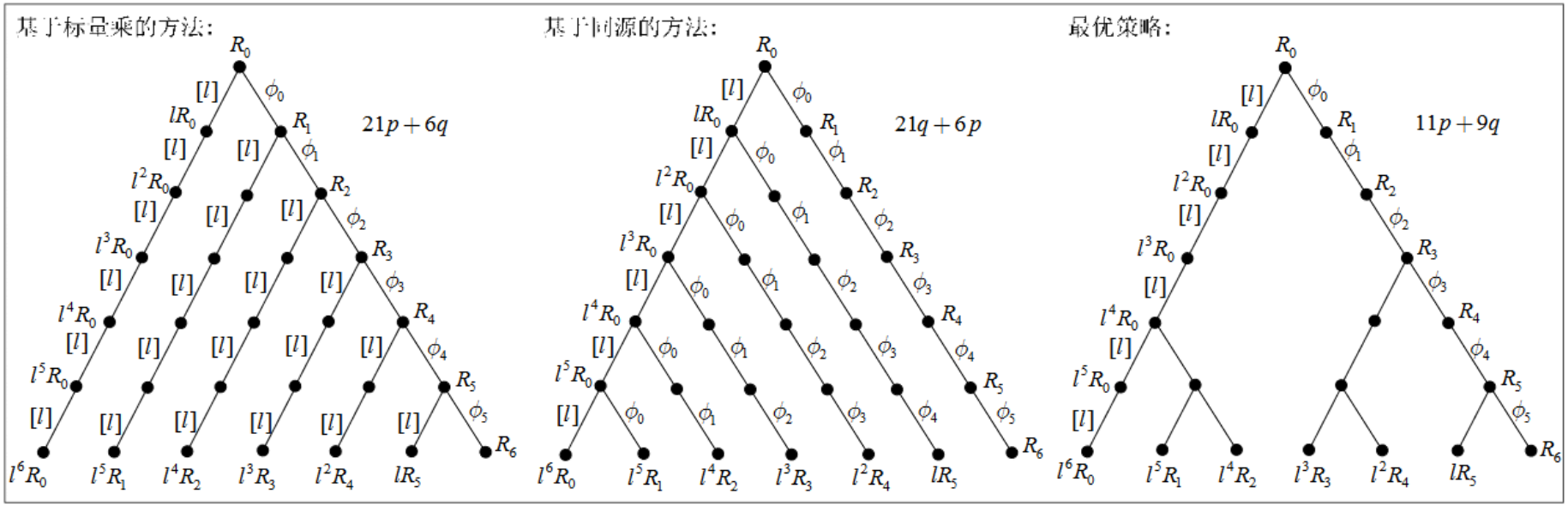


Fig.2 Compute ℓ^7-isogeny

上图的根节点是 l^e-isogeny 的 kernel 的生成元，每个叶子节点都是一个 l-isogeny 的 kernel 的生成元，这样它们复合起来能得到最终的 l^e-isogeny 和 image。isogeny 计算策略的目标就是从根节点出发，以最小代价得到所有叶子节点。

$$\{O\} = \langle 2^{e_A} R_A \rangle \subset \langle 2^{e_A - 1} R_A \rangle \subset \ldots \subset \langle R_A \rangle = A$$

Vélu's Formulas

Vélu's Formulas 是一种根据 kernel 算 isogeny 的具体方法：
https://www.mariascrs.com/2020/11/07/velus-formulas.html
在实践中，经过拆分后每个 kernel 的阶为 l_A, l_B，即 2 或 3，因此计算速度很快。当然还有其他计算 isogeny 的方法（Kohel's Algorithm，但也是通过 kernel 计算）

运行速度

我用个人电脑粗略地测试了一下 SIDH 的 100 次的平均运行速度，大概在 250000ms
而 ECDH 的 100 次的平均运行速度，大概在 35000ms 代码分别来自
https://github.com/sidh-crypto/sidh-c-reference
https://github.com/kokke/tiny-ECDH-c

程序流程图

自己画了个程序流程图，具体步骤参考的是 [3]
https://m0d1.top/2022/05/10/sidhcode/

Table of Contents

代码优化

代码优化的几个切入点 [2, 6]：

1. 有限域的运算：任何加速底层的有限域的基本算术方法都能加快 SIDH 的实现
2. 曲线及坐标类型：不同的曲线类型以及不同的坐标形式，其上的点加、倍点、同源和同源曲线的代数操作的计算量是不同的
3. 压缩公钥和恢复公钥的计算量：与所选曲线和坐标类型有关
4. 生成元的计算：线性无关的判断方法目前主要采用双线性对的计算
5. 倍点运算：SIDH 中涉及到的标量乘主要的形式包括 P+kQ 和 lR，其中,P、Q、R 均为椭圆曲线上的点,k、l 均为正整数. 对于 P+kQ 的计算，主要是利用 Ladder 算法进行优化计算. 对于 lR 的计算，目前的方法主要是利用加法链进行优化计算.
6. isogeny 和同源曲线计算：在 SIDH 中，Alice 和 Bob 均需要计算 l^e-isogeny 和同源曲线。对于 l^e-isogeny 的计算，需要进行 e 次 l-isogeny 的复合。显然复合次数越少，计算速度越快。对于同源曲线的计算，当 l 较大时，直接利用同源曲线公式计算效率较低。

Reference I

[1] Galbraith, S. D., Petit, C., Shani, B., and Ti, Y. B.
On the security of supersingular isogeny cryptosystems.
In *International Conference on the Theory and Application of Cryptology and Information Security* (2016), Springer, pp. 63–91.

[2] Jao, D.
Supersingular isogeny key encapsulation-spec.
[EB/OL].
`https://sike.org/files/SIDH-spec.pdf` Accessed October 1, 2020.

[3] Jao, D., and Feo, L. D.
Towards quantum-resistant cryptosystems from supersingular elliptic curve isogenies.
In *International Workshop on Post-Quantum Cryptography* (2011), Springer, pp. 19–34.

Reference II

[4] LB Wang.
A concrete introduction to number theory and algebra.
https://github.com/lbwang/CINTA-cn/blob/master/CINTA-cn.pdf
Accessed April 26, 2022.

[5] Pereira, G. C., and Barreto, P. S.
Isogeny-based key compression without pairings.
In *IACR International Conference on Public-Key Cryptography* (2021), Springer, pp. 131–154.

[6] 黄艳; 张方国;.
椭圆曲线同源的有效计算研究进展.
软件学报 *32* (2021), 1151–1164.

Thank you!